

Sandbox detection and evasion techniques. How malware has evolved over the last 10 years

 ptsecurity.com/ww-en/analytcs/antisandbox-techniques/

■ positive technologies

Contents

Introduction

In most cases, hackers "case out" their targets before attacking. They do this by collecting information about the system and internal network, which gives an idea of how they can profit from an attack and helps to plan further actions. Of course, the attackers need to be sure they have accessed a real workstation on a company's infrastructure, and not a mere sandbox—a virtual environment designed to analyze the behavior of executable files. That is why modern malware has capabilities for detecting and evading protection mechanisms, as well as for hiding malicious functionality if run in a sandbox or code analyzer.

We have analyzed 36 malware families used by at least 23 APT groups around the world during the period from 2010 through the first half of 2020. The selection was made based on [MITRE](#) data and information about new malware samples analyzed by the [PT Expert Security Center](#).

In this research, we will show how sandbox evasion techniques have evolved in the last 10 years.



What is a sandbox

Sandboxes run a file in an isolated virtual environment, analyze the actions performed by the file, and issue a verdict that answers the all-important question: is the file safe or not?

Sandboxes can be agent-based or agentless.

Agent-based sandboxes The virtual machine has a built-in agent (special process) that manages the system, in addition to getting and passing events and artifacts of interest. When a new process is generated, the sandbox intercepts API function calls (changes to an address in process memory or changes to code in a function body).

This approach has one significant drawback: the sandbox needs to conceal and protect agent-related objects from malware.

Agentless sandboxes These sandboxes use second level address translation (SLAT), a form of hardware-assisted virtualization built into CPUs. AMD processors support SLAT through Rapid Virtualization Indexing (RVI), while Intel's implementation is known as Extended Page Table (EPT).

Extended page tables are nested between the guest physical memory and the host virtual memory. This allows to do the following:

- Examine memory pages of the guest machine.
- Identify important parts (for example, parts containing addresses or code of kernel functions).
- Mark selected pages to separate EPT memory access rights from guest machine access rights.
- Intercept attempts to access marked memory regions (if this happens, an EPT violation error will occur and the guest machine will be stopped).
- Analyze the memory state and extract information about an event.
- Mark the memory page anew to return it to the correct state.
- Restore operation of the guest machine..

All these actions are observed from outside the sandbox: malware cannot detect that it is being watched.

Executive summary

Sandbox evasion and anti-analysis techniques are found most frequently in remote access tools (accounting for 56% of the malware in our dataset) and loaders (14%). These types of malware are used to perform reconnaissance and gather information about the target

system. If attackers spot that their malware is running inside a virtual environment, they will not continue their attack and will not download the payload. Instead, they will attempt to maintain stealth by terminating execution of the malware.

Figure 1. Types of malware

69 percent of the malware samples in our dataset were used for cyberespionage. Such attacks require staying invisible on a victim's system for extended periods, which is why attackers look for ways to maintain long-term, stealthy persistence.

Figure 2. Attacker motives

To detect virtual machines (sandboxes), attackers send WMI queries (25% of malware in the dataset), perform other environment checks (33%), or check which processes are running (19%). Attackers can also use information about the virtualization environment to plan their future efforts.

Figure 3. Popular sandbox evasion and anti-analysis methods (percentage of malware)

It is getting more and more difficult to perform static analysis of malicious files by matching suspicious files with known signatures and hash sums, because malware authors are using code obfuscation to impede analysis attempts. That is why we recommend analyzing file behavior in a sandbox.

Evolution of anti-sandbox techniques

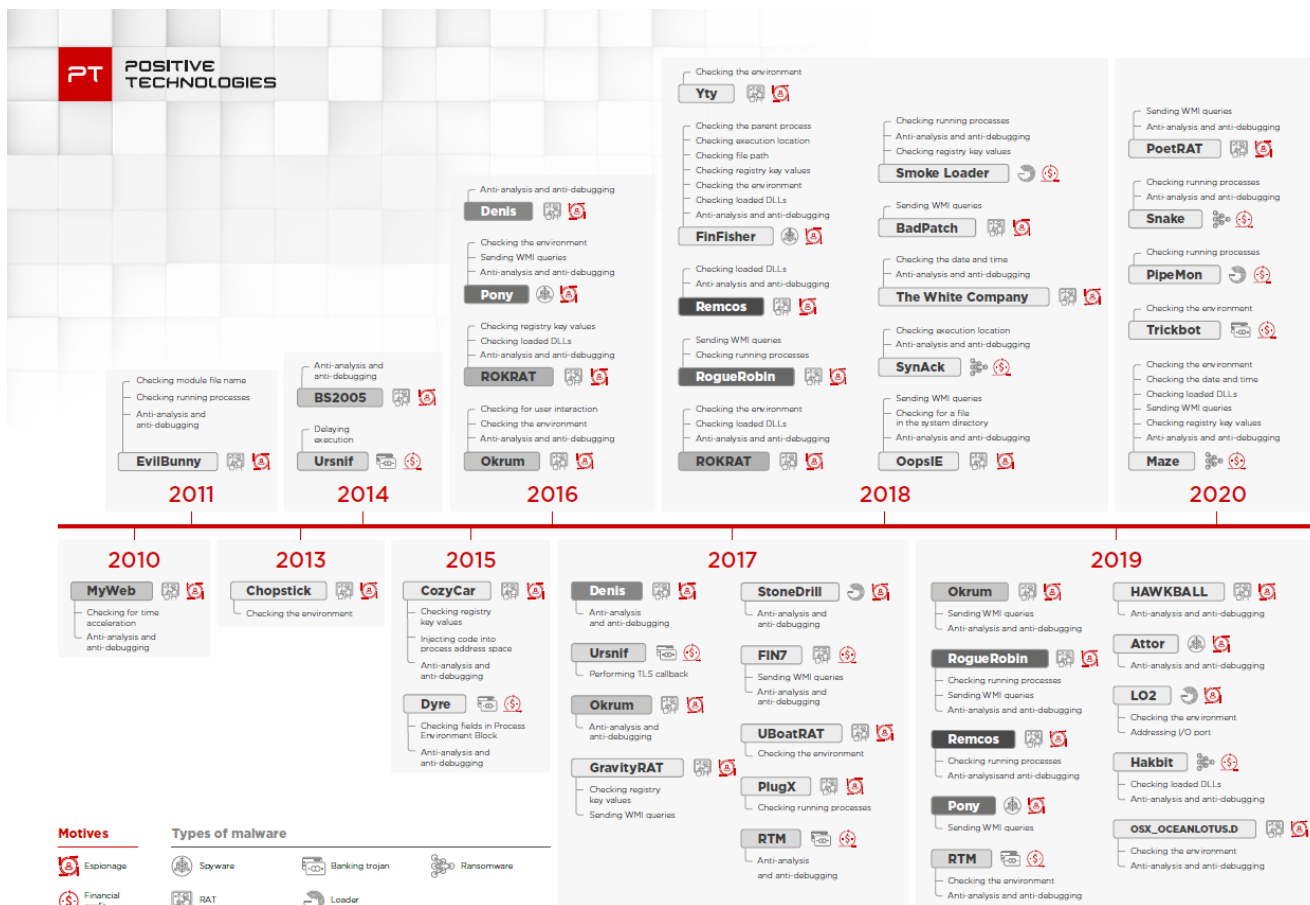


Figure 4. Sandbox evasion and anti-analysis methods used by malware in 2010–2020
 Download Timeline

To evade sandboxes and analysis tools, the same malware may use different methods in different years. Threat actors also try to combine multiple methods. If one method does not work and is intercepted by the sandbox, the malware can still use other signs to determine whether it is running in a virtual environment and, if so, terminate itself in time to avoid discovery.



APT group [Ke3chang](#) (also known as APT15) used the [MyWeb](#) backdoor in 2010, [BS2005](#) in 2014, and [Okrum](#) in 2016–2019. They used a number of methods for evading and detecting virtual machines and code analyzers. They checked for human interaction by waiting for three left clicks or locating the cursor position, and determined the amount of physical memory by calling `GetGlobalMemoryStatusEx` or by sending WMI queries.

Here are several more examples:

ROKRAT

Type: RAT

Group: APT37, active since 2012

Target: organizations in South Korea

Infection vector: phishing (phishing emails with malicious HWP attachments exploiting vulnerability CVE-2013-0808)

Motive: espionage

Sandbox evasion and anti-analysis methods used	2016	2018
Checking whether the SbieDll.dll, Dbghelp.dll, Api_log.dll, or Dir_watch.dll libraries are loaded	+	+
Obtaining the value of the SystemBiosVersion key of the HARDWARE\DESCRIPTION\System registry branch	+	-
Using the NOP (No Operation) instruction as padding in self-modifying code to protect from debuggers	+	-
Calling the IsDebuggerPresent function to detect debugging	-	+
Calling the GetTickCount function twice to check for step-through execution	-	+
Checking for the file C:\Program Files\VMware\VMware Tools\vmtoolsd.exe	-	+

RogueRobin

Type: RAT

Group: DarkHydrus, active since 2016

Target: government agencies and educational institutions in the Middle East

Infection vector: phishing (malicious Microsoft Office documents distributed via Google Drive)

Motive: espionage

Sandbox evasion and anti-analysis methods used	2018	2019
Sending WMI queries to get the BIOS version and manufacturer	+	+

Sandbox evasion and anti-analysis methods used	2018	2019
Sending WMI queries to check the number of CPU cores; the value must exceed 1	+	+
Sending WMI queries to check the amount of physical memory; the value must be at least 2,900,000,000 bytes	+	+
Checking the number of running processes for Wireshark and Sysinternals	+	+
Obfuscating a PowerShell script with Invoke-Obfuscation	+	+
Checking for debuggers in each DNS request	-	+

Remcos

Type: RAT

Group: [Gorgon Group](#), active since 2018

Target: government organizations in Russia, the United Kingdom, Spain, and the United States

Infection vector: phishing (emails with malicious Microsoft Word documents exploiting vulnerability CVE-2017-0199)

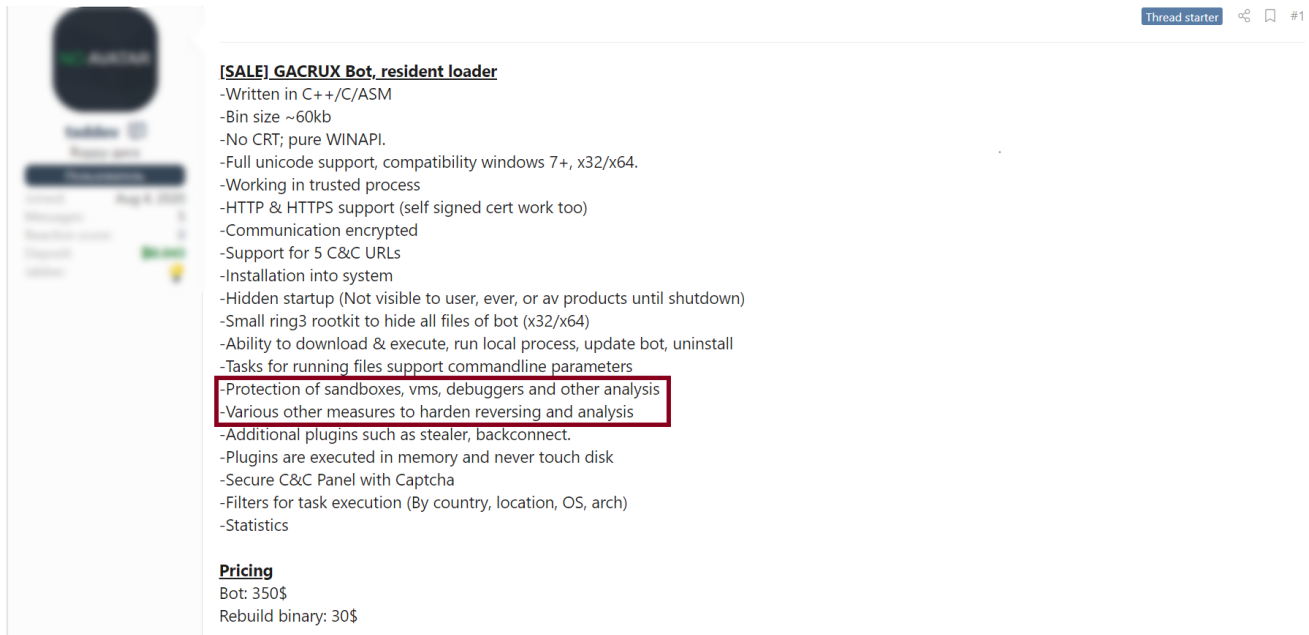
Motive: espionage

Sandbox evasion and anti-analysis methods used	2018	2019
Checking for obsolete SbieDll.dll system artifact	+	-
Encrypting source code with RC4 and Base64 algorithms	+	+
Checking active processes for vmtoolsd.exe and vbox.exe	-	+
Calling the IsDebuggerPresent function to check whether the process that calls the function is being run in a debugger context	-	+



In 2018–2019, the number of malware families using sandbox evasion techniques increased. However, this statistic is likely explained by the fact that experts now investigate malware samples more often.

Attackers who sell malware on the darkweb also offer functionality for detecting and evading sandboxes and antivirus tools, as well as for countering analysis and debugging. The starting price for malware with built-in sandbox evasion is \$30. Additional protection from detection by sandboxes and antivirus solutions costs \$20.



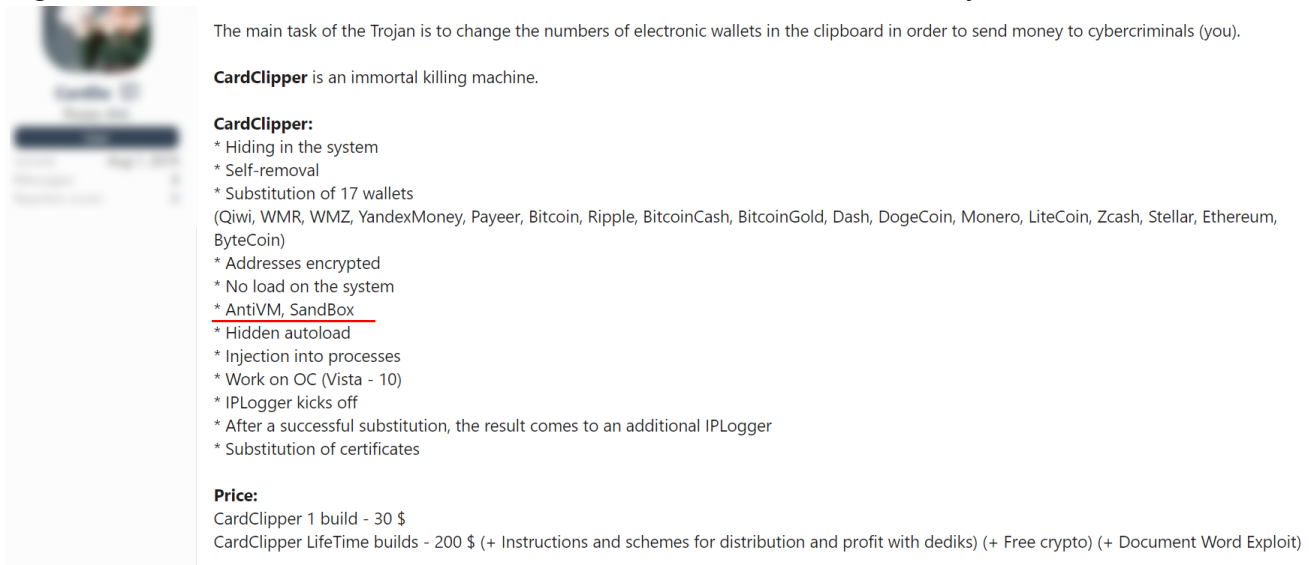
The image shows a screenshot of a dark web forum post. On the left is a user profile for 'Trojan 02'. The main text is a list of features for a malware loader. Two items are highlighted with a red box: '-Protection of sandboxes, vms, debuggers and other analysis' and '-Various other measures to harden reversing and analysis'. Below the list is a 'Pricing' section with 'Bot: 350\$' and 'Rebuild binary: 30\$'. In the top right corner, there is a 'Thread starter' label and a '#1' indicator.

[SALE] GACRUX Bot, resident loader

- Written in C++/C/ASM
- Bin size ~60kb
- No CRT; pure WINAPI.
- Full unicode support, compatibility windows 7+, x32/x64.
- Working in trusted process
- HTTP & HTTPS support (self signed cert work too)
- Communication encrypted
- Support for 5 C&C URLs
- Installation into system
- Hidden startup (Not visible to user, ever, or av products until shutdown)
- Small ring3 rootkit to hide all files of bot (x32/x64)
- Ability to download & execute, run local process, update bot, uninstall
- Tasks for running files support commandline parameters
- Protection of sandboxes, vms, debuggers and other analysis
- Various other measures to harden reversing and analysis
- Additional plugins such as stealer, backconnect.
- Plugins are executed in memory and never touch disk
- Secure C&C Panel with Captcha
- Filters for task execution (By country, location, OS, arch)
- Statistics

Pricing
Bot: 350\$
Rebuild binary: 30\$

Figure 5. Advertisement for a loader with built-in evasion functionality



The image shows a screenshot of a dark web forum post. On the left is a user profile for 'Trojan 02'. The main text describes the 'CardClipper' malware. It lists various features such as 'Hiding in the system', 'Self-removal', and 'Substitution of 17 wallets'. A 'Price' section at the bottom lists 'CardClipper 1 build - 30 \$' and 'CardClipper LifeTime builds - 200 \$ (+ Instructions and schemes for distribution and profit with dediks) (+ Free crypto) (+ Document Word Exploit)'. The text is formatted with bold and italicized words.

The main task of the Trojan is to change the numbers of electronic wallets in the clipboard in order to send money to cybercriminals (you).

CardClipper is an immortal killing machine.

CardClipper:

- * Hiding in the system
- * Self-removal
- * Substitution of 17 wallets (QIWI, WMR, WMZ, YandexMoney, Payeer, Bitcoin, Ripple, BitcoinCash, BitcoinGold, Dash, DogeCoin, Monero, Litecoin, Zcash, Stellar, Ethereum, ByteCoin)
- * Addresses encrypted
- * No load on the system
- * AntiVM, SandBox
- * Hidden autoload
- * Injection into processes
- * Work on OC (Vista - 10)
- * IPLogger kicks off
- * After a successful substitution, the result comes to an additional IPLogger
- * Substitution of certificates

Price:
CardClipper 1 build - 30 \$
CardClipper LifeTime builds - 200 \$ (+ Instructions and schemes for distribution and profit with dediks) (+ Free crypto) (+ Document Word Exploit)

Figure 6. Advertisement for services to protect malware from sandbox detection

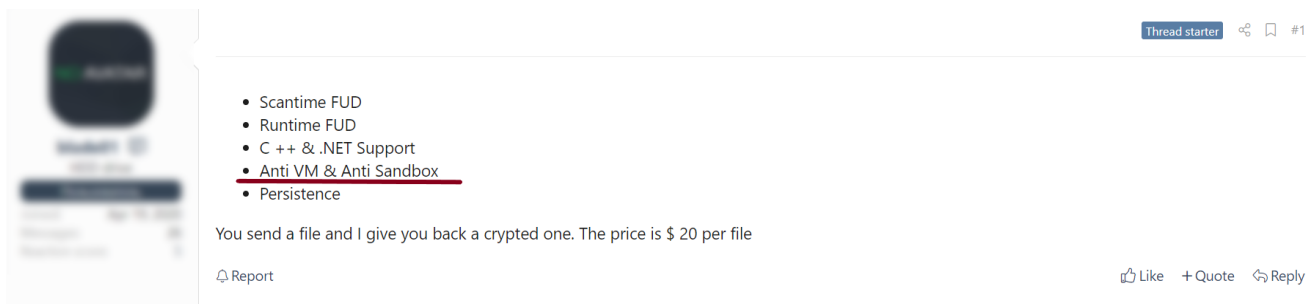


Figure 7. Advertisement for malware with built-in virtualization evasion functionality

Popular virtualization evasion techniques

Checking running processes



Used in EvilBunny, FinFisher, PlugX, Remcos, RogueRobin, Smoke Loader, PipeMon, Snake

One fifth of malware analyzes the list of running processes to detect a virtual environment. For example:

- The EvilBunny RAT continues execution only if at least 15 processes are running.
- PlugX (a backdoor widely used by APT groups over the last 10 years) checks if VMware Tools are running in background by searching for processes named "vmttoolsd".
- Remcos, used by Gorgon Group in phishing attacks against governments, searches for "vmttoolsd" and "vbox.exe" in the list of active processes.

```

Main(): void X
1 // Program
2 // Token: 0x06000001 RID: 1 RVA: 0x00002050 File Offset: 0x00000250
3 public static void Main()
4 {
5     Process[] processesByName = Process.GetProcessesByName("vmttoolsd");
6     if (processesByName.Length > 0)
7     {
8         Application.Exit();
9     }
10    else
11    {
12        Console.WriteLine("WARNING");
13        VH7kNabNuEHrGHdc8v.EHrGHDbNu();
14        VH7kNabNuEHrGHdc8v.EHrGHD();
15    }

```

Figure 8.

Searching for vmttoolsd (Remcos)

WMI queries



Used in BadPatch, Fin7, GravityRAT, OopsIE, Pony, RogueRobin

Since 2016, malware developers have been actively using WMI queries. Windows Management Instrumentation (WMI) is a technology for centralized management of Windows-based infrastructures. It allows access to devices, accounts, services, processes, network interfaces, and other programs. Of the malware in question, 25 percent makes use of them. In most cases, the attackers are trying to find out the model of hard drive or motherboard, as well as OS and BIOS versions.

GravityRAT uses an interesting method to detect virtual environments. By sending the `SELECT * FROM MSAcpi_ThermalZoneTemperature` WMI query, it checks the CPU temperature: if the malware is being run on a physical machine, the temperature value will be returned. But if the system responds with "ERROR" or "Not Supported," this means that the malware is running in a virtual environment.

```
C:\WINDOWS\system32>wmic /namespace:\\root\WMI path MSAcpi_ThermalZoneTemperature get CurrentTemperature
CurrentTemperature
3062
2732
2911
2922
2922
2932
2921
```

Figure 9. Output for the query `SELECT * FROM MSAcpi_ThermalZoneTemperature` on a physical machine

```
C:\Windows\system32>wmic /namespace:\\root\WMI path MSAcpi_ThermalZoneTemperature get CurrentTemperature
Node - DESKTOP-FU7K5UL
ERROR:
Description = Not supported
```

Figure 10. Output for the query `SELECT * FROM MSAcpi_ThermalZoneTemperature` in a virtual environment

WMI queries have also been used by the OilRig_group (APT34, Helix Kitten), which for more than five years has targeted a variety of industries, including government, finance, energy, and telecommunications, primarily in the Middle East. The group's backdoor OopsIE sends the WMI query `SELECT * FROM Win32_Fan` to check the state of the CPU fan. This query should return a class that provides statistics on the CPU fan. The backdoor checks whether the response is empty, which would indicate a virtual environment.

```

Select Administrator: Windows PowerShell
PS C:\Windows\system32> $q = "Select * from Win32_Fan"
PS C:\Windows\system32> Get-WmiObject -Query $q

__GENUS                : 2
__CLASS                : Win32_Fan
__SUPERCLASS           : CIM_Fan
__DYNASTY               : CIM_ManagedSystemElement
__RELPATH               : Win32_Fan.DeviceID="root\\cimv2 0"
__PROPERTY_COUNT       : 22
__DERIVATION            : {CIM_Fan, CIM_CoolingDevice, CIM_LogicalDevice, CIM_LogicalElement...}
__SERVER               :
__NAMESPACE            : root\cimv2
__PATH                 : \\.\root\cimv2:Win32_Fan.DeviceID="root\\cimv2 0"
ActiveCooling          : True
Availability            : 1
Caption                : Cooling Device
ConfigManagerErrorCode :
ConfigManagerUserConfig :
CreationClassName      : Win32_Fan
Description            : Cooling Device
DesiredSpeed           :
DeviceID               : root\cimv2 0
ErrorCleared           :
ErrorDescription       :
InstallDate            :
LastErrorCode          :
Name                   : Cooling Device
PNPDeviceID            :
PowerManagementCapabilities :
PowerManagementSupported :
Status                 : other
StatusInfo             : 2
SystemCreationClassName : Win32_ComputerSystem
SystemName              :
VariableSpeed          :
PSComputerName         :

```

Figure 11. Output for the query SELECT * FROM Win32_Fan on a physical machine

```

Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> $q = "Select * from Win32_Fan"
PS C:\Windows\system32> Get-WmiObject -Query $q
PS C:\Windows\system32>

```

Figure 12. Output for the query SELECT * FROM Win32_Fan in a virtual environment

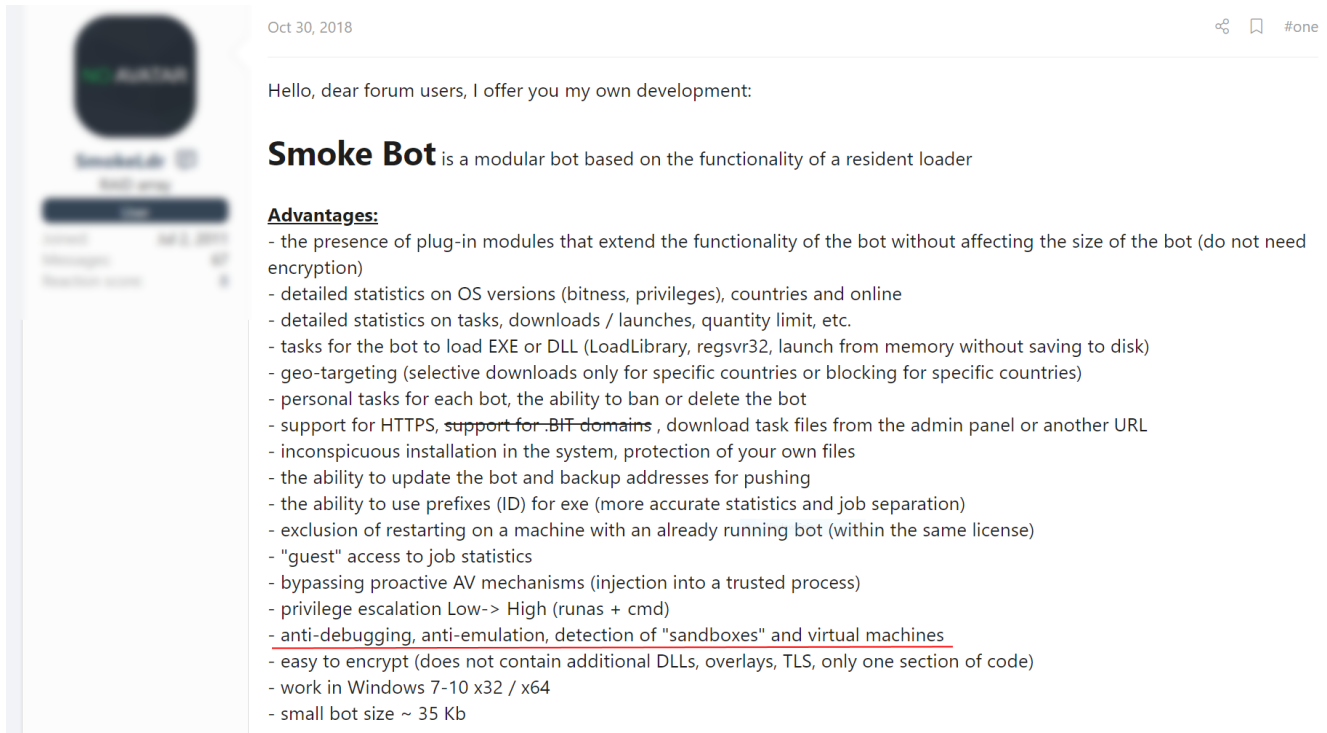
Registry key values checks



Used in CozyCar, Smoke Loader, FinFisher, GravityRAT, ROKRAT

Some malware (14 percent) reads registry key values and looks for substrings in them that suggest a virtual machine. For example:

The Smoke Loader banking trojan, used by TA505, checks registry key values in System\CurrentControlSet\Enum\IDE and System\CurrentControlSet\Enum\SCSI to search for substrings that match QEMU, VirtualBox, VMware, or Xen virtualization products. Smoke Loader (Smoke Bot) is offered for sale on the darkweb. The complete malware package costs \$1,650.



Oct 30, 2018 🔊 📌 #one

Hello, dear forum users, I offer you my own development:

Smoke Bot is a modular bot based on the functionality of a resident loader

Advantages:

- the presence of plug-in modules that extend the functionality of the bot without affecting the size of the bot (do not need encryption)
- detailed statistics on OS versions (bitness, privileges), countries and online
- detailed statistics on tasks, downloads / launches, quantity limit, etc.
- tasks for the bot to load EXE or DLL (LoadLibrary, regsvr32, launch from memory without saving to disk)
- geo-targeting (selective downloads only for specific countries or blocking for specific countries)
- personal tasks for each bot, the ability to ban or delete the bot
- support for HTTPS, ~~support for .BIT domains~~, download task files from the admin panel or another URL
- inconspicuous installation in the system, protection of your own files
- the ability to update the bot and backup addresses for pushing
- the ability to use prefixes (ID) for exe (more accurate statistics and job separation)
- exclusion of restarting on a machine with an already running bot (within the same license)
- "guest" access to job statistics
- bypassing proactive AV mechanisms (injection into a trusted process)
- privilege escalation Low-> High (runas + cmd)
- anti-debugging, anti-emulation, detection of "sandboxes" and virtual machines
- easy to encrypt (does not contain additional DLLs, overlays, TLS, only one section of code)
- work in Windows 7-10 x32 / x64
- small bot size ~ 35 Kb

Figure 13. Smoke Bot banking malware for sale

Cost:

- BOT - \$ 400
- STEALER - 100 \$
- FORM GRABBER - 300 \$
- PASS SNIFFER - 100 \$
- FAKE DNS - 100 \$
- DDOS - 200 \$
- HIDDEN TV - 150 \$
- KEYLOGGER- 100 \$
- PROCMON - 50 \$
- FILE SEARCH - 50 \$
- EMAIL GRABBER - 100 \$
- bot rebuild - 30 \$
- updates: minor fixes - free, the rest is discussed separately

Figure 14. Price of Smoke Bot banking malware on the darkweb

- FinFisher verifies that HKLM\SOFTWARE\Microsoft\Cryptography\MachineGuid does not equal "6ba1d002-21ed-4dbe-afb5-08cf8b81ca32"; HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\DigitalProductId does not equal "55274-649-6478953-23109", "A22-00001", or "47220", and that HARDWARE\Description\System\SystemBiosDate does not contain "01/02/03".

- CozyCar used by APT29, checks the registry key values in SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall for security products to avoid.

Other environment checks

In addition to checking running processes, registry key values, and sending WMI queries, malefactors can check the environment in other ways. For example, the RTM (Redaman) banking trojan checks for the following files and directories on C:\ and D:\ drives:

- cuckoo,
- fake_drive,
- perl,
- strawberry,
- targets.xls,
- tsl,
- wget.exe,
- *python*.

The existence of any of these files or directories indicates that the malware is running in a sandbox or a code analyzer.

APT37 (also known as ScarCruft, Group123, and TEMP.Reaper) has modified its ROKRAT backdoor over the last several years. In addition to checking registry key values, this malware also checks whether the file C:\Program Files\VMware\VMware Tools\vmtoolsd.exe exists and whether the following code analyzer and debugger DLLs have been loaded:

- SbieDll.dll,
- Dbghelp.dll,
- Api_log.dll,
- Dir_watch.dll.

Correctly configuring virtual machines is enough to stop the following attacker technique. PoetRAT, remote access malware, used in targeted attacks against ICS and SCADA systems in the energy sector, checks the hard disk size to determine whether it is running in a sandbox environment. Since the malware assumes that sandboxes have hard drives of less than 62 GB, it can be tricked by allocating more space for the virtual machine.

Does a sandbox have to detect all evasion techniques

Not all sandbox evasion methods are easy to detect. Some checks—such as file path, MAC address, date and time, and operation execution time—strongly resemble legitimate actions. Detection may generate a large number of false positives and interfere with proper functioning of other programs. This, however, does not mean that malware will remain totally invisible. Sandboxes do not have to catch each and every evasion technique, since malware

has many other attributes that can be detected at other stages of operation. That said, the more techniques the sandbox sees, the greater the chances of detecting new malware samples and applying this information to counter cyberthreats.

Anti-analysis and anti-debugging

To slip past antivirus programs for as long as possible, malefactors try to prevent analysis of malware by security professionals. They do so by using code obfuscation and anti-debugging techniques.

In 2019, the Remcos RAT added an anti-debugging method to its arsenal. If the loader detects a debugger in the system after calling the `IsDebuggerPresent` function, it displays the message "This is a third-party compiled Autolt script" and terminates execution.

The authors of FinFisher spyware went to great effort to obfuscate malicious code and impede analysis. For example, opcode `0x1A` should represent a `JB` (Jump if below) function, but is implemented through a set carry (`STC`) instruction followed by a `JMP` to the dispatcher code which will verify the carry flag condition set by `STC`.

```

pop     eax
lea    eax, [eax-1A0h]
mov    [edi+4AEDE7D5h], eax ; MOV [EDI+6], EAX
                                ; MOV EAX, 403f0a
mov    word ptr [edi+4AEDE7D9h], 0E0FFh ; JMP EAX
mov    al, [ecx] ; Load one byte from the opcode
mov    [edi+4AEDE7CFh], al ; Store the first byte
mov    eax, [ebx+2Ch] ; Load the previous ESP
lea    edi, [edi+4AEDE7CFh] ; Deobfuscate EDI
push   dword ptr [eax] ; PUSH the previous EFLAGS
popf   ; Load the flags
jmpf   edi ; Go to the extracted code
endp
; JMP  02 F8 B8 F9  <-- Placeholder
; MOV EAX, VmJbDispatcher
; JMP EAX

```

Generated code:

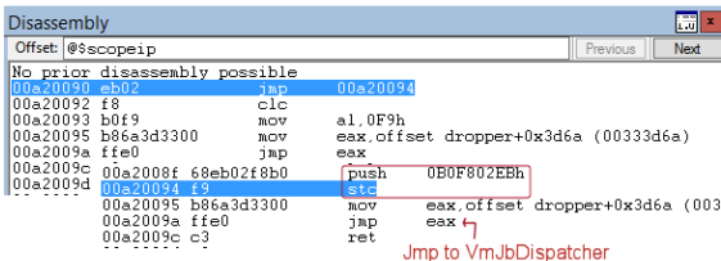


Figure 15. Example of an obfuscation technique (FinFisher)

To check for the step-through execution normally used by debuggers, EvilBunny calls `NtQuerySystemTime`, `GetSystemTimeAsFileTime`, and `GetTickCount`. It calls each function twice to calculate a delta and performs a sleep operation between the first and second calls. If any of the three deltas is below 998 milliseconds, execution will terminate.

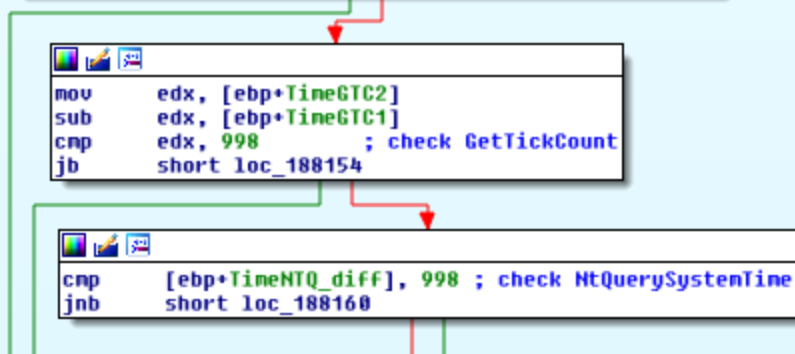
```

cnp     esi, esp
call   __RTC_CheckEsp
mov     esi, esp
lea    eax, [ebp+SystemTimeAsFileTime]
push   eax             ; lpSystemTimeAsFileTime
call   ds:GetSystemTimeAsFileTime
cnp     esi, esp
call   __RTC_CheckEsp
mov     esi, esp
call   ds:GetTickCount
cnp     esi, esp
call   __RTC_CheckEsp
mov     [ebp+TimeGTC1], eax
mov     esi, esp
push   1000           ; dwMilliSeconds
call   ds:Sleep
cnp     esi, esp
call   __RTC_CheckEsp
mov     esi, esp
lea    ecx, [ebp+TimeNTQ2]
push   ecx
call   [ebp+NtQuerySystemTime]
cnp     esi, esp
call   __RTC_CheckEsp
mov     esi, esp
lea    edx, [ebp+var_450]
push   edx             ; lpSystemTimeAsFileTime
call   ds:GetSystemTimeAsFileTime
cnp     esi, esp
call   __RTC_CheckEsp
mov     esi, esp
call   ds:GetTickCount
cnp     esi, esp
call   __RTC_CheckEsp
mov     [ebp+TimeGTC2], eax
mov     eax, [ebp+TimeNTQ2]
sub    eax, [ebp+TimeNTQ1]
xor    edx, edx
mov    ecx, 10000
div    ecx
mov    [ebp+TimeNTQ_diff], eax
mov    eax, [ebp+var_450.dwLowDateTime]
sub    eax, [ebp+SystemTimeAsFileTime.dwLowDateTime]
xor    edx, edx
mov    ecx, 10000
div    ecx
cnp    eax, 998       ; check GetSystemTimeAsFileTime
jb     short loc_188154

```

C
C
C
M
C
J

Figure 16. Double calling of



functions (EvilBunny)

It is getting more and more difficult to perform static analysis of malware and identify attributes of malicious files by matching them with signatures and hash sums. That is why, in addition to static analysis, we recommend running suspicious files in a virtual environment to analyze their behavior.

Conclusion

Attackers constantly modify their malware to evade detection for as long as possible. In this respect, APT groups do especially well. To collect information about the victim's infrastructure, attackers prefer using malware with built-in functionality for detecting and evading virtual machines and code analyzers. In addition, loaders and remote access tools sold on the darkweb have built-in basic sandbox evasion functions, or at least this is what their sellers claim.

In recent years, malware authors have been trying especially hard to evade code analyzers. Hackers do all they can to hide malicious functionality from security researchers and minimize the likelihood of detection of malware based on known indicators of compromise. Traditional defenses may not be able to detect malicious programs. To detect modern malware, we recommend analyzing file behavior in a secure virtual environment. By using a sandbox, you also enrich your IOC database and can leverage this information to better respond to cyberthreats. By updating all your protection tools with the latest IOCs, you can detect even new malware versions if hackers attempt a second attack on your infrastructure. For example, if attackers compile a new malware version but forget to change the command and control (C2) address, the newer malware will still be detected because of the identical address.

Sandboxes have already learned to thwart the majority of popular evasion techniques. Even if hackers use methods that resemble legitimate processes, such as checking the current date and time, malware will most likely reveal itself by other signs a sandbox will be able to detect. Hackers constantly refine their tools, change sandbox detection techniques, and use multiple techniques at the same time. In parallel, sandboxes must be flexible enough and easily adapt to new challenges by imitating a real workstation. A sandbox must hide its presence well in order to prevent malware from terminating early and successfully collect indicators of compromise.