SEGURIDAD EN REDES Y SISTEMAS INFORMÁTICOS

Ana González (Kao)

UNIDAD 5. Seguridad en aplicaciones

Ana González (Kao)

ÍNDICE UNIDAD 5

UNIDAD 5. SEGURIDAD EN APLICACIONES.

Ana González (Kao)

1. Fases de desarrollo del software

Ya que en este tema se va a hablar de desbordamiento de pilas y programación segura es justo antes conocer cuales son las fases de desarrollo del software para poder entender cómo y porqué llegan a producirse los fallos en la programación que luego dan lugar a exploits.

Análisis de requisitos

Extraer los requisitos de un producto de software es la primera etapa para crearlo.

Mientras que los clientes piensan que ellos saben lo que el software tiene que hacer, se requiere de habilidad y experiencia en la ingeniería de software para reconocer requisitos incompletos, ambiguos o contradictorios.

Ana González (Kao)

El resultado del análisis de requisitos con el cliente se plasma en el documento ERS, Especificación de Requerimientos del Sistema, cuya estructura puede venir definida por varios estándares, tales como CMM-I. Asimismo, se define un diagrama de Entidad/Relación, en el que se plasman las principales entidades que participarán en el desarrollo del software. La captura, análisis y especificación de requisitos (incluso pruebas de ellos), es una parte crucial; de esta etapa depende en gran medida el logro de los objetivos finales. Se han ideado modelos y diversos procesos de trabajo para estos fines. Aunque aun no está formalizada, ya se habla de la Ingeniería de Requisitos. La IEEE Std. 830-1998 normaliza la creación de las Especificaciones de Requisitos Software (Software Requirements Specification).

Diseño y arquitectura

Se refiere a determinar como funcionará de forma general sin entrar en detalles. Consiste en incorporar consideraciones de la implementación tecnológica, como el hardware, la red, etc. Se definen los Casos de Uso para cubrir las funciones que realizará el sistema, y se transforman las entidades definidas en el análisis de requisitos en clases de diseño, obteniendo un modelo cercano a la programación orientada a objetos.

Programación

Reducir un diseño a código puede ser la parte más obvia del trabajo de ingeniería de software, pero no es necesariamente la porción más larga. La complejidad y la duración de esta etapa está íntimamente ligada al o a los lenguajes de programación utilizados.

Pruebas

Consiste en comprobar que el software realice correctamente las tareas indicadas en la especificación. Una técnica de prueba es probar por separado cada módulo del software, y luego probarlo de forma integral para así llegar al objetivo. Se considera una buena practica el que las pruebas sean efectuadas por alguien distinto al desarrollador que la programó, idealmente un área de pruebas; sin perjuicio de lo anterior el programador debe hacer sus propias pruebas. En general hay dos grandes formas de organizar un área de pruebas, la primera es que esté compuesta por personal inexperto y que desconozca el tema de pruebas, de esta forma se evalúa que la documentación entregada sea de calidad, que los procesos descritos son tan claros

Ana González (Kao)

que cualquiera puede entenderlos y el software hace las cosas tal y como están descritas. El segundo enfoque es tener un área de pruebas conformada por programadores con experiencia, personas que saben sin mayores indicaciones en que condiciones puede fallar una aplicación y que pueden poner atención en detalles que personal inexperto no consideraría.

Documentación

Todo lo concerniente a la documentación del propio desarrollo del software y de la gestión del proyecto, pasando por modelaciones (UML), diagramas, pruebas, manuales de usuario, manuales técnicos, etc; todo con el propósito de eventuales correcciones, usabilidad, mantenimiento futuro y ampliaciones al sistema.

Mantenimiento

Mantener y mejorar el software para enfrentar errores descubiertos y nuevos requisitos. Esto puede llevar más tiempo incluso que el desarrollo inicial del software. Alrededor de 2/3 de toda la ingeniería de software tiene que ver con dar mantenimiento. Una pequeña parte de este trabajo consiste en arreglar errores, o *bugs*. La mayor parte consiste en extender el sistema para hacer nuevas cosas. De manera similar, alrededor de 2/3 de toda la ingeniería civil, arquitectura y trabajo de construcción es dar mantenimiento.

Y si se tiene tan bien planificado el desarrollo, ¿porque se producen errores?:

1. Presión excesiva en el tiempo de ejecución.

2. Cambios en las especificaciones del proyecto.

3. Ausencia de especificaciones técnicas.

4. Ausencia de un proyecto documentado o correctamente documentado.

5. y 6. Demasiadas innovaciones superficiales.

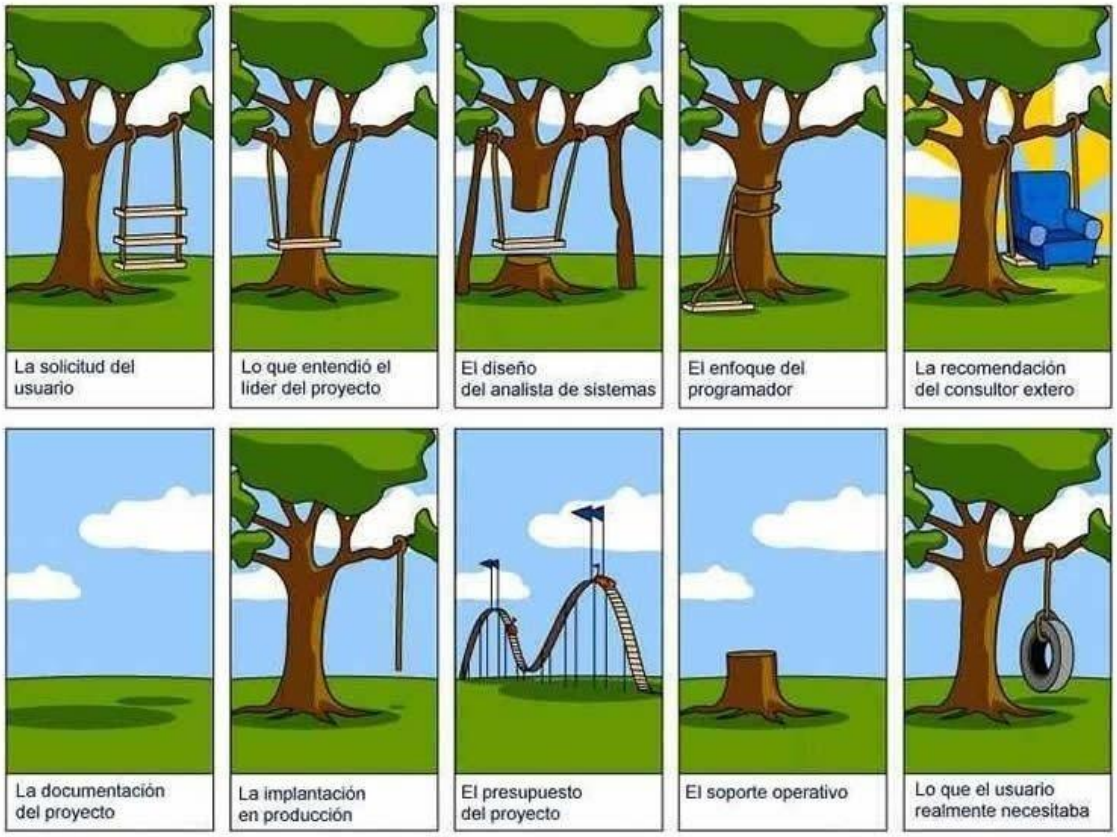
7. Añadir en el desarrollo funcionalidades que no estaban originalmente. (features creep ó requirements creep).

8. Ausencia del método científico.

9. Ignorar lo obvio.

Ana González (Kao)

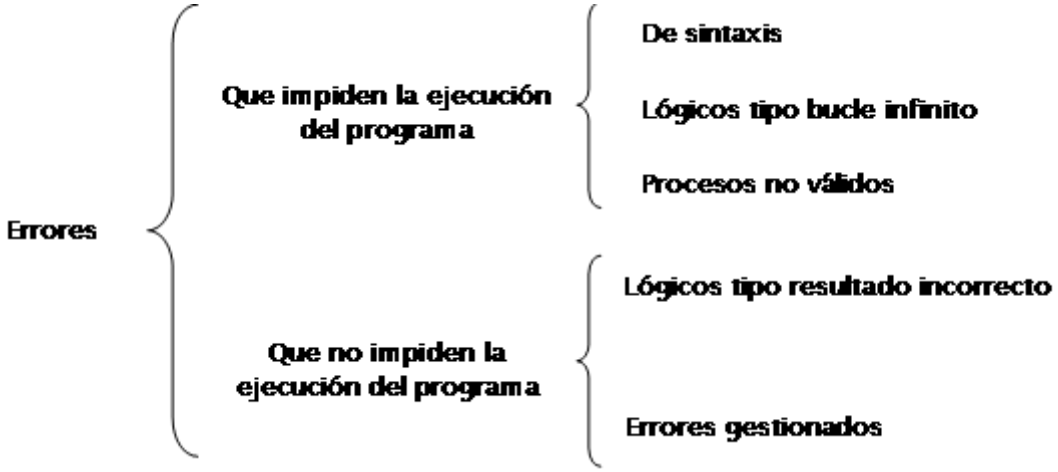
10. Comportamiento poco ético.



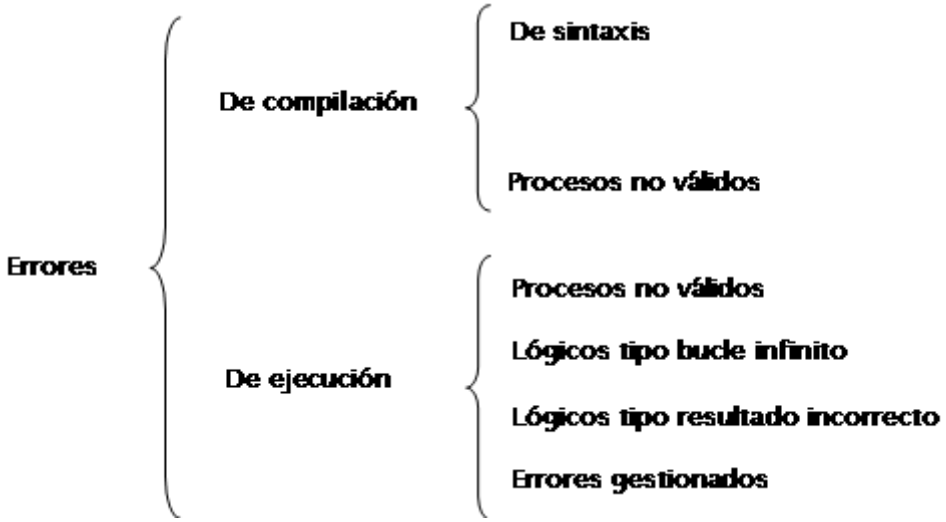
2. CLASIFICACIÓN DE ERRORES.

Los errores en un programa o algoritmo se pueden clasificar en distintos tipos. Atendiendo a los efectos que ocasionan se podría hablar de errores que impiden la ejecución de un programa y errores que no impiden la ejecución de un programa. Atendiendo al momento en que se producen podríamos hablar de errores de compilación y errores de ejecución. Lo vemos en forma de esquemas:

Atendiendo a los efectos que ocasionan:



Atendiendo al momento en que se producen:



Cuando una vez tenemos escrito el código del programa y ordenamos su ejecución, se produce una "lectura de interpretación" previa llamada compilación. Recordemos que el ordenador no interpreta directamente las órdenes que le damos sino que necesita una traducción. Si durante esa traducción se detecta un problema el programa no comienza a ejecutarse. Lo más habitual es que se detecten fallos de sintaxis, ciertos procesos no válidos e incluso errores lógicos tipo bucle infinito en algunas

Ana González (Kao)

circunstancias. Si el programa no compila estamos obligados a realizar las correcciones oportunas antes de poder ejecutarlo.

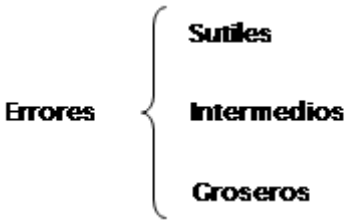
Durante la ejecución del programa pueden producirse errores previsibles porque se derivan del código o imprevisibles por ser su origen externo (entradas incorrectas de usuario, problemas con ficheros, etc.).

Un error de ejecución puede ser gestionado (vía detección o vía lógica) pero uno de compilación no.

Atendiendo a la naturaleza del error los clasificaremos en:

Y según el tratamiento que reciben:

Por su facilidad de detección tendríamos:



Hay errores cuya clasificación no es sencilla. Por ejemplo, si al usuario se le pide un número entero durante la ejecución del programa, pero introduce uno real, se puede producir un error de ejecución por proceso no válido. Sin embargo, el trasfondo del

Ana González (Kao)

error es lógico: el programa no está preparado para reaccionar ante una situación que es posible. A estos errores los llamaremos errores de fondo lógico.

3. Programación segura

Aunque se ha avanzado mucho en los mecanismos para escribir código libre de fallos, todavía se siguen creando programas incorrectos.

Se estima que por cada 1000 líneas de código que se escriben, aparecen entre 5 y 10 errores.

Solo una pequeña parte de esos errores de programación se pueden utilizar como vulnerabilidades de seguridad.

MITRE (http://cwe.mitre.org/top25/index.html) mantiene una lista de los bugs de programación mas peligrosos.

Desbordamiento de pila

De entre todos los tipos de errores de programación, los que con mayor facilidad se convierten en fallos (bugs) explotables mediante exploits son los conocidos como desbordamiento de buffer o Buffer Overflow (BO).

El problema del desbordamiento de buffer se produce cuando un programa reserva un cierto espacio para una variable o un array y trata de escribir más bytes de los inicialmente reservados, con lo que se sobrescriben zonas de memoria que contienen otros datos.

Normalmente el exploit consiste en pasar al programa erróneo parámetros incorrectos (p.e. URL o nombre de fichero muy largo) para que este ejecute el código deseado por el atacante.

Los fallos de seguridad causados por desbordamiento de buffer son responsabilidad directa de los errores del programador, y no fallos de administración o configuración.

La mejor forma de evitar estos errores es siguiendo una serie de reglas de programación

Ana González (Kao)

Variables, el poder de variar

Cuando se elabora un programa puede surgir la necesidad de esperar por datos que sean proporcionados desde una fuente externa, para lograrlo se hace uso de algo que en el argot de programación se conoce como “variable”.

Una variable de programación, al igual que una variable en una función matemática, puede tomar cualquier valor. Tomemos justamente el ejemplo matemático para entenderlo y pensemos en la función: y = x + 5.

Supongamos ahora que deseamos hacer un programa que nos diga cuánto vale “y” una vez que determinemos el valor de “x”. Dicho programa diría algo como: “Hola, tengo un severo problema y necesito de tu ayuda. Debo saber el valor de “y” que resulta de la función “x + 5” pero yo no sé cuánto vale “x” ¿Podrías indicarlo?”

Así pues, sabemos que un programa puede recibir información de una fuente ajena al programa (el usuario, en el caso de nuestro ejemplo) y que esa información será almacenada en variables, pero, ¿cómo se hace esto? Sencillo, el programa, al ser ejecutado toma espacio de memoria, de todo ese espacio que tomó deja una parte pequeña de memoria para los valores que desconoce (las variables) y que está a la espera de que alguna fuente externa se los proporcione.

Arreglos

Dentro de la programación existen varios tipos de variables. Están las que llegan a almacenar valores llamados enteros, las que llegan a almacenar valores flotantes o las que almacenan solo un carácter alfanumérico, entre otras. Pero hay un tipo de variable a la que le debemos prestar especial atención: los arreglos.

¿Qué es un arreglo? Imaginemos un largo estante, tan largo como lo deseemos, que tiene cajones y dentro de cada uno podemos guardar lo que sea, lo que queramos, y a ese estante le asignamos un nombre con el cual lo identificamos fácil y rápidamente. Un arreglo es algo similar, es un gran estante que llevará un nombre y que usaremos para guardar valores ya sea de tipo entero, flotante, caracteres, etcétera. Cuando necesitemos un valor en específico solo bastará ir al “cajón” indicado del estante y tomarlo.

La manera en que un programa maneja un arreglo que no ha sido inicializado (es decir, que no tiene valores) es similar a la manera que maneja las variables, les deja un espacio de memoria, pero esta vez la deja de manera continua: si el arreglo es de cinco

Ana González (Kao)

elementos dejará el espacio para cinco variables, si es de diez, dejará diez espacios, de acuerdo a lo que haya sido definido por el programador.

El error que se comete al momento de programar

Imaginemos que es necesario hacer una aplicación que guarde cinco números proporcionados de manera externa. Existen dos opciones, una es declarar cinco variables o hacer uso de un arreglo que conste de cinco elementos. Después de una no tan difícil deliberación decidimos usar el arreglo y corremos la aplicación para probarla. Llegamos al punto en donde el programa solicita introducir los cinco datos y empezamos a proporcionar uno por uno hasta llegar al quinto dato, llegado este punto lo normal sería que ya no se pudieran introducir más datos, sin embargo el programa permite un sexto dato, es decir, meter más datos de los que debería guardar. Es justo ahí donde hemos cometido un error al momento de desarrollar el programa, que más que un error es un descuido, ya que no se tomaron las debidas precauciones para evitar introducir más datos de los que el programa puede manejar de acuerdo con su diseño y es esto justamente la esencia de la vulnerabilidad llamada desbordamiento de buffer.

Cuando un programa nos permite introducir más datos de los que espera y puede manejar, sucede que existe la posibilidad de escribir en zonas de memoria donde se ubica información que ayuda al control y flujo del programa. Ahora entonces, la pregunta lógica es ¿Qué es y cómo funciona esa zona de memoria que controla la ejecución del programa?

El stack

La cosa es muy simple, un programa que es inicializado necesita tomar un espacio de memoria que usará para, llegado el momento, poner en ese espacio el valor de las variables que no fueron declaradas desde el código del programa y que será conocido una vez que el programa esté corriendo. Este espacio de memoria se conoce con el nombre de *stack*, o “pila” si se prefiere la traducción al español.

¿Y cómo funciona el *stack*?, para ilustrar este concepto hay un sinfín de objetos con los cuales se pude ejemplificar, pero tomemos uno de mis favoritos, los libros. Imaginemos un libro sobre una mesa, y sobre ese libro ponemos otro, y sobre ese libro otro más, y así hasta que tenemos una columna de cincuenta libros. Si eventualmente

Ana González (Kao)

necesitamos tomar el primero, aquel que está justo entre la mesa y los cuarenta y nueve restantes, no podremos hacerlo sin antes quitar los que están encima de este. De hecho, no podemos tomar el libro cuarenta y nueve sin antes haberle quitado de encima el libro cincuenta. Entonces, para llegar al de hasta abajo tengo que retirar el libro cincuenta, luego el cuarenta y nueve, luego el cuarenta y ocho, y así hasta llegar al primero.

El *stack* es algo muy similar, es una gran columna (virtual) de memoria donde se almacenará información, ya sea de control o de variables, y, a diferencia del ejemplo de los libros, no es que exista la posición uno, la dos o la tres; lo que existe es una dirección de memoria para cada elemento.

En el ejemplo se señaló que si quería llegar al primer libro, deberían quitarse uno por uno los cuarenta y nueve libros que estaban arriba. Este concepto de ir quitando uno por uno empezando desde el último tiene un nombre y se llama LIFO (*last in*, *first out*), que quiere decir que el último que entra será el primero en salir.

El *stack* es una sección de memoria que suele utilizar el concepto LIFO (aunque no todos los *stack* operan así) para introducir datos que pueden ser de control y flujo del programa o el valor de variables que no fueron declaradas en el código del programa. Ahora, ¿cómo sabe la computadora en qué sección de memoria debe introducir información? La respuesta tiene un acrónimo y es SP.

SP es el acrónimo para *Stack Pointer* y su función, como lo dice su nombre, es apuntar a una dirección de memoria de la *pila*, de hecho la del último elemento. Es importante entender que el SP siempre se encontrará apuntando al último elemento, porque justamente así es como se determinan dos operaciones que se utilizan para manipular datos en la pila: *push* para introducirle datos y *pop* para sacar.

Es más fácil de lo que parece

Aunque parezca complicado conseguir llegar a desbordar la pila y ejecutar nuestro código (como atacantes), es posible.

http://www.projectshellcode.com/

A pesar de los mecanismos de prevención, que veremos, todavía es uno de los fallos más peligrosos: http://cwe.mitre.org/top25/index.html

3. Programación segura

Como evitarlos los desbordamientos

Existen dos posibles estrategias:

1. Abordar el problema utilizando una metodología de programación que elimine los errores de programación. Generar código correcto.

2. Utilizar mecanismos de protección para que en tiempo de ejecución detecten los posibles fallos e impidan crear exploits. Impedir que código erróneo se convierta en fallos de seguridad.

Consejos de programación

Una breve lista de cuestiones a tener presentes siempre que se programa:

Utiliza lenguajes de programación que permitan verificar rangos de vectores en tiempo de compilación: Java, ADA, etc.

Valida TODOS los datos que introduce el usuario: valores le dos de ficheros, nombres de fichero, datos de teclado, direcciones URL, mensajes de red, variables de entorno.

No utilices funciones de manipulación de cadenas que no comprueben la longitud de la cadena destino.

sprintf, fscanf, scanf, sscanf, vsprintf, gets, strcpy, strcat, etc.

Verifica todos los valores de error devueltos por las llamadas a sistema y la librería.

Herramientas para auditar código

Para auditoría web podéis usar vega o acunetix:

http://www.securitybydefault.com/2013/06/vega-herramienta-para-auditar-websites.html

http://subgraph.com/vega_download.php

Para auditar el fuente:

http://hispalinux.es/node/722

http://www.binaryanalysis.org/en/home

Stack protect

Es una modificación del compilador para insertar justo antes de la dirección de retorno un entero (bien aleatorio o bien que contenga algún byte a cero) llamado \canario" y que es comprobado antes de retornar de la función.

La última versión del compilador de GCC (4.1) dispone de esta característica. Se activa con el ag de compilacion: -fstack-protect. Aunque por defecto suele estar activada.

Pila no ejecutable

Si la memoria donde reside la pila no tiene permisos de ejecución, entonces se podrá sobrescribir con código malicioso pero no se podrá ejecutar. Por lo que el error no se convierte en un fallo de seguridad.

Los principales problemas son:

El exploit puede preparar la pila con los parámetros oportunos y saltar a una función legítima (execl) localizada dentro de la zona de memoria de código.

También requiere modificar el compilador ya que el propio compilador utiliza la pila para ejecutar código.

No es necesario que el código que se ejecute resida en la pila.

Pila en posiciones aleatorias

El shell-code se inserta en la pila.

Para ejecutar el shell-code, se tiene que sobreescribir la dirección de retorno con la dirección absoluta del shell-code.

Si la dirección de la pila es diferente en cada ejecución del programa, entonces un exploit genérico no acertara a ejecutar el shell-code.

Esta técnica se conoce como: “Address space layout randomization".

Se suelen modificar las posiciones del código, datos, pila y librerías.

4. Sandboxing

Y, ¿qué es el *Sandboxing*?.

Este "cajón de arena" (traducción literal al castellano) es un **aislamiento de procesos**, es decir, un mecanismo que implementan varias aplicaciones para ejecutar aplicaciones y programas con seguridad y "aislarlas" del resto del sistema dentro de una especie de contenedor virtual desde el cual controlar los distintos recursos que solicita dicha aplicación (memoria, espacio en disco, privilegios necesarios, etc).

Este férreo control al que se somete el proceso sirve para discernir si el código a ejecutar es malicioso o no puesto que, por norma general, se restringirá cualquier tipo de acceso a dispositivos de entrada o de inspección del sistema anfitrión.

Gracias al *sandboxing*, por ejemplo en Google Chrome, el navegador de Google es capaz de "aislar" pestañas de navegación entre sí y, además, impedir que una página web con contenido malicioso intente instalar cualquier tipo de software en nuestro sistema, monitorizar lo que estamos haciendo o acceder a información alojada en nuestro disco duro (y entre las aplicaciones que aisla, Flash es una de ellas). De hecho, este mecanismo también se incluyó en Adobe Reader X porque uno de los grandes "coladeros" de Adobe era la posibilidad de esconder código en los archivos PDF y que éste se ejecutase al abrir un "documento malicioso" porque la aplicación no impedía ni controlaba peticiones que iban más allá de mostrar el contenido del archivo (y se daban casos de cambios en el registro de Windows o la instalación de software en el sistema)

. Además de proteger a los usuarios, **esta técnica también es utilizada por equipos de seguridad, por ejemplo, para estudiar *malware*** dentro de un entorno controlado y ver qué efectos tiene en un sistema para proceder a su caracterización, también podemos utilizar aplicaciones para "aislar" a otras y probarlas de manera segura (importante si sospechamos de ellas aunque, ante la duda, es mejor no instalarlas aunque usando Sandboxie o Glipse podremos mejorar la seguridad de nuestras pruebas). A mayor escala, la virtualización de un sistema operativo dentro de otro es también una forma práctica de *sandboxing*.

Ana González (Kao)

5. Glosario de Seguridad

Guía de terminología relacionada con la seguridad que puede encontrar en el Informe sobre las Amenazas a la Seguridad en Internet de Symantec y en otros materiales relacionados con la seguridad informática.

Adware

Adware es un software, generalmente no deseado, que facilita el envío de contenido publicitario a un equipo.

Amenaza

Una amenaza informática es toda circunstancia, evento o persona que tiene el potencial de causar daño a un sistema en forma de robo, destrucción, divulgación, modificación de datos o negación de servicio (DoS).

Amenazas polimorfas

Las amenazas polimorfas son aquellas que tienen la capacidad de mutar y en las cuales cada instancia del malware es ligeramente diferente al anterior a este. Los cambios automatizados en el código realizados a cada instancia no alteran la funcionalidad del malware, sino que prácticamente inutilizan las tecnologías tradicionales de detección antivirus contra estos ataques.

Antispam

Antispam es un producto, herramienta, servicio o mejor práctica que detiene el spam o correo no deseado antes de que se convierta en una molestia para los usuarios. El antispam debe ser parte de una estrategia de seguridad multinivel.

Antivirus

Antivirus es una categoría de software de seguridad que protege un equipo de virus, normalmente a través de la detección en tiempo real y también mediante análisis del

sistema, que pone en cuarentena y elimina los virus. El antivirus debe ser parte de una estrategia de seguridad estándar de múltiples niveles.

Aplicaciones engañosas

Las aplicaciones engañosas son programas que intentan engañar a los usuarios informáticos para que emprendan nuevas acciones que normalmente están encaminadas a causar la descarga de malware adicional o para que los usuarios divulguen información personal confidencial. Un ejemplo es el software de seguridad fraudulento, que también se denomina scareware.

Ataques multi-etapas

Un ataque en múltiples etapas es una infección que normalmente implica un ataque inicial, seguido por la instalación de una parte adicional de códigos maliciosos. Un ejemplo es un troyano que descarga e instala adware.

Ataques Web

Un ataque Web es un ataque que se comete contra una aplicación cliente y se origina desde un lugar en la Web, ya sea desde sitios legítimos atacados o sitios maliciosos que han sido creados para atacar intencionalmente a los usuarios de ésta.

Blacklisting o Lista Negra

La lista negra es el proceso de identificación y bloqueo de programas, correos electrónicos, direcciones o dominios IP conocidos maliciosos o malévolos.

Bot

Un bot es una computadora individual infectada con malware , la cual forma parte de una red de bots (bot net).

Botnet

Conjunto de equipos bajo el control de un bot maestro, a través de un canal de mando y control. Estos equipos normalmente se distribuyen a través de Internet y se utilizan para actividades malintencionadas, como el envío de spam y ataques distribuidos de negación de servicio. Las botnet se crean al infectar las computadoras con malware, lo cual da al atacante acceso a las máquinas. Los propietarios de computadoras infectadas generalmente ignoran que su máquina forma parte de una botnet, a menos que tengan software de seguridad que les informe acerca de la infección.

Caballo de Troya

Son un tipo de código malicioso que parece ser algo que no es. Una distinción muy importante entre troyanos y virus reales es que los troyanos no infectan otros archivos y no se propagan automáticamente. Los caballos de troya tienen códigos maliciosos que cuando se activan causa pérdida, incluso robo de datos. Por lo general, también tienen un componente de puerta trasera, que le permite al atacante descargar amenazas adicionales en un equipo infectado. Normalmente se propagan a través de descargas inadvertidas, archivos adjuntos de correo electrónico o al descargar o ejecutar voluntariamente un archivo de Internet, generalmente después de que un atacante ha utilizado ingeniería social para convencer al usuario de que lo haga.

Canal de control y comando

Un canal de mando y control es el medio por el cual un atacante se comunica y controla los equipos infectados con malware, lo que conforma un botnet.

Carga destructiva

Una carga destructiva es la actividad maliciosa que realiza el malware. Una carga destructiva es independiente de las acciones de instalación y propagación que realiza el malware.

Crimeware

Software que realiza acciones ilegales no previstas por un usuario que ejecuta el software. Estas acciones buscan producir beneficios económicos al distribuidor del software.

Ciberdelito

El ciberdelito es un delito que se comete usando una computadora, red o hardware. La computadora o dispositivo puede ser el agente, el facilitador o el objeto del delito. El delito puede ocurrir en la computadora o en otros lugares.

Definiciones de virus

Una definición de virus es un archivo que proporciona información al software antivirus, para identificar los riesgos de seguridad. Los archivos de definición tienen protección contra todos los virus, gusanos, troyanos y otros riesgos de seguridad más recientes. Las definiciones de virus también se denominan firmas antivirus.

Descarga inadvertida

Una descarga inadvertida es una descarga de malware mediante el ataque a una vulnerabilidad de un navegador Web, equipo cliente de correo electrónico o plug-in de navegador sin intervención alguna del usuario. Las descargas inadvertidas pueden ocurrir al visitar un sitio Web, visualizar un mensaje de correo electrónico o pulsar clic en una ventana emergente engañosa.

Economía clandestina

La economía clandestina en línea es el mercado digital donde se compran y se venden bienes y servicios obtenidos a través de la ciberdelincuencia, con el fin de cometer delitos informáticos. Dos de las plataformas más comunes a disposición de los participantes en la economía clandestina en línea son los canales en servidores IRC y foros Web. Los dos tienen grupos de discusión que utilizan participantes para comprar y vender bienes y servicios fraudulentos. Los artículos vendidos son datos de tarjetas de crédito, información de cuentas bancarias, cuentas de correo electrónico y toolkits de creación de malware. Los servicios incluyen cajeros que pueden transferir fondos de cuentas robadas en moneda real, phishing y hosting de páginas fraudulentas y anuncios de empleo para cargos como desarrolladores de fraude o socios de phishing.

Encriptación

La encriptación es un método de cifrado o codificación de datos para evitar que los usuarios no autorizados lean o manipulen los datos. Sólo los individuos con acceso a una contraseña o clave pueden descifrar y utilizar los datos. A veces, el malware utiliza la encriptación para ocultarse del software de seguridad. Es decir, el malware cifrado revuelve el código del programa para que sea difícil detectarlo.

Exploits o Programas intrusos

Los programas intrusos son técnicas que aprovechan las vulnerabilidades del software y que pueden utilizarse para evadir la seguridad o atacar un equipo en la red.

Filtración de datos

Una filtración de datos sucede cuando se compromete un sistema, exponiendo la información a un entorno no confiable. Las filtraciones de datos a menudo son el resultado de ataques maliciosos, que tratan de adquirir información confidencial que puede utilizarse con fines delictivos o con otros fines malintencionados

Firewall

Un firewall es una aplicación de seguridad diseñada para bloquear las conexiones en determinados puertos del sistema, independientemente de si el tráfico es benigno o maligno. Un firewall debería formar parte de una estrategia de seguridad estándar de múltiples niveles.

Firma antivirus

Una firma antivirus es un archivo que proporciona información al software antivirus para encontrar y reparar los riesgos. Las firmas antivirus proporcionan protección contra todos los virus, gusanos, troyanos y otros riesgos de seguridad más recientes. Las firmas antivirus también se denominan definiciones de virus.

Greylisting o Lista Gris

La lista gris es un método de defensa para proteger a los usuarios de correo electrónico contra el spam. Los mensajes de correo electrónico son rechazados

Ana González (Kao)

temporalmente de un remitente que no es reconocido por el agente de transferencia de correos. Si el correo es legítimo, el servidor de origen tratará de nuevo y se aceptará el correo electrónico. Si el correo es de un remitente de spam, probablemente no se reintentará su envío y por lo tanto, no logrará pasar el agente de transferencia de correos.

Gusanos

Los gusanos son programas maliciosos que se reproducen de un sistema a otro sin usar un archivo anfitrión, lo que contrasta con los virus, puesto que requieren la propagación de un archivo anfitrión infectado.

Ingeniería Social

Método utilizado por los atacantes para engañar a los usuarios informáticos, para que realicen una acción que normalmente producirá consecuencias negativas, como la descarga de malware o la divulgación de información personal. Los ataques de phishing con frecuencia aprovechan las tácticas de ingeniería social.

Lista blanca o Whitelisting

La lista blanca es un método utilizado normalmente por programas de bloqueo de spam, que permite a los correos electrónicos de direcciones de correo electrónicos o nombres de dominio autorizados o conocidos pasar por el software de seguridad.

Keystroke Logger o Programa de captura de teclado (Keylogger)

Es un tipo de malware diseñado para capturar las pulsaciones, movimientos y clics del teclado y del ratón, generalmente de forma encubierta, para intentar robar información personal, como las cuentas y contraseñas de las tarjetas de crédito.

Malware

El malware es la descripción general de un programa informático que tiene efectos no deseados o maliciosos. Incluye virus, gusanos, troyanos y puertas traseras. El malware a menudo utiliza herramientas de comunicación populares, como el correo electrónico

Ana González (Kao)

y la mensajería instantánea, y medios magnéticos extraíbles, como dispositivos USB, para difundirse. También se propaga a través de descargas inadvertidas y ataques a las vulnerabilidades de seguridad en el software. La mayoría del malware peligroso actualmente busca robar información personal que pueda ser utilizada por los atacantes para cometer fechorías.

Mecanismo de propagación

Un mecanismo de propagación es el método que utiliza una amenaza para infectar un sistema.

Negación de servicio (DoS)

La negación de servicio es un ataque en el que el delincuente intenta deshabilitar los recursos de una computadora o lugar en una red para los usuarios. Un ataque distribuido de negación de servicio (DDoS) es aquel en que el atacante aprovecha una red de computadoras distribuidas, como por ejemplo una botnet, para perpetrar el ataque.

Pharming

Método de ataque que tiene como objetivo redirigir el tráfico de un sitio Web a otro sitio falso, generalmente diseñado para imitar el sitio legítimo. El objetivo es que los usuarios permanezcan ignorantes del redireccionamiento e ingresen información personal, como la información bancaria en línea, en el sitio fraudulento. Se puede cometer pharming cambiando el archivo de los equipos anfitriones en la computadora de la víctima o atacando una vulnerabilidad en el software del servidor DNS.

Phishing

A diferencia de la heurística o los exploradores de huella digital, el software de seguridad de bloqueo de comportamiento se integra al sistema operativo de un equipo anfitrión y supervisa el comportamiento de los programas en tiempo real en busca de acciones maliciosas. El software de bloqueo de comportamiento bloquea acciones potencialmente dañinas, antes de que tengan oportunidad de afectar el sistema. La

protección contra el comportamiento peligroso debe ser parte de una estrategia de seguridad estándar de múltiples niveles.

Protección heurística (Heuristics-Based Protection)

Forma de tecnología antivirus que detecta las infecciones mediante el escrutinio de la estructura general de un programa, las instrucciones de sus computadoras y otros datos contenidos en el archivo. Una exploración heurística hace una evaluación sobre la probabilidad de que el programa sea malicioso con base en la aparente intención de la lógica. Este plan puede detectar infecciones desconocidas, ya que busca lógica generalmente sospechosa, en lugar de huellas específicas de malware, tales como los métodos tradicionales de antivirus de firmas. La protección heurística debería hacer parte de una estrategia de seguridad estándar de múltiples niveles

Redes punto a punto (P2P)

Red virtual distribuida de participantes que hacen que una parte de sus recursos informáticos estén a disposición de otros participantes de la red, todo sin necesidad de servidores centralizados. Las redes puntos a punto son utilizadas para compartir música, películas, juegos y otros archivos. Sin embargo, también son un mecanismo muy común para la distribución de virus, bots, spyware, adware, troyanos, rootkits, gusanos y otro tipo de malware.

Rootkits

Componente de malware que utiliza la clandestinidad para mantener una presencia persistente e indetectable en un equipo. Las acciones realizadas por un rootkit, como la instalación y diversas formas de ejecución de códigos, se realizan sin el conocimiento o consentimiento del usuario final.
Los rootkits no infectan las máquinas por sí mismos como lo hacen los virus o gusanos, sino que tratan de proporcionar un entorno indetectable para ejecutar códigos maliciosos. Los atacantes normalmente aprovechan las vulnerabilidades en el equipo seleccionado o utilizan técnicas de ingeniería social para instalar manualmente los rootkits. O, en algunos casos, los rootkits pueden instalarse automáticamente al ejecutarse un virus o gusano o incluso simplemente al navegar en un sitio Web malicioso.
Una vez instalados, el atacante puede realizar prácticamente cualquier función en el

sistema, incluyendo acceso remoto, intercepción de comunicaciones, así como procesos de ocultamiento, archivos, claves de registro y canales de comunicación.

Seguridad basada en la reputación

La seguridad basada en la reputación es una estrategia de identificación de amenazas que clasifica las aplicaciones con base en ciertos criterios o atributos para determinar si son probablemente malignas o benignas. Estos atributos pueden incluir diversos aspectos como la edad de los archivos, la fuente de descarga de los archivos y la prevalencia de firmas y archivos digitales. Luego, se combinan los atributos para determinar la reputación de seguridad de un archivo. Las calificaciones de reputación son utilizadas después por los usuarios informáticos para determinar mejor lo que es seguro y permitirlo en sus sistemas. La seguridad basada en la reputación debe ser parte de una estrategia de seguridad estándar de múltiples niveles.

Sistema de detección de intrusos

Un sistema de detección de intrusos es un servicio que monitorea y analiza los eventos del sistema para encontrar y proporcionar en tiempo real o casi real advertencias de intentos de acceso a los recursos del sistema de manera no autorizada. Es la detección de ataques o intentos de intrusión, que consiste en revisar registros u otra información disponible en la red. Un sistema de detección de intrusos debe ser parte de una estrategia de seguridad estándar de múltiples niveles.

Sistema de prevención de intrusos

Un sistema de prevención de intrusos es un dispositivo (hardware o software) que supervisa las actividades de la red o del sistema en busca de comportamiento no deseado o malicioso y puede reaccionar en tiempo real para bloquear o evitar esas actividades. Un sistema de prevención de intrusos debe ser parte de una estrategia de seguridad estándar de múltiples niveles.

Software de seguridad fraudulento (rogue)

Un programa de software de seguridad rogue es un tipo de aplicación engañosa que finge ser software de seguridad legítimo, como un limpiador de registros o detector

antivirus, aunque realmente proporciona al usuario poca o ninguna protección y, en algunos casos, puede de hecho facilitar la instalación de códigos maliciosos contra los que busca protegerse.

Spam

También conocido como correo basura, el spam es correo electrónico que involucra mensajes casi idénticos enviados a numerosos destinatarios. Un sinónimo común de spam es correo electrónico comercial no solicitado (UCE). El malware se utiliza a menudo para propagar mensajes de spam al infectar un equipo, buscar direcciones de correo electrónico y luego utilizar esa máquina para enviar mensajes de spam. Los mensajes de spam generalmente se utilizan como un método de propagación de los ataques de phishing

Spyware

Paquete de software que realiza un seguimiento y envía información de identificación personal o información confidencial a otras personas. La información de identificación personal es la información que puede atribuirse a una persona específica, como un nombre completo. La información confidencial incluye datos que la mayoría de personas no estaría dispuesta a compartir con nadie e incluye datos bancarios, números de cuentas de tarjeta de crédito y contraseñas. Los receptores de esta información pueden ser sistemas o partes remotas con acceso local.

Toolkit

Paquete de software diseñado para ayudar a los hackers a crear y propagar códigos maliciosos. Los toolkits frecuentemente automatizan la creación y propagación de malware al punto que, incluso los principiante delincuentes cibernéticos son capaces de utilizar amenazas complejas. También pueden utilizarse toolkits para lanzar ataques web, enviar spam y crear sitios de phishing y mensajes de correo electrónico.

Variantes

Las variantes son nuevas cepas de malware que piden prestado códigos, en diversos grados, directamente a otros virus conocidos. Normalmente se identifican con una

Ana González (Kao)

letra o letras, seguido del apellido del malware; por ejemplo, W32.Downadup.A, W32.Downadup.B y así sucesivamente.

Vector de ataque

Un vector de ataque es el método que utiliza una amenaza para atacar un sistema.

Virus

Programa informático escrito para alterar la forma como funciona una computadora, sin permiso o conocimiento del usuario. Un virus debe cumplir con dos criterios:

Debe ejecutarse por sí mismo: generalmente coloca su propio código en la ruta de ejecución de otro programa.

Debe reproducirse: por ejemplo, puede reemplazar otros archivos ejecutables con una copia del archivo infectado por un virus. Los virus pueden infectar computadores de escritorio y servidores de red.

Muchos de los virus actuales están programados para operar sigilosamente la computadora del usuario con el fin de robar información personal y utilizarla para cometer delitos. Otros menoscaban el equipo dañando los programas, eliminando archivos o volviendo a formatear el disco duro. Aún existen otros que no están diseñados para causar daño, aunque simplemente se reproducen y hacen manifiestan su presencia presentando mensajes de texto, video y audio, aunque este tipo de ataques de notoriedad no son tan comunes, puesto que los autores de virus y demás malware tiene como fin obtener ganancias ilegales.

Virus más propagado

Amenaza que se dice está en su apogeo e indica que ya se está extendiendo entre los usuarios informáticos.

Vulnerabilidad

Una vulnerabilidad es un estado viciado en un sistema informático (o conjunto de sistemas) que afecta las propiedades de confidencialidad, integridad y disponibilidad (CIA) de los sistemas. Las vulnerabilidades pueden hacer lo siguiente:

- Permitir que un atacante ejecute comandos como otro usuario
- Permitir a un atacante acceso a los datos, lo que se opone a las restricciones específicas de acceso a los datos
- Permitir a un atacante hacerse pasar por otra entidad
- Permitir a un atacante realizar una negación de servicio