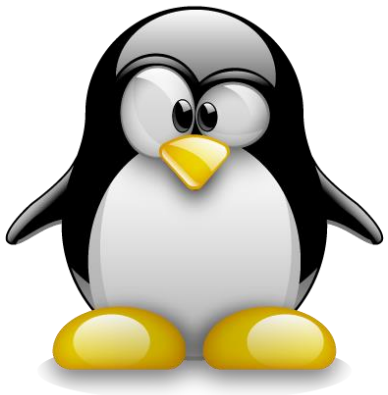
Threat not found!

@UnaPibaGeek

./shei

Sheila Ayelen Berta a.k.a Shei Winker

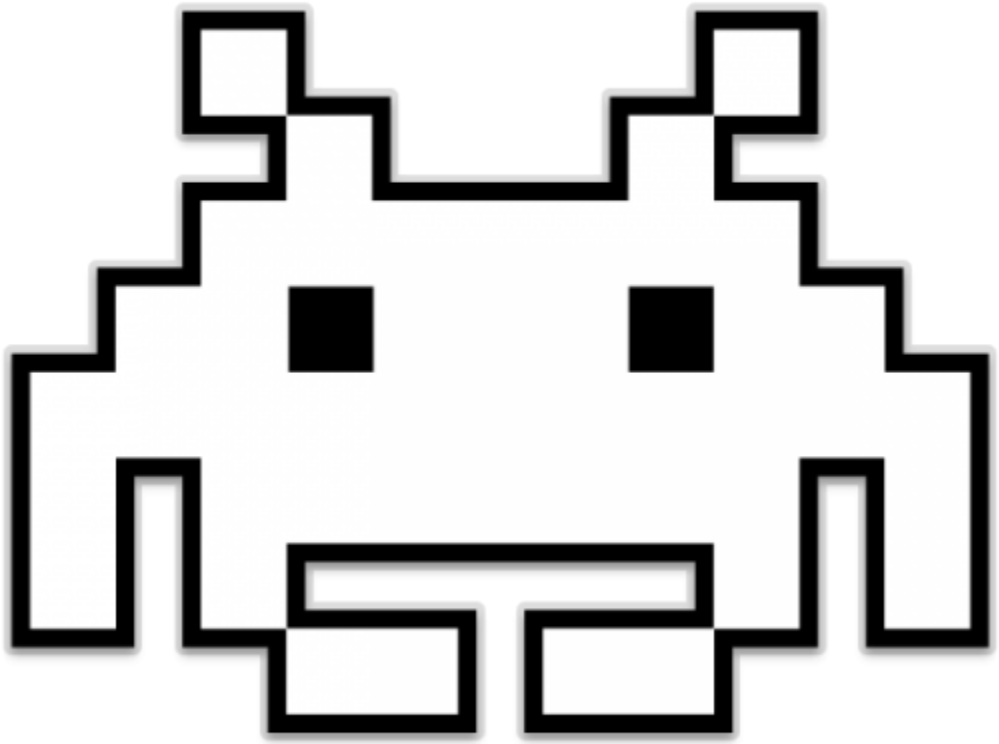
- 20 años – Apasionada del hacking desde los 12.
- Programadora en...
  - ASM (Microcontroladores y 32bits).
  - AutoIT.
  - C/C++ y JAVA (Android).
  - Python.
  - Tecnologías Web (PHP, JS, Ajax, SQL, etc.).



- Escritora de un libro de Web Hacking de 320 páginas.
- Profesora en la facultad de Ingeniería – UdeMM / Da Vinci.
- Security Research.
- Blog personal: www.SeMeCayoUnExploit.com.

@UnaPibaGeek

Start!



@UnaPibaGeek

Funcionamiento de los Antivirus

- Hoy en día son, en realidad, soluciones antimalware.
- Pueden bloquear un malware al momento de intentar infectar un equipo, o bien, realizar el proceso de desinfección si el mismo ya había sido comprometido.
- En los dos casos anteriores el primer paso es detectar el malware, para ello los antivirus utilizan dos métodos diferentes de detección llamados reactiva y proactiva.

Detección Reactiva

- Funciona a base de firmas. ¿Qué es una firma? Una pequeña parte del código del malware.
- Para crear una firma el laboratorio del antivirus debe recibir una copia del malware.
- La firma debe ubicarse donde no genere falsos positivos ni permita que el malware siga funcionando si alguien la modifica.
- En el equipo del usuario, el antivirus compara cada archivo a analizar con las firmas de la base de datos. Si encuentra coincidencias mostrará un alerta.

Rompiendo firmas I:
Recursos del ejecutable y DLL/OCX

- Una firma puede estar ubicada en los recursos del ejecutable (por ejemplo en el icono) o en los metadatos.
- Los nombres de las DLL y OCX también podrían ser usados para ubicar una firma.

```
B9 02 57 72 69 74 65 46 69 6C 65 00 4B 45 52 4E  ¹.WriteFile.KERN
45 4C 33 32 2E 64 6C 6C 00 00 57 53 4F 43 4B 33  EL32.dll..WSOCK3
32 2E 64 6C 6C 00 00 00 00 00 00 00 00 00 00 00  2.dll...........
```

Malwarebytes	Backdoor.Tiny

http://goo.gl/Wr9fU2

```
B9 02 57 72 69 74 65 46 69 6C 65 00 6B 65 72 6E  ¹.WriteFile.kern
65 6C 33 32 2E 44 4C 4C 00 00 77 73 6F 63 6B 33  el32.DLL..wsock3
32 2E 44 4C 4C 00 00 00 00 00 00 00 00 00 00 00  2.DLL...........
```

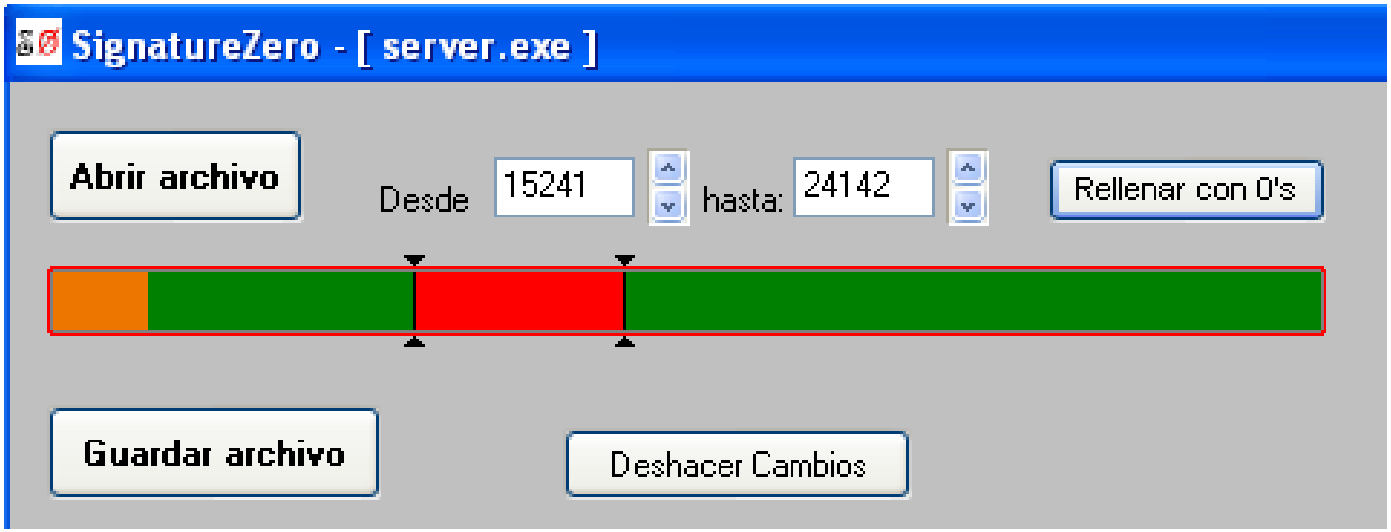
Malwarebytes	✔

http://goo.gl/vNVgw4

@UnaPibaGeek

Rompiendo firmas II:
Acorralar con 00's

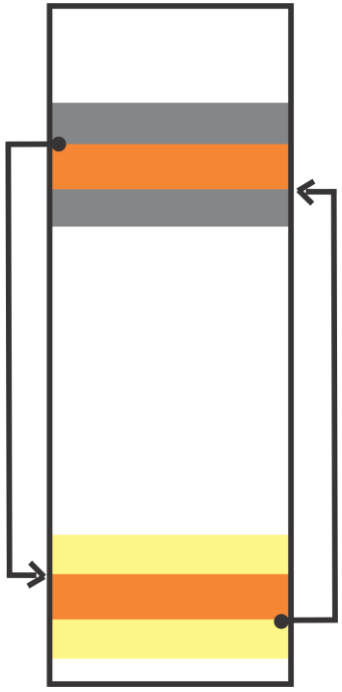
- Es posible encontrar la ubicación exacta de la firma dentro del malware.
- Hay diferentes métodos para lograrlo, uno es acorralar la firma llenando de ceros diferentes partes del ejecutable, hasta que damos con la locación exacta de la misma.



Rompiendo firmas III:

...y no morir en el intento.

- Al rellenar la firma con ceros cabe la posibilidad de que el malware deje de funcionar, aunque solo hayamos modificado una parte mínima de su código.



Para evitar esto, en lugar de llenar la firma con ceros copiaremos su código o parte del mismo y lo moveremos a otro sector del ejecutable.
De este modo, el antivirus no encontrará la firma y el malware seguirá funcionando ya que solo alteramos el flujo de ejecución (*).

Firma
Código movido
Espacio libre (NOPs)

* Las flechas indican cómo se altera el flujo de ejecución.

@UnaPibaGeek

JMP's con OllyDBG

Paso 1: Localizar la firma en el OllyDBG:

```
00401015   .  E8 D0020000    CALL <JMP.&WSOCK32.#23>
0040101A   .  A3 A2314000    MOV DWORD PTR DS:[4031A2],EAX
0040101F   .  66:C705 A6314(
```

Paso 2: Mover la firma a un espacio libre o "hueco de NOP's" e insertar un JMP a la dirección inmediatamente posterior a la firma (JMP 0040101F):

```
004012F2      00              DB 00
004012F3   >  A3 A2314000     MOV DWORD PTR DS:[4031A2],EAX
004012F8   .  66:C705 A6314(  MOV WORD PTR DS:[4031A6],2
00401301   .  C705 AA314000   MOV DWORD PTR DS:[4031AA],0
0040130B   .  66:C705 A8314(  MOV WORD PTR DS:[4031A8],611E
00401314   .^ E9 06FDFFFF     JMP tini-oll.0040101F
00401319      00              DB 00
```

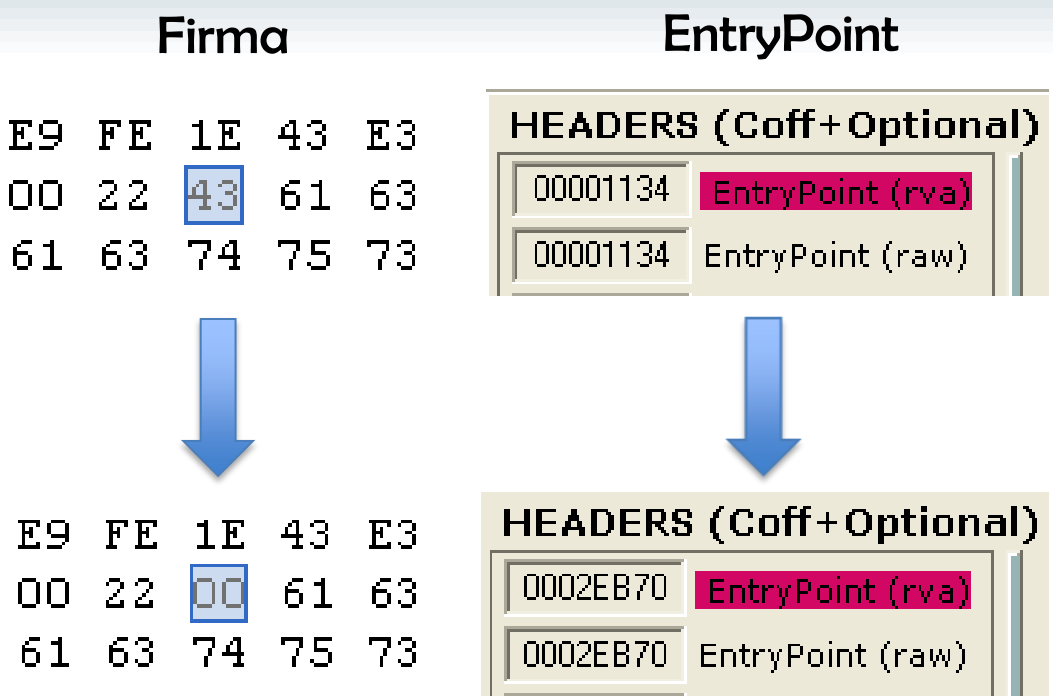
Paso 3: Reemplazar la firma por un JMP a la nueva ubicación del código de la misma (JMP 004012F3):

```
00401015   .  E8 D0020000    CALL <JMP.&WSOCK32.#23>
0040101A   .v E9 D4020000    JMP tini-oll.004012F3
0040101F   >  66:C705 A6314(  MOV WORD PTR DS:[4031A6],2
```

@UnaPibaGeek

Alternativa: MOVE BYTE PTR

¿Qué sucede si la firma no se encuentra en el disco pero si en la memoria? Evadimos la protección reactiva del AV al romper la firma, y el malware se ejecutaría correctamente ya que en memoria su código sigue siendo el mismo.




```
MOV BYTE PTR[0040116B],43
PUSH 00401134
RETN
```

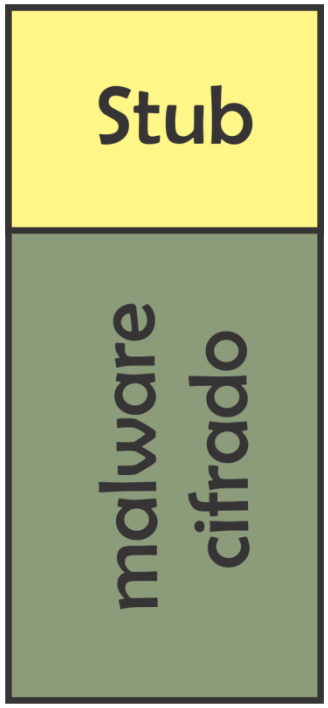
```
0042EB70  $ C605 6B114000  MOV BYTE PTR DS:[40116B],43
0042EB77  . 68 34114000    PUSH cactus-m.00401134
0042EB7C  . C3             RETN
0042EB7D    90             NOP
0042EB7E    90             NOP
```

@UnaPibaGeek

¿No era más fácil cifrar el malware?

Existen programas conocidos como “crypters” que facilitan el uso de algoritmos simétricos, para cifrar Malware.

En el momento que se cifra el malware todas las firmas de los antivirus se rompen, ya que el contenido de ese ejecutable pasa a ser otro: todo el malware cifrado.



Sin embargo, hoy en día cifrar un malware posee al Menos dos grandes desventajas si se trata de evadir AVs:

- El “stub”, la parte que se añade al malware cifrado para poder descifrarlo y ejecutarlo en memoria, es difícil de hacer pasar desapercibido al AV.
- Un malware cifrado levanta muchas más sospechas a la detección proactiva del AV.

Detección Proactiva

- Surgió para dar respuesta en situaciones donde la detección reactiva no puede. Por ejemplo, cuando aparece un malware nuevo y aun no existen firmas para detectarlo.

- Se basa en heurística para determinar si un archivo es ofensivo: Compara el contenido del mismo con comportamientos típicos de un malware y por cada semejanza "levanta una bandera".

- Hay tres tipos de heurística / algoritmos heurísticos:
  - Heurística genérica.
  - Heurística pasiva.
  - Heurística activa (sandbox).

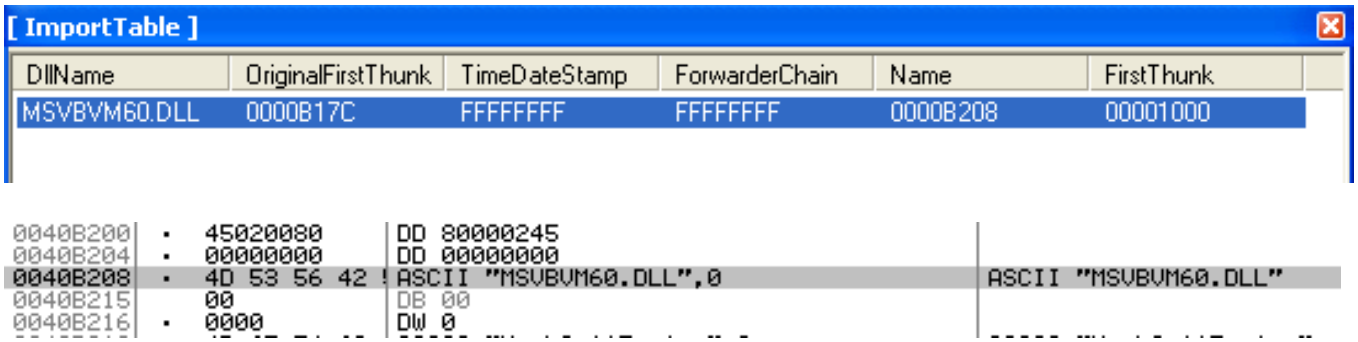
Heurística pasiva

Explora pasivamente (sin ejecutar) el contenido del archivo en busca de condiciones que indican un posible malware, por ejemplo:

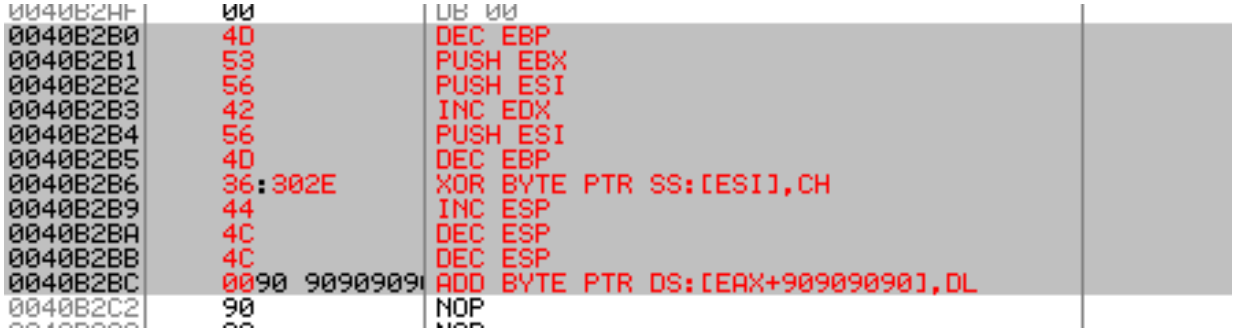
- Empaquetamiento / Cifrado: intención de ocultar el verdadero comportamiento del malware.
- Entry Point inusual: el inicio del ejecutable con instrucciones para escribir la memoria o realizar CALL's extraños, entre otras acciones fuera de lo común.
- Llamadas externas: librerías/APIs (DLL) llamadas por el ejecutable, que son comúnmente utilizadas por malware.

Evasión de heurística pasiva: Moviendo la Import Table (Parte I)

Paso 1: obtener la dirección de memoria de la llamada externa:



Paso 2: Mover las instrucciones a otro sector del ejecutable:



@UnaPibaGeek

Evasión de heurística pasiva: Moviendo la Import Table (Parte II)

Paso 3: Sustituir la locación original por NOP's o código basura:

```
0040B208    90          NOP
0040B209    90          NOP
0040B20A    90          NOP
0040B20B    90          NOP
0040B20C    90          NOP
0040B20D    90          NOP
0040B20E    90          NOP
0040B20F    90          NOP
0040B210    90          NOP
0040B211    90          NOP
0040B212    90          NOP
0040B213    90          NOP
0040B214    90          NOP
0040B215    00          DB 00
```

Paso 4: Cargar la nueva dirección en la Import Table:

[ImportTable]

DllName	OriginalFirstThunk	TimeDateStamp	ForwarderChain	Name	FirstThunk
▮▮▮▮▮▮▮▮▮▮▮	0000B17C	FFFFFFFF	FFFFFFFF	0000B208	00001000

[ImportTable]

DllName	OriginalFirstThunk	TimeDateStamp	ForwarderChain	Name	FirstThunk
MSVBVM60.DLL	0000B17C	FFFFFFFF	FFFFFFFF	0000B2B0	00001000

@UnaPibaGeek

Heurística activa (sandbox)

Ejecuta el archivo a analizar en un entorno virtualizado (también conocido como sandbox) en el cuál puede determinar con mayor precisión el verdadero comportamiento del ejecutable.

Las acciones comúnmente necesarias por los malware son aquellas que despiertan la sospecha de la heurística activa.

Por ejemplo: establecer conexiones remotas, iniciarse junto al sistema operativo, autocopiarse a carpetas del sistema, etcétera.

Desventajas:

- Alto consumo de recursos del sistema.
- Falsos positivos.

Evasión de heurística activa:
Identificar comportamientos "sospechosos"

- Conexión Remota.
- Edición del registro.
- Inyección en proceso.
- Descifrado y escritura en memoria.
- Editar archivo hosts.
- Acceder a SAM.
- Sobrescribir archivos.
- Deshabilitar funciones del sistema operativo.
- Descargar archivos.
- Borrar DLL y OCX.
- Autoborrado (melt).
- Copiarse a carpetas ocultas y del sistema.

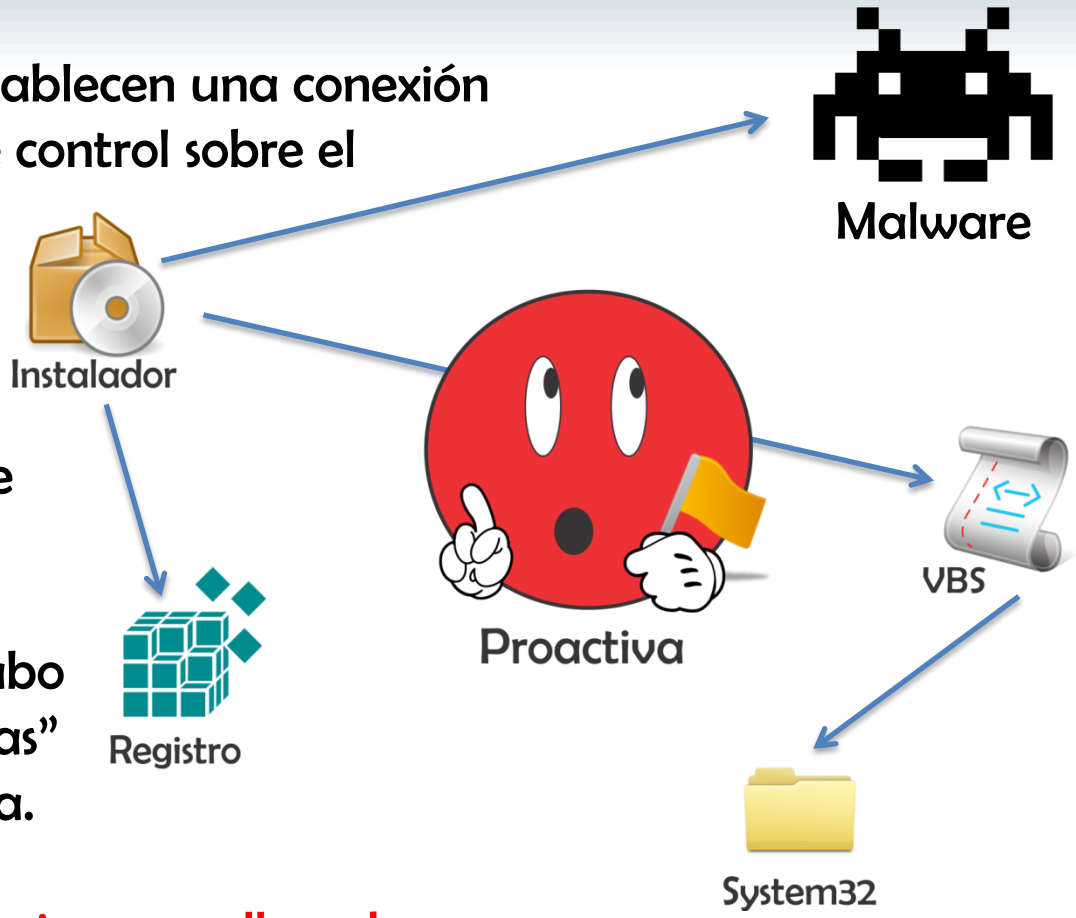
El antivirus puede mostrar una alerta si el ejecutable lleva a cabo una o varias de las acciones mencionadas arriba. La rapidez del antivirus para marcar la amenaza dependerá del nivel de gravedad de cada acción.

Evasión de heurística activa:
Distribuir comportamientos

La mayoría de los malware establecen una conexión remota para darle al atacante control sobre el equipo comprometido.

Además de eso añaden una entrada en el registro de Windows para iniciarse siempre que el equipo se encienda.

Si un solo ejecutable lleva a cabo todas esas acciones "sospechosas" el antivirus mostrará una alerta.



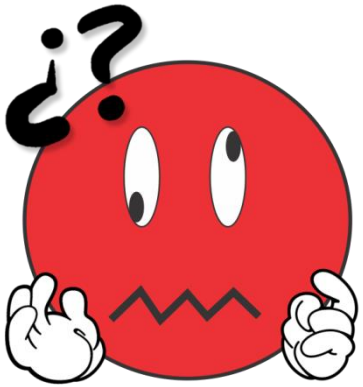
¿Pero que pasa si las acciones son llevadas a cabo por diferentes ejecutables y archivos?.

@UnaPibaGeek

Confundir a la proactiva

Soy un reproductor de música realizo conexiones remotas para "descargar canciones".

Soy una "aplicación de limpieza" por eso edito el registro.



Abro todos los archivos porque soy un "antivirus" y necesito escanear.

Antes que nada voy a bruteforcear un hash durante 5 minutos, espero no te canses de observarme.

No te toco, no te toco el aire es libre ;)

Para los programadores es fácil jugar en aquella delgada línea que existe en la detección proactiva entre un malware y un falso positivo.

@UnaPibaGeek

Malware en código: AutoIt, "el sucesor"

AutoIT Malware infected thousands of computers worldwide

September 25, 2014 By Pierluigi Paganini

G+1 8

My Page Like 57

A Greek security researche

is a combination of AutoIT

named Limitless Keylogge

welivesecurity

Noticias, opiniones y análisis de la comunidad de seguridad de ESET en Español

Últimos Posts | Tutoriales | Opinión | Videos | Artículos | Nuestros Expertos | Glosario

Cómo entender códigos maliciosos escritos en AutoIt

POR CAMILO GUTIÉRREZ AMAYA PUBLICADO 6 FEB 2014 - 07:36PM

TUTORIALES | TAGS AUTOIT DEBUGGING INJECTOR AUTOIT DECOMPILADOR EXE2AUT WIN

McAfee Labs

AutoIt and Malware: What's the Connection?

By Intel Security, Inc. on Aug 28, 2012

August 25, 2015 , 11:13 am

> Malware > AutoIt Used To Spread Malware and

blog.trendmicro.com

Toolsets

AutoIt Used To Spread Malware and Toolsets

May 6

6:59 am (UTC-7) | by Kyle Wilhoit (Senior Threat Researcher)

@UnaPibaGeek

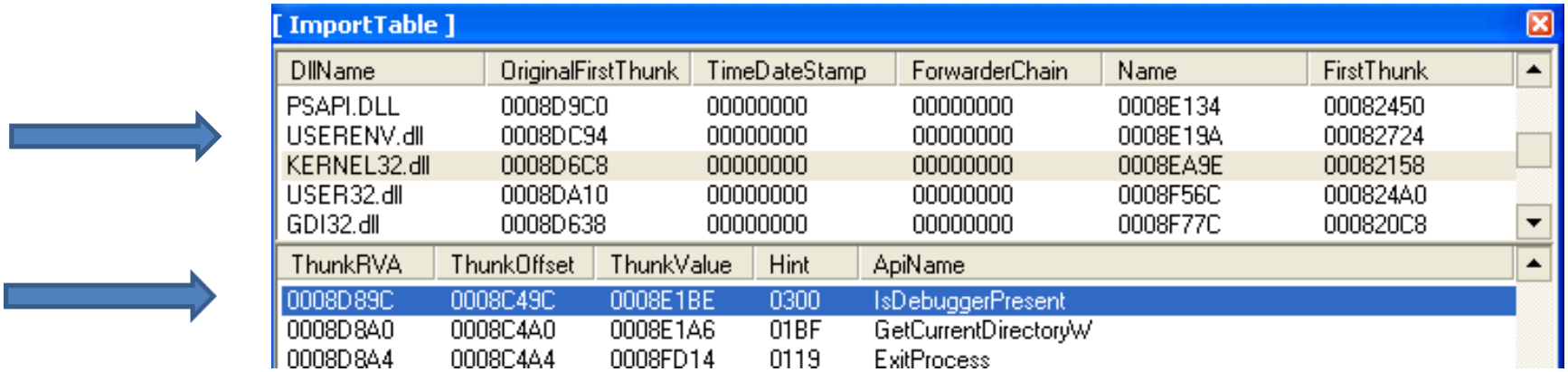
Binarios de AutoIT

AutoIt es un lenguaje de programación con una sintaxis muy similar a Visual Basic, incluso se lo conoce como *"Visual Basic killer"*.

Algunas de las características de este lenguaje por las cuales muchos desarrolladores de malware lo han elegido son:

- Orientación a eventos.
- Compatibilidad con todas las versiones de Windows.

¿Dónde está el truco?



@UnaPibaGeek

Evasión de antivirus en Android:
Migrando conceptos a otras plataformas

Al día de hoy la mayoría de las veces que un antivirus para Android logra detectar un malware, lo hace a través de una técnica similar a la detección reactiva.

Técnicas utilizadas para evitar el análisis e identificación:

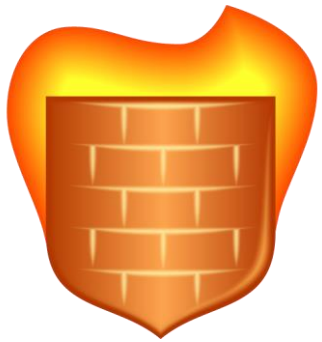
- AntiEmulación.
- AntiDebugging.
- AntiAV / AVkiller.

El uso de Broadcast Receivers en un malware para Android, podría confundir a los antivirus y complicar tanto su análisis como su detección.

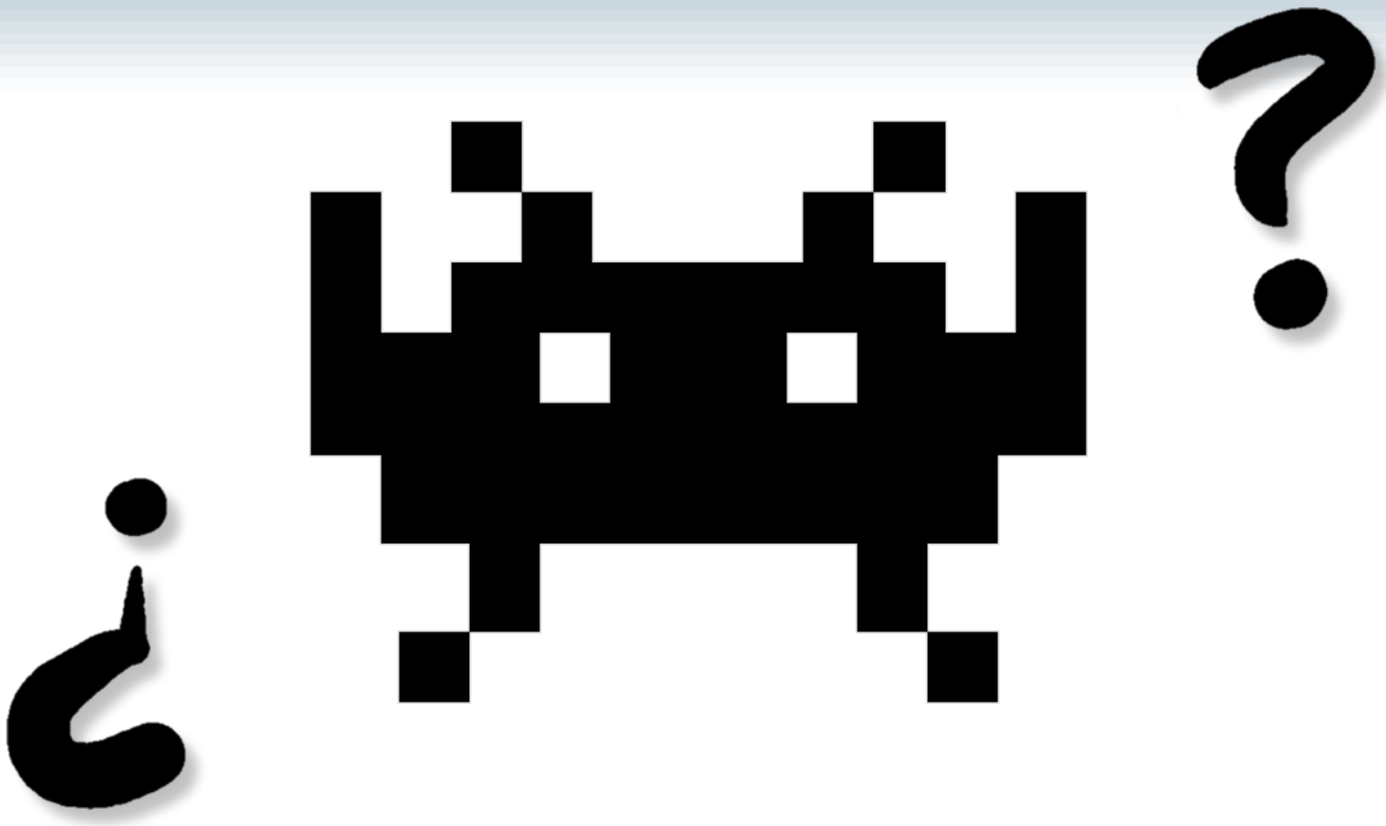
Conclusión

Si bien el antivirus nos protege de miles y miles de amenazas ya conocidas (firmas) y algunas otras no conocidas (heurística) su trabajo se complica ante el malware con ciertas características para evadirlo.

Por lo tanto es recomendable que complementemos la seguridad del AV con, al menos, un Firewall.



...¿Preguntas?...



Por Sheila A. Berta... @UnaPibaGeek