Análisis del Malware Tema 2

Sandboxes

La idea de las sandbox es poder analizar dinámicamente el malware en un entorno aislado. Estas Herramientas registran toda la actividad del malware y generan un reporte. Para esto también se puede usar un KVM (Kernel Virtual Machine)que nos permite arrancar una máquina virtual para analizar automáticamente la actividad de un malware, generar un reporte y recuperar la máquina a una snapshot anterior.

Cuckoo Sandbox

Inicia varias máquinas virtuales conectadas a una red virtual para analizar el comportamiento de un malware. Lo primero que se debe hacer tras instalar Cuckoo es configurar la aplicación, para ello vamos a la carpeta conf:

- cuckoo.conf: contiene la config de la VM, permite elegir entre KVM y virtualbox
- virtualbox.conf: configuración que usará el virtualbox como la plataforma y la ip
- kvm.conf: lo mismo que virtualbox.conf pero para kvm
- reporting.conf: Para definir el formato del reporte

Para inicializar cuckoo primero se debe instalar el agente en la máquina virtual (agent/agent.py). Finalmente se debe realizar una snapshot con el agente arrancado, de forma que cuckoo pueda restaurar la máquina virtual tras cada análisis. Para iniciar cuckoo se debe ejecutar el script cuckoo.py en el host. Tras eso envías el archivo que se quiere analizar con submit.py, que se encuentra en la carpeta utils. Se pueden seleccionar los formatos con el flag -package.

Los resultados del análisis se guardan en la carpeta storage/analysis:

- files: Archivos creados cuando se ejecutó el malware
- logs: Actividad del malware como llamadas a librerías del sistema dll
- reports: reporte con el formati anteriormente mencionado

Cuckoo tiene algunas limitaciones ya que al volverse muy popular algunos malwares traen contramedidas como sistemas anti análisis. Lo que hacen estos sistemas es:

- Revisar la resolución de pantalla
- Revisar el número de núcleos disponibles
- Revisar el historial del navegador (Suele estar vacío en las máquinas virtuales)

Debido a esto cuckoo está en constante evolución, al igual que el malware.

REMnux

Kit de herramientras para hacer ingeniería inversa y revisar software malicioso. Provee de una colección de herramientras creadas por la comunidad que se pueden usar para el análisis del

malware. También ofrece imágenes de docker con herramientas de análisis de malware populares.

FlareVM Sandbox

Colección de scripts que permiten montar y mantener un ambiente de ingeniería inversa en una máquina virtual. Depende de 2 principales tecnologías:

- Chocolatey: Sistema de gestión de paquetes para windows donde un paquete es un archivo zip que contiene un script de instalación de PowerShell que descarga y configura herramientas específicas.
- Boxstarter: Usa los paquetes de chocolatey para automatizar la instalación de software y crear ambientes repetibles automatizados.

Contramedidas del Malware

El malware puede tener numerosas medidas contra la ingeniería inversa:

- Ofuscación
- Ocultar información de configuración
- Encriptación de la comunicación de red
- Codificación de datos.

Ofuscación

Transformar el programa dejando las funcionalidades intactas para dificultar el entendimiento de su funcionamiento. Normalmente la ofuscación se usa para:

- Proteger propiedad intelectual: para evitar el plagio
- Hacer los antivirus menos efectivos: evita la detección de malware por firma digital
- Ralentizar el análisis de malware: Para mantener el código del malware más tiempo en funcionamiento.

Codificación de datos

Cifrado simple

Hay muchos cifrados simples que aplican operaciones de bit sobre los registros de datos:

- ADD y SUB: Añade o elimina caracteres del set de caracteres
- ROL y ROR: Rota hacia la derecha o la izquierda los bots de ciertos sets de caracteres
- ROT: Intecambia 13 veces cada caracter del alfabeto
- XOR

Algoritmos standar de ciptografía

La forma más fácil de descubrir el algoritmo es usando uno estándar:

- Mirar los strings e imports para detectar el cifrado de APIs
- Detectar el uso de constantes mágicas de encriptado.
- Analizar la entropía de un archivo para detectar partes que han sido comprimidas o cifradas.

Custom Encoding

Sistemas de codificación caseros:

- Combinar varias capas de multiples metodos de codificados
- Algoritmos completamente customizados creados por el autor del malware.

Reverse Encoding

Procedimiento estándar para encontrar que codificado se ha utilizado y deducir como descifrarlo:

1. Trazar la ejecución del programa buscando funciones de codificado y decodificado
2. Deducir cuando y como se usan estas funciones.
3. Usar el malware contra sí mismo: reprogramar las funcioanes, usarlas como son en el malware…

Usando el malware contra sí mismo

Arrancar el malware en un debugger y establecer breakpointas antes y despues del codificado o decodificado. Tiene sus problemas:

- Puede que el malware no desencripte la información que nos interesa
- No se puede deducir como hacer que el malware ejecute la función de desencriptado

Empaquetado del malware

Los diseñadores de malware suelen añadir un componente llamado run-time packer que son aplicaciones que comprimen el código como otras aplicaciones típicas como Zip o RAR. Cuando estas aplicaciones son descomprimidas con el empaquetador, la aplicación será descomprimida en la memoria de sistema en vez de en el sistema de archivos. La ventaja de esto es que el código de un malware es más pequeño y menos detectable, lo que dificulta su detección por parte de antivirus. Otra medida que se puede tomar es encriptar el malware con el empaquetador, de forma que ni los antivirus puedan desempaquetar el código.

Procedimiento

1. Descomprimir el código empaquetado

2. cargar el código empaquetado en memoria
3. Resolver las importaciones del ejecutable original
4. Transferir la ejecución al OEP (Original entry point.)

Normalmente se realizan tareas anti análisis en cada paso.

Empaquetando el Malware

En los empaquetadores, un estub es una pieza de código que contiene la rutina de decompresión o descifrado que actua de cargador y se ejecuta antes de ejecutar el malware.

Detección

Hay varios indicadores de que podemos estar tratando con un ejecutable empaquetado:

- PE Header: Diferentes tamaños entre datos y tamaño virtual
- Acceso Limitado a los strings e imports
- Entropía: tiene una entropía alta

Desempaquetando el código

Si este fuera un caso fácil, para desempaquetar el código pueden usarse métodos como XOR, AES, etc... A veces se mantiene la tabla de importación intacta de forma que el cargador de windows puede resolver los imports. Si fuera un caso difícil no hay imports y hacerlo manualmente es un dolor de cabeza.

Desempaquetado automatizado

Existen herramientas como UPX que permiten realizar los desempaquetados, pero no siempre sirven. A veces se crean scripts o herramientas propias para empaquetar, lo que hace difícil su desempaquetado. El desempaquetado automatizado no siempre funciona debido a esto.

Cuando falla el desempaquetado automatizado

La dificultad y tiempo requeridos depende de:

- El empaquetador
- Las opciones de empaquetado
- El método de desempaquetado
- La presencia de múltiples niveles de empaquetado.

En estos casos se pueden aplicar 2 métodos:

- Crear una herramienta para deshacer cada paso realizado por el empaquetador y hacer ingeniería inversa del proceso (nadie lo hace)

- Arrancar el programa empaqutado para encontrar el OEP y dimpear todo el proceso desde memoria (lo común)

El proceso

1. Encontrar el punto de entrada original (OEP): La dirección de memoria donde el programa comienza es movido en la sección empaquetada. El analista tiene que recuperar el archivo original
2. Reconstruir la tabla de direcciones importadas (IAT): Esta referencia las funciones usadas por el programa disponibles en la API de windows. Durante la ejecución, el IAT es resuelto dinámicamente por el malware y lo que hace mucho más difícil trazar el código.

Encontrando el OEP: Saltos

Buscar por instrucciones de salto (JMP) al final de cada sección de código:

- Salto del stub de desempaquetado al OEP
- Muy visible en Stubs de desempaquetado simples

El OEP puede estar en diferentes secciones de un stub de desempaquetado:

- Establecer breakpoints en otras secciones
- La ejecución de código se parará en el momento en que cambies de sección

Encontrando el OEP: PUSHA/POPA

- PUSHA es utilizado para guardar registros de estado al principio del stub
- POPA se usa para restaurar el estado del registro al final del stub
- Se pone un breakpoint de memoria en uno de los registros de valor del stack
- Se romperá la ejecución del código cuando POPA cambie el valor
- Normalmente este código suele estar cerca del OEP.

Encontrando el OEP: Llamadas a la API

Bloquear la tabla de imports nos acerca al final del stub:

- Breakpoint en LoadLibrary/GetProcAdress
- Buscar cerca

Buscar llamadas API de Windows como GetModuleHandle o GetCommandLine:

- Rompe una de estas funciones
- OEP suele estar encima del pariente.

Bloqueando Imports

- El stup de empaquetado actua como un proxy que intercepta cada llamada a la API. La llamada

se traduce de la API al stub y ejecutada por una payload
- Se debe evitar hacerlo a mano.

Antianálisis

Técnicas usadas por diseñadores de malware para evitar que este sea detectado usando desensamblado, Debug, Máquinas virtuales o antivirus.

Anti-Desensamblado

Muchos desensambladores son lineares (Procesan una instrucción a la vez) y orientados al flujo (Examina cada instrucción y construyen una lista de localizaciones para desensamblar considerando saltos, llamads, etc..)

El malware se aprovehca de la heurística del desemsamblado, explotando las cosas que da por hecho un ensamblador. El malware introduce código para confundir el análisis de frame del stack.

Anti-Debugging

El malware puede usar el API de Windows para detectar si está siendo debuggeado:

- isDebuggerPressent: revisa el ambiente de bloque de proceso (PEB)
- CheckRemoteDebuggerPresent: Busca por un proceso en la máquina local
- NtQueryInformationProcess: Puede ser usado para revisar si se está ejecutando un debugger
- OutputDebugString: Envía un string al debugger para mostrarlo.

El malware puede hacer revisiones manuales:

- Revisa el flag BeingDebugged en 0x02 en el PEB
- Revisa el flag ProcessHeap en 0x18 en el PEB
- Revisa el flag NTGlobal localizado en 0x68 en el PEB.

El malware también puede buscar por residuos del sistema:

- Claves de Registro:
- Buscar en el sistema por archivos o directorios especificos
- Buscar en el proceso actual una lista para detectar debuggers populares
- Usar FindWindow para localizar un debugger

El malware puede buscar su propio código para detectar interrupts (INT 3, breakpoint o 0xCC). Esto puede ser evitado usando breakpoints de hardware usando registros (D0-D3), pero el malware también puede detectarlos. También puede detectar el debugger ya que cuando está en debug corre más lento. El malware puede contar Ticks usando QueryPerformanceCounter, GetTickCount o la instrucción rdtsc.

Anti-Máquina Virtual

El malware puede usar herramientas como scoopyNG para detectar que está en una máquina virtual. También puede detectar que está en una máquina virtual ya que hay instrucciones con semántica diferente en una máquina virtual que en hardware real. Por suerte gran parte del código anti máquina virtual puede ser parcheado fácilmente, pero el malware puede aprovechar exploits en la máquina para escapar al host.

Anti-Antivirus

Un antivirus realiza la siguientes operaciones detectables:

- Antes de la ejecución: Revisa hashes y firmas, usa un escaneo de emulador y análisis de código heurístico.
- Durante la ejecución: Pone anzuelos en las librerías del sistema, monitoriza el kernel, la red y el sistema de archivos.

El Malware puede usar las mismas estrategias que se usa para el anti-análisis par contrarrestar un antivirus:

- Revisa por procesos de antivirus o si ha sido instalado en una máquina virtual
- Examina las librerías para detectar los anzuelos
- Se usa ofuscación de código para dificultar la detección
- Usa empaquetadores para cambiar su firma hash.

Los

Detección y Eliminación

En esta parte se ve como detectar y eliminar malware mediante el uso de Antivirus, Yara, Firewalls y proxies.

base de datos indicadora de compromiso

Esta base de datos exite para revisar por malware no detectado o nuevo, aislar infecciones y borrar malware presente en una intranet. Estas bases de datos deben ser compartidas entre compañías para prevenir la difusión de nuevo malware.

Indicadores que caracterizan el malware:

- Dirección IP, nombre de dominio, claves de registro, hashes, nombres de archivo, etc…
- Tan complejo como una regla IDS para detectar una familia de malware.

El diseño de este tipo de base de datos es todavía un arte y debe evitar tanto falsos positivos como falsos negativos. Para evitar eso se usan numerosas herramientras como antivirus, firewalls, proxies, etc…

Defensa de Malware

Para proteger una red, se deben establecer múltiples capas de defensa:

- Antivirus
- Firewall
- Proxy
- Sistema de detección/prevencion de intrusiones (UDS/IPS)

Antivirus

Programas diseñados para detectar y eliminar viruses, pueden enfrentarse a tipos de malware como spyware, adware, gusanos, troyanos, rootkits... Generalmente para la defensa se recomiendan los siguientes mecanismos:

- Firma digital: Busca por hashes, firmas de hosts o redes y los compara con una base de datos de firmas de malware.
- Heurística: revisa los archivos por patrones de códigos similares a los presentes en una familia de viruses.
- Detección de Rootkits: En ocasiones los rootkits pueden hacer los antivirus inefectivos, además de ser difíciles de eliminar, muchas veces siendo necesario una reinstalación completa del sismtema
- Protección en Tiempo Real: Monitoriza los sistemas del equipo para detectar actividad sospechosa mientras los datos son cargados a la memoria activa del equipo.

El problema de los antivirus es que muchas veces pueden dar falsos positivos, que hacen que un archivo o programa seguros sean identificados como virus, lo que puede producir problemas en el sistema operativo o hacer algunos programas inutilizables. Además, si se arranca un antivirus con protección en tiempo real pueden haber conflictos o problemas de rendimiento:

- Conflictos: Muchas veces se recomienda apagar estos antivirus cuando se instala actualizaciones del sistema o drivers
- Rendimiento: Juegos y programas de tiempo real pueden sufrir una gran pérdida de calidad.

ClamAV es un antivirus de la empresa Sourcefire que es open source pensado para sistemas Unix, aunque en la actualidad tiene compatibilidad también con windows. Una vez instalado, usar ClamAV es muy simple, solo es necesario actualizar la base de datos de firmas de malware y escanear el sistema. Para actualizar la base de datos se usa el comando “freshclam” y para ejecutar un análisis del sistema se usa “clamscan”. También podemos crear nuestras propias firmas con la siguiente sintaxis:

```
Name:Type:Offset:HexadecimalCode
```

- Nombre corresponde al nombre del malware
- Tipo corresponde con el tipo de archivo (0 para cualquiera, 1 para binarios windows, 2 para scripts, 3 para HTML, 4 para email, 5 para multimedia, 6 para binarios de linux y 7 para ASCII)
- Offset corresponde a la dirección inicial para buscar por la firma presente en el último campo, se puede usar * para evitarlo
- El código hexadecimal se refiere a la firma por la que se debe buscar

Firmas

Una firma ideal identifica un archivo de forma exacta y tiene la propiedad de que dos archivos diferentes no pueden compartir el hash y que no se puede obtener el archivo del hash. Pueden ser usadas para revisar si un archivo ha sido modificado o corrompido durante una transmisión de datos usando funciones como MD5, SHA4 o SHA256. De todas formas los codificadores de malware pueden usar bytes aleatorios son modificar el comportamiento del código para producir firmas diferentes con el mismo código.

Firmas Yara y Reglas

YARA (Yet Another Ridiculous Acronym) es el nombre de una herramienta usada para la detección de malware e investigación. YARA proporciona un sistema de reglas para crear descripciones de familias de malware basado en texto o patrones binarios. Una descripción es una regla Yara y una regla que consiste en sets de cadenas de texto y una expresión booleana. Yara Puede ser combinado con otras herramientras, trayendo de serie módulos para procesar PE, ELF analisis, además de tener soporte para el sandbox Cuckoo.

Un ejemplo de yara podría ser el siguiente:

```
rule silent_banker : banker
  {
  meta:
    description = "This is just an example"
    threat_level = 3

    in_the_wild = true
  strings:
    $a = {6A 40 68 00 30 00 00 6A 14 8D 91}
    $b = {8D 4D B0 2B C1 C0 27 99 4E 59 F7 F9}
    $c = "UVODFRYSIHLNWPEJXQZAKCBGMT"
  condition:
    $a or $b or $c
  }
```

Firewall

Sistema de seguridad de red que monitoriza un controla el tráfico de red que entra y sale del sistema. Se basa en reglas de seguridad predeterminadas. Normalmente establecen una barrera entre una red interna confiable y una red externa no confiable como internet. Existen varios tipos de firewall:

- Firewall de red: Filtra el tráfico entre 2 o mas redes y funciona en hardware.
- Firewall a nivel de host: Funciona dentro de un ordenador y controla el tráfico de red que entra y sale de dicha máquina. Pude funcionar como un servicio del sistema operativo o como una aplicación endpoint security.

En un firewall existen varias capas que siguen las políticas basadas en reglas:

- Capa de Red: Funciona a nivel TCP/IP y filtra los paquetes de entrada y salida por IP, Puetos TCP/UDP, dominio, MAC…
- Capa de Aplicación: Funciona con protocolos de aplicación como HTTP, HTTPS, FTP…

Servidor Proxy

Un servidor proxy es un programa o dispositivo que actua como intermediario en las peticiones realizadas de un cliente a otro server. El servidor proxy más popular es aquel que soporta búsqueda web con diferentes propósitos. Hay 2 tipos de proxys dependiendo de como sean implementados:

- Local Proxy: Está en la misma máquina que el cliente que hace la petición. Se usan para que el cliente pueda controlar el trafico y establecer reglas de filtrado.
- Network Proxy o Proxy externo: Implementado por una entidad externa. Implementa bloqueo de contenido, control de tráfico, compartición de IP, etc…
- Forward Proxy: Suele estar cerca de los clientes y puede ser configurado manualmente en el buscador para reducir el tiempo de acceso, actuando como memoria cache compartida y capturar tráfico web.
- Reverse Proxy: Localizado en el servidor, el tráfico a los servidores web es distribuido.

Squid:

- Es un software gratuito usado para capturar tráfico HTTP. Puede ser usado para mejorar la velocidad de servidores web, DNS y otras redes.

Sistemas de detección de intrusión

Es un programa usado para detectar acceso no autorizado a un ordenador o una red. Normalmente tienen sensores virtuales para descibrir anomalías que puedan indicar la presencia de malware. La operación de estas herramientras está basada en un análisis detallado del tráfico de red para compararlo con firmas de ataques conocidos o para detectar comportamiento extraño. Normalmente funciona integrado con un firewall, siendo una herramienta defensiva batante potente.

Mecanismos IDS

- basado en patrón: analiza paquetes en la red para compararlos con ataques conocidos y preconfigurados.
- Basados en heurística: Determina la actividad de red normal como el ancho de banda usado, protocolos, puertos y dispositivos que suelen estar interconectados.

Tipos de IDS

- HostIDS (HIDS): Trata de detectar modificaciones que afectan a un nodo en particular.
- NetworkIDS (NIDS): Trata de analizar ataques a un segmento entero de red, capturando todo el tráfico de red.

Limitaciones del IDPS

- Muchos ataques reales pueden ser ignorados si no mandan mucha carga al sistema.
- Para IDS basados en firmas hay mucho tiempo entre el descubrimiento de una amenaza y esta siendo notificada al IDS.
- Los paquetes cifrados no son procesados por la mayoría de IDPS.
- Los NIDS no pueden compensar la autenticación débil para debilidades en los protocolos de red, por lo que pueden ser susceptibles a ataques basados en protocolo.
- Un IDS puede ser de hardware, software o ambos. Los elementos de hardware son muy útiles debido a los requerimientos del procesador en redes de altro tráfico.

Técnicas de evasión de IDPS

- Fragmentación: envía paquetes fragmentados para que el IDPS no pueda detectar la firma del ataque.
- Evitar predeterminados: Cambiar los puertos usados habitualmente para ataques
- Ataques coordinados de baja banda ancha: Ataque de escaneo simultáneo por parte de múltiples atacantes que dificulta al IDS correlacionar paquetes capturados y deducir que hay un escaneo de red en proceso
- Spoofing de dirección
- Evasión de cambio de patron

Snort

Es un detector de intrusiones open source. Fue desarrollado para analizar el tráfico de red tanto en tiempo real como de forma forense. Puede ser usado para detectar ataques, incluyendo fingerprinting de sistemas operativos. Snort tiene 3 modos de operación:

- Sniffer Mode: Captura los paquetes de red y los muestra en la consola.

```
./snort -dev
```

- Packet Logger Mode: Guarda los apquetes en el disco duro

```
./snort -dev
./snort -dev -l
```

- Intrusion detection mode: realiza detección

```
./snort -d -l ./log -h 192.168.1.0 -c snort.conf
```