Table of Contents

[Consolidation of Various Anti-Techniques 2](#__RefHeading__149_1539073869)

[Anti-Virtualization Techniques 2](#__RefHeading__160_1539073869)

[VirtualBox 4.1.12 2](#__RefHeading__153_1539073869)

[VMWare Workstation 9.0.0 build-812388 2](#__RefHeading__155_1539073869)

[QEMU 5](#__RefHeading__162_1539073869)

[VirtualPC (Still working on it) 5](#__RefHeading__164_1539073869)

[Anti-Debugging Techniques 6](#__RefHeading__166_1539073869)

[Windows API 6](#__RefHeading__274_1539073869)

[Deeper Check 6](#__RefHeading__276_1539073869)

[Checking for certain strings in Malware 7](#__RefHeading__278_1539073869)

[Checking for Open Windows or Running processes (FindWindows) 7](#__RefHeading__280_1539073869)

[Anti-Monitoring Techniques 8](#__RefHeading__309_1539073869)

[Check for running process 8](#__RefHeading__311_1539073869)

[Anti-Sandbox Techniques 8](#__RefHeading__313_1539073869)

[Check for Registry 8](#__RefHeading__315_1539073869)

[Check for dll 8](#__RefHeading__317_1539073869)

[Anti-Virtualization Techniques 2](#__RefHeading__149_1539073869)

[VirtualBox 4.1.12 2](#__RefHeading__149_1539073869)

[VMWare Workstation 9.0.0 build-812388 3](#__RefHeading__149_1539073869)

[QEMU 5](#__RefHeading__149_1539073869)

[VirtualPC (Still working on it) 5](#__RefHeading__149_1539073869)

[Anti-Debugging Techniques 6](#__RefHeading__149_1539073869)

[Windows API 6](#__RefHeading__149_1539073869)

[Deeper Check 6](#__RefHeading__149_1539073869)

[Checking for certain strings in Malware 7](#__RefHeading__149_1539073869)

[Checking for Open Windows or Running processes (FindWindows) 7](#__RefHeading__149_1539073869)

[Anti-Monitoring Techniques 8](#__RefHeading__149_1539073869)

[Check for running process 8](#__RefHeading__149_1539073869)

[Anti-Sandbox Techniques 8](#__RefHeading__149_1539073869)

[Check for Registry 8](#__RefHeading__149_1539073869)

[Check for dll 8](#__RefHeading__149_1539073869)

# Consolidation of Various Anti-Techniques

## Anti-Virtualization Techniques

### **VirtualBox 4.1.12**

|  |  |
| --- | --- |
| HKLM\HARDWARE\DEVICEMAP\Scsi\Scsi Port 0\Scsi Bus 0\Target Id 0\Logical Unit Id 0 | **Identifier:** VBOX HARDDISK |
| HKLM\HARDWARE\Description\System | **SystemBiosSystem:**  VBOX - 1  **VideoBiosSystem:**  Oracle VM VirtualBox Version 4.1.12\_Ubuntu VGA BIOS  Oracle VM VirtualBox Version 4.1.12\_Ubuntu VGA BIOS  Oracle VM VirtualBox Version 4.1.12\_Ubuntu  Oracle VM VirtualBox |
| HKLM\SOFTWARE\Oracle\VirtualBox Guest Addition | Exists = VirtualBox |
| HKLM\SYSTEM\CurrentControlSet\Services  HKLM\SYSTEM\ControlSet001\Services | VBoxGuest  VBoxMouse  VBoxSF  VBoxService |
| HKLM\SYSTEM\CurrentControlSet\Services\Disk\Enum  HKLM\SYSTEM\ControlSet001\Services\Disk\Enum | **0:**  DiskVBOX\_HARDDISK |
| C:\Windows\system32\drivers | VBoxMouse.sys  VBoxGuest.sys  VBoxSF.sys |
| C:\Windows\system32\ | VBoxService.exe |
| [C:\Program](file:///../C:/Program) Files\Oracle\VirtualBox Guest Additions | VBoxControl.exe  VBoxTray.exe  VBoxVideo.sys |
| Running Processes | VBoxService.exe  VBoxTray.exe |
| MAC Address | 080027, 08:00:27, 08-00-27 |

### **VMWare Workstation 9.0.0 build-812388**

|  |  |
| --- | --- |
| HKLM\HARDWARE\DEVICEMAP\Scsi\Scsi Port \*x\*\Scsi Bus 0\Target Id 0\Logical Unit Id 0 | **(Scsi Port 1) Identifier:**  NECVMWar VMware IDE CDR10  **(Scsi Port 2) Identifier:**  VMware, VMware Virtual S1.0 |
| HKLM\SOFTWARE\VMWare\VMTools | Exists = VMWare |
| HKLM\SYSTEM\CurrentControlSet\Services | Vmci  vmhgfs  VMMEMCTL  vmmouse  VMSCSI  VMTools  vmusbmouse  VMware Physical Disk Helper Service  vmx\_svga  vmxnet |
| HKLM\SYSTEM\CurrentControlSet\Services\Disk\Enum  HKLM\SYSTEM\ControlSet001\Services\Disk\Enum | **0:**  \*Ven\_VMware\_&Prod\_VMware\_Virtual\* |
| HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Reinstall\0000] | **DisplayName:**  VMware Pointing Device  **Mgf:**  VMware, Inc.  **ProviderName:**  VMware, Inc.  **ReinstallString:**  C:\WINDOWS\system32\ReinstallBackups\0000\DriverFiles\vmmouse.inf |
| C:\Windows\system32\drivers | vmxnet.sys  vmx\_svga.sys  vmusbmouse.sys  vmscsi.sys  vmmouse.sys  vmhgfs.sys  vmci.sys |
| [C:\Program](file:///../C:/Program) Files\VMware\VMware Tools\ | vmacthlp.exe vmtoolsd.exe |
| C:\Program Files\Common Files\VMware\Drivers\memctl | vmmemctl.sys |
| Running Processes | Vmacthlp.exe  vmtoolsd.exe |
| SIDT, SLDT, SGDT, STR, SMSW | Significant different from host machine. Addresses are at different position (one at the top, one at the bottom)  (See redpill.c, checkvm.c, vmdetects) |
| IN Instruction | VMWare uses IN instruction in special port (VX). It is supposed to generate an exception in host machine.  The malware can contain this snippet:  push ebx  mov eax, 0x564D5868  mov ebx, 0x8685D465  mov ecx, 0x0A  mov dx, 0x5658  in eax, dx  mov \_EBX, ebx  pop ebx  If \_EBX == 0x564D5868, means this is in VMWare. |
| MAC Addresses | 00-05-69  00:05:69  000569  00-0C-29  00:0c:29  000c29  00-1C-14  00:1C:14  001C14  08-00-27  08:00:27  080027 |

### QEMU

|  |  |
| --- | --- |
| HKLM\HARDWARE\DEVICEMAP\Scsi\Scsi Port \*x\*\Scsi Bus 0\Target Id 0\Logical Unit Id 0 | **(Scsi Port 1) Identifier:**  QEMU QEMU DVD-ROM |

### VirtualPC (Still working on it)

|  |  |
| --- | --- |
| Invalid Instruction | This ““0x0F 0x3F 0x07 0x0B” can be used to detect VirtualPC. This pair of instruction does not generate an exception in VirtualPC. |

## Anti-Debugging Techniques

It is important to note that some of the anti-debugging techniques here might generate a false positive. Some of the techniques are not meant to use to detect whether a debugger exists, but more of a workaround to check whether a debugger is attached to the process.

### Windows API

1. **IsDebuggerPresent**[[1]](#footnote-2)**:** Search Process Environment Block. Return 1 if the process is being debugged.
2. **CheckRemoteDebuggerPresent:** Same as above.
3. **NtQueryInformationProcess:** Return the information about a given process. Can use this API to check for “ProcessDebugPort” to check whether the current Process is being debugged.
4. **NtQuerySystemInformation**: Can be used to check whether kernel-mode debugger exists by checking “SystemKernelDebuggerInformation” with certain flags.
5. **OutputDebugString:** Malware can make use of this API and “GetLastError()” to check whether a debugger is attached to the process.
6. **CreateProcessA, CreateProcessW, WaitForDebugEvent**[[2]](#footnote-3)**:** A process can spawn a child process to debug its parent process. This will prevent any other debuggers to attach to the parent process.
7. **UnhandledExceptionFilter**[[3]](#footnote-4)**:** Will not proceed if a debugger is attached. Hence, the process will exit.
8. **OpenProcess, SeDebugPrivilege:** OllyDbg and WinDbg requires SeDebugPrivilege. It will take control over CSRSS.exe. If we can open CSRSS.exe, this means that the process is attached to the debugger.
9. **BlockInput()**: Malware writer may use this API to block keyboard and mouse from reaching the application. Hence, we can see whether BlockInput() API exists in the malware.
10. **CreateDesktopA, CreateDesktopW, SwitchDesktop:** Malware writer may create many desktops for the process to run. This will make the debugging process harder.
11. **FindWindows:** Checking for OllyDbg, WinDbg or so.

### Deeper Check

1. **BeingDebugged** (Used by PEB): Offset 2 from PEB, which can be pushed or move into EAX. Compare the offset value with 0 or 1.
2. **PEB NtGlobalFlag:** Offset 0x68 in PEB. Because there are more flags created when the process is run with Debugger. So the value at this location will be different if the process is running with debugger. The value will be 0x70.
3. **PEB ProcessHeap, ForceFlag Field**: Offset 0x18 in PEB. Contains a header to check whether the heap is created within a debugger. Offset 0x10 (Windows XP) from the field.

### Checking for certain strings in Malware

1. **dbghelp.dll:** Microsoft Debugger DLL
2. **syser.sys**: Syser
3. **hanolly.sys:** Use by Themida to check for OllyDbg

### Checking for Open Windows or Running processes (FindWindows)

1. ollydbg.exe
2. IDAPro
3. WinDbg
4. SoftICE
5. SyserApp.exe

## Anti-Monitoring Techniques

### Check for running process

1. procmon.exe
2. procexp.exe
3. autoruns.exe

## Anti-Sandbox Techniques

### Check for Registry

1. HKLM\Software\Microsoft\Windows\CurrentVersion
   * 76487-644-3177037-23510
   * 55274-640-2673064-23950
   * 76487-337-8429955-22614

### Check for dll

1. Sbiedll.dll (Sandboxie)

1. http://msdn.microsoft.com/en-us/library/windows/desktop/ms680345%28v=vs.85%29.aspx [↑](#footnote-ref-2)
2. http://www.codeproject.com/Articles/30815/An-Anti-Reverse-Engineering-Guide#SelfDebugging [↑](#footnote-ref-3)
3. http://evilcodecave.wordpress.com/2008/07/24/setunhandledexception-filter-anti-debug-trick/ [↑](#footnote-ref-4)