

IX JORNADAS STIC CCN-CERT

DETECCIÓN E INTERCAMBIO, FACTORES CLAVE

Madrid, 10 y 11 de diciembre 2015

Desactivación de infecciones, malware y cryptolockers: Prueba de Concepto



CENTRO CRIPTOLÓGICO NACIONAL





- Pablo San Emeterio
- Telefónica
- pablo.sanemeteriolopez@telefonica.com
-  @psaneme
- Román Ramírez
- RootedCON
- rramirez@rootedcon.com
-  @patowc

Índice

1. **Quienes somos**
2. **¿Por qué esta charla?**
3. **Trabajo anterior y otros proyectos**
4. **Técnicas identificadas**
5. **Demostración**
6. **Conclusiones**

Quienes somos

Román y Pablo

- Román es uno de los organizadores de RootedCON, colaborador de distintos proyectos y miembro activo de las diversas comunidades a las que pertenece.
- Pablo es un apasionado de la seguridad informática, trabaja en Telefónica y es conocido por sus trabajos sobre vulnerabilidades en plataformas de mensajería instantánea y técnicas de hooking.

Índice

1. Quienes somos
2. ¿Por qué esta charla?
3. Trabajo anterior y otros proyectos
4. Técnicas identificadas
5. Demostración
6. Conclusiones



¿Por qué esta charla?



¿Por qué esta charla?

- En algunos casos, las estrategias de prevención se adhieren a premisas equivocadas.
- Hay multitud de alternativas a la hora de mitigar el impacto o la evolución de una infección, ¿por qué todo parece que tiene que basarse en una *firma*?
- ¿Y si pongo en mi plataforma de correo un filtro que vaya identificando correos iguales y los ponga en una cola? ¿No es esa una mitigación para la entrada de cryptolockers?

¿Por qué esta charla?

- Un idea que tuvimos fue la de **hacer creer** e los “bichos” que están en una **sandbox** o una **VM**: en muchos casos, se desactivan sin dar mayores problemas.
- Así que pensamos en cómo hacerlo de forma estructurada.
- De eso va esta charla, de hacerte “**inmune**” a infecciones peligrosas sin necesidad de firmas, o complejos sistemas **heurísticos**.





Índice

1. Quienes somos
2. ¿Por qué esta charla?
- 3. Trabajo anterior y otros proyectos**
4. Técnicas identificadas
5. Demostración
6. Conclusiones

Trabajo anterior y otros proyectos

Jordi Vazquez (@jordisk)

- Como con toda buena idea, siempre tiene que venir un “*salao*” a estropearte la alegría por lo *superlisto* que eres...
- Jordi, con cariño, búscate otro sector. No vas a volver a trabajar en el de la Seguridad...

Jordi Vazq

- Como c
estropo

- Jordi, c
de la S

proyectos

ao” a

oajar en el

Kaspersky® Academy
CyberSecurity
for the Next Generation
International Student Conference

▶ Emulate virtual machines
To avoid malware infections

Jordi Vázquez Aira
Institut Obert of Catalonia (Barcelona, Spain)

KASPERSKY



POLITECNICO
DI MILANO

<http://kasperskycontenthub.com/kasperskyacademy/files/2013/12/aira.pdf>

Introduction

Trabajo anterior y otros proyectos

Yago Jesús

- Antiransom es una herramienta alucinante (os recomendamos su uso), y tiene muy, muy buenas ideas. Probadas con éxito en NN
- De hecho, lo hablamos con Yago y fue él quien nos señaló que @Jordisk ya había hablado de esta idea “tan original” ☹️ (Yago, a ti también te odiamos profundamente).



Índice

1. Quienes somos
2. ¿Por qué esta charla?
3. Trabajo anterior y otros proyectos
4. **Técnicas identificadas**
5. Demostración
6. Conclusiones

Técnicas identificadas

Flag de **DEBUG** en el sistema

- Uno de los mecanismos más básicos de detección pasa por identificar que nuestro proceso está dentro de un debugger:

```
BOOL WINAPI IsDebuggerPresent(void) ;
```

```
BOOL WINAPI CheckRemoteDebuggerPresent(__in HANDLE  
hProcess, __inout PBOOL pbDebuggerPresent) ;
```

```
void WINAPI OutputDebugString(__in_opt LPCTSTR  
lpOutputString) ;
```

Técnicas identificadas

Nombres de procesos EVIDENTES

- Algunos “bichos” buscan cadenas de texto conocidas en la lista de procesos:
- Ollydbg.exe, idag.exe, immunitydebugger.exe, windbg.exe
- Wireshark.exe, procexp.exe, procmon.exe.
- vboxservice.exe, vboxtray.exe
- vmware-authd.exe, vmnat.exe, vmwaredhcp.exe, vmware-usbarbitrator64.exe,...

Técnicas identificadas

Ventanas de procesos EVIDENTES

- Algunos “bichos” buscan cadenas de texto conocidas en las ventanas de las aplicaciones en ejecución (OllyDBG, Windbg, IDA,...):

- | | | |
|-------------|--------------------------------|---|
| • OllyDbg | WndClass: OLLYDBG, | txt: OllyDbg |
| • Immunity | WndClass: ID (Visible) | txt:Immunity Debugger - [CPU] |
| • IDA | WndClass: TIDAWindow | txt: The interactive disassembler |
| • WinDBG | WndClass: WinDbgFrameClass | txt:WinDbg:6.12.0002.633 X86 |
| • Procexp | WndClass: PROCEXPL | txt: Process Explorer – Sysinternals... |
| • Procmon | WndClass: PROCMON_WINDOW_CLASS | txt: Process Monitor - ... |
| • Wireshark | WndClass: gdkWindowToplevel | txt: The Wireshark Network Analyzer. |

Técnicas identificadas

Algunas características que los bichos examinan

- Por ejemplo, que corre en una máquina con muy poca RAM (posiblemente una máquina virtual).
 - GlobalMemoryStatusEx memoria < 1G
- O extraer el número de micros (si es menor de 2,...)
 - GetSystemInfo numero de procesadores < 2
- O espacio en disco por debajo de un tamaño sospechoso
 - GetDiskFreeSpaceExA < 40G

Algunas características que los bichos examinan

Técnicas identificadas

- Nombre del equipo en el que ejecutan.
 - SANDBOX
 - TRENDMICRO
 - Xp-sp3-template
 - MALTEST
 - VIRUS
 - MALWARE
- Path en el que estan ejecutando)
 - C:\sample\pos.exe
 - C:\Analysis\Sandboxstarter.exe
 - C:\Analysis
 - C:\Insidetm
 - C:\Sample

Técnicas identificadas

Crear ficheros “sospechosos habituales”

- Los Kaiser Söze de las diferentes *sandbox* y herramientas de virtualización:
- C:\\WINDOWS\\system32\\drivers\\VBoxMouse.sys, C:\\WINDOWS\\system32\\drivers\\VBoxGuest.sys, C:\\WINDOWS\\system32\\drivers\\VBoxSF.sys, C:\\WINDOWS\\system32\\drivers\\VBoxVideo.sys, C:\\WINDOWS\\system32\\vboxdisp.dll, C:\\WINDOWS\\system32\\vboxhook.dll, C:\\WINDOWS\\system32\\vboxmrxnp.dll, C:\\WINDOWS\\system32\\vboxogl.dll, C:\\WINDOWS\\system32\\vboxoglarrayspu.dll, C:\\WINDOWS\\system32\\vboxoglcrutil.dll, C:\\WINDOWS\\system32\\vboxoglerrorsru.dll, C:\\WINDOWS\\system32\\vboxoglfeedbackspu.dll, C:\\WINDOWS\\system32\\vboxoglpacspu.dll, C:\\WINDOWS\\system32\\vboxoglpassthroughspu.dll, C:\\WINDOWS\\system32\\vboxservice.exe, C:\\WINDOWS\\system32\\vboxtray.exe, C:\\WINDOWS\\system32\\VBoxControl.exe, C:\\program files\\oracle\\virtualbox guest additions\\, C:\\WINDOWS\\system32\\drivers\\vmmouse.sys, C:\\WINDOWS\\system32\\drivers\\vmhgfs.sys,...

Técnicas identificadas

Hookear funciones en todos los procesos

- Por ejemplo, alterar funciones que permitan obtener información de la hora, reloj y el paso del tiempo (acelerarlas para los procesos y que sospechen de un entorno en sandbox).
- Simular claves de registro
- Simular servicios

Técnicas identificadas

Claves de registro “cantosas”

- Otro punto de control de algunos bichos pasa por identificar claves de registro concretas:

```
SYSTEM\\ControlSet001\\Services\\VBoxGuest,      SYSTEM\\ControlSet001\\Services\\VBoxMouse,
SYSTEM\\ControlSet001\\Services\\VBoxService,    SYSTEM\\ControlSet001\\Services\\VBoxSF,
SYSTEM\\ControlSet001\\Services\\VBoxVideo
```

```
HKEY_LOCAL_MACHINE, "HARDWARE\\DEVICEMAP\\Scsi\\Scsi Port 0\\Scsi Bus 0\\Target Id 0\\Logical
Unit Id 0", "Identifier", "QEMU"
```

```
HKEY_LOCAL_MACHINE, "HARDWARE\\Description\\System", "SystemBiosVersion", "QEMU"
```

```
HKEY_LOCAL_MACHINE, "HARDWARE\\DEVICEMAP\\Scsi\\Scsi Port 0\\Scsi Bus 0\\Target Id 0\\Logical
Unit Id 0", "Identifier", "VBOX") VBOX HARDDISK ----- Type DiskPeripheral
```

```
HKEY_LOCAL_MACHINE, "HARDWARE\\Description\\System", "SystemBiosVersion", "VBOX" VBOX - 1
```

```
HKEY_LOCAL_MACHINE, "SOFTWARE\\Oracle\\VirtualBox Guest Additions"
```

```
HKEY_LOCAL_MACHINE, "HARDWARE\\Description\\System", "VideoBiosVersion", "VIRTUALBOX"
```

Técnicas identificadas

Claves de registro “cantosas”

- SystemBiosDate siempre es interesante y algunas de VMWARE:

```
HKEY_LOCAL_MACHINE, "HARDWARE\\DESCRIPTION\\System",  
"SystemBiosDate", "06/23/99"
```

```
HKEY_LOCAL_MACHINE, "HARDWARE\\DEVICEMAP\\Scsi\\Scsi Port 0\\Scsi  
Bus 0\\Target Id 0\\Logical Unit Id 0", "Identifier", "VMWARE"  
HKEY_LOCAL_MACHINE, "HARDWARE\\DEVICEMAP\\Scsi\\Scsi Port 1\\Scsi  
Bus 0\\Target Id 0\\Logical Unit Id 0", "Identifier", "VMWARE"  
HKEY_LOCAL_MACHINE, "HARDWARE\\DEVICEMAP\\Scsi\\Scsi Port 2\\Scsi  
Bus 0\\Target Id 0\\Logical Unit Id 0", "Identifier", "VMWARE"
```

Técnicas identificadas

Direcciones MAC de productos de virtualización...

- Identificar el fabricante de la tarjeta de red es otro camino para levantar la liebre sobre un entorno virtualizado:

08:00:27 ... (vbox)

00:05:69 ... (vmware)

00:0C:29 ...

00:1C:14 ...

00:50:56 ...

Técnicas identificadas

PAFISH

- Programa desarrollado por **Alberto Ortega** que detecta estas técnicas

```
C:\Users\Alberto\Downloads\pafish.exe
* Pafish <Paranoid fish> *

Some anti(debugger/VM/sandbox) tricks
used by malware for the general public.

[*] Windows version: 6.1 build 7601
[*] CPU vendor: GenuineIntel

[-] Debuggers detection
[*] Using IsDebuggerPresent() ... OK

[-] CPU information based detections
[*] Checking the difference between CPU timestamp counters <rdtsc> ... OK
[*] Checking the difference between CPU timestamp counters <rdtsc> forcing UM ex
it ... OK
[*] Checking hypervisor bit in cpuid feature bits ... OK
[*] Checking cpuid vendor for known VM vendors ... OK

[-] Generic sandbox detection
[*] Using mouse activity ... OK
[*] Checking username ... OK
[*] Checking file path ... OK
[*] Checking common sample names in drives root ... OK
[*] Checking if disk size <= 60GB via DeviceIoControl() ... OK
[*] Checking if disk size <= 60GB via GetDiskFreeSpaceExA() ... OK
[*] Checking if Sleep() is patched using GetTickCount() ... OK
[*] Checking if NumberOfProcessors is < 2 via raw access ... OK
[*] Checking if NumberOfProcessors is < 2 via GetSystemInfo() ... OK

[-] Hooks detection
[*] Checking function DeleteFileW method 1 ... OK

[-] Sandboxie detection
[*] Using GetModuleHandle(<sbiedll.dll>) ... OK

[-] Wine detection
[*] Using GetProcAddress(wine_get_unix_file_name) from kernel32.dll ... OK

[-] VirtualBox detection
[*] Scsi port->bus->target id->logical unit id-> 0 identifier ... OK
[*] Reg key (HKLM\HARDWARE\Description\System "SystemBiosVersion") ... OK
[*] Reg key (HKLM\SOFTWARE\Oracle\VirtualBox Guest Additions) ... OK
[*] Reg key (HKLM\HARDWARE\Description\System "VideoBiosVersion") ... OK
[*] Reg key (HKLM\HARDWARE\ACPI\DSDT\UBOX) ... OK
[*] Reg key (HKLM\HARDWARE\ACPI\FADT\UBOX) ... OK
[*] Reg key (HKLM\HARDWARE\ACPI\RSDT\UBOX) ... OK
[*] Reg key (HKLM\SYSTEM\ControlSet001\Services\UBox*) ... OK
[*] Reg key (HKLM\HARDWARE\DESCRIPTION\System "SystemBiosDate") ... OK
[*] Driver files in C:\WINDOWS\system32\drivers\UBox* ... OK
[*] Additional system files ... OK
[*] Looking for a MAC address starting with 08:00:27 ... OK
[*] Looking for pseudo devices ... OK
[*] Looking for UBoxTray windows ... OK
[*] Looking for UBox network share ... OK
[*] Looking for UBox processes (vboxservice.exe, vboxtray.exe) ... OK

[-] VMware detection
[*] Scsi port 0,1,2 ->bus->target id->logical unit id-> 0 identifier ... OK
[*] Reg key (HKLM\SOFTWARE\VMware, Inc.\VMware Tools) ... OK
[*] Looking for C:\WINDOWS\system32\drivers\vmtoolsd.sys ... OK
[*] Looking for C:\WINDOWS\system32\drivers\vmhgfs.sys ... OK

[-] Qemu detection
[*] Scsi port->bus->target id->logical unit id-> 0 identifier ... OK
[*] Reg key (HKLM\HARDWARE\Description\System "SystemBiosVersion") ... OK

[-] Feel free to RE me, check log file for more information.
```

```
D:\ut06269\Downloads\pafish-master\pafish-master\pafish.exe

[*] Checking function DeleteFileW method 1 ... OK
[*] Checking function ShellExecuteExW method 1 ... OK
[*] Checking function CreateProcessA method 1 ... OK

[-] Sandboxie detection
[*] Using GetModuleHandle(<sbiedll.dll>) ... traced!

[-] Wine detection
[*] Using GetProcAddress(wine_get_unix_file_name) from kernel32.dll ... OK
[*] Reg key (HKCU\SOFTWARE\Wine) ... traced!

[-] VirtualBox detection
[*] Scsi port->bus->target id->logical unit id-> 0 identifier ... traced!
[*] Reg key (HKLM\HARDWARE\Description\System "SystemBiosVersion") ... traced!
[*] Reg key (HKLM\SOFTWARE\Oracle\VirtualBox Guest Additions) ... traced!
[*] Reg key (HKLM\HARDWARE\Description\System "VideoBiosVersion") ... traced!
[*] Reg key (HKLM\HARDWARE\ACPI\DSDT\UBOX) ... traced!
[*] Reg key (HKLM\HARDWARE\ACPI\FADT\UBOX) ... traced!
[*] Reg key (HKLM\HARDWARE\ACPI\RSDT\UBOX) ... traced!
[*] Reg key (HKLM\SYSTEM\ControlSet001\Services\UBox*) ... traced!
[*] Reg key (HKLM\HARDWARE\DESCRIPTION\System "SystemBiosDate") ... traced!
[*] Driver files in C:\WINDOWS\system32\drivers\UBox* ... OK
[*] Additional system files ... OK
[*] Looking for a MAC address starting with 08:00:27 ... traced!
[*] Looking for pseudo devices ... OK
[*] Looking for UBoxTray windows ... traced!
[*] Looking for UBox network share ... OK
[*] Looking for UBox processes (vboxservice.exe, vboxtray.exe) ... traced!
[*] Looking for UBox devices using WMI ... OK

[-] VMware detection
[*] Scsi port 0,1,2 ->bus->target id->logical unit id-> 0 identifier ... traced!
[*] Reg key (HKLM\SOFTWARE\VMware, Inc.\VMware Tools) ... traced!
[*] Looking for C:\WINDOWS\system32\drivers\vmtoolsd.sys ... OK
[*] Looking for C:\WINDOWS\system32\drivers\vmhgfs.sys ... OK
[*] Looking for a MAC address starting with 00:05:69, 00:0C:29, 00:1C:14 or 00:5
0:56 ... OK
[*] Looking for network adapter name ... OK
[*] Looking for pseudo devices ... traced!
[*] Looking for VMware serial number ... OK

[-] Qemu detection
[*] Scsi port->bus->target id->logical unit id-> 0 identifier ... traced!
[*] Reg key (HKLM\HARDWARE\Description\System "SystemBiosVersion") ... traced!
[*] cpuid CPU brand string 'QEMU Virtual CPU' ... OK

[-] Bochs detection
[*] Reg key (HKLM\HARDWARE\Description\System "SystemBiosVersion") ... traced!
[*] cpuid AMD wrong value for processor name ... OK
[*] cpuid Intel wrong value for processor name ... OK

[-] Cuckoo detection
[*] Looking in the TLS for the hooks information structure ... OK

[-] Feel free to RE me, check log file for more information.
```



¡ESTÁ VIVO!
¡VIVOOOO!

Índice

1. Quienes somos
2. ¿Por qué esta charla?
3. Trabajo anterior y otros proyectos
4. Técnicas identificadas
5. Demostración
6. Conclusiones

Conclusiones

- Realmente estamos convencidos de que la idea es buena (tu idea, Jordi ☹).
- El driver funcionando, no tiene impacto alguno en el funcionamiento del sistema: no consume recursos, no hace nada “extraño”...
- Obviamente hay riesgos (siempre los hay):
 - **¿Qué ocurre si el bicho en vez de desactivarse decide asesinar nuestro equipo? ¿Detectado en sandbox a la entrada?**
 - **Hay que probarlo con TODAS las aplicaciones que vayas a usar. Nunca se sabe si has roto algo y todavía no te has dado cuenta...**

Conclusiones

- Esta idea la vemos integrada en la herramienta de **antiransom** de Yago Jesús... o dentro de los **antivirus**.
- No entendemos la razón de que algo tan evidente no esté implementado en ninguno de los motores más conocidos. Los modelos orientados a firmas son **poco efectivos** (todos hemos vivido el resurgir de un gusano de hace cinco años y que nuestro antivirus no lo detecte porque no es el *hype* actual...).
- PERO sobre todo: hay muchas estrategias disponibles para ganar en seguridad **GRATIS**. No viváis en el mundo de las máquinas, las suscripciones, los mantenimientos y ese largo etcétera...

Agradecimientos

- Jordi Vazquez (@jordisk)
- Yago Jesus (@YJesus)
- Alberto Ortega (@A0rtega)
- Marc Ribero (@Seifreed)







E-Mails

- > info@ccn-cert.cni.es
- > ccn@cni.es
- > sondas@ccn-cert.cni.es
- > redsara@ccn-cert.cni.es
- > organismo.certificacion@cni.es



Websites

- > www.ccn.cni.es
- > www.ccn-cert.cni.es
- > www.oc.ccn.cni.es



Síguenos en Linked in

