

Forum:[Hack Forums](#)

Thread: [C# AntiVM](#)

Kryptc - June 9, 2020, 4:22 pm

What would be a good way to detect a VM and/or sandboxie? Maybe even VT scans if possible?

Accord - June 9, 2020, 7:42 pm

This is a good read if you want to know the technique,

<https://www.cyberbit.com/blog/endpoint-security/anti-vm-and-anti-sandbox-explained/>

VisualSoft - June 10, 2020, 12:22 am

(Jun 9, 2020 07:42 PM)Accord Wrote: [\[CITE\]](#) This is a good read if you want to know the technique, <https://www.cyberbit.com/blog/endpoint-security/anti-vm-and-anti-sandbox-explained/> nice find, hop op can make working source from it.

__Digital__ - June 10, 2020, 8:11 am

(Jun 9, 2020 04:22 PM)Kryptc Wrote: [\[CITE\]](#) What would be a good way to detect a VM and/or sandboxie? Maybe even VT scans if possible?
check this out and don't forget to call it in your program

<https://pastebin.com/zAmFnNui>

If virtual machine detected you can also tell it to do something.

Kryptc - June 10, 2020, 8:42 am

Ill check it out and lyk!

__Digital__ - June 10, 2020, 8:49 am

(Jun 10, 2020 08:42 AM)Kryptc Wrote: [\[CITE\]](#) Ill check it out and lyk!
if you need any help feel free to drop me a pm.

Kryptc - June 10, 2020, 10:04 am

(Jun 10, 2020 08:49 AM)__Digital__ Wrote: [\[CITE\]](#) if you need any help feel free to drop me a pm.
Will do!

Grooty - May 9, 2021, 3:30 pm

Found on Russian-language forums

Code namespace AntiVM

```
{
    using System;
    using System.Runtime.InteropServices;

    internal static class NativeMethods
    {
        [DllImport("kernel32.dll", BestFitMapping = false, CharSet = CharSet.Unicode)]
        internal static extern IntPtr GetModuleHandle(string lpModuleName);
    }
}
```

```

namespace AntiVM
{
    using System;
    using System.Collections.Generic;
    using System.Diagnostics;
    using System.Linq;
    using System.Management;
    using System.Windows.Forms;

    public static class CheckVirtual
    {
        /* Created r3xq1 */

        /// <summary>
        /// <b>????? ??? ?????????? ?? ?????? ? ?????? RDP</b> (Remote Desktop Protocol)
        /// </summary>
        public static bool IsRdpAvailable => SystemInformation.TerminalServerSession == true;

        /// <summary>
        /// <b>????? ??? ?????????? ?????????? ? ???????????.</b>
        /// <para>????? ?????????????? ??? ???? ???? ???? ???? ???? ???? ???? ???? ????
        <b>true</b> ???? <b>false</b></para>
        /// </summary>
        /// <returns><b>true</b> ??? <b>false</b></returns>
        public static bool SandBoxes => Process.GetProcessesByName("SbieCtrl").Length > 0 &&
        NativeMethods.GetModuleHandle("SbieDll.dll") != IntPtr.Zero;

        /// <summary>
        /// <b>????? ??? ?????????? VMware. Inc ?? ?????? ???????????????.</b>
        /// <para>????????????? ?? ?????? <b>VirtualNames</b> ? ??? ???? ???? ???? ???? ????
        ?????????? <b>true</b> ???? <b>false</b></para>
        /// </summary>
        /// <returns><b>true</b> ??? <b>false</b></returns>
        public static bool CheckWMI()
        {
            var VirtualNames = new List<string>
            {
                "virtual", "vmbox", "vmware", "virtualbox", "box",
                "thinapp", "VMXh", "innotek gmbh", "tpvcgateway",
                "tpautoconnsvc", "vbox", "kvm", "red hat"
            };
            List<string> list = GetModelsAndManufactures();
            foreach (string spisok in list)
            {
                return VirtualNames.Contains(spisok); // true
            }
            return false;
        }

        /// <summary>
        /// <b>????? ??? ?????????? ?????????? ? ?????????????? ? ?????? ?????? ?? ??????
        ???????????????.</b>

```

```

/// </summary>
/// <returns>?????????? ?????? ? ??????????????? ???????????.</returns>
private static List<string> GetModelsAndManufactures()
{
    var ModMan = new List<string>();
    try
    {
        using var searcher = new ManagementObjectSearcher(@"root\CIMV2", "SELECT *
FROM Win32_ComputerSystem");
        using var items = searcher.Get().OfType<ManagementObject>().Where(p => p !=
null).FirstOrDefault();
        ModMan.Add(items["Manufacturer"]?.ToString().ToLower());
        ModMan.Add(items["Model"]?.ToString().ToLower());
    }
    catch { }
    return ModMan;
}

/// <summary>
/// <b>?????????????? ?????????? ???? ????????? ?? ????????? ????????? ? ?????? VMware</b>
(Virtual Machine)
/// </summary>
/// <returns></returns>
public static bool Inizialize() => SandBoxies || IsRdpAvailable || CheckWMI();
}
}

namespace AntiVM
{
    using System;

    internal static class Program
    {
        [STAThread]
        public static void Main()
        {
            Console.Title = "Anti Virtual Machine - (VM)";

            if (CheckVirtual.Inizialize())
            {
                Console.WriteLine("????????????? ?????????????? ????????");
            }
            else
            {
                Console.WriteLine("?? ??????????? ?? ????????? ????????");
            }

            Console.Read();
        }
    }
}

```

check mac address (vmare/virtual-box always use same MAC), check for debugger, check to see if CD/DVD drive is physically installed (vms dont), check HDD size 20+gb, and check for virtualbox related processes