Stopping trojans and spywares

CONVISO
APPLICATION SECURITY

Whoami

Antonio Costa "CoolerVoid"
Just another computer programmer.

https://github.com/CoolerVoid/curriculum_cooler

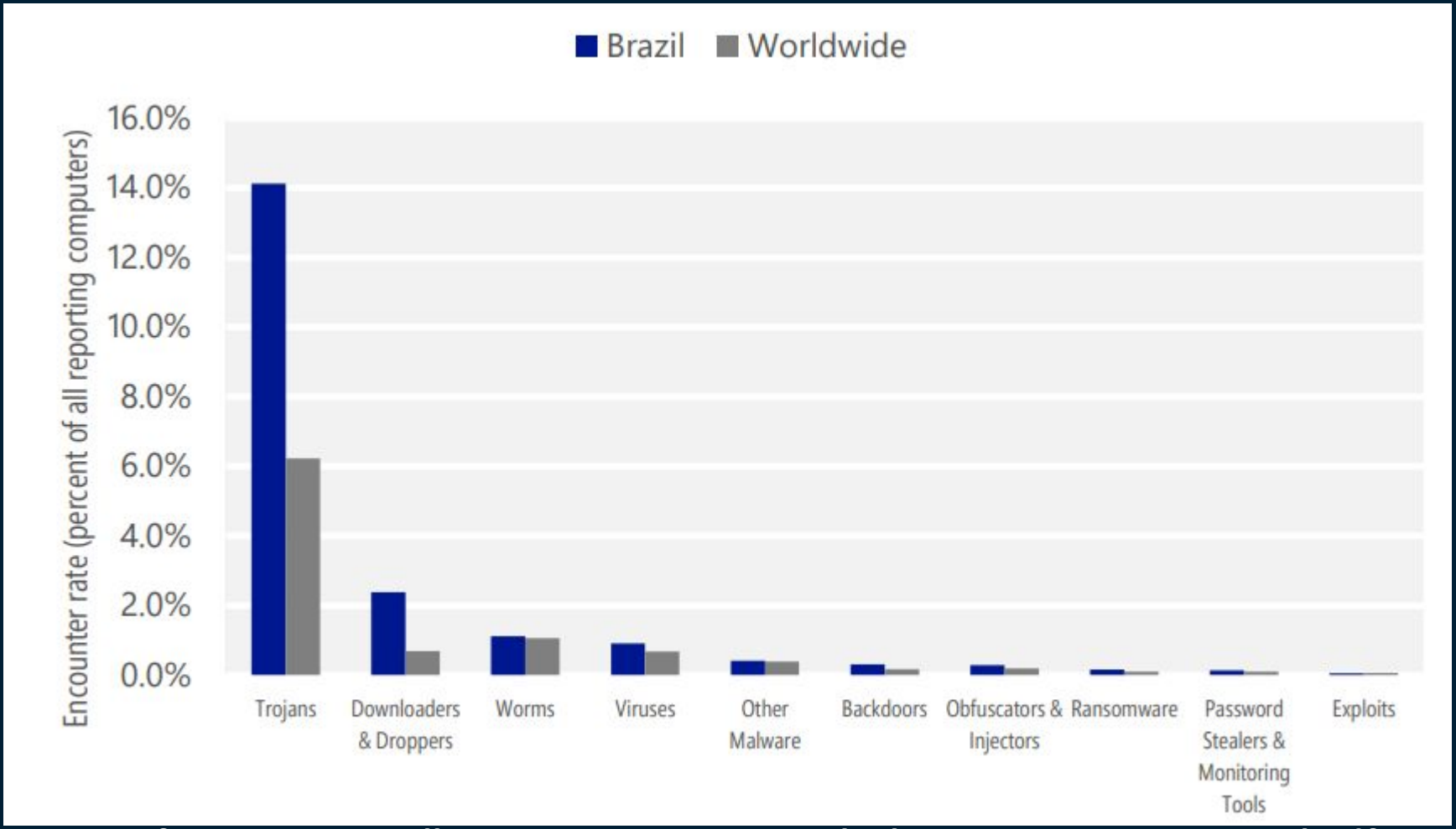
Order-paper

- Common attacks
- The User pitfalls
- Keyloggers types + Stopping Keyloggers
- Screenloggers types + Stopping Screenloggers
- Stopping Pharming

- The global hooking - brainstorms

- Future insights
- The end

Brazil status in 2017



Microsoft_Security_Intelligence_Report_Regional_Threat_Assessment_Brazil.pdf

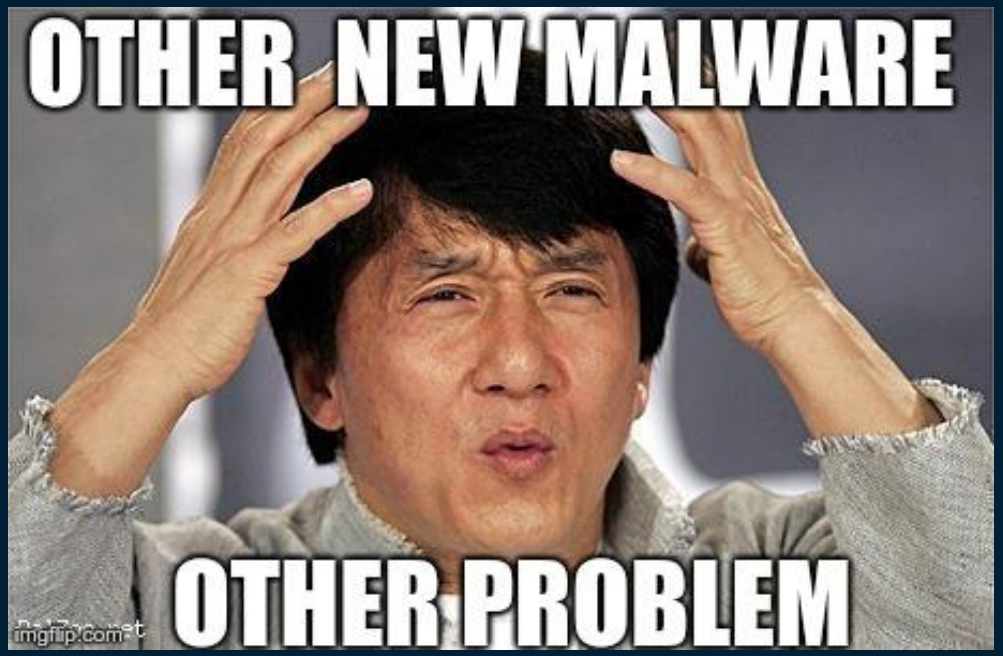
https://www.microsoft.com/en-us/security/intelligence-report

Common attacks

- Web plugins
- Pharming
- Form grabbing
- Keyloggers and screenloggers
- Browser certificates

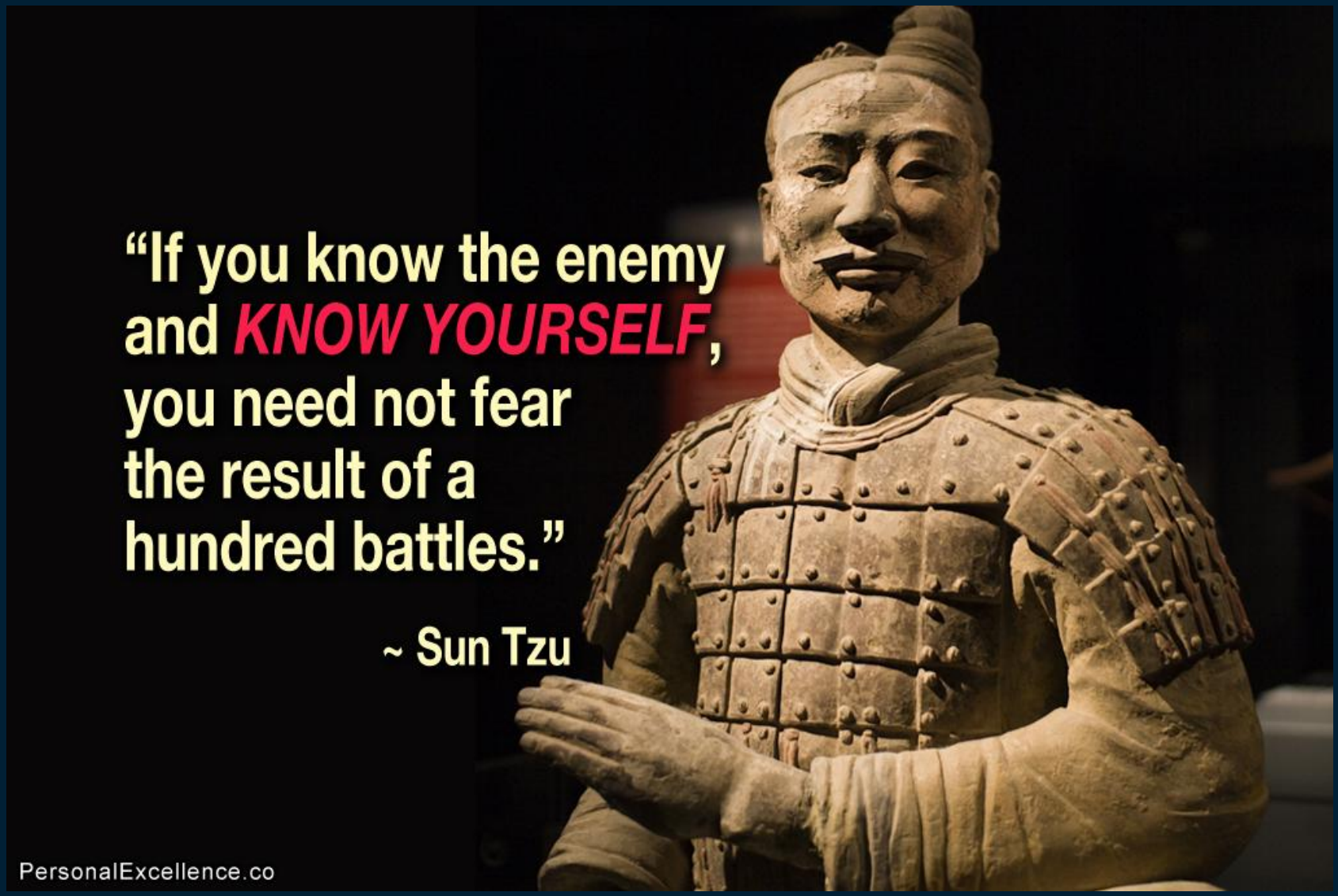
Pitfalls:

- AntiVirus, Firewall, Social engineering...



STOP

BLOB


"If you know the enemy
and KNOW YOURSELF,
you need not fear
the result of a
hundred battles."
~ Sun Tzu
PersonalExcellence.co

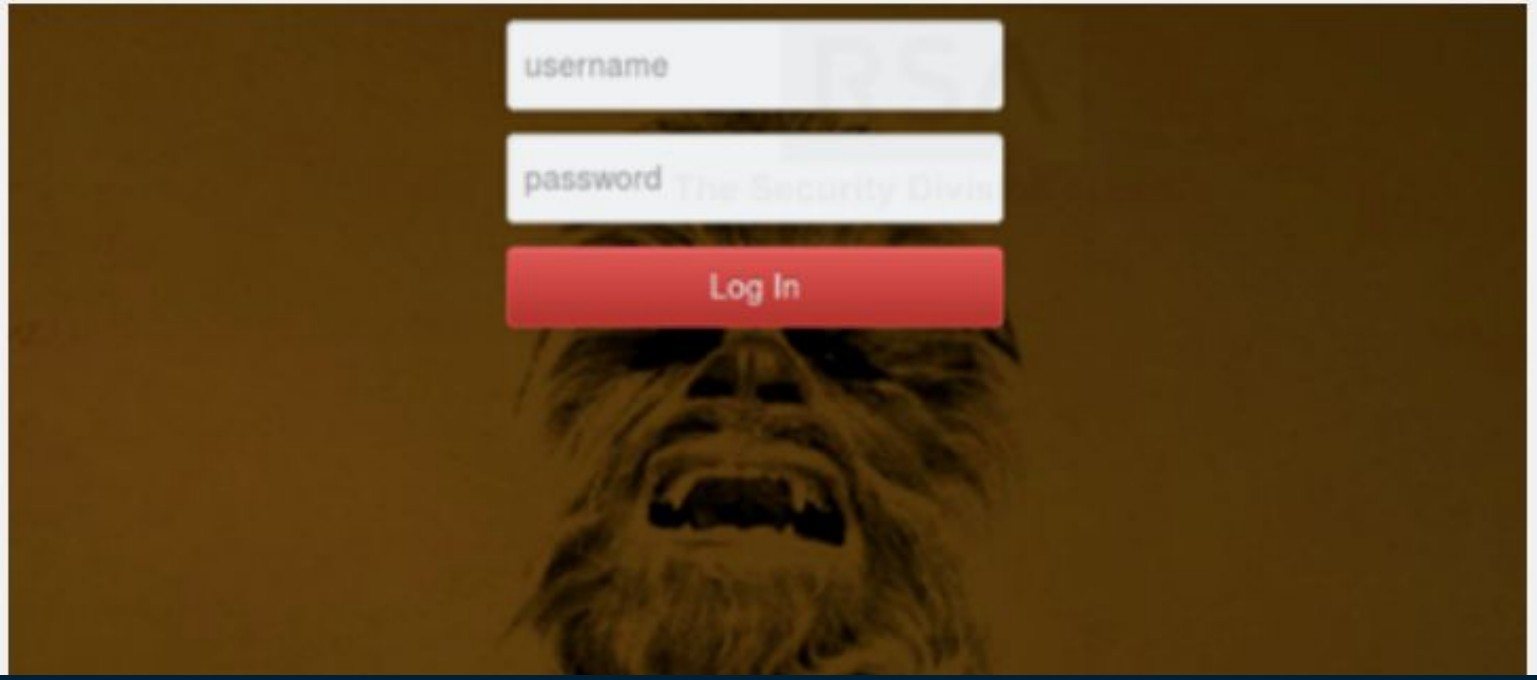
i5-46

BIZ & IT —

Meet “ChewBacca,” the point-of-sale malware that infected dozens of retailers

Memory-scraping malware used TOR anonymity service to conceal itself.

DAN GOODIN - 1/31/2014, 5:20 PM



Chewbaca, Zeus, Dyre, Chupacabra, SpyEye...

"ChewBacca," as the crimeware is dubbed, scrapes large chunks of computer memory from infected terminals and dumps them to a file, a researcher from RSA reported in a blog post published Thursday. It then uses regular expressions and other programming techniques to extract data that was copied from credit and debit cards. ChewBacca also captures sensitive data using a generic keylogger.

"The ChewBacca trojan appears to be a simple piece of malware that, despite its lack of sophistication and defense mechanisms, succeeded in stealing payment card information from several dozen retailers around the world in a little more than two months," Yotam Gottesman, a senior security researcher on RSA's FirstWatch team, wrote. Researchers found that beginning in late October, ChewBacca had logged track 1 and 2 data of payment cards scanned on infected

https://github.com/ytisf/theZoo

Chewbaca, Zeus, Dyre, Chupacabra, SpyEye...

"ChewBacca," as the crimeware is dubbed, scrapes large chunks of computer memory from infected terminals and dumps them to a file, a researcher from RSA reported in a blog post published Thursday. It then uses regular expressions and other programming techniques to extract data that was copied from credit and debit cards. ChewBacca also captures sensitive data using a generic keylogger.

"The ChewBacca trojan appears to be a simple piece of malware that, despite its lack of sophistication and defense mechanisms, succeeded in stealing payment card information from several dozen retailers around the world in a little more than two months," Yotam Gottesman, a senior security researcher on RSA's FirstWatch team, wrote. Researchers found that beginning in late October, ChewBacca had logged track 1 and 2 data of payment cards scanned on infected

https://github.com/ytisf/theZoo

Process Monitor

Introduction

Process Monitor is an advanced monitoring tool for Windows that shows real-time file system, Registry and process/thread activity. It combines the features of two legacy Sysinternals utilities, *Filemon* and *Regmon*, and adds an extensive list of enhancements including rich and non-destructive filtering, comprehensive event properties such session IDs and user names, reliable process information, full thread stacks with integrated symbol support for each operation, simultaneous logging to a file, and much more. Its uniquely powerful features will make Process Monitor a core utility in your system troubleshooting and malware hunting toolkit.

docs.microsoft.com/en-us/sysinternals/downloads/procmon

Keyloggers:

keyloggers are applications that monitor a user's keystrokes and then send this information back to the malicious user.

https://attack.mitre.org/wiki/Technique/T1056

Keyloggers types

One of common approach in a keylogger development is using function **GetAsyncKeyState() or GetKeyState()**:

Determines whether a key is up or down at the time the function is called, and whether the key was pressed after a previous call to **GetAsyncKeyState**.

Syntax

C++

```
SHORT WINAPI GetAsyncKeyState(
  _In_ int vKey
);
```

https://msdn.microsoft.com/pt-br/library/windows/desktop/ms646293(v=vs.85).aspx

You can found in user32.dll

Keyloggers types

Second common approach in a keylogger development is using function **SetWindowsHookEx() with first argument called be "WH_KEYBOARD_**LL" to get all inputs of keyboard.

Installs an application-defined hook procedure into a hook chain. You would install a hook procedure to monitor the system for certain types of events. These events are associated either with a specific thread or with all threads in the same desktop as the calling thread.

Syntax

C++

```
HHOOK WINAPI SetWindowsHookEx(
  _In_ int       idHook,
  _In_ HOOKPROC  lpfn,
  _In_ HINSTANCE hMod,
  _In_ DWORD     dwThreadId
);
```

https://msdn.microsoft.com/pt-br/library/windows/desktop/ms644990(v=vs.85).aspx

You can found in kernel32.dll

Keyloggers:

That is a C/C++ functions, they don't run in C#, Delphi, VB…

That is not TRUE !!!

Remember:

GetAsyncKeyState() and GetKeyState() can found in **user32.dll**

SetWindowsHookEx() can found in **kernel32.dll**

C#, VB, Delphi…

uses kernel32.dll and user32.dll

Stopping keyloggers:

One way to block a keylogger is to intercept a function like GetAsyncKeyState() , GetKeyState() and SetWindowsHookEx() using a simple hooking, and return null value.

Using easyhook is not hard, but you will need to construct your strategy, one mistake in a DLL file, the programm will crash and wont work.

Hooking:

The Easyhook example uses popular technique "System-wide Windows Hooks" for injecting DLL into a targeted process relies on provided by Windows Hooks.

That hook technique is normally implemented in a DLL in order to meet the basic requirement for system-wide hooks. The basic of that technique is the sort of hooks is that the hook callback procedure is executed in the address spaces of each hooked up process in the system. To install a hook you call **SetWindowsHookEx()** with your particular parameters. While the application installs a system-wide hook, the operating system try to map the DLL into the particular address space in each of its client processes. Anyway, global variables within the DLL will be "per-process" and cannot be shared among the processes that have loaded the hook DLL**.**

Stopping keyloggers:

```
SHORT WINAPI GetAsyncKeyStateHook( _In_ int vKey )
{
    MessageBox(NULL, L"Keylogger blocked !!!", L"Anti-key-keeper", MB_OK);
    return 0;
}
```

```
if (inRemoteInfo->UserDataSize == sizeof(DWORD))
    gFreqOffset = *reinterpret_cast<DWORD *>(inRemoteInfo->UserData);

// Perform hooking
HOOK_TRACE_INFO hHook = { NULL }; // keep track of our hook

                                  // Install the hook
NTSTATUS result = LhInstallHook(
    GetProcAddress(GetModuleHandle(TEXT("user32")), "GetAsyncKeyState"),
    GetAsyncKeyStateHook,
    NULL,
    &hHook);
if (FAILED(result))
{
    std::wstring s(RtlGetLastErrorString());
    std::wcout << "Failed to install hook in antikeykeeper.dll: ";
    std::wcout << s;
```

Stopping keyloggers:

If the first argv is “WH_KEYBOARD_LL” block him...

```
HHOOK WINAPI SetWindowsHookExHook(_In_ int idHook, _In_ HOOKPROC lpfn, _In_ HINSTANCE hMod, _In_ DWORD dwThreadId)
{
    if (idHook == WH_KEYBOARD_LL)
    {
        MessageBox(NULL, L"Keylogger blocked !!!", L"Anti-key-keeper", MB_OK);
        return 0;
    }
    else
        return SetWindowsHookEx(idHook, lpfn, hMod, dwThreadId);
}
```

```
NTSTATUS result2 = LhInstallHook(
    GetProcAddress(GetModuleHandle(TEXT("kernel32")), "SetWindowsHookEx"),
    SetWindowsHookExHook,
    NULL,
    &hHook2);
```

Stopping keyloggers:

Proof of concept:

Screenloggers:

Screenlogger is used to steal the user's password when him uses the virtual keypad of bank. The function uses screen-shots of the full desktop and with the function GetCursorPos() of GDI windows api gets the X and Y positions of the mouse to find the mouse pointer and cuts a square to get an image...

Stopping Screenloggers:

Its not hard, you need to use easyhook in the GetCursorPos() function, to always return value zero, follow this path and you can block.

Retrieves the position of the mouse cursor, in screen coordinates.

Syntax

C++

```
BOOL WINAPI GetCursorPos(
  _Out_ LPPOINT lpPoint
);
```

msdn.microsoft.com/pt-br/library/windows/desktop/ms648390(v=vs.85).aspx

Stopping Screenloggers

Video of proof of concept:

Stopping Pharming

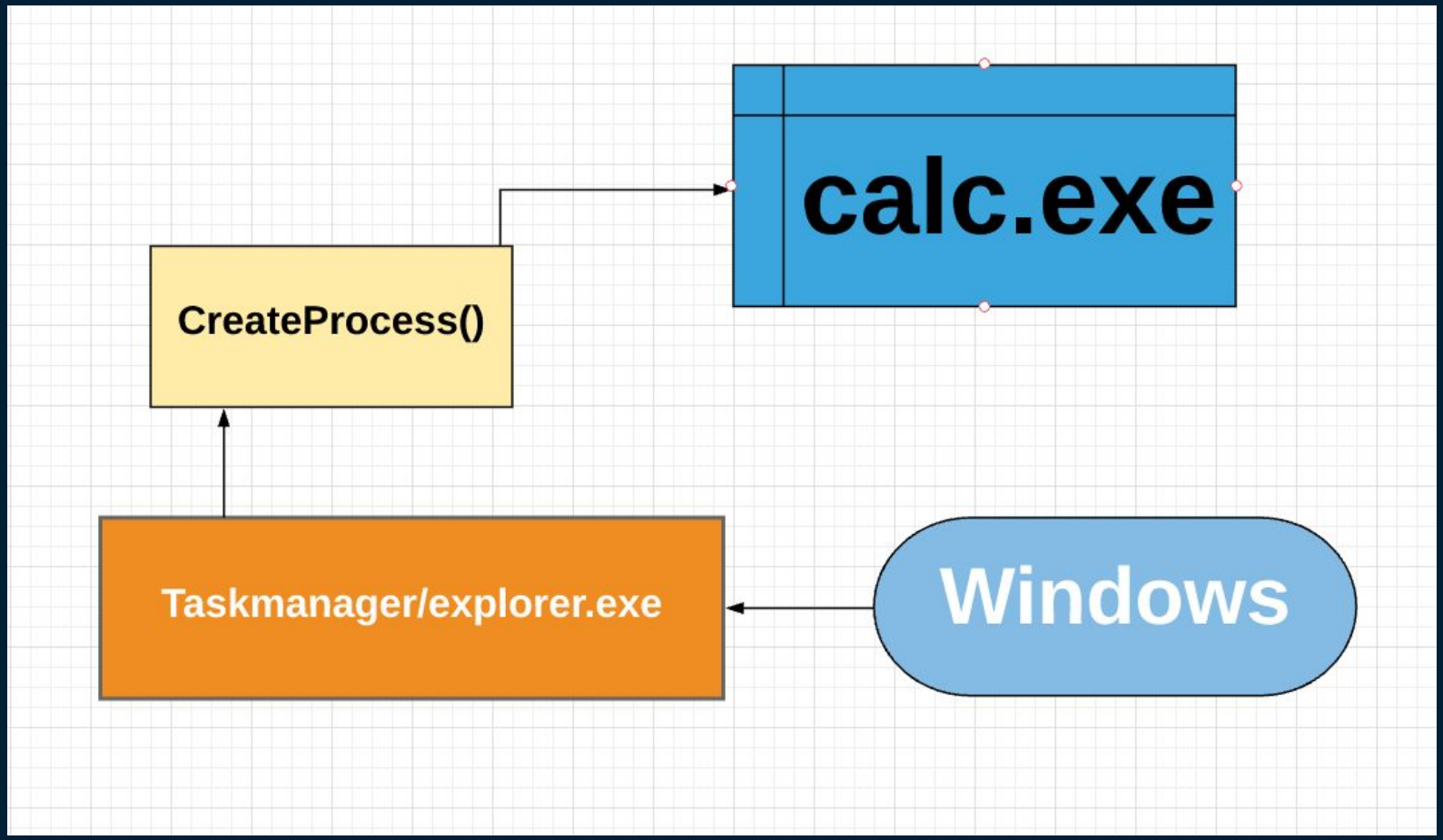
Stopping pharming attacks need 2 steps:

1. Global hooking using CreateProcess() to intercept all programs before they call WriteMemory(), and inject DLL of hooking in each program…

2. Intercept with hooking the function WriteFile(), if the first argument is “hosts.txt”(DNS redirect config) return zero and block.

That approach its common, the attackers keep using redirect with pharming is a fatal combo with phishing to get credentials.

Global Hooking

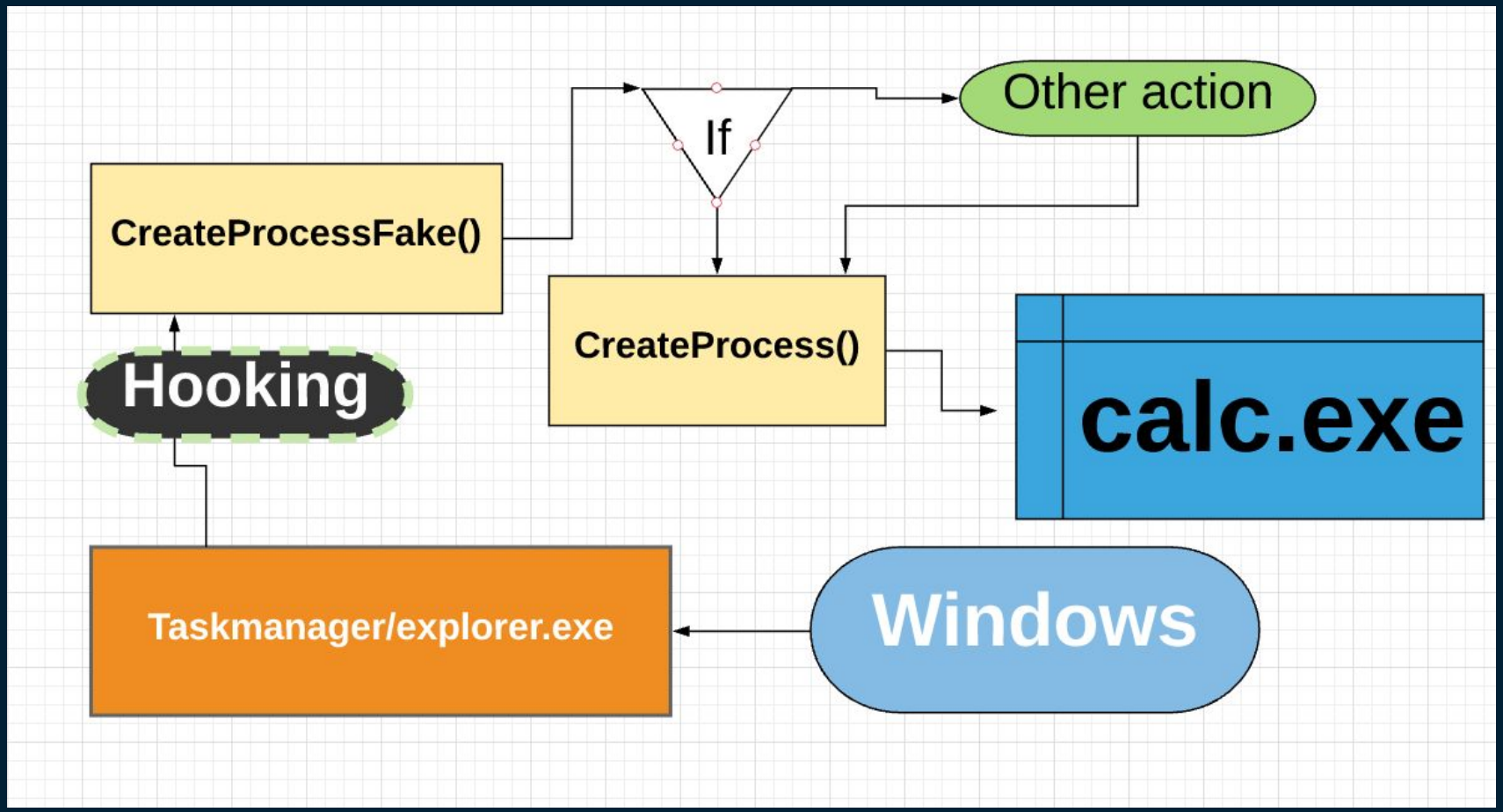


Global Hooking

Global hooking using **CreateProcess()** to intercept all programs before they call **WriteMemory()**, and inject DLL of hooking with protection functions in each program…

That technique, you can use to create your antivirus, is a important step intercept all programs before execution, you can send the binary to sandbox (to make analysis) before of execution.

Global Hooking



Future insights

If you do not control the enemy, the enemy will control you

~ Miyamoto Musashi

AZ QUOTES

Future insights - Fake information

Fake information about the bank's window title to mislead the malware...

```
32  string GetActiveWindowTitle()
33  {
34          char wnd_title[256];
35
36          HWND hwnd=GetForegroundWindow();
37          GetWindowText(hwnd,wnd_title,sizeof(wnd_title));
38
39          return wnd_title;
40  }
```

Future insights - Fake information

Fake information about the operational system, to mislead the malware.

```
#include <windows.h>

// i improve this func https://msdn.microsoft.com/en-us/library/windows/desktop/ms724834%28v=
// return windows version
char * OperatingSystemVersion()
{
    OSVERSIONINFOEX VersionOS;
    VersionOS.dwOSVersionInfoSize = sizeof(OSVERSIONINFOEX);

    if(GetVersionEx((OSVERSIONINFO*)&VersionOS))
    {
        if((VersionOS.dwMajorVersion == 5) && (VersionOS.dwMinorVersion == 0))
            return "WINDOWS 2000";
        else if((VersionOS.dwMajorVersion ==5) && (VersionOS.dwMinorVersion == 1))
            return " WINDOWS XP";
        else if((VersionOS.dwMajorVersion == 5) && (VersionOS.dwMinorVersion == 2))
        {
```

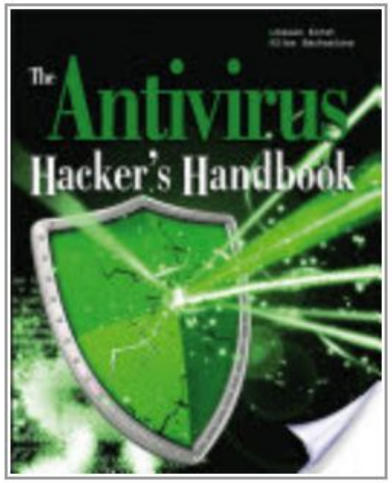
Future insights

Bypass Problems to mitigation:

Unhooking
Non polymorphic
Doesn`t have killer protection
Anti-debug protection
Anti tampering etc…
Windows Powershell
VB script

Reference:

The Antivirus Hacker's Handbook



Joxean Koret, Elias Bachaalany

John Wiley & Sons, 28 de set de 2015 - 384 páginas

★★★★★

0 Resenhas

G+

Hack your antivirus software to stamp out future vulnerabilities

The Antivirus Hacker's Handbook guides you through the process of reverse engineering antivirus software. You explore how to detect and exploit vulnerabilities that can be leveraged to improve future software

Does not have anti spyware content, but have good stuff on hookings.

Thank you

Greetz: Justin Stenning aka spazzarama, Tsuda Kageyu, M0nad, p4ck4g3, Welias, RaphaelSC and Moh4med37.

https://github.com/CoolerVoid

Twitter: @Cooler_freenode

acosta@conviso.com.br

coolerlair@gmail.com

Entre em contato:

0800 003 0304

CONTATO COMERCIAL
comercial@conviso.com.br

ENDEREÇO:

Curitiba - PR
Rua Coronel Assumpção, 74
Alto da XV - CEP: 80045-355

São Paulo - SP
Rua Funchal, 203 - Edifício Concorde
1º andar, Conjunto 11 - Vila Olimpia
CEP: 04551-904

Siga no Twitter
twitter.com/conviso

Curta a Fanpage no Facebook
fb.com/convisoappsec

Conheça o nosso blog
blog.conviso.com.br

CONVISO
APPLICATION SECURITY