Desactivación de infecciones, malware y cryptolockers

Pablo San Emeterio
Román Ramírez

cyber camp
CYBERSECURITY EVENT
2016

GOBIERNO DE ESPAÑA
MINISTERIO DE ENERGÍA, TURISMO Y AGENDA DIGITAL

incibe_
TRABAJANDO POR LA CONFIANZA DIGITAL

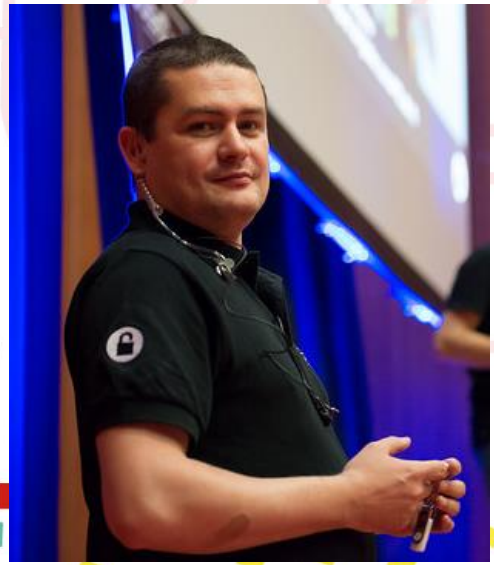
CyberCamp.es

Pablo San Emeterio

- Telefónica
- pablo.sanemeteriolopez@telefonica.com

@psaneme

Román Ramírez



- RootedCON
- rramirez@rootedcon.com

@patowc

Índice

¿Por qué este trabajo?

Trabajo anterior y otros proyectos

Técnicas reconocimiento

Diseño

Demostración

Conclusiones

Índice

¿Por qué este trabajo?

Trabajo anterior y otros proyectos

Técnicas reconocimiento

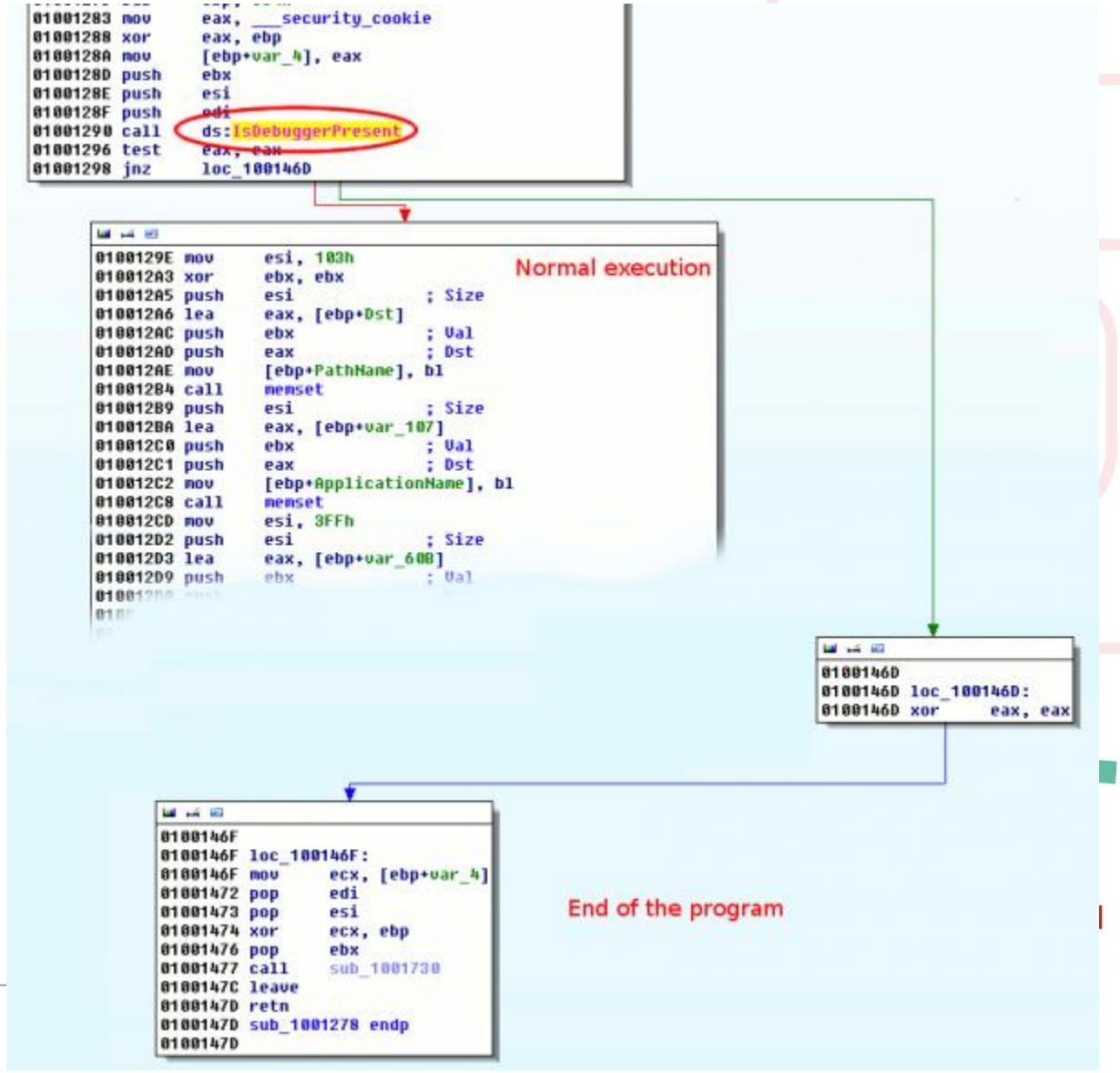
Diseño

Demostración

Conclusiones

¿Por qué de este trabajo?

¿Por qué de este trabajo?



incibe_
2006-2016
TRABAJANDO POR LA CONFIANZA DIGITAL

¿Por qué de este trabajo?

¿Por qué de este trabajo?

- **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.**

- **Si lo combinamos con tecnologías de sandboxing puede dar un ratio de protección muy alto.**

¿Por qué de este trabajo?

Índice

Trabajos anteriores

Existen trabajos previos desde 2008

Xu Chen, Jon Andersen, Z.Morley Mao, Michael Bailey, Jose Nazario.

Towards an Understanding of Anti-virtualization and Anti-debugging Behaivior 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

Índice

¿Por qué este trabajo?

Trabajo anterior y otros proyectos

Técnicas reconocimiento

Diseño

Demostración

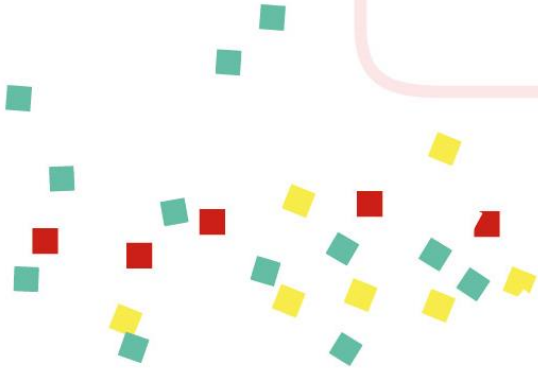
Conclusiones

Técnicas reconocimiento

- Detección de depuración
- Detección de entornos de análisis de malware
- Detección de VM

Técnicas reconocimiento

- Detección de depuración
- Detección de entornos de análisis de malware
- Detección de VM

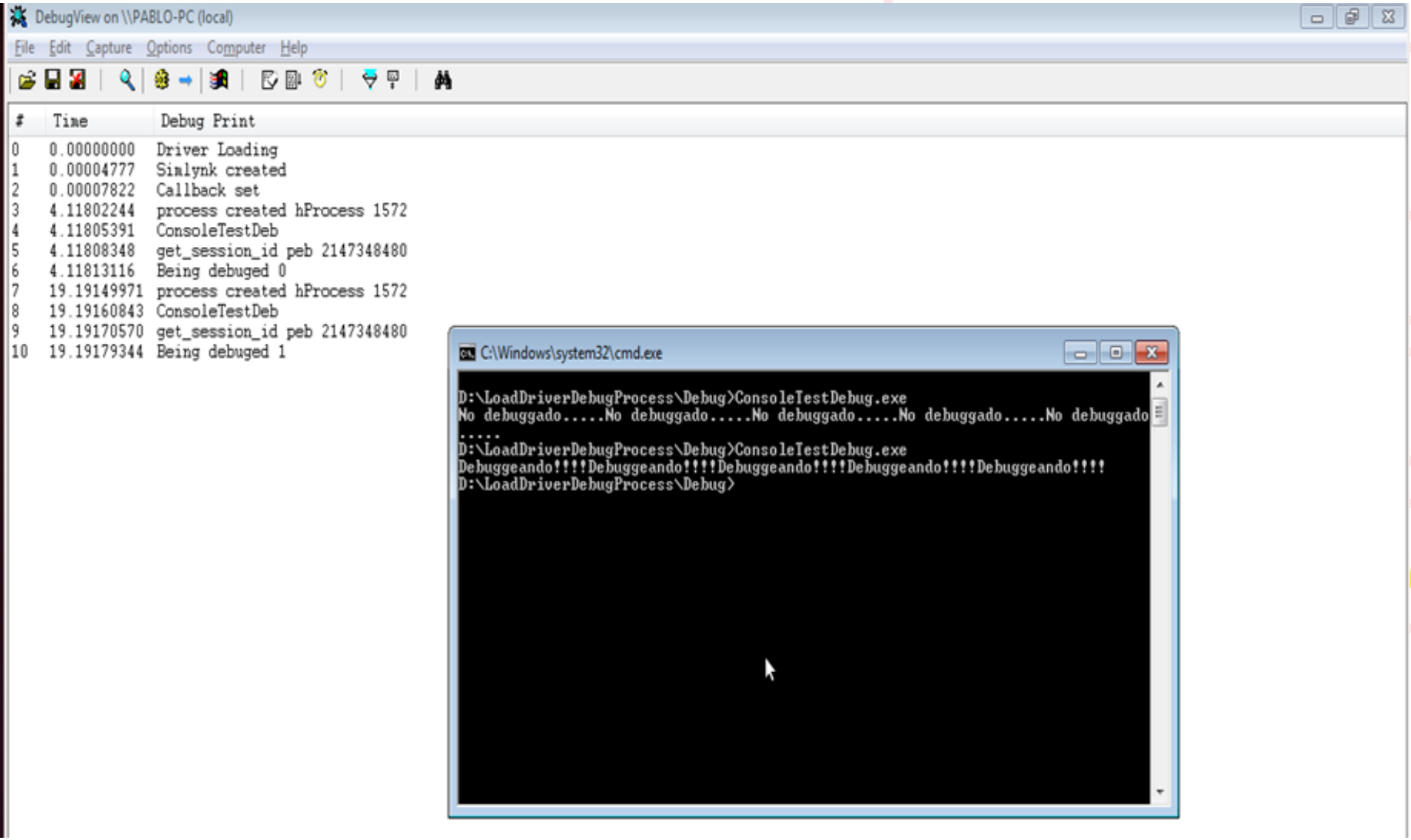


Técnicas reconocimiento

Flag de Debug

```
BOOL WINAPI IsDebuggerPresent(void);

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

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

Técnicas reconocimiento



Técnicas reconocimiento

OutputDebugString

```
DWORD Val = 123;

SetLastError(Val);

OutputDebugString(L"random");

if(GetLastError() == Val) {

    //Debugger Detected - Do Something Here

} else {

    //No Debugger Detected - Continue

}
```

Técnicas reconocimiento

Debug

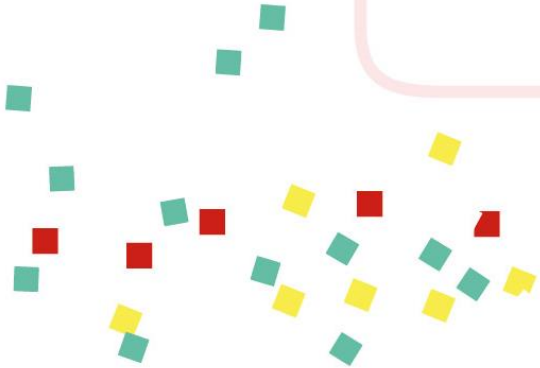
- Falta ampliar esta detección a los **Heaps** del proceso
- Otros puntos

Técnicas reconocimiento

- Detección de depuración

- Detección de entornos de análisis de malware

- Detección de VM



Técnicas reconocimiento

Nombres de procesos

- Ollydbg.exe, idag.exe, immunitydebugger.exe, windbg.exe
- Wireshark.exe, procexp.exe,…

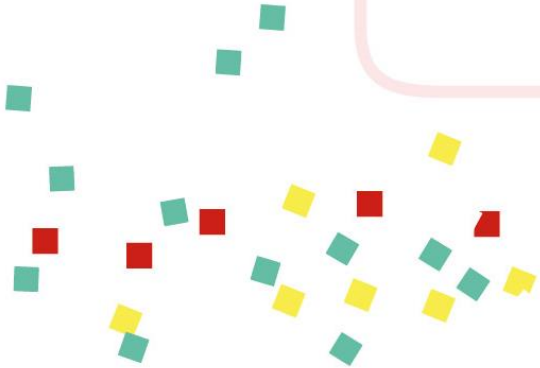
Técnicas reconocimiento

Nombres de ventanas

- OllyDbg WndClass: OLLYDBG, txt: OllyDbg
- Inmunity WndClass: ID (Visible) txt:Immunity Debugger - [CPU]
- IDA WndClass: TIDAWindow txt: The interactive disassembler
- WinDBG WndClass: WinDbgFrameClass txt:WinDbg: 6
- Procexp WndClass: PROCEXPL txt: Process Explorer - Sysinternals
- Wireshark WndClass: gdkWindowToplevel txt: The Wireshark Network Analyzer

Técnicas reconocimiento

- Detección de depuración
- Detección de entornos de análisis de malware
- **Detección de VM**



Técnicas reconocimiento

Características de la máquina

- **RAM**
  - GlobalMemoryStatusEx memoria < 1G
- **Número de micros**
  - GetSystemInfo numero de procesadores < 2
- **Espacio en disco**
  - GetDiskFreeSpaceExA < 60G

Técnicas reconocimiento

Nombres de procesos

- vboxservice.exe, vboxtray.exe ..
- vmware-authd.exe, vmnat.exe, vmwaredhcp.exe, vmware-usbarbitrator64.exe,…
- Agent.py

Técnicas reconocimiento

Nombres de ventanas

- **Productos de virtualización:**

- **h1 = FindWindow("VBoxTrayToolWndClass", NULL);**
- **h2 = FindWindow(NULL, "VBoxTrayToolWnd");**

Técnicas reconocimiento

Paths de ficheros conocidos

```
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\vboxoglerrorspu.dll,
C:\WINDOWS\system32\vboxoglfeedbackspu.dll,
C:\WINDOWS\system32\vboxoglpackspu.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 reconocimiento

Claves de registro conocidos

**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 reconocimiento

Claves de registro conocidos

```
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" o "VIRTUALBOX"

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 reconocimiento

MAC Address

```
08:00:27 …  (vbox)

00:05:69 …  (vmware)
00:0C:29 …
00:1C:14 …
00:50:56 …
```

Técnicas reconocimiento

Otros

Nombre de equipo

- SANDBOX
- TRENDMICRO
- Xp-sp3-template
- MALTEST
- VIRUS
- MALWARE

Path de ejecución

- C:\sample\pos.exe
- C:\Analysis\Sandboxstarter.exe
- C:\Analysis
- C:\Insidetm
- C:\Sample

Nombre de unidades de disco

- SANDBOX

Índice

Diseño

Apps de “pega”

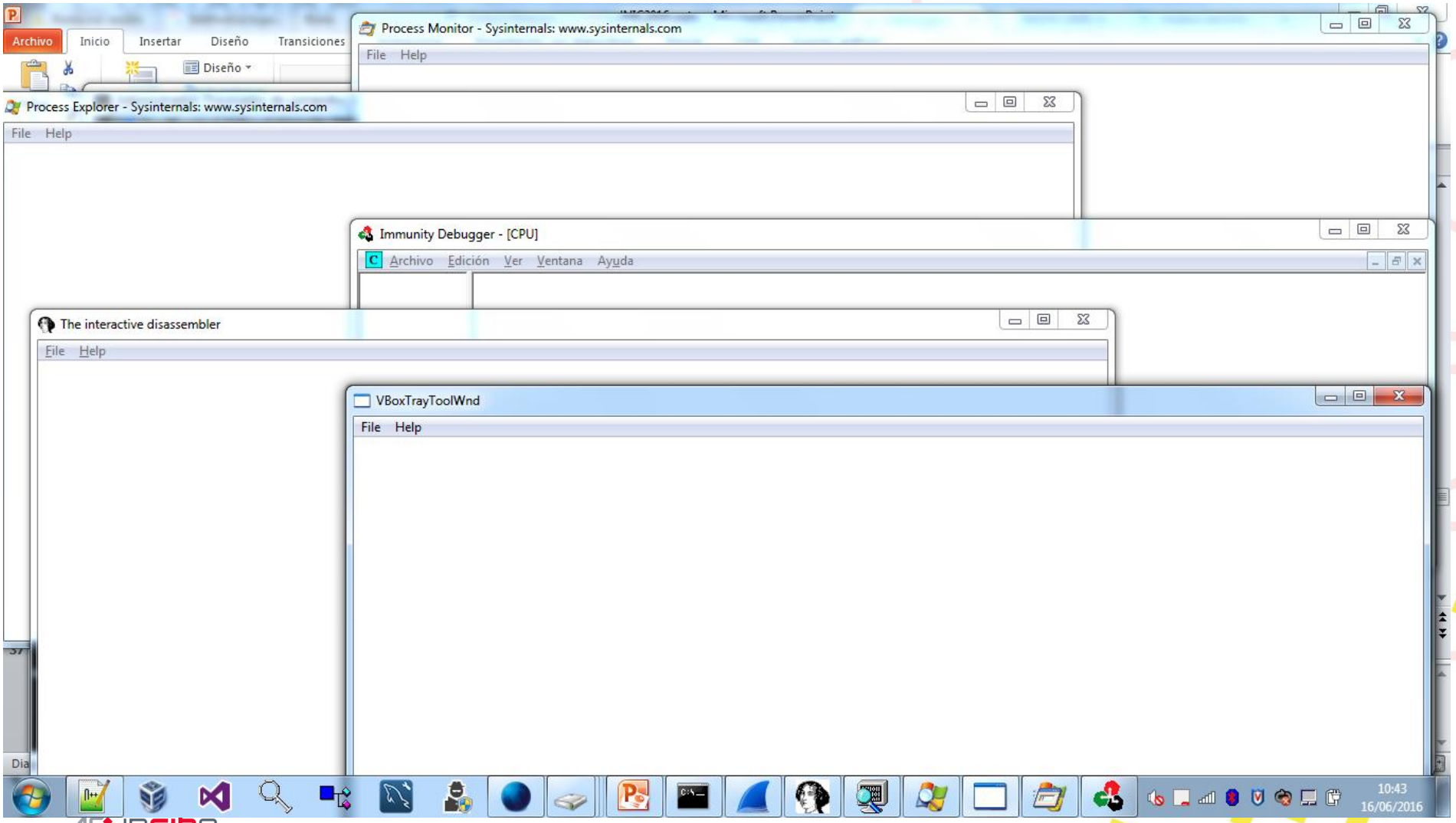
Aplicaciones simulan ser las apps de un analista de malware.

Tienen el mismo proccessName

Tienen el mismo WndClass y WndText

Si se analizan firmas del sw se ve que es diferente

Diseño


Archivo
Inicio
Insertar
Diseño
Transiciones
Diseño
Process Monitor - Sysinternals: www.sysinternals.com
File Help
Process Explorer - Sysinternals: www.sysinternals.com
File Help
Immunity Debugger - [CPU]
Archivo Edición Ver Ventana Ayuda
The interactive disassembler
File Help
VBoxTrayToolWnd
File Help
10:43
16/06/2016

Diseño

Driver

Monitoriza creación de procesos. (PsSetCreateProcessNotifyRoutine)

A cada proceso nuevo :

Le pone el flag de debug a True

Le inyecta una dll con el nobre **sandboxie.dll**

La dll hookea funciones para simular claves de registro, ficheros, etc

Diseño - Driver

PsSetCreateProcessNotifyRoutine routine

The **PsSetCreateProcessNotifyRoutine** routine adds a driver-supplied callback routine to, or removes it from, a list of routines to be called whenever a process is created or deleted.

Syntax

C++

```
NTSTATUS PsSetCreateProcessNotifyRoutine(
  _In_ PCREATE_PROCESS_NOTIFY_ROUTINE NotifyRoutine,
  _In_ BOOLEAN                        Remove
);
```

Diseño - Driver

ProcDebugger.c - ProcessCallback

```
peb=PsGetProcessPeb(ep);

if (peb)
{
    DbgPrint("get_session_id peb %d",peb);

    KeStackAttachProcess(ep,&ka_state);
    isDebug=(int)peb->BeingDebugged;
    DbgPrint("Being debuged %d",(int)isDebug);
    if (!isDebug){
        peb->BeingDebugged =1;
    }
    DbgPrint("FINAL Being debuged %d",(int)peb->BeingDebugged);

    KeUnstackDetachProcess(&ka_state);

}
```

Diseño - Driver

PsSetLoadImageNotifyRoutine routine

The **PsSetLoadImageNotifyRoutine** routine registers a driver-supplied callback that is subsequently notified whenever an image is loaded (or mapped into memory).

Syntax

C++

```
NTSTATUS PsSetLoadImageNotifyRoutine(
  _In_ PLOAD_IMAGE_NOTIFY_ROUTINE NotifyRoutine
);
```

Diseño - Driver

ProcDebugger.c - DllMapCallback

```
RtlInitUnicodeString( &ustrNtdll, L"KERNEL32.DLL" );
stringsDifference = (upperFullName.Length - ustrNtdll.Length) /2;
compare = RtlCompareMemory(upperFullName.Buffer+stringsDifference, ustrNtdll.Buffer, ustrNtdll.Length);

if (compare == ustrNtdll.Length)
{
    DbgPrint("loaded %wZ into Process: %d, FullImageLength:%d ntdllLength:%d",FullImageName,ProcessId,FullI
    DbgPrint ("NTDLL found Process %d, ", PsGetCurrentProcessId());
    //obtener base dll
    pAddr = GetModuleExportAddress(ImageInfo->ImageBase,"NtCreateThreadEx");
    if (pAddr != NULL)
    {
        DbgPrint("Found funcion NtCreateThreadEx address %d indexCall %d", pAddr,*(INT*)(((INT)pAddr)+1));
    }

    pAddr = GetModuleExportAddress(ImageInfo->ImageBase,"LoadLibraryA");
    if (pAddr != NULL)
    {
        DbgPrint("Found function, address %d", pAddr);

        EncolarRespuesta(ProcessId,pAddr); //Guardamos el PID en una Queue para ir sacandolos segun los sol
        CancelPendingIRP(); //Cancelar alguna IRP pendiente para lanzar el PID a userSpace
```

Diseño - LoadDriver

LoadDriver.cpp - LanzaProcesosSimulacion

```
char * Process []={"idag.exe","ImmunityDebugger.exe","OLLYDBG.EXE","procexp.exe"
"Procmon.exe","vboxtray.exe","windbg.exe","Wireshark.exe",NULL};

char command[1024];

memset(PIDs,sizeof(PIDs),0);

for (int i =0; i<MAX_PROCESS;i++)
{
    if (Process[i] == NULL)
        break;

    memset(command,0,sizeof(command));
    sprintf_s(command,sizeof(command),PROCESS_PATH,Process[i]);
    PIDs[i] = StartProcess(command);

}
```

Diseño - LoadDriver

LoadDriver.cpp – InjectDll

```
void InjectDll (HANDLE PID, PVOID pAddr)
{
    //printf("Received Process %d Address %x\n", callbackInfo.hProcessId,callbackInfo

    HANDLE process = OpenProcess(PROCESS_ALL_ACCESS, FALSE, (DWORD) PID);
    if(process == NULL)
    {
        printf("Error: the specified process couldn't be found.\n");

        return ;
    }
```

OpenProcess, VirtualAlloc, WriteProcessMemory,CreateRemoteThread

Diseño - SbieDll

Sólo tiene IATHooking recursivo

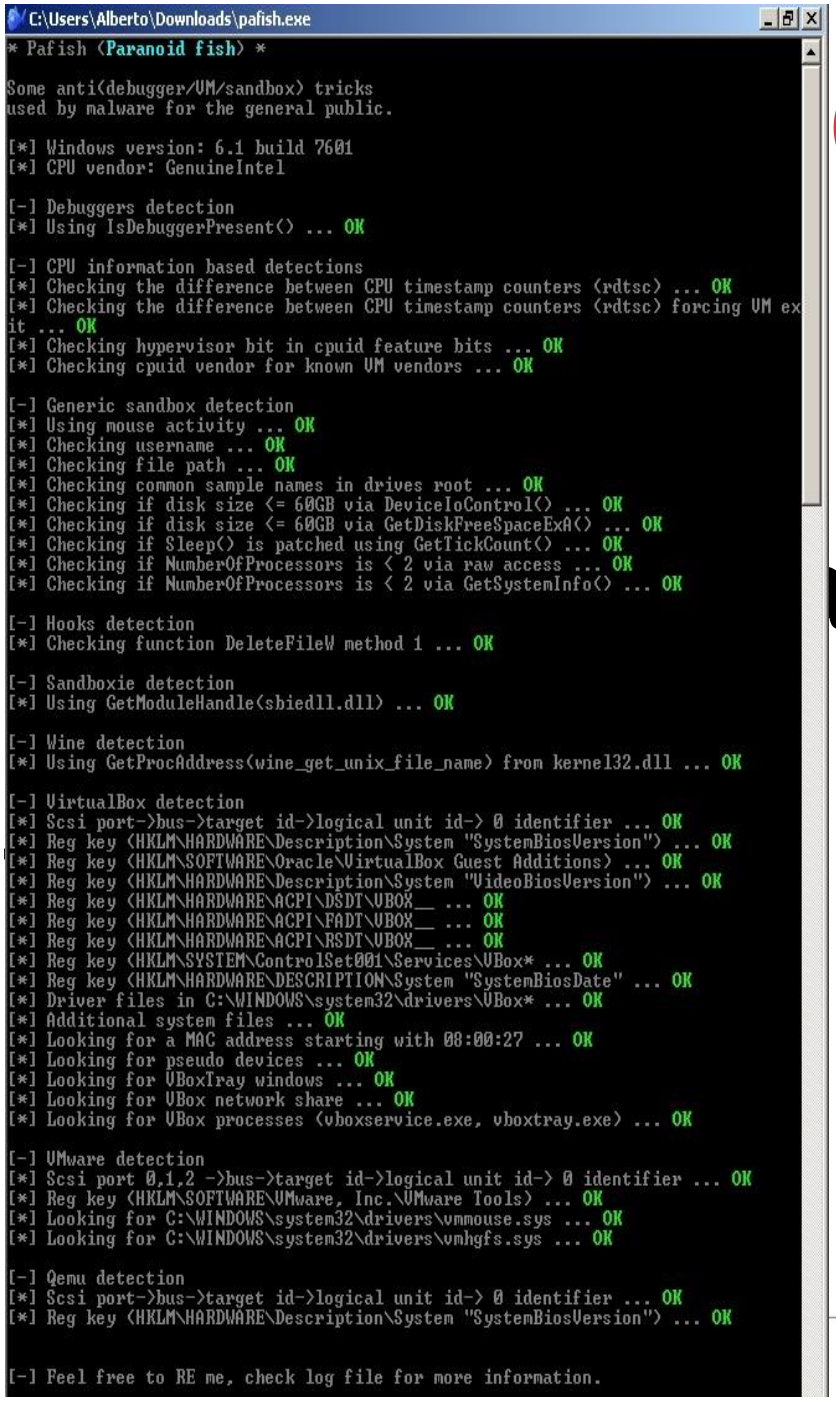
dllmain.cpp – DllMain

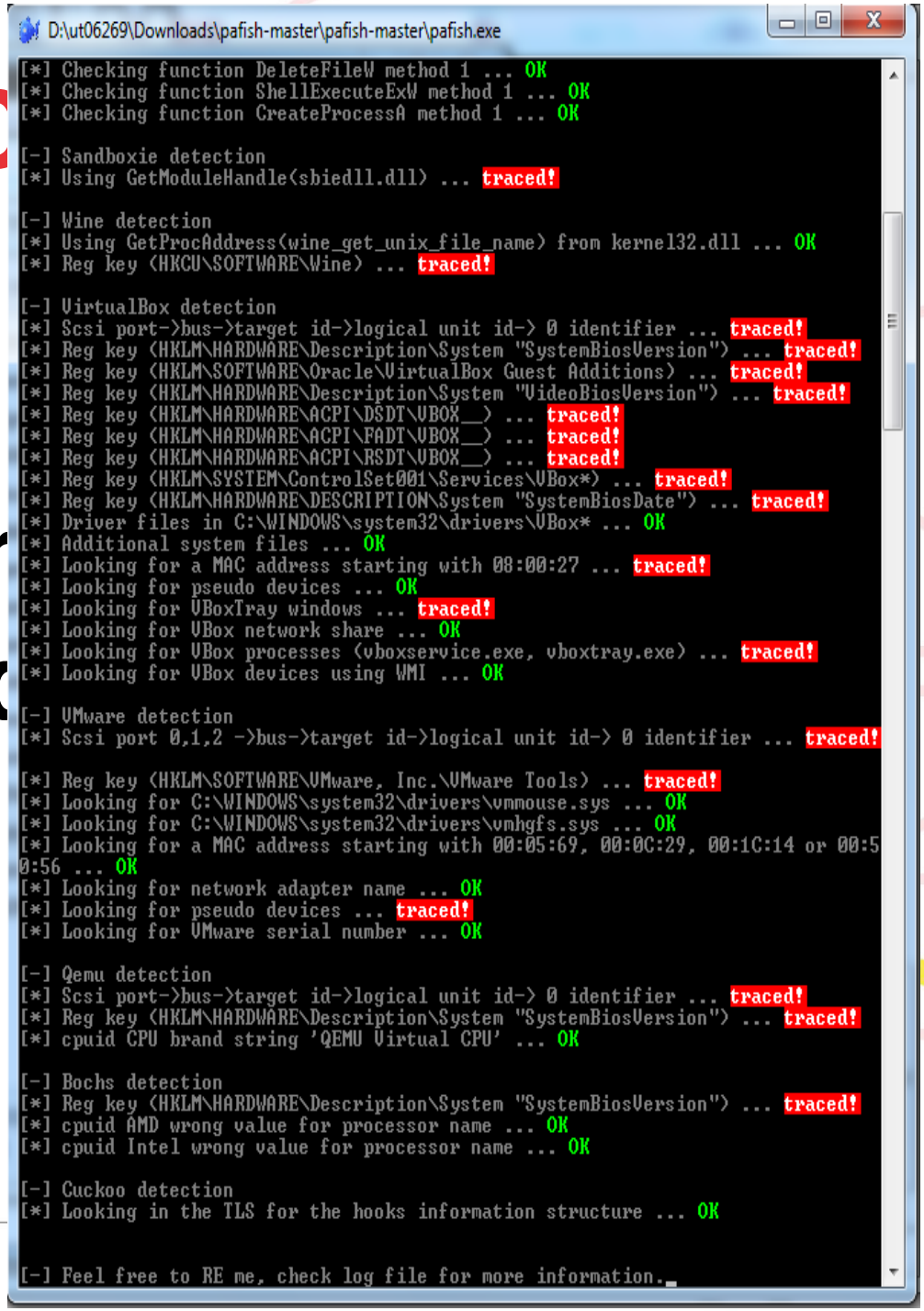
```
BOOL APIENTRY DllMain( HINSTANCE hModule, DWORD fdwReason, LPVOID lpReserved )
{
    switch (fdwReason)
    {
        case DLL_PROCESS_ATTACH:
        {
            hDLL = hModule;
//          dllFilter = NULL;
        ■   ■   ■
        case DLL_PROCESS_DETACH:
        {
            hDLL = hModule;

            // We don't need thread notifications for what we're doing.  Thus, get
            // rid of them, thereby eliminating some of the overhead of this DLL
            DisableThreadLibraryCalls( hModule );

            HookCalls(false);


            break;
        }
        case DLL_THREAD_ATTACH:
        case DLL_THREAD_DETACH:
            break;
```

Diseño - SbieDll

```
SDLLHook SOutputDebugString =
{
    "Kernel32.dll",
    false, NULL,        // Default hook disabled, NULL function pointer.
    {
        { "OutputDebugStringA", MyOutputDebugStringA, NULL },
        { "OutputDebugStringW", MyOutputDebugStringW, NULL },
        { NULL, NULL,NULL }
    }
};


void WINAPI MyOutputDebugStringA( LPCTSTR lpOutputString )
{
    int i = 0;
    i+=1;
    return;
}
void WINAPI MyOutputDebugStringW( LPCTSTR lpOutputString )
{
    int i =0;
    i+=2;
    return;
}
```

Índice


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 VM exit ... 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\VBOX__ ... OK
[*] Reg key (HKLM\HARDWARE\ACPI\FADT\VBOX__ ... OK
[*] Reg key (HKLM\HARDWARE\ACPI\RSDT\VBOX__ ... OK
[*] Reg key (HKLM\SYSTEM\ControlSet001\Services\VBox* ... OK
[*] Reg key (HKLM\HARDWARE\DESCRIPTION\System "SystemBiosDate" ... OK
[*] Driver files in C:\WINDOWS\system32\drivers\VBox* ... 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
[-] 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
[-] 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\VBOX__) ... traced!
[*] Reg key (HKLM\HARDWARE\ACPI\FADT\VBOX__) ... traced!
[*] Reg key (HKLM\HARDWARE\ACPI\RSDT\VBOX__) ... traced!
[*] Reg key (HKLM\SYSTEM\ControlSet001\Services\VBox*) ... traced!
[*] Reg key (HKLM\HARDWARE\DESCRIPTION\System "SystemBiosDate") ... traced!
[*] Driver files in C:\WINDOWS\system32\drivers\VBox* ... OK
[*] Additional system files ... OK
[*] Looking for a MAC address starting with 08:00:27 ... traced!
[*] Looking for pseudo devices ... OK
[*] Looking for VBoxTray windows ... traced!
[*] Looking for VBox network share ... OK
[*] 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 ... 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 ... 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.

Índice

Conclusiones

- Combinado con tecnología de sandboxing puede detener muchos ataques de Ransomware

- Junto con un elemento que analice gmail, hotmail, etc.

- VT, Yara, etc..

Conclusiones

- **Convencidos de que la idea es buena.**
- **El driver funcionando no consume recursos y no hace “extraños”**

- **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 se han roto.**

ToDoS

Una dll por tecnología de sandboxing ✓

Lista blanca de procesos / Directorio protegido

Alerta/Log llamadas de procesos a funciones

Hookear más funciones

CPUID – RedPill - NoPill ✓

Ocultarse ante busquedas

Reparcheos

Agradecimientos

- Ricardo J. Rodriguez
- (@RicardoJRdez)
- Jordi Vazquez (@jordisk)
- Yago Jesus (@YJesus)
- Alberto Ortega (@A0rtega)
- Marc Ribero (@Seifreed)

@patowc

@psaneme

cyber camp
CYBERSECURITY EVENT
2016

Gracias por su atención

GOBIERNO DE ESPAÑA
MINISTERIO DE ENERGÍA, TURISMO Y AGENDA DIGITAL

incibe_
2006-2016
TRABAJANDO POR LA CONFIANZA DIGITAL