

Seguridad Informática

Tema 6 – Malware



Universidad
Rey Juan Carlos

Isaac Lozano Osorio isaac.lozano@urjc.es

04/03/2024



- **Definición y características.**
- Tipos de Malware.
- Rentabilidad.
- Contramedidas.
- Técnicas de detección.
- Adversarial examples.

- Malware, *malicious software*, es toda pieza de software que tiene como objetivo instalarse en el sistema y realizar cualquier tipo de acción malintencionada, o destructiva, sin el consentimiento del usuario.
- Esa acción abarca desde los daños a algún componente físico de la máquina, hasta daños en la información contenida en ella, o cambios su comportamiento habitual.
- Las características básicas del malware son:
 - Replicación.
 - Métodos de ocultamiento.
 - Activación.
 - Manifestación.
 - Explotación y escalamiento de privilegios.

- Se refiere a la capacidad para duplicarse a sí mismo.
- Si un programa cuenta con esta característica se le considera virus, sin importar que tenga alguna otra característica.
- Al principio, estos programas se difundían mediante dispositivos de almacenamiento externo y se activaban con ayuda del usuario.
- En la actualidad los códigos pueden expandirse y auto-propagarse a través de Internet, al aprovechar fallos en los sistemas operativos, malas configuraciones o la ingenuidad de los usuarios.

- Es la capacidad que tiene el código malicioso para evitar las protecciones comunes de los sistemas operativos, detectores de intrusos y sistemas antivirus.
 - Cifrado (polimorfismo básico).
 - Polimorfismo Avanzado o Metamorfismo.
 - Cambio de extensiones.
 - Hidra o Multi-infector.
- Activación es la capacidad del código malicioso para activarse:
 - Contador.
 - Doble Click.
 - Encender el equipo.
 - Ejecutar un programa.

- Es la capacidad que hará notar al usuario del equipo infectado que tiene un código malicioso en la máquina.
 - Enviar un mensaje.
 - Formatear el disco.
 - Borrar información.
 - Modificar la información.
 - Deteriorar el rendimiento de la infraestructura de la red.
 - Deteriorar el rendimiento de la máquina.

- Trivial:
 - Sólo replicarse
- Moderado:
 - Destrucción de la información.
- Mayor:
 - Corrupción de archivos o información.
- Severo:
 - Alteración o modificación de la información.
- Ilimitado:
 - Robo de información.

- Definición y características.
- **Tipos de Malware.**
- Rentabilidad.
- Contramedidas.
- Técnicas de detección.
- Adversarial examples.

- Debido al gran desarrollo dentro de esta área, el concepto de *malware* abarca una amplia variedad de desarrollos software:
 - Adware.
 - Attack kit.
 - Auto-rooter.
 - Backdoor.
 - Downloaders.
 - Drive-by-download.
 - Exploits.
 - Flooders.
 - Keyloggers.
 - Logic bomb.
 - Macro virus.
 - Mobile code.
 - Rootkit.
 - Spammer.
 - Spyware.
 - RAT (Remote Access Trojan).
 - Troyano.
 - Virus.
 - Gusano.
 - Zombie.

- Una manera de clasificarlos en teniendo en cuenta tres factores que definen su ciclo de vida:
 - Su método de propagación.
 - Su método de ocultación.
 - Su objetivo una vez activado.
- También veremos una clasificación desde el punto de vista de la estructura del ejecutable.

- Desde el punto de vista de su modo de propagación distinguimos:
 - **Virus:** es el más conocido. Tiene por objetivo alterar el funcionamiento normal del ordenador sin el conocimiento ni autorización del usuario. Se instalan dentro del código de un programa del equipo objetivo, y cuando este se ejecuta, infecta a otros archivos del sistema. Requieren de la interacción del usuario para infectar a nuevos equipos.
 - **Gusanos (Worms):** es un tipo de malware que se replica y se mueve por las redes de ordenadores de forma autónoma, buscando nuevas víctimas. Los gusanos no requieren interacción con el usuario para propagarse, pero sí para instalarse.

- El modo de ocultación hace referencia a cómo el malware se hace persistente en el sistema y se asegura el acceso al equipo infectado. En este caso hablamos de:
 - **Backdoors:** una vez infectado el ordenador, el atacante logra la persistencia instalando métodos alternativos de acceso.
 - **Rootkits:** hacen posible que otro malware se oculte en su dispositivo y pueden dificultar, incluso imposibilitar, la limpieza de la infección.
 - **Trojanos:** se hacen pasar por software inofensivo, pero realmente incluyen una parte maliciosa oculta. Generalmente se usan para robar información.
 - **Drive-by-downloads:** la infección se produce cuando el usuario visita un sitio web infectado, el código se descarga e instala.
 - **Rogue antivirus:** el malware muestra una alerta mientras navegamos por internet sobre un virus detectado en el sistema, y se ofrece la posibilidad de descargar un antivirus para realizar un análisis en profundidad.



- El siguiente paso es obtener beneficio de la máquina:
 - **Spyware/adware:** se centran en recolectar datos del comportamiento de usuario durante la navegación (spyware), o en mostrar ventanas emergentes con anuncios (adware). No se pueden considerar malware como tal porque no destruyen datos, pero sí se instalan usando estrategias engañosas.
 - **Keyloggers:** este software registra y almacena en un fichero todas las pulsaciones de teclado realizadas.
 - **Bots:** añade al ordenador a una red de ordenadores “zombies” controlados por un servidor central gestionado por el atacante.
 - **Malvertising:** aparecen en forma de anuncio, banner de propaganda o ventana emergente, que al cargarse, redirigen o cargan contenido de otro sitio web que tiene el malware.
 - **Ransomware:** “secuestra” los datos cifrando todo el contenido del disco duro y solicita un rescate.

Tipos de Malware

La UOC sufre un ransomware que afecta a su campus virtual: así es el último ciberataque que golpea a una universidad española



3 Enero 2022 - Actualizado 3 Enero 2022, 14:04

 33 Comentarios


servicio a la UAB

Barcelona (UAB) sufrió un ciberataque en sus servicios virtuales.


B
atònoma
lona



Your network has been infected!



Your documents, photos, databases and other important files encrypted



To decrypt your files you need to buy our special software - **General-Decryptor**



Follow the instructions below. But remember that you do not have much time

General-Decryptor price
the price is for all PCs of your infected network

You have **8 days, 19:07:29**

- * If you do not pay on time, the price will be doubled
- * Time ends on Mar 28, 16:30:11

Current price

214151 XMR
≈ 50,000,000 USD

After time ends

428302 XMR
≈ 100,000,000 USD



- Otra clasificación puede ser teniendo en cuenta la estructura del ejecutable y la capacidad de **cambiar su estructura** en cada ejecución.
- Existe malware que es capaz de mutar su estructura para evitar ser detectado:
 - **Polimórfico**: son gusanos que incluyen una serie de funciones que modifican el ejecutable del malware para evitar ser detectado. La función y el comportamiento (el núcleo) no cambia, sólo cambia el fichero ejecutable.
 - **Metamórfico**: se recodifican. Su código cambia (se reorganiza) cada vez que se propagan o cada vez que son distribuidos. Como su código cambia, un antivirus basado en firmas podría no detectarlo.

- Definición y características.
- Tipos de Malware.
- **Rentabilidad.**
- Contramedidas.
- Técnicas de detección.
- Adversarial examples.

- El desarrollo de malware se ha profesionalizado y se ha convertido en una fuente de ingresos.
- La mayor parte del malware que se desarrolla tiene como objetivo obtener beneficios económicos:
 - **Click fraud**: utilizan el modelo de negocio de los clicks y visitas a ciertas páginas. Se puede alquilar una botnet cuya tarea es hacer clicks en ciertos enlaces, generando beneficios.
 - **Extorsión**: es el caso del ransomware. Se extorsiona a la víctima impidiéndole el acceso a todos los archivos almacenados en el equipo.
 - **Spam**: se contratan botnets para realizar campañas de envío de spam.
 - **MaaS (*Malware as a Service*)**: se alquila/vende el uso de botnets, creación de exploit-kits a medida, en los que se alquila la infraestructura de la botnet y servicios de soporte técnico.

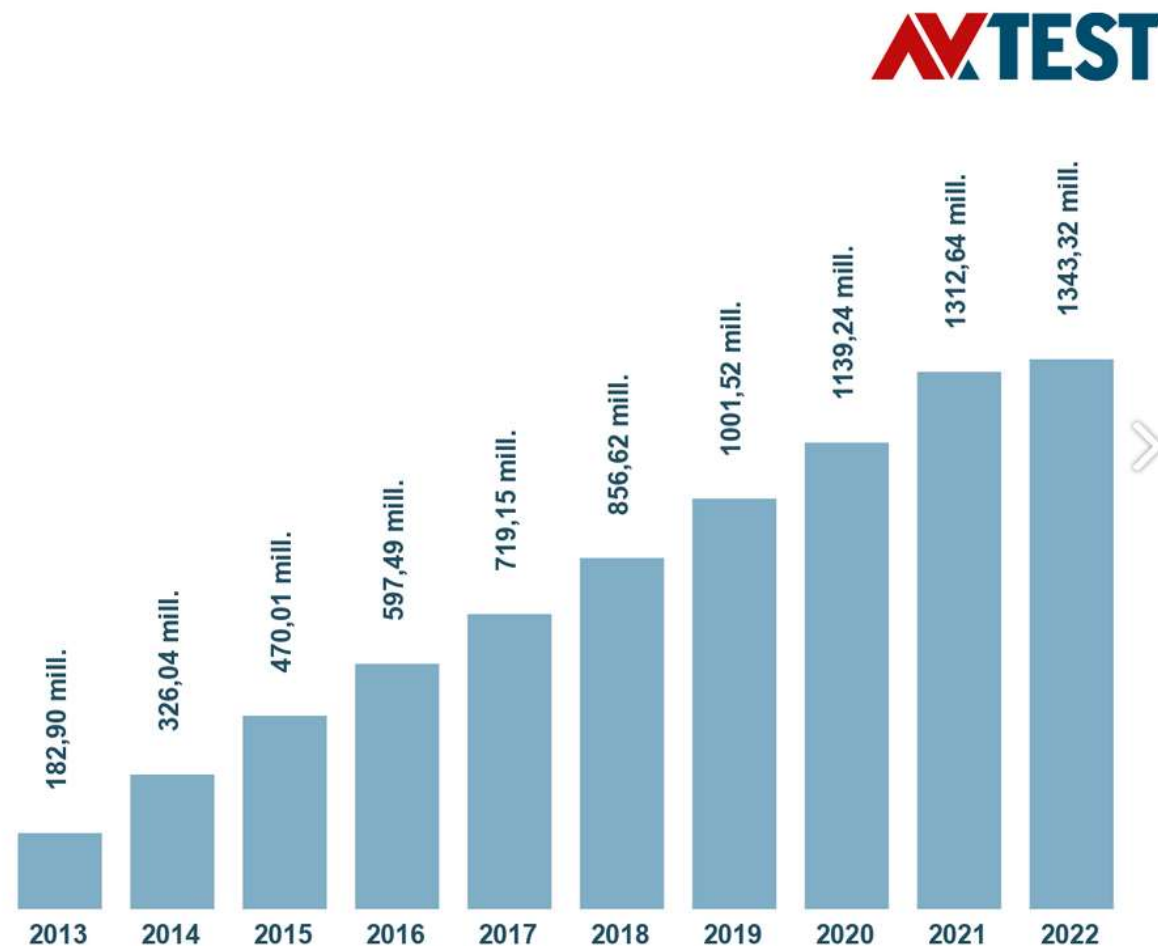
- Definición y características.
- Tipos de Malware.
- Rentabilidad.
- **Contramedidas.**
- Técnicas de detección.
- Adversarial examples.

- Existen algunas contramedidas conocidas por todos para evitar que se instale malware en las máquinas.
- Mantener el software actualizado.
- Si una aplicación no se utiliza, mejor desinstalarla.
- Tener cuidado con los programas que se instalan y las fuentes de donde se obtienen.
- No abrir emails de remitentes desconocidos. Cuidado con los emails que tienen un asunto llamativo.
- No proporcionar datos personales por correo electrónico.
- Mantener actualizado el sistema operativo y el antivirus.

- Definición y características.
- Tipos de Malware.
- Rentabilidad.
- Contramedidas.
- **Técnicas de detección.**
 - Análisis estático.
 - Análisis dinámico.
- Adversarial examples.

- El principal problema que hay a día de hoy con el malware es:

Cantidad total de malware



Última actualización: 06 de April de 2022

Copyright © AV-TEST GmbH, www.av-test.org



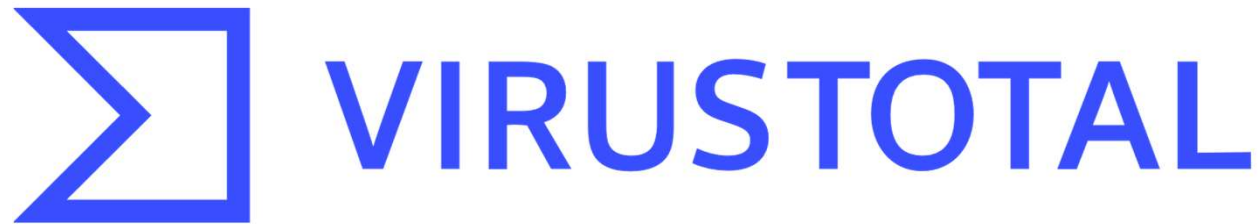
- Se necesita del desarrollo de técnicas rápidas para la identificación de malware.
- Este problema se puede “reducir” a un problema de *Machine Learning*, más concretamente de clasificación.
- Dado un código ejecutable, queremos diseñar un sistema que nos diga si ese ejecutable es malware, o no.
- Se realiza usando dos enfoques diferentes:
 - Análisis estático.
 - Análisis dinámico.

- Definición y características.
- Tipos de Malware.
- Rentabilidad.
- Contramedidas.
- **Técnicas de detección.**
 - **Análisis estático.**
 - Análisis dinámico.
- Adversarial examples.


- Trata de determinar la naturaleza del programa mediante el análisis del código, o la estructura del programa.
- El programa no se ejecuta en ningún momento.
- Se utilizan diferentes técnicas:
 - Análisis de los metadatos.
 - Escaneo con un antivirus.
 - Hashing.
 - Búsqueda de cadenas de texto.
 - Empaquetado y ofuscación.
 - Formato de fichero Portable Executable.

Escaneo con un antivirus

- Es el primer indicio, ejecutar uno (o varios) programas antivirus sobre el archivo sospechoso.
- Los antivirus se basan en bases de datos que contienen fragmentos de código de malware ya conocidos (*firmas*) y también contienen procedimientos de detección basados en el comportamiento o patrones en el código.
- El problema es que los que generan malware pueden fácilmente modificar su código, cambiando la *firma del programa* para evadir los antivirus.



41
/ 63



Community Score

⚠️ 41 security vendors and no sandboxes flagged this file as malicious


af2a2be20d7bbec0a9bb4a4dfa898aa18ef4994a9791d7cf37b7b62b379992ac

systemd-daemon

elf 64bits malware self-delete

127.02 KB
Size

2023-02-20 10:10:35 UTC
28 days ago



DETECTION

DETAILS

RELATIONS

BEHAVIOR

COMMUNITY 10

Join the VT Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Popular threat label ⚠️ trojan.linux/rotajakiro


Threat categories trojan

Family labels linux rotajakiro

Security vendors' analysis ⓘ

Do you want to automate checks?

AhnLab-V3	⚠️ Backdoor/Linux.RotaJakiro.130064	ALYac	⚠️ Backdoor.Linux.Agent
Antiy-AVL	⚠️ Trojan/Linux.Generic	Arcabit	⚠️ Trojan.Linux.Backdoor.DJ
Avast	⚠️ ELF:RotaJakiro-A [Trj]	AVG	⚠️ ELF:RotaJakiro-A [Trj]



Seguridad Informática

27

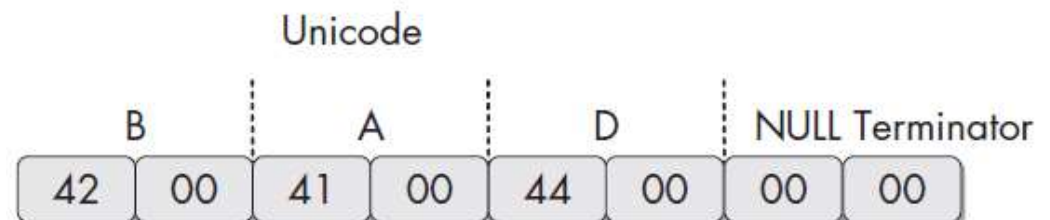
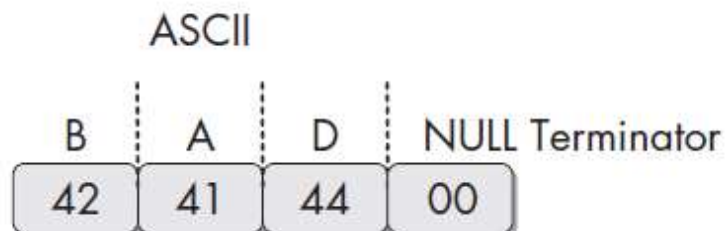
Hashing

- El hashing es una estrategia común para identificar de manera unívoca el malware.
- Se genera una hash del archivo ejecutable.
- Esa hash se usa como etiqueta.
- Se comparte esa etiqueta entre el grupo de personas interesadas en la detección.
- Se busca esa hash online para comprobar si el archivo ya se ha identificado.

```
C:\>md5deep c:\WINDOWS\system32\sol.exe  
373e7a863a1a345c60edb9e20ec3231 c:\WINDOWS\system32\sol.exe
```

Búsqueda de cadenas de texto

- Todos los programas tienen “Strings” si imprimen un mensaje, se conectan a una URL o copian archivos a una dirección concreta.
- La búsqueda de esas cadenas de caracteres nos proporciona información acerca del programa que estamos analizando.
- Existen programas que extraen esos Strings teniendo en cuenta que tanto en ASCII como en Unicode, los Strings terminan con el carácter NULL.



Búsqueda de cadenas de texto

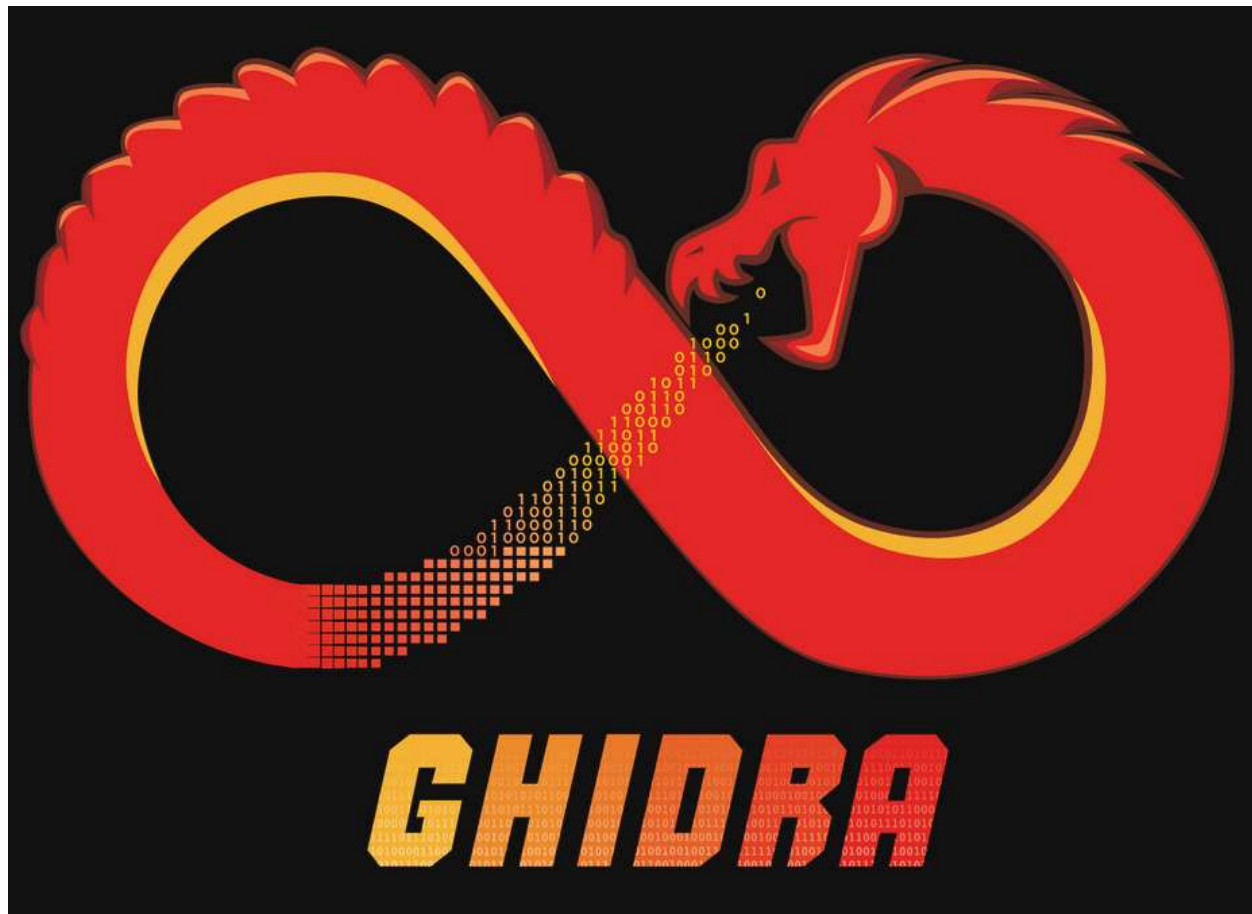
- Estos programas extraen las cadenas de texto que terminan con el carácter NULL pero es tarea del analista determinar si son correctas o no.

```
C:>strings bp6.ex_  
VP3  
VW3  
t$@  
D$4  
99.124.22.1  
e-@  
GetLayout  
GDI32.DLL  
SetLayout  
M}C  
Mail system DLL is invalid.!Send Mail failed to send message.
```

Búsqueda de cadenas de texto

- Otro enfoque consiste en la definición del Grafo de Flujo de Control para los fragmentos de código.
- Se extraen los métodos o la funcionalidad de un código.
- Por cada método, se construye en CFG (*Control Flow Graph*).
- Después se parsea cada fragmento de código con una gramática CFG para obtener una representación simbólica del código ejecutable.
- Por último se comparan esas representaciones simbólicas.

- Programa *open source* para realizar reversing:



Empaquetado y ofuscación

- Es común que los desarrolladores de malware usen técnicas de empaquetado y ofuscación para hacer más difícil la tarea de detección.
- Código ofuscado es aquel cuya funcionalidad se ha tratado de ocultar.
- Las técnicas de empaquetado son aquella en las que el malware está comprimido y no se puede analizar.
- Por lo general, el código legítimo suele contener muchos Strings, mientras que el malware ofuscado apenas tiene Strings.

Formato de fichero *Portable Executable*

- Las técnicas de detección vistas hasta ahora no analizan nada del formato del archivo.
- El formato de fichero *Portable Executable* (PE) es una estructura de datos que contiene la información necesaria para que el cargador de programas de Windows (*Windows loader*) sea capaz de ejecutar el código.
- Los archivos PE empiezan con una cabecera que incluye información sobre el código, las librerías necesarias y requisitos de espacio.

Formato de fichero *Portable Executable*

- Una fuente de información muy valiosa es la lista de funciones que se importan en el código.
- Las librerías se pueden enlazar (*linked*) dentro de un código de manera estática, dinámica o en tiempo de ejecución.
 - *Static linking*: es la manera menos usada. Consiste en copiar todo el código de la librería dentro del ejecutable. Si se analiza el código es difícil diferenciar el código de la librería del código de la aplicación.
 - La mayoría de los malware usan el enlace en tiempo de ejecución, *runtime linking*. Estos códigos se “conectan” a las librerías solo cuando la función se ejecuta.
 - *Dynamic linking*: el sistema operativo host se encarga de buscar las librerías cuando el programa se carga en memoria.

- Es posible hacerse una idea del programa en cuestión si prestamos atención a las DLLs que utiliza:

DLLs	Descripción
Kernel32.dll	DLL muy común que contiene funcionalidad básica como acceso y manipulación de memoria, archivos o recursos hardware.
Advapi32.dll	Proporciona acceso a funcionalidad más avanzada como el Gestor de Servicios o el Registro.
User32.dll	Contiene funcionalidad relacionada con la GUI como botones, barras de scroll, y componentes para controlar y responder a las acciones del usuario.
Gdi32.dll	Esta librería contiene funcionalidad para visualizar y manipular gráficos.
Ntdll.dll	Esta DLL es la interfaz con el kernel de Windows. Los ejecutables normalmente no importan este archivo directamente, sino que utilizan Kernel32.dll. Si un código importa esta librería significa que el autor quiere acceder a funcionalidad que no está incluida en los programas típicos de Windows. Con esta librería se puede ocultar cierta funcionalidad o manipular los procesos.
WSock32.dll y Ws2_32.dll	Son DLLs para el trabajo en red. Si un programa usa estas librerías, seguramente se vaya a conectar a una red o a realizar tareas en red.
Wininet.dll	Esta DLL contiene funcionalidad de red de alto nivel. Por ejemplo, implementa protocolos como FTP, HTTP o NTP.

- El archivo PE contiene una cabecera y una serie de secciones.
- La cabecera contiene metadatos sobre el fichero, y las secciones más interesantes son las siguientes:
 - `.text`: contiene las instrucciones que va a ejecutar la CPU. El resto de secciones contiene datos e información para ayudar a las instrucciones. Esta sección es la única que debería tener código ejecutable.
 - `.rdata`: contiene información de las librerías que se importan y sobre las funciones que se exportan. A veces esta información está incluida en las secciones `.idata` y `.edata`.
 - `.data`: contiene los datos globales de la aplicación, los que son accesibles desde cualquier punto del programa.
 - `.rsrc`: contiene los recursos que necesita el programa y que no se consideran parte del ejecutable, por ejemplo iconos, las imágenes, o las cadenas de caracteres.

Herramientas de análisis estático

- Información del PE File:
 - Detect it easy
 - PEiD
 - CFFExplorer
- Debuggers:
 - Windbg
 - dnspy

sonarcloud

 Reliability

38 Bugs ?

E

 Maintainability


673 Code Smells ?

A

 Security

30 Vulnerabilities ?

E

 Security Review

82 Security Hotspots ?  0.0% Reviewed

E

Coverage

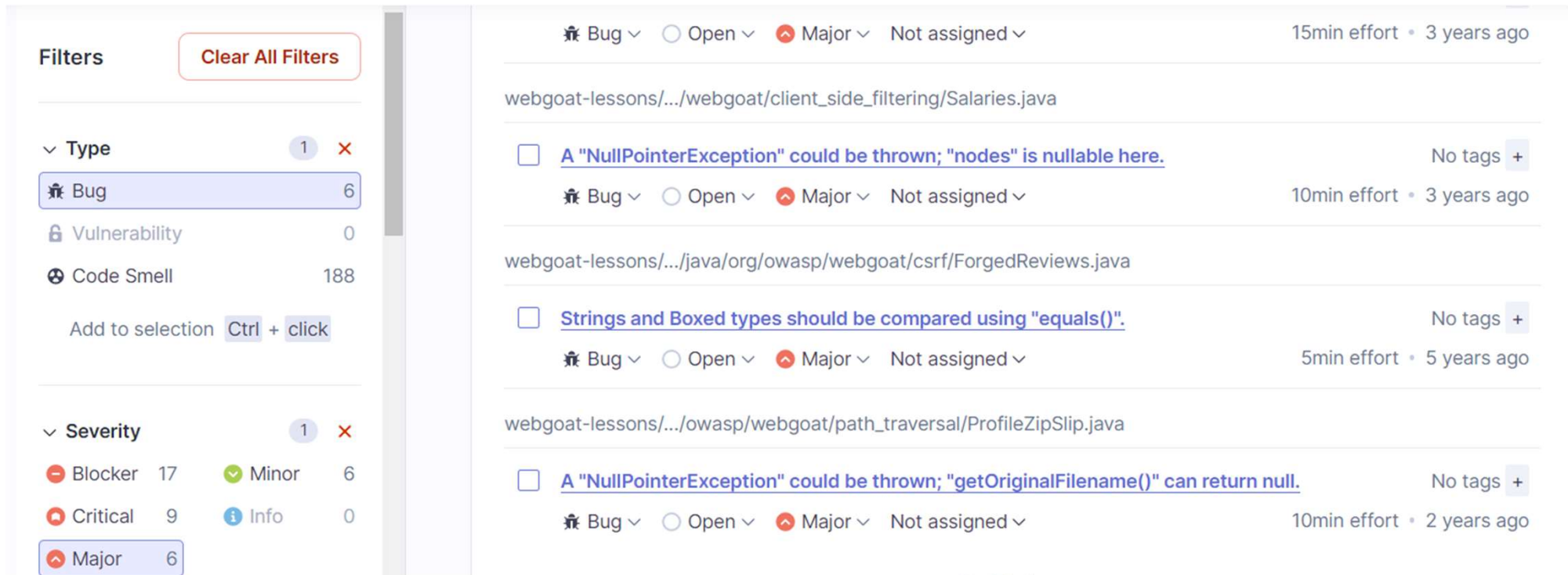
0.0% Coverage ?



Duplications

1.6% Duplications ?





The screenshot displays the SonarCloud web interface. On the left, a 'Filters' sidebar allows users to refine search results. It includes a 'Clear All Filters' button and two main sections: 'Type' and 'Severity'. The 'Type' section shows 'Bug' selected with 6 results, while 'Vulnerability' (0) and 'Code Smell' (188) are also listed. The 'Severity' section shows 'Major' selected with 6 results, alongside 'Blocker' (17), 'Critical' (9), 'Minor' (6), and 'Info' (0). The main area on the right lists specific code quality issues. Each issue entry includes a file path, a checkbox for selection, a brief description of the issue, and metadata such as effort and time since the issue was reported. The issues shown are:

- Issue 1:** File: `webgoat-lessons/.../webgoat/client_side_filtering/Salaries.java`. Description: `A "NullPointerException" could be thrown; "nodes" is nullable here.`. Metadata: 15min effort, 3 years ago.
- Issue 2:** File: `webgoat-lessons/.../java/org/owasp/webgoat/csrf/ForgedReviews.java`. Description: `Strings and Boxed types should be compared using "equals()".`. Metadata: 10min effort, 3 years ago.
- Issue 3:** File: `webgoat-lessons/.../owasp/webgoat/path_traversal/ProfileZipSlip.java`. Description: `A "NullPointerException" could be thrown; "getOriginalFilename()" can return null.`. Metadata: 5min effort, 5 years ago.
- Issue 4:** File: `webgoat-lessons/.../owasp/webgoat/path_traversal/ProfileZipSlip.java`. Description: `A "NullPointerException" could be thrown; "getOriginalFilename()" can return null.`. Metadata: 10min effort, 2 years ago.

- Definición y características.
- Tipos de Malware.
- Rentabilidad.
- Contramedidas.
- **Técnicas de detección.**
 - Análisis estático.
 - **Análisis dinámico.**
- Adversarial examples.

- Por análisis dinámico entendemos cualquier tipo de análisis realizado después de ejecutar el código.
- Es el segundo paso en el análisis de malware y se realiza después del análisis estático.
- Generalmente se suelen recurrir a estas técnicas cuando debido a la ofuscación, el empaquetado, o cualquier otra razón, el análisis estático no ha tenido éxito.
- La detección dinámica permite observar la verdadera funcionalidad del malware, porque la existencia de un string sospechoso no significa que dicho string se ejecute.
- La idea es ejecutar el código y analizar las trazas del programa, los paquetes de red, la modificación de memoria, ...

Comportamientos típicos del malware:

- Modificar el sistema de archivos.
- Modificar los registros de Windows para cambiar configuraciones.
- Cargar drivers.
- Acciones sobre la red.

- Este tipo de detección tiene un riesgo y es que se ejecuta el código, por lo que se puede poner en riesgo el equipo y la red.
- Además, tiene limitaciones:
 - No todos los caminos de ejecución pueden ejecutarse si se ejecuta el malware.
 - Por ejemplo, existe malware que usa la línea de comandos y necesitan argumentos que van a ejecutar diferente funcionalidad. Si no conocemos esas opciones no podremos evaluar toda esa funcionalidad.
 - En este caso se suele recurrir a técnicas avanzadas de análisis dinámico.

Sandboxes

- Son un elemento clave en este tipo de detección.
- Es un mecanismo de protección para ejecutar código desconocido en un entorno seguro donde el sistema real no se verá afectado por los programas del sandbox.
- Proporcionan entornos virtualizados que simulan los servicios de red para asegurarse de que el software, o el malware, que se está evaluando funcionará correctamente.
- Algunos ejemplos: Norman SandBox, GFI Sandbox, Anubis, Joe Sandbox, ThreatExpert, BitBlaze, o Comodo Instant Malware Analysis.

- Los sistemas de sandbox presentan algunos problemas:
 - Sólo lanza el archivo ejecutable, sin opción de línea de comandos. Si el malware necesita opciones por línea de comandos, no se ejecutará nada.
 - Si el malware necesita un paquete de *command-and-control* antes de ejecutar un backdoor, el backdoor nunca se lanzará.
 - El sandbox no registra todos los eventos porque entonces la tarea se alargaría mucho en el tiempo. Por ejemplo: si el malware hace un sleep por un día antes de realizar la actividad maliciosa.
 - Existe malware que detecta si está siendo ejecutado en un sandbox o virtual machine, y en ese caso no se ejecuta, o realiza otras actividades.
 - El malware puede necesitar algunos registros o archivos que sólo están disponibles en el host.

Análisis dinámico de código

JoeSandbox Cloud BASIC

Overview ▾

Signatures ▾

Process Tree

Domains / IPs

Dropped

Static

Network ▾

Stats

Behavior ▾

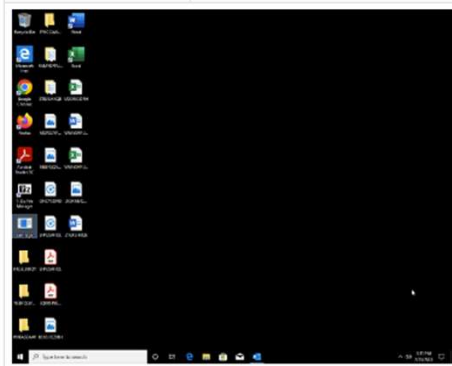
Disassembly ▾



Overview

General Information

Sample Name:	cartridge.exe
Analysis ID:	808524
MD5:	01c0ffbf4899dd96...
SHA1:	fef363e5680db20f...
SHA256:	45407220b71e13...
Infos:	



Detection

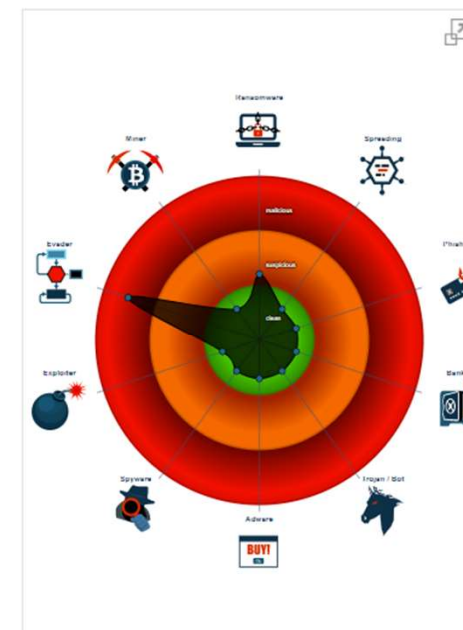


Score:	76
Range:	0 - 100
Whitelisted:	false
Confidence:	100%

Signatures

- Machine Learning detection for sample
- PE file contains section with special chars
- Yara signature match
- Uses code obfuscation techniques (call, p...
- Checks if the current process is being de...
- PE file contains sections with non-standa...
- Contains capabilities to detect virtual mac...
- Detected potential crypto function
- Sample execution stops while process w...
- PE file contains more sections than normal
- Program does not show much activity (idle)
- Entry point lies outside standard sections

Classification



- Definición y características.
- Tipos de Malware.
- Rentabilidad.
- Contramedidas.
- Técnicas de detección.
- **Adversarial examples.**

- Ataques de evasión.
- Consiste en desarrollar un sistema que sea capaz de crear “modificaciones” del malware de tal forma que un clasificador no sea capaz de clasificarlo.
- Puede clasificarlo como que es otro tipo, o familia, de malware.
- O directamente no clasificarlo como malware.

Desde el punto de vista del defensor

- Construir un sistema que es capaz de crear adversarial examples tiene muchas ventajas.
- Las más significativas:
 - Exponer vulnerabilidades del detector.
 - Mejorar la robustez del sistema de detección (generalmente una red neuronal o Deep Learning)

- La dificultad de determinar automáticamente que algo es malware.
- Que algo sea malware o no depende de la intención.
- Por ejemplo, un gestor remoto (e.g., TeamViewer) no es malware si te lo instalas tú, pero se considera como malicioso si te lo instalan sin que te enteres.
- La mayoría de las aplicaciones del móvil te cogen tantos datos que serían malware si no fuera porque nosotros les damos permiso y aceptamos las condiciones.
- Por ejemplo, subir un instalador a una plataforma de análisis de malware (como Hybrid Analysis) suele dar un resultado malicioso si no es un software puesto en una lista blanca de software conocido.

- Imaginad que construimos un sistema que supone un avance significativo en el campo de la seguridad informática.
- Por ejemplo:
 - Desarrollamos un sistema que genera adversarial examples bastante valiosos que son difíciles de detectar.
 - Desarrollamos un algoritmo capaz de descifrar un disco duro afectado por un ransomware.
- ¿Creéis que se debe publicar este sistema? Desde el punto de vista del OpenSource, o con artículos de investigación...