



Malware Kung Fu++: *solo no puedes, con amigos sí*

DIEZ AÑOS FORTALECIENDO LA
CIBERSEGURIDAD NACIONAL



- Román Ramírez
- RootedCON
- rramirez@rootedcon.com



@patowc



- Pablo San Emeterio
- Telefónica
- pablo.sanemeteriolopez@telefonica.com



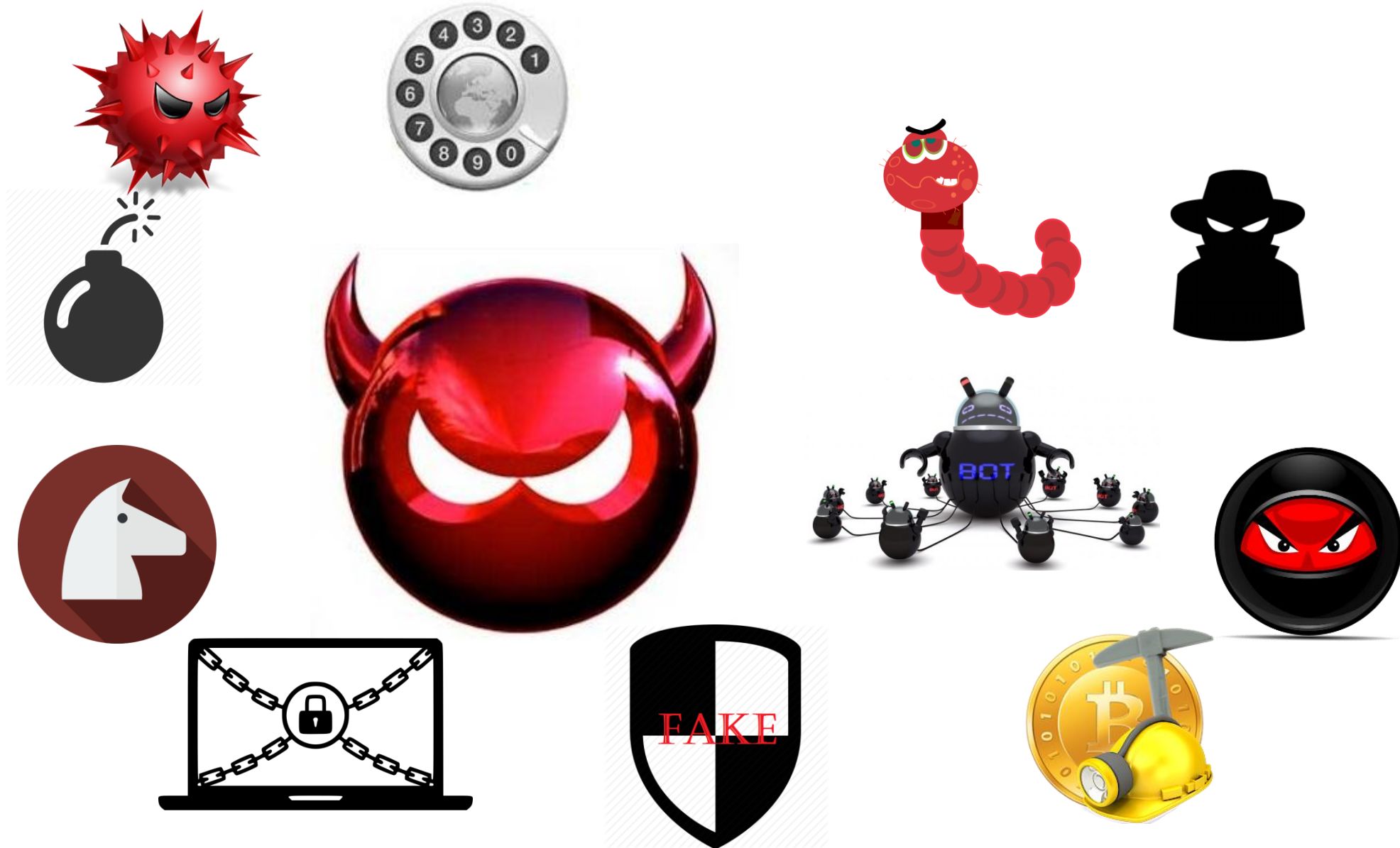
@psaneme

Índice

- 1. Qué os contamos el año pasado**
- 2. Seguimos convencidos de la idea**
- 3. Qué hemos añadido este año**
- 4. Colaboración entre elementos de seguridad SiBUS**
- 5. Conclusiones**

Qué os contamos el año pasado...









Qué os contamos el años pasado



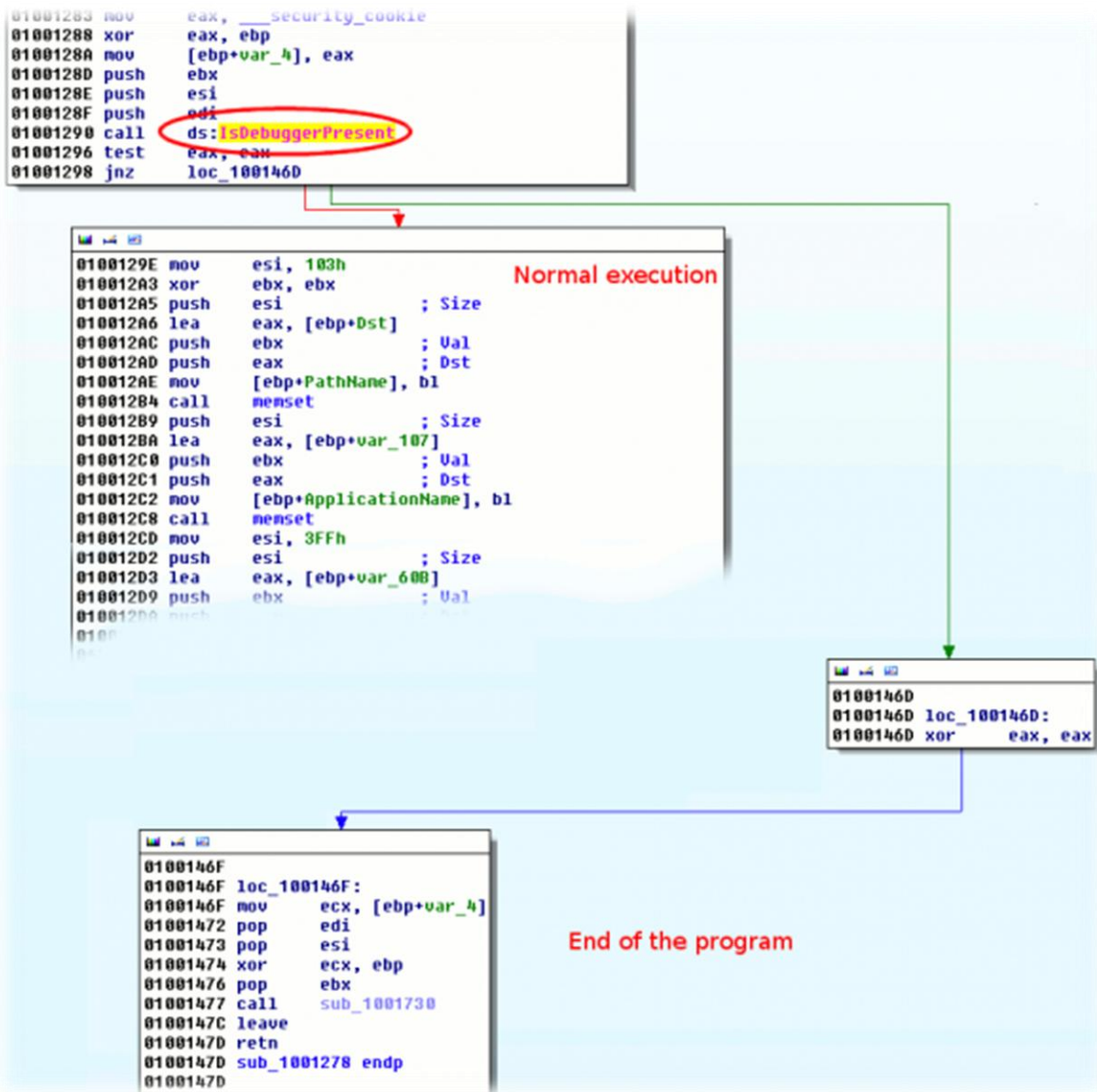
cuckoo 



vmware®



Qué os contamos el años pasado



Qué os contamos el años pasado



Qué os contamos el años pasado

- **Objetivo:** hacer creer a los “bichos” que están dentro de un entorno de análisis de malware (virtualizado, debugging o de sandbox)
- **Resultado:** hemos observado que en diversos casos se desactivan sin mayores complicaciones.

Este año.....

- Si combinamos esta idea con otras tecnologías (p.ej. **sandboxing**), el ratio de protección puede ser alto.

Qué os contamos el años pasado

Resumen (rápido) de todas las técnicas

- **Detección de depuración:** isDebuggerPresent, CheckRemoteDebuggerPresent, OutputDebugString
- **Detección de entorno Sandbox/análisis de malware:**
 - nombres de **procesos** evidentes (OllyDbg.exe, windbg.exe, immunity, procexp, wireshark,...)
 - Nombres de **ventanas** (OllyDbg, Immunity Debugger,wireshark...)

Qué os contamos el años pasado

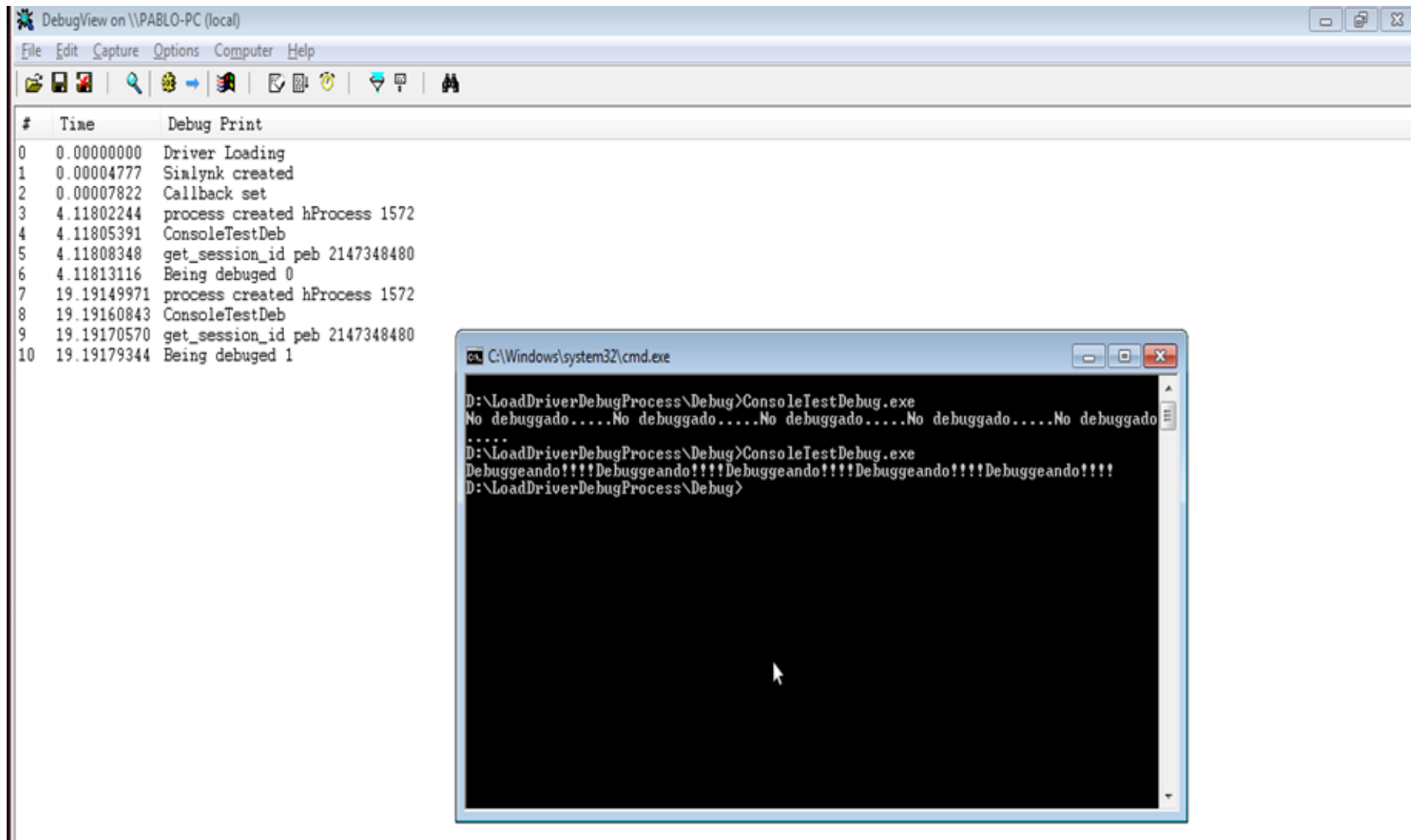
Resumen (rápido) de todas las técnicas

- Detección de entorno virtualizado:
 - **GlobalMemoryStatusEx** memoria < 1G, **GetSystemInfo** numero de procesadores < 2, **GetDiskFreeSpaceExA** < 60G
 - **Direcciones MAC** : '08:00:27, 00:05:69, 00:0c:29, 00:1c:14, 00:50:56
 - **Nombres de procesos**: vboxservice.exe, vboxtray.exe, vmware-authd.exe, vmnat.exe, vmwaredhcp.exe, vmware-usbarbitrator64.exe, Agent.py

Qué os contamos el años pasado

Resumen (rápido) de todas las técnicas

- **Detección de entorno virtualizado:**
 - **Ficheros conocidos en VM** C:\WINDOWS\system32\drivers\VBBoxMouse.sys, C:\WINDOWS\system32\drivers\VBBoxGuest.sys, C:\WINDOWS\system32\drivers\vmmouse.sys, etc
 - **Claves de registro de windows:** SYSTEM\\ControlSet001\\Services\\ VBBoxGuest o VBBoxMouse, HKEY_LOCAL_MACHINE, "HARDWARE\\Description\\System", "SystemBiosVersion", "VBOX" VBOX – 1, "HARDWARE\\Description\\System", "VideoBiosVersion", "VIRTUALBOX", etc
 - **Nombres de equipos:** sandbox, trendmicro, xp-sp3-template, maltest, virus, malware, sandbox ;)
 - **Nombre de unidades de disco:** sandbox.
 - **Instrucciones x86:** CPUID, RedPill (SIDT) o NoPill (SGDT, SLDT)



The screenshot shows a Windows XP desktop with two open windows. The background window is 'DebugView on \\PABLO-PC (local)', which displays a log of system events. The foreground window is a command prompt titled 'C:\Windows\system32\cmd.exe', showing the execution of 'ConsoleTestDebug.exe' and its output.

DebugView Log:

#	Time	Debug Print
0	0.00000000	Driver Loading
1	0.00004777	Simlynk created
2	0.00007822	Callback set
3	4.11802244	process created hProcess 1572
4	4.11805391	ConsoleTestDeb
5	4.11808348	get_session_id peb 2147348480
6	4.11813116	Being debugged 0
7	19.19149971	process created hProcess 1572
8	19.19160843	ConsoleTestDeb
9	19.19170570	get_session_id peb 2147348480
10	19.19179344	Being debugged 1

Command Prompt Output:

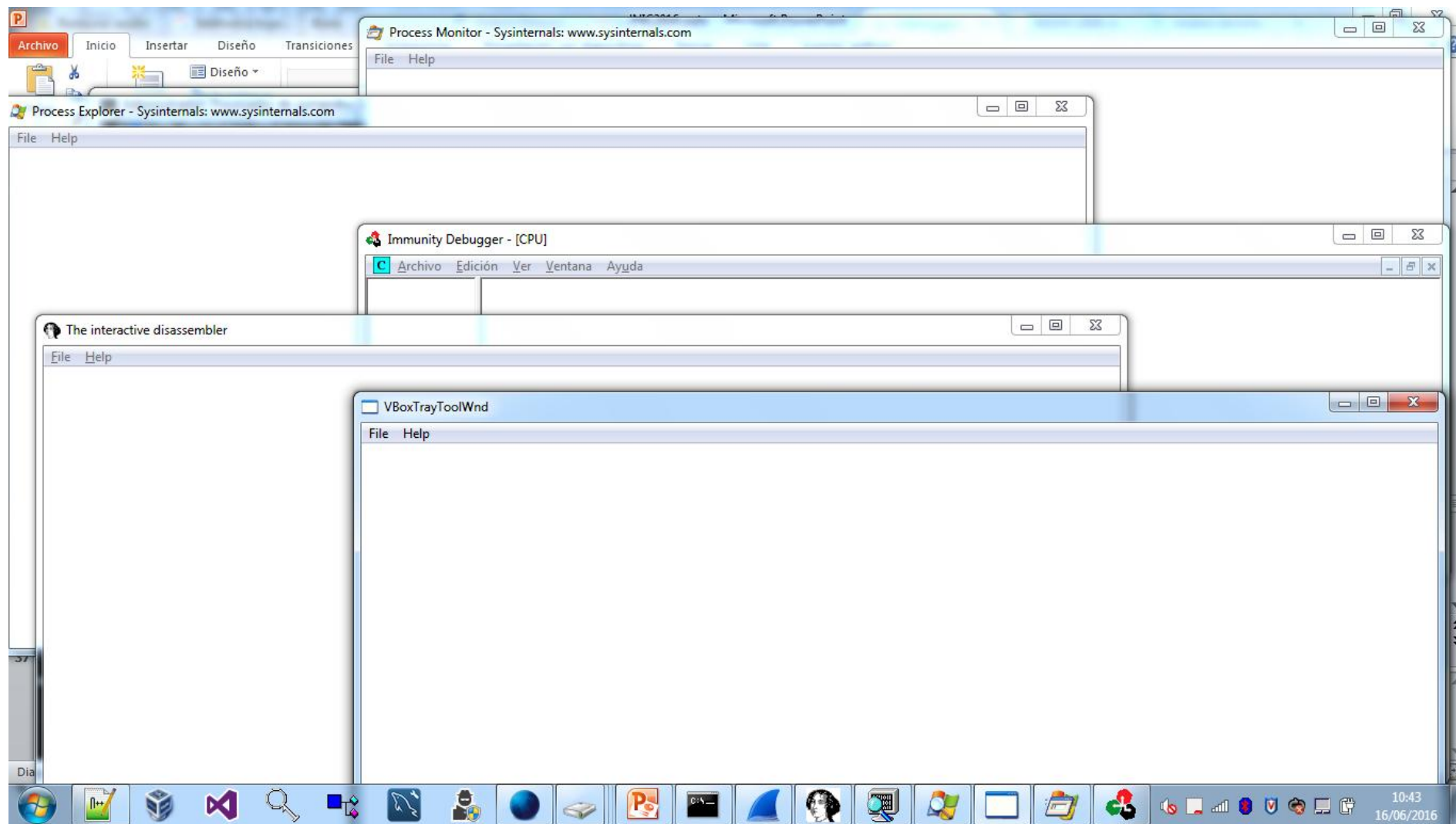
```

D:\LoadDriverDebugProcess\Debug>ConsoleTestDebug.exe
No debuggado.....No debuggado.....No debuggado.....No debuggado.....No debuggado
.....
D:\LoadDriverDebugProcess\Debug>ConsoleTestDebug.exe
Debuggeando!!!!Debuggeando!!!!Debuggeando!!!!Debuggeando!!!!Debuggeando!!!!
D:\LoadDriverDebugProcess\Debug>
  
```



Apps de “pega”

- Aplicaciones simulan ser las apps de un analista de malware.
- Tienen el mismo proccessName
- Tienen el mismo WndClass y WndText
- Si se analizan firmas del sw se ve que es diferente



Simulación de entorno de análisis

- Simular ficheros.
- Simular claves de registro
- Simular interfaces de red
- Simular procesos
- Simular espacio en Disco
- Simular RAM disponible
- Simular N° de CPUsg

```
C:\WINDOWS\system32\cmd.exe
* Pafish (Paranoid fish) *

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

[*] Windows version: 6.1 build 7601
[*] CPU: GenuineIntel
    CPU brand: Intel(R) Core(TM) i5-3320M CPU @ 2.60GHz

[-] 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 exit ... OK
[*] Checking hypervisor bit in cpuid feature bits ... OK
[*] Checking cpuid hypervisor 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
[*] Checking if physical memory is < 1Gb ... OK
[*] Checking operating system uptime using GetTickCount() ... OK
[*] Checking if operating system IsNativeUhdBoot() ... OK

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

[-] Sandboxie detection
[*] Using GetModuleHandle(sbidll.dll) ... OK

[-] Wine detection
[*] Using GetProcAddress(wine_get_unix_file_name) from kernel32.dll ... OK
[*] Reg key (HKCU\SOFTWARE\Wine) ... 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\RSMT\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 VBoxTray windows ... OK
[*] Looking for VBox network share ... OK
[*] Looking for VBox processes (vboxservice.exe, vboxtray.exe) ... OK
[*] Looking for VBox devices using WMI ... 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\vmmouse.sys ... OK
[*] Looking for C:\WINDOWS\system32\drivers\vmhgfs.sys ... OK
[*] Looking for a MAC address starting with 08:05:69, 00:0C:29, 00:1C:14 or 00:50:56 ... OK
[*] Looking for network adapter name ... OK
[*] Looking for pseudo devices ... OK
[*] Looking for VMware serial number ... OK

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

[-] Bochs detection
[*] Reg key (HKLM\HARDWARE\Description\System "SystemBiosVersion") ... OK
[*] 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.
```

```
C:\WINDOWS\system32\cmd.exe
* Pafish (Paranoid fish) *

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

[*] Windows version: 6.1 build 7601
[*] CPU: GenuineIntel
    Hypervisor: UBoxUBoxUBox
    CPU brand: Intel(R) Core(TM) i5-3320M CPU @ 2.60GHz

[-] Debuggers detection
[*] Using IsDebuggerPresent() ... traced!

[-] CPU information based detections
[*] Checking the difference between CPU timestamp counters (rdtsc) ... traced!
[*] Checking the difference between CPU timestamp counters (rdtsc) forcing UM exit ... traced!
[*] Checking hypervisor bit in cpuid feature bits ... traced!
[*] Checking cpuid hypervisor vendor for known VM vendors ... traced!

[-] Generic sandbox detection
[*] Using mouse activity ... traced!
[*] Checking username ... traced!
[*] Checking file path ... traced!
[*] Checking common sample names in drives root ... traced!
[*] Checking if disk size <= 60GB via DeviceIoControl() ... traced!
[*] Checking if disk size <= 60GB via GetDiskFreeSpaceExA() ... traced!
[*] Checking if Sleep() is patched using GetTickCount() ... traced!
[*] Checking if NumberOfProcessors is < 2 via raw access ... traced!
[*] Checking if NumberOfProcessors is < 2 via GetSystemInfo() ... traced!
[*] Checking if physical memory is < 1Gb ... traced!
[*] Checking operating system uptime using GetTickCount() ... traced!
[*] Checking if operating system IsNativeUhdBoot() ... OK

[-] Hooks detection
[*] Checking function ShellExecuteExW method 1 ... traced!
[*] Checking function CreateProcessA method 1 ... traced!

[-] Sandboxie detection
[*] Using GetModuleHandle(sbidll.dll) ... traced!

[-] Wine detection
[*] Using GetProcAddress(wine_get_unix_file_name) from kernel32.dll ... traced!
[*] 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\RSMT\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* ... traced!
[*] Additional system files ... traced!
[*] Looking for a MAC address starting with 08:00:27 ... traced!
[*] Looking for pseudo devices ... traced!
[*] Looking for VBoxTray windows ... traced!
[*] Looking for VBox network share ... traced!
[*] Looking for VBox processes (vboxservice.exe, vboxtray.exe) ... traced!
[*] Looking for VBox 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\vmmouse.sys ... traced!
[*] Looking for C:\WINDOWS\system32\drivers\vmhgfs.sys ... traced!
[*] Looking for a MAC address starting with 08:05:69, 00:0C:29, 00:1C:14 or 00:50:56 ... OK
[*] Looking for network adapter name ... traced!
[*] 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.
```

Seguimos convencidos de la idea

Qué os contamos el años pasado

Trabajos anteriores

- Xu Chen, Jon Andersen, Z.Morley Mao, Michael Bailey, Jose Nazario, “Towards an Understanding of Anti-virtualization and Anti-debugging Behaviour in Modern Malware”, 2008
- Jordi Vazquez
<http://kasperskycontenthub.com/kasperskyacademy/files/2013/12/aira.pdf>
- Rodriguez, R.J., Rodriguez-Gastón, I. & Alonso, J., “Towards the Detection of Isolation-Aware Malware”, 2016

<https://webdiis.unizar.es/~ricardo/files/papers/RRA-LATAM-16.pdf>

Homeopatía...

Algunos comentarios que hemos recibido

- Nos hizo gracia que hablando en un entorno de protección antimalware alguien dijera que le parecía **“homeopatía”**.
- Seguimos pensando que se pueden aplicar muchos mecanismos de defensa en esta línea.
- Obviamente, infalible no hay absolutamente nada. Y menos una única tecnología de protección

Soluciones similares...



Hack Players

Participa Retos Blogroll en español Blogroll in English Herramientas Afiliados

AntiFooling, simula que tu máquina es virtual para que el malware no se ejecute

PUBLICADO POR VICENTE MOTOS ON DOMINGO, 4 DE DICIEMBRE DE 2016 ETIQUETAS: [ANTIVIRUS](#) , [FORTIFICACIÓN](#) , [HERRAMIENTAS](#) , [MALWARE](#) , [TÉCNICAS](#)

Hoy nos llegaba una petición de un amigo en nuestro [grupo de Telegram](#) para que echáramos un vistazo a una herramienta publicada por Scorpio en [Indetectables](#), concretamente AntiFooling.

AntiFooling es una sencilla pero útil herramienta hecha con AutoIt para **simular que un sistema Windows es una máquina virtual**, pero ¿para qué?. Pues hoy en día la mayoría del malware es analizado dinámicamente en máquinas virtuales y sandboxes, por lo que **el software malicioso suele implementar medidas que lo detectan y detienen su ejecución** en tal caso. Por lo tanto, si "engañamos" al malware para que crea que está en un entorno de análisis de este tipo podremos conseguir que tampoco se ejecute en nuestra máquina, obteniendo así un eficaz "antivirus".



AntiFooling v1.0.0

AntiFooling is a tool designed for preventing the malware. This is possible through the exploitation of the most common exceptions Anti-Emulation and Anti-Debugging techniques used by the Malware, that stop the execution if it detects

Virtual Machines

☒ VirtualBox ☐ VMWare

☐ Run on System Startup

Enable Disable

Soluciones similares...



Hack Players

Participa Retos Blogroll en español Blogroll in English Herramientas Afiliados

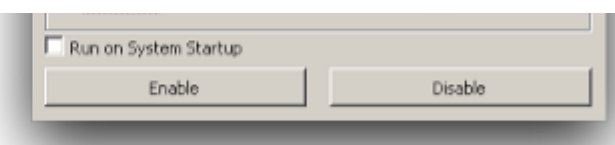
AntiFooling, malware no [ramsal](#) 5 de diciembre de 2016, 19:43

PUBLICADO POR VICENTE / MALWARE, TÉCNICAS

Hoy nos llegaba un amigo en nuestro para que echáramos una herramienta publicada en Indetectables, con AntiFooling.

AntiFooling es una sencilla pero útil herramienta hecha con AutoIt para **simular que un sistema Windows es una máquina virtual**, pero ¿para qué?. Pues hoy en día la mayoría del malware es analizado dinámicamente en máquinas virtuales y sandboxes, por lo que **el software malicioso suele implementar medidas que lo detectan y detienen su ejecución** en tal caso. Por lo tanto, si "engañamos" al malware para que crea que está en un entorno de análisis de este tipo podremos conseguir que tampoco se ejecute en nuestra máquina, obteniendo así un eficaz "antivirus".

[Responder](#)



Soluciones similares...

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 lab

POLITECNICO DI MILANO

Introduction

ramienta que hacía lo mismo.
virtual y tras la instalación del
e todo indicaba que era virtual,

Qué hemos añadido este año

Driver

Monitoriza creación de procesos.

(PsSetCreateProcessNotifyRoutine)

A cada proceso nuevo :

- Le pone el flag de debug a True
- Le inyecta una dll con el nombre sandboxie.dll
- La dll hooke funciones para simular claves de registro, ficheros, etc
- **Suspende el proceso**
- **Mandar info al bus**

Instrumentación con PIN

- Puedes controlar el comportamiento del ejecutable de muchas maneras.
- Tiene bastantes contras en el mundo real.

```
* Pafish <Paranoid fish> *
```

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

```
[*] Windows version: 6.1 build 7601  
[*] CPU: GenuineIntel  
    CPU brand:      Intel(R) Core(TM) i5-3320M CPU @ 2.60GHz
```

```
[*] 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 VM exit ... OK
```

```
[*] Checking hypervisor bit in cpuid feature bits ... OK
```

```
[*] Checking cpuid hypervisor vendor for known VM vendors ... OK
```

```
[*] Generic sandbox detection
```

```
[*] Using mouse activity ... OK
```

```
* Pafish <Paranoid fish> *
```

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

```
[*] Windows version: 6.1 build 7601
```

```
[*] CPU: GenuineIntel  
    Hypervisor: VBoxVBoxVBox  
    CPU brand:      Intel(R) Core(TM) i5-3320M CPU @ 2.60GHz
```

```
[*] Debuggers detection
```

```
[*] Using IsDebuggerPresent() ... OK
```

```
[*] CPU information based detections
```

```
[*] Checking the difference between CPU timestamp counters (rdtsc) ... traced!
```

```
[*] Checking the difference between CPU timestamp counters (rdtsc) forcing VM exit ... traced!
```

```
[*] Checking hypervisor bit in cpuid feature bits ... traced!
```

```
[*] Checking cpuid hypervisor vendor for known VM vendors ... traced!
```

Administrator: Procesador de comandos de Windows - pin.exe - - pin.vbox.dll -- d:\pafish.exe

* Pafish <Paranoid fish> *

Some anti<debugger/UM/sandbox> tricks
used by malware for the general public.

[*] Windows version: 6.1 build 7601

[*] CPU: GenuineIntel
Hypervisor: VBoxVBoxVBox
CPU brand: Intel(R) Core(TM) i5-3320M CPU @ 2.60GHz

[+] Debuggers detection

[*] Using IsDebuggerPresent() ... **traced!**

[+] CPU information based detections

[*] Checking the difference between CPU timestamp counters <rdtsc> ... **traced!**

[*] Checking the difference between CPU timestamp counters <rdtsc> forcing VM exit ... **traced!**

[*] Checking hypervisor bit in cpuid feature bits ... **traced!**

[*] Checking cpuid hypervisor vendor for known VM vendors ... **traced!**

[+] Generic sandbox detection

[*] Using mouse activity ... **traced!**

[*] Checking username ... **traced!**

[*] Checking file path ... **traced!**

[*] Checking common sample names in drives root ... **traced!**

[*] Checking if disk size <= 60GB via DeviceIoControl() ... **traced!**

[*] Checking if disk size <= 60GB via GetDiskFreeSpaceEx() ... **traced!**

[*] Checking if Sleep() is patched using GetTickCount() ... **traced!**

[*] Checking if NumberOfProcessors is < 2 via raw access ... **traced!**

[*] Checking if NumberOfProcessors is < 2 via GetSystemInfo() ... **traced!**

[*] Checking if physical memory is < 1Gb ... **traced!**

[*] Checking operating system uptime using GetTickCount() ... **traced!**

[*] Checking if operating system IsNativeUhdBoot() ... **OK**

[+] Hooks detection

[*] Checking function ShellExecuteExW method 1 ... **traced!**

[*] Checking function CreateProcessA method 1 ... **traced!**

[+] Sandboxie detection

[*] Using GetModuleHandle(sbie.dll) ... **traced!**

[+] Wine detection

[*] Using GetProcAddress(wine_get_unix_file_name) from kernel32.dll ... **traced!**

[*] 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* ... **traced!**

[*] Additional system files ... **traced!**

[*] Looking for a MAC address starting with 08:00:27 ... **traced!**

[*] Looking for pseudo devices ... **traced!**

[*] Looking for VBoxTray windows ... **traced!**

[*] Looking for UBox network share ... **traced!**

[*] 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\vmmouse.sys ... **traced!**

[*] Looking for C:\WINDOWS\system32\drivers\vmhgfs.sys ... **traced!**

[*] Looking for a MAC address starting with 00:05:69, 00:0C:29, 00:1C:14 or 00:50:56 ... **OK**

[*] Looking for network adapter name ... **traced!**

[*] 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**

e Virtualizacion

ologias de virtualizacion a la vez

cmd: Administrador: Procesador de comandos de Windows - pin.exe -t pin.vbox.dll -- d:\pafish.exe

* Pafish <Paranoid fish> *

Some anti<debugger/UM/sandbox> tricks
used by malware for the general public.

[*] Windows version: 6.1 build 7601

[*] CPU: GenuineIntel
Hypervisor: VBoxVBoxVBox
CPU brand: Intel(R) Core(TM)

[~] Debuggers detection

[*] Using IsDebuggerPresent() ... **traced!**

[~] CPU information based detections

[*] Checking the difference between CPU

[*] Checking the difference between CPU

[*] Checking hypervisor bit in cpuid f

[*] Checking cpuid hypervisor vendor f

[~] Generic sandbox detection

[*] Using mouse activity ... **traced!**

[*] Checking username ... **traced!**

[*] Checking file path ... **traced!**

[*] Checking common sample names in dr

[*] Checking if disk size <= 60GB via

[*] Checking if disk size <= 60GB via

[*] Checking if Sleep() is patched usin

[*] Checking if NumberOfProcessors is

[*] Checking if NumberOfProcessors is

[*] Checking if physical memory is < 1G

[*] Checking operating system uptime us

[*] Checking if operating system IsNat

[~] Hooks detection

[*] Checking function ShellExecuteExW

[*] Checking function CreateProcessA n

[~] Sandboxie detection

[*] Using GetModuleHandle(sbxiedll.dll)

[~] Wine detection

[*] Using GetProcAddress(wine_get_unix

[*] 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\RSMT\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* ... **traced!**

[*] Additional system files ... **traced!**

[*] Looking for a MAC address starting with 08:00:27 ... **traced!**

[*] Looking for pseudo devices ... **traced!**

[*] Looking for VBoxTray windows ... **traced!**

[*] Looking for VBox network share ... **traced!**

[*] Looking for VBox processes (vboxservice.exe, vboxtray.exe) ... **traced!**

[*] Looking for VBox 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\vmmouse.sys ... **traced!**

[*] Looking for C:\WINDOWS\system32\drivers\vmhgfs.sys ... **traced!**

[*] Looking for a MAC address starting with 00:05:69, 00:0C:29, 00:1C:14 or 00:50:56 ... **OK**

[*] Looking for network adapter name ... **traced!**

[*] 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**

D:\LoadDriverDebugProcess\Release>LoadDriver.exe -h

Kung-Fu Malware Help

Command [option] [--no-sbxie]

-h	--help	:	prints help
-a	--all	:	simulate all sandboxes
-b	--bochs	:	simulate bochs
-q	--qemu	:	simulate qemu
-v	--vmware	:	simulate vmware
-x	--vbox	:	simulate virtualbox
-w	--wine	:	simulate wine
	--no-sbie	:	no sandboxie check

* Pafish (Paranoid fish) *

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

[*] Windows version: 6.1 build 7601

[*] CPU: GenuineIntel
Hypervisor: UBoxUBoxUBox
CPU brand: Intel(R) Core(TM) i5-3320M CPU @ 2.60GHz

[!] Debuggers detection

[*] Using IsDebuggerPresent() ... **traced!**

[!] CPU information based detections

[*] Checking the difference between CPU timestamp counters (rdtsc) ... **traced!**

[*] Checking the difference between CPU timestamp counters (rdtsc) forcing VM exit ... **traced!**

[*] Checking hypervisor bit in cpuid feature bits ... **traced!**

[*] Checking cpuid hypervisor vendor for known VM vendors ... **traced!**

[!] Generic sandbox detection

[*] Using mouse activity ... **traced!**

[*] Checking username ... **traced!**

[*] Checking file path ... **traced!**

[*] Checking common sample names in drives root ... **traced!**

[*] Checking if disk size <= 60GB via DeviceIoControl() ... **traced!**

[*] Checking if disk size <= 60GB via GetDiskFreeSpaceEx() ... **traced!**

[*] Checking if Sleep() is patched using GetTickCount() ... **traced!**

[*] Checking if NumberOfProcessors is < 2 via raw access ... **traced!**

[*] Checking if NumberOfProcessors is < 2 via GetSystemInfo() ... **traced!**

[*] Checking if physical memory is < 1Gb ... **traced!**

[*] Checking operating system uptime using GetTickCount() ... **traced!**

[*] Checking if operating system IsNativeUhdBoot() ... **OK**

[!] Hooks detection

[*] Checking function ShellExecuteExW method 1 ... **traced!**

[*] Checking function CreateProcessA method 1 ... **traced!**

[!] Sandboxie detection

[*] Using GetModuleHandle(sbiid11.dll) ... **OK**

[!] Wine detection

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

[*] Reg key <HKCU\SOFTWARE\Wine> ... **OK**

[!] 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\RSMT\UBOX> ... **traced!**

[*] Reg key <HKLM\SYSTEM\ControlSet001\Services\UBox*> ... **traced!**

[*] Reg key <HKLM\SYSTEM\Description\System "SystemBiosDate"> ... **traced!**

[*] Driver files in C:\WINDOWS\system32\drivers\UBox* ... **traced!**

[*] Additional system files ... **traced!**

[*] Looking for a MAC address starting with 08:00:27 ... **traced!**

[*] Looking for pseudo devices ... **traced!**

[*] Looking for UBoxTray windows ... **traced!**

[*] Looking for UBox network share ... **traced!**

[*] 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 ... **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**

[*] Looking for a MAC address starting with 00:05:69, 00:0C:29, 00:1C:14 or 00:50:56 ... **OK**

[*] Looking for network adapter name ... **OK**

[*] Looking for pseudo devices ... **OK**

[*] Looking for VMware serial number ... **OK**

[!] Qemu detection

[*] Scsi port->bus->target id->logical unit id-> 0 identifier ... **OK**

[*] Reg key <HKLM\HARDWARE\Description\System "SystemBiosVersion"> ... **OK**

[*] cpuid CPU brand string 'QEMU Virtual CPU' ... **OK**

[!] Bochs detection

[*] Reg key <HKLM\HARDWARE\Description\System "SystemBiosVersion"> ... **OK**

[*] 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**

* Pafish (Paranoid fish) *

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

[*] Windows version: 6.1 build 7601

[*] CPU: GenuineIntel
Hypervisor: VMwareVMware
CPU brand: Intel(R) Core(TM) i5-3320M CPU @ 2.60GHz

[!] Debuggers detection

[*] Using IsDebuggerPresent() ... **traced!**

[!] CPU information based detections

[*] Checking the difference between CPU timestamp counters (rdtsc) ... **traced!**

[*] Checking the difference between CPU timestamp counters (rdtsc) forcing VM exit ... **traced!**

[*] Checking hypervisor bit in cpuid feature bits ... **traced!**

[*] Checking cpuid hypervisor vendor for known VM vendors ... **traced!**

[!] Generic sandbox detection

[*] Using mouse activity ... **traced!**

[*] Checking username ... **traced!**

[*] Checking file path ... **traced!**

[*] Checking common sample names in drives root ... **traced!**

[*] Checking if disk size <= 60GB via DeviceIoControl() ... **traced!**

[*] Checking if disk size <= 60GB via GetDiskFreeSpaceEx() ... **traced!**

[*] Checking if Sleep() is patched using GetTickCount() ... **traced!**

[*] Checking if NumberOfProcessors is < 2 via raw access ... **traced!**

[*] Checking if NumberOfProcessors is < 2 via GetSystemInfo() ... **traced!**

[*] Checking if physical memory is < 1Gb ... **traced!**

[*] Checking operating system uptime using GetTickCount() ... **traced!**

[*] Checking if operating system IsNativeUhdBoot() ... **OK**

[!] Hooks detection

[*] Checking function ShellExecuteExW method 1 ... **traced!**

[*] Checking function CreateProcessA method 1 ... **traced!**

[!] Sandboxie detection

[*] Using GetModuleHandle(sbiid11.dll) ... **OK**

[!] Wine detection

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

[*] Reg key <HKCU\SOFTWARE\Wine> ... **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\RSMT\UBOX> ... **OK**

[*] Reg key <HKLM\SYSTEM\ControlSet001\Services\UBox*> ... **OK**

[*] Reg key <HKLM\SYSTEM\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**

[*] 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 ... **traced!**

[*] Looking for C:\WINDOWS\system32\drivers\vmhgfs.sys ... **traced!**

[*] Looking for a MAC address starting with 00:05:69, 00:0C:29, 00:1C:14 or 00:50:56 ... **traced!**

[*] Looking for network adapter name ... **traced!**

[*] Looking for pseudo devices ... **traced!**

[*] Looking for VMware serial number ... **OK**

[!] Qemu detection

[*] Scsi port->bus->target id->logical unit id-> 0 identifier ... **OK**

[*] Reg key <HKLM\HARDWARE\Description\System "SystemBiosVersion"> ... **OK**

[*] cpuid CPU brand string 'QEMU Virtual CPU' ... **OK**

[!] Bochs detection

[*] Reg key <HKLM\HARDWARE\Description\System "SystemBiosVersion"> ... **OK**

[*] 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**



Eleven Paths

A Telefonica COMPANY

Colaboración entre elementos de seguridad

SiBUS

(“solo no puedes, con amigos sí”)

Equipo rojo

Equipo azul

27001:201 Cumplimiento

Auditoría interna

Auditoría externa

Tengo hijas

Tengo un

hámster

ENCS

CAB/Cambios

NIS

Parche rechazado por impacto

SCIIF

LECRim

DPA

FCSEs

Auditoría de licencias

Presupuestos

Más

Mi jefe

presupuestos

El jefe de mi

jefe

Revisar informes

Acudir a

Soportar eventos

Soportar eventos

Acudir a

Acudir a

Acudir a

Acudir a

Acudir a

Acudir a

Acudir a

Acudir a

Acudir a

Simplemente necesito encontrar **UNA** debilidad.

La gente que no
entiende lo que es un
certificado y afirma
que trabaja en
tecnología/seguridad

“Pues vaya mierda de hacker
que eres, Chema dice que...”

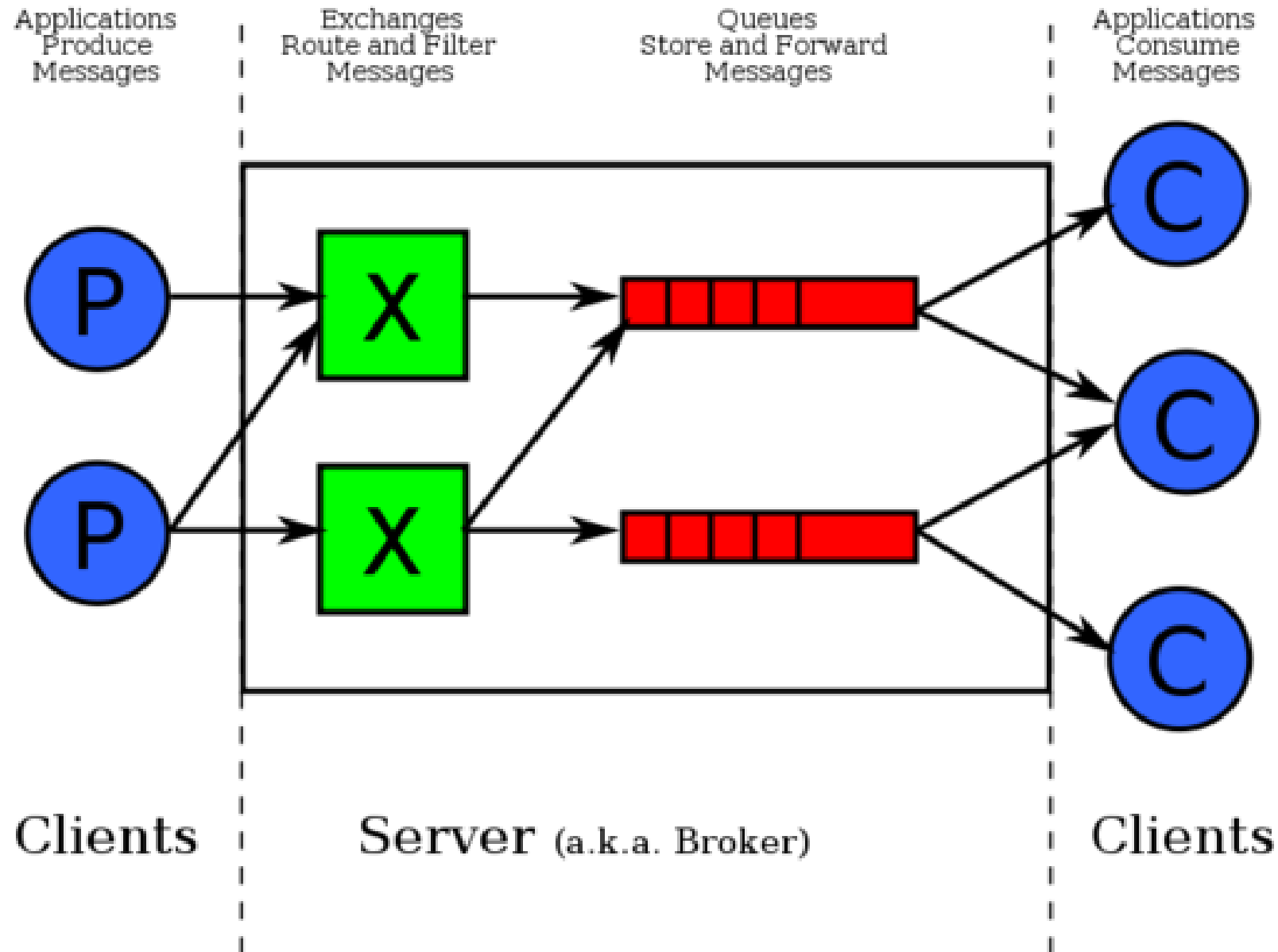
Gestión de
Identidad

BUS de mensajes de seguridad

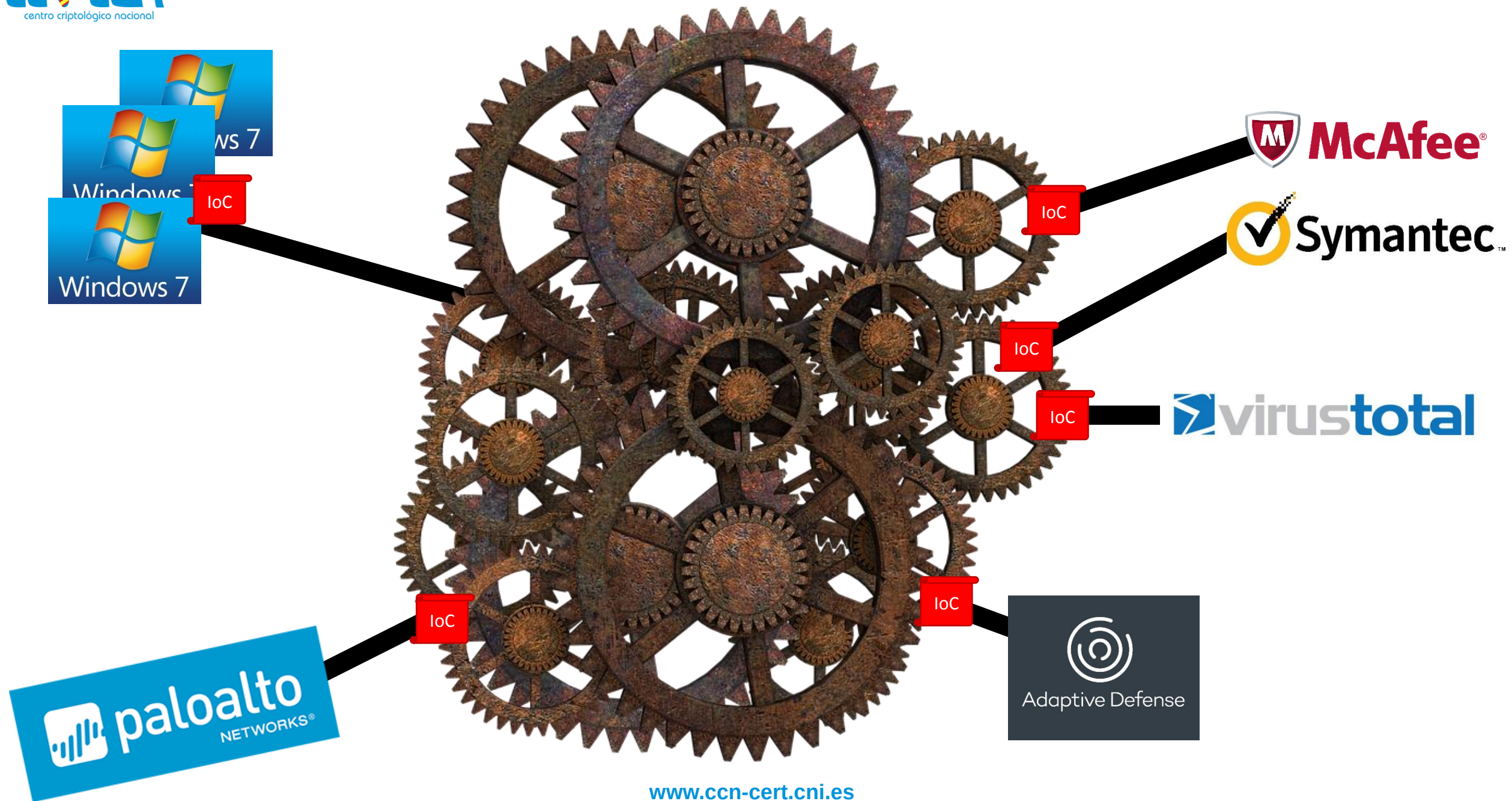
- BUS basado en mensajes en AMQP (demo con RabbitMQ).
- Si un elemento detecta un IoC del tipo que sea, debería compartirlo de inmediato con el BUS.
- Algunos fabricantes ya tratan de seguir este modelo: ¿problema? Debes adaptarte a **SU** estándar.
- Un estándar controlado por **UN** fabricante, no es un estándar, es una “**licencia**”.

BUS

- E
- S
- A
- l



)).



STIX, OpenIOC, MAEC, CYBIX...

- Da igual el formato o el estándar, lo relevante es la arquitectura.

OpenIOC

An Open Framework for Sharing Threat Intelligence

Sophisticated Threats Require Sophisticated Indicators

[Overview](#)
[Why OpenIOC?](#)
[Schema](#)
[Tools](#)
[OpenIOC FAQ](#)
[Resources](#)

Overview

In the current threat environment, rapid communication of pertinent threat information is the key to quickly detecting, responding and containing targeted attacks. OpenIOC is designed to fill a void that currently exists for organizations that want to share threat information both internally and externally in a machine-digestible format. OpenIOC is an extensible XML schema that enables you to describe the technical characteristics that identify a known threat, an attacker's methodology, or other evidence of compromise.

OpenIOC was originally designed to enable MANDIANT's products to codify intelligence in order to rapidly search for potential security breaches. Now, in response to requests from across the user community, MANDIANT has standardized and open sourced the OpenIOC schema and is releasing tools and utilities to allow communication of threat information at machine speed.

STIX

Getting Started - Documentation - Data Model - Community - About

Lookup STIX Data Types

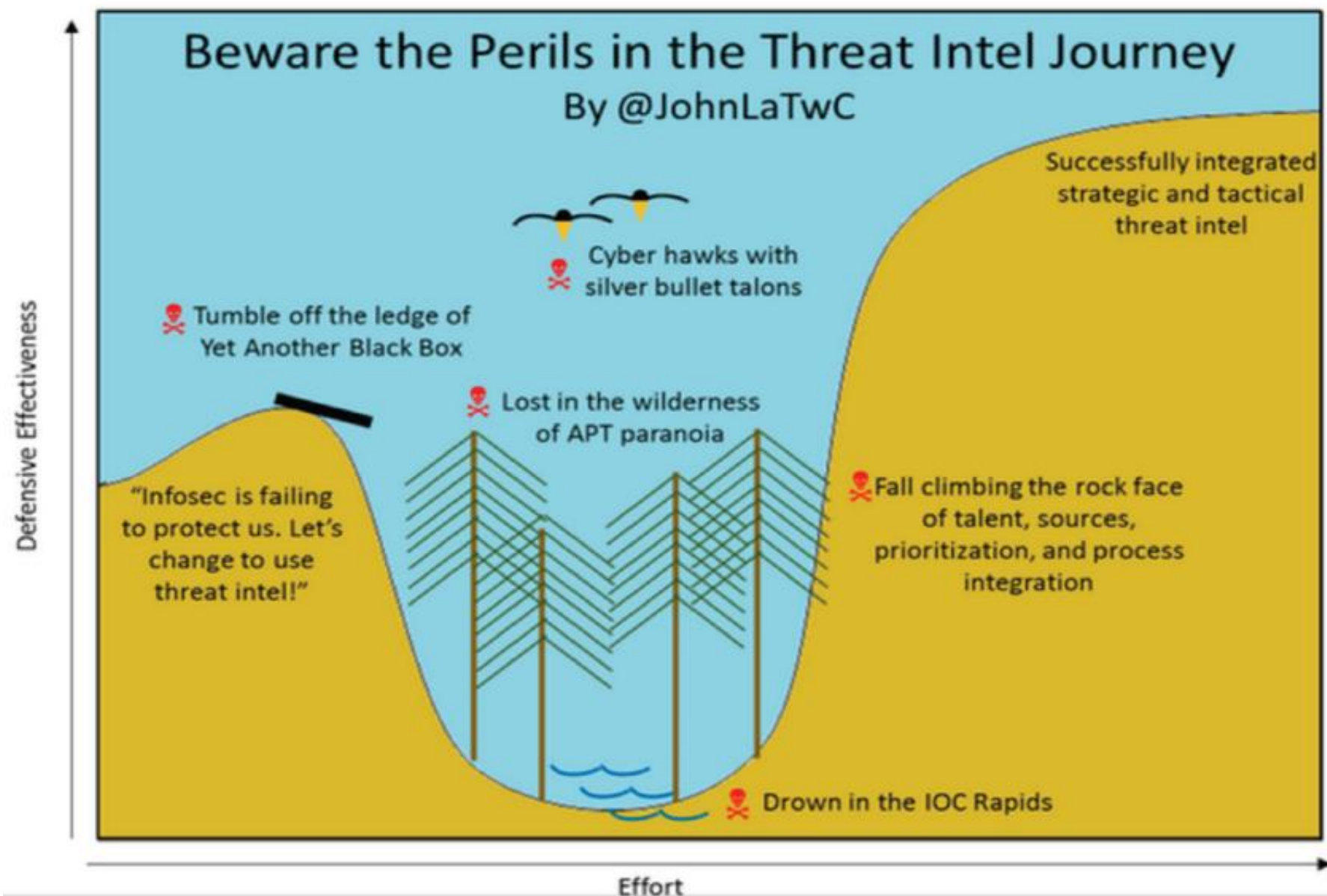
About STIX

Structured Threat Information Expression (STIX™) is a structured language for describing cyber threat information so it can be shared, stored, and analyzed in a consistent manner.

The STIX [whitepaper](#) describes the motivation and architecture behind STIX. At a high level the STIX language consists of 9 key constructs and the relationships between them:

- **Observables** describe what has been or might be seen in cyber
- **Indicators** describe patterns for what might be seen and what they mean if they are
- **Incidents** describe instances of specific adversary actions
- **Adversary Tactics, Techniques, and Procedures** describe attack patterns, malware, exploits, kill chains, tools, infrastructure, victim targeting, and other methods used by the adversary
- **Exploit Targets** describe vulnerabilities, weaknesses, or configurations that might be exploited
- **Courses of Action** describe response actions that may be taken in response to an attack or as a preventative measure
- **Campaigns** describe sets of incidents and/or TTPs with a shared intent
- **Threat Actors** describe identification and/or characterization of the adversary

• **Reports** collect related STIX content and give them shared context



Autor: John Lambert
www.ccn-cert.cni.es

Conclusiones

- La **unión hace la fuerza**, si unimos motores de AV, sandboxing, defensas perimetrales y técnicas de contrainteligencia es más difícil tener éxito en el ataque
- Necesidad de **estándar de comunicaciones** entre tecnologías
- Creemos que es una idea muy poderosa

The screenshot shows the GitHub repository page for 'psaneme / Kung-Fu-Malware'. The browser address bar shows the URL 'https://github.com/psaneme/Kung-Fu-Malware'. The repository has 71 commits, 1 branch, 0 releases, and 1 contributor. The commit history table shows four entries, all dated '4 months ago': 'LoadDriverDebugProcess' (Add files via upload), 'ProcWatcherDebugDriver' (Delete tr), 'bin/i386' (Delete one), and 'README.md' (Initial commit). Below the commit history, the 'README.md' file is displayed with the title 'Kung-Fu-Malware'.

psaneme / **Kung-Fu-Malware**

Watch 2 Star 1 Fork 1

Code Issues 1 Pull requests 0 Projects 0 Pulse Graphs

No description or website provided.

71 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Find file Clone or download

Commit Message	Action	Time
psaneme committed on GitHub Delete tr		Latest commit c3b0822 on 4 Aug
LoadDriverDebugProcess	Add files via upload	4 months ago
ProcWatcherDebugDriver	Delete tr	4 months ago
bin/i386	Delete one	4 months ago
README.md	Initial commit	4 months ago

README.md

Kung-Fu-Malware

e AV,

s de

logías



➤ E-Mails

- info@ccn-cert.cni.es
- ccn@cni.es
- sat-inet@ccn-cert.cni.es
- sat-sara@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

