Universidad Nacional Autónoma de México
Facultad de Ciencias
Análisis de Software Malicioso 2017-1
Proyecto01

Profesor: Jonathan Banfi Vázquez
Alumno: José Ricardo Rodríguez Abreu

1. Implementación

La implementación de este proyecto fue generada en el lenguaje de programación C con uso de bibliotecas Windows.h y las bibliotecas de uso general de C.

Para el progreso de la implementación y comodidad del programador, decidí mostrar el desarrollo del proyecto con apoyo de la herramienta Git y el servicio de github.com.

Desde el sitio https://github.com/ricardorodab/malwareEjemplo podemos acceder al repositorio del proyecto que muestra con 6 commits con el procedimiento modular usado para la realización del programa.

1. Commit 1: 61febe4650796f130cfc3ed6f8a7c18a7f734cc0
   (a) Creación del archivo malwareEjemplo.c
   (b) Creación de un método main
   (c) Redacción de la licencia de uso del software con propósitos educativos
   (d) Estructuración de la portabilidad del software con la definición de variables del sistema para localización de ambientes tipo Linux, Windows, Unix y Apple.
2. Commit 2: 02a7ed23f78703ae5d8b4dd1ef5b75ebfc43dcdc
   (a) Modificación del archivo MAKEFILE para compilar en sistemas Unix
   (b) Creación funciones:
      i. cmplocation - Compara la ubicación del archivo.
      ii. mueveArchivo - Mueve el archivo a la variable %tmp%
      iii. ejecutarDeNuevo - Llama a una instancia del programa en la nueva ubicación
      iv. so - Es la función principal sobrecargada en todos los Sistemas Operativos posibles

(c) Finaliza el punto uno del los requisitos del proyecto: Ruta particular de ejecución.

3. Commit 3: f165eb04be1e014612270c2f8c66b1ed0d3f81b2

   (a) El malware genera una llave en HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentV
   (b) Es modificado para que la ejecución del programa pueda ser llevado acabo mediante dos medios: Desde el directorio con la linea de comandos (cmd) o con doble click.
   (c) Creación de funciones:
      i. getNombre - En lugar de siempre sobrecargar la llamada de las funciones para conocer el nombre del programa, se moduló y se creó esta función que nos regresa el nombre.
      ii. registry_app - Crea la llave de la persistencia.

4. Commit 4: bd869ae642796fcb88a3bc2be153f53dc58454f5

   (a) Creamos la conexión a el sitio del dominio y la petición HTTP
   (b) Creación de funciones:
      i. casoMalicioso - Realiza llamadas a funciones del sistema para probar las posibles acciones maliciosas a realizar.
      ii. ejecutaAccion - Dado una cadena, decide que acción debe tomar.
      iii. order66 - Es la función que se conecta al servidor y nos regresa el archivo comando.txt en forma de cadena.

5. Commit 5: e1b39eebc1065ab76ce5cd22d2ae31887618a129

   (a) El archivo makefie es modificado quitando los comentarios.
   (b) El malware además de la llave que generaba ahora genera la de persistencia real en HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Ru
   (c) Se crea el archivo comando.txt de ejemplo
   (d) Se termina de comentar el código del archivo malwareEjemplo.c
   (e) En general casi todas las funciones son modificadas para el correcto comportamiento final del programa.
   (f) Se decidió la creación de una función llamada mySystemShutdown para pedir permisos para las acciones maliciosas.
   (g) Se agregaron hasta 7 casos de acciones maliciosas y una acción de seguridad.
   (h) Se creó el binario malwareEjemplo.exe
   (i) Se creó el archivo malwareEjemplo.exe.manifest para que al ser ejecutado el .exe reciba una petición de permisos de administrador.

6. Commit 6: Readme

   (a) Agrega un Readme y estructura al repositorio.

2. Referencias:

Si alguna función o fragmento de código fue sacado de internet, éste se encuentra propiemente commentado con el link y las convenciones de las licencias GNU y ANSI C.

- http://stackoverflow.com/
- https://msdn.microsoft.com/en-us/library/windows/desktop