

My Big Brother Microsoft

by Randy Charles Morin

If you watched the movie “Net” you’ll remember the premise of the movie was that a large software-giant had malicious code that could be used to comprise the computers where the code is installed. It has always been rumored that Microsoft has been using similar techniques in order to maintain their monopoly. The technique is called tentacling, that is, Microsoft is using the installed software on your system to control your system remotely. This belief led to the current implementation of Microsoft Windows Update.

“This is done without sending any information to Microsoft.”

-Product Updates @ Windows Update

If you read the about Windows Update text, then you’ll note that Microsoft is attempting to reassure the Internet community that it is not tentacling.

“This scan is done on your computer to ensure the safety and privacy of your system information. None of this information is sent to Microsoft or over the Internet.”

-How does it work @ about Windows Update

This article describes some information that I have accumulated that proves that Microsoft is scanning your hard drive when you perform certain tasks.

In order for the reader to following the article, his level of expertise with a computer will have to be quite high. In the experimenting, I’ll be using three tools to verify what Microsoft’s product is doing.

- Network Monitor
- Registry Monitor
- File Monitor

Let me begin by describing these three tools.

Note

Everything in this article assumes you are using Windows 2000 Professional with SP2 installed.

NetMon

The first tool, the Network Monitor or NetMon for short, was produced by Microsoft and is available in different forms on various Microsoft products. The best redistributed-form of this tool (IMHO) is the one found on the Microsoft System Management Server 2.0 Service Pack 3.

When you enter the SMS CD, it will prompt you to install SMS. Ignore that and kill SMS setup. Rather, if you navigate the CD in Windows Explorer, you will see a folder called NMEXT. Find the appropriate setup.exe file and run this setup.

When the installation is complete, you can run the tool from Administrative Tools folder found in the Control Panel.

RegMon

A couple programmers named Mark Russinovich and Bryce Cogswell produce the second and third tools. They have a website <http://www.sysinternals.com/> that has some of the best tools available to programmers and hackers alike. I won't describe all their tools as they have produced quite an abundance of good tools, but I will describe my two favorites, RegMon and FileMon.

RegMon or the Registry Monitor captures all traffic involving the Windows Registry. This tool can prove very valuable to those writing software installation products and utilities. It can also provide invaluable insight into how a particular program might be using the Windows Registry.

You can download RegMon from <http://www.sysinternals.com/ntw2k/source/regmon.shtml>. There is no installation or setup file. You'll have to unzip the contents of this package into a separate folder and create your own shortcut to the REGMON.EXE executable file.

FileMon

The second tool from SysInternals is FileMon. This tool, very similar to RegMon, monitors all traffic involving your hard drive. Whenever someone reads-write a directory or file, the traffic is logged by the tool.

You can download FileMon from <http://www.sysinternals.com/ntw2k/source/filemon.shtml>. Again there is no installation or setup file. You unzip the contents of the package into a folder and run the FILEMON.EXE file from there.

Get familiar with these tools as you read this article.

Unbearable Waiting

What first clued me into Microsoft's hard disk scanning was when a friend of mine told me that the SOAP Toolkit 2.0 installation package was taking a very long time. If you double-clicked the setup file, then you waited a varying but long amount of time before the install wizard prompted for input from the user.

You can download the installation package from <http://download.microsoft.com/download/xml/soap/2.0/W98NT42KMe/EN-US/SoapToolkit20.exe>. It is highly likely that this toolkit won't exist there for that long. If it's not there, then try <http://msdn.microsoft.com/soap/>. I thought maybe Microsoft was up to something, but I didn't really take the time to find out what.

Then I read the following post on the Microsoft SOAP newsgroup.

From: [yufu](#)

Subject: SOAP Installataion

Newsgroups: microsoft.public.xml.soapsdk

Date: 2001-10-30 04:34:12 PST

Does anyone know why that stupid SOAPSDK SP2 take so much time to install, I thought it had been dead!

I had two reasons to believe something was at fault. So I immediately sat down at my computer and ran the SOAP Toolkit installer. Nothing. I waited for 5 to 10 minutes and nothing happened. Then the installation wizard popped up. Weird. What was going on?

I ran the installer again. This time I notice two dialogs appears and disappeared quite quickly telling me that the Installer Package was loading. But in no time (couple of seconds), the system returned to nothingness. This time, I opened the clock and recorded when the installer was started. One minute passed and nothing. Two minutes passed and nothing. Three. Four. Five. Six. Seven. Eight.

Eight minutes and forty-five seconds after I started the installer, the installation wizard popped up asking me if I wanted to install the SOAP Toolkit on my computer. Very weird. What was the installer doing for those eight minutes?

In my next test, I started the task manager first to determine what the task activity level was. I started the soaptoolkit20.exe installer and immediately this process appeared in the process list. It also launched a secondary process WScript.exe. Normally, I would kill any WScript.exe processes that run on my computer. Many a virus has been based on this nasty interpreter.

WScript for those that don't know is the command-interpreter for the Visual Basic Script language. When you run files with the .vbs extension, this interpreter reads the file, interprets the code and executes the code.

In this case, I let the WScript run as I was quite confident that it was just some script file used to launch the installation of the SOAP Toolkit 2.0. But even with that confidence, I took the time to open the soaptoolkit20.exe executable file to visit its contents.

You might ask how you could read a self-installing package. Well, open the file with WinZip. WinZip is quite intelligent enough to realize the package is an archive of other files. So it just opens the package and displays the list of included files.

- soapsdk.msi
- setup.vbs

There's your vbs file.

Listing Error! Bookmark not defined.: setup.vbs

```
' For use with Windows Scripting Host, CScript.exe or WScript.exe
' Copyright (c) 2001, Microsoft Corporation
'
Option Explicit
Const productCode = "{36BEAD11-8577-49AD-9250-E06A50AE87B0}" 'The ProductCode for the
Microsoft SOAP Toolkit
Const msiInstallStateUnknown = -1 'The product is neither advertised nor installed.
Const msiUILevelFull = 5 'Authored UI with wizards, progress, and errors.

On Error Resume Next
```

```
'Connect to Windows Installer object
Dim installer : Set installer = Nothing
Set installer = Wscript.CreateObject("WindowsInstaller.Installer")

If Err <> 0 Then
    Wscript.Echo "Could not load the Windows Installer engine. Please make sure you have
the Windows Installer installed on your computer."
End If

'Set UI Level
installer.UILevel = msiUILevelFull

'Logic to use the correct command line arguments
If installer.ProductState(productCode) <> msiInstallStateUnknown Then
    If LCase(installer.ProductInfo(productCode, "ProductName")) = LCase("Microsoft SOAP
Toolkit 2.0") Then
        'RTM version installed - execute Small Upgrade
        installer.InstallProduct "soapsdk.msi",
"REINSTALL=SDKFeature,SMOGFeature,DocumentationFeature,WSDLGenFeature,ISAPIFeature,DebugU
tilFeature,CPPSupportFeature,MSXMLFeature REINSTALLMODE=vomus"
    Else
        If LCase(installer.ProductInfo(productCode, "ProductName")) = LCase("Microsoft SOAP
Toolkit 2.0 SP2") Then
            'SP2 version installed - execute Maintenance mode
            installer.InstallProduct "soapsdk.msi"
        Else
            'Neither RTM nor SP2 - unsupported version and needs to be un-installed before
installing SP2
            Wscript.Echo "The installed version of the Microsoft SOAP Toolkit can not be
upgraded to Service Pack 2." &_
vbNewLine & vbNewLine & " Please un-installed the existing version from
Add/Remove Programs on Control Panel" &_
vbNewLine & " before installing the Microsoft SOAP Toolkit 2.0 SP2 release."
        End If
    End If
Else
    'Neither RTM nor SP2 - execute install for a clean installation
    installer.InstallProduct "soapsdk.msi", "ADDLOCAL=ALL"
End If
```

I quick look reveals that the script is harmless. It starts the windows installer component and runs the installation package, soapsdk.msi.

Now back to the unbearable waiting. You might want to know what the Task Manager was reporting. The soaptoolkit20.exe process was running at 0% CPU. It was delegating to the WScript. The WScript.exe process was idling (near idle) between 0 and 15%. It sat there for the entire eight minutes. A little CPU here, a little CPU there, but always close to 0% CPU usage.

Monitor

The next step was to monitor the network, file and registry activity during this idling stage. I started the three tools mentioned above and captured enormous amounts of data as the toolkit installation ran.

I made the mistake the first time of running the toolkit without first disabling all other processes that generate similar traffic. This made the output of the monitor quite difficult to read. I was in a hurry, so I didn't really take the time to perform a more controlled test.

In a brief overview of the output, I quickly determined that the installation package was navigating every folder on my hard drive, system folders and private folders alike. Now

this caught my attention. We had source code in those folders. What was Microsoft doing reading those folders? Clearly they had little to do with installing soap.

At this point, I actually started to believe that Microsoft was stupid enough to use such unsophisticated techniques. I still really only had a base level of proof that wouldn't stand on its own. So, I decided that I would respond to Yufu's posting and send a copy to a couple of Microsoft employees.

From: Randy Charles Morin (rmorin@kbcafe.com)
Subject: Re: SOAP Installataion
Newsgroups: microsoft.public.xml.soapsdk
Date: 2001-10-30 18:02:47 PST

I looked into this and found some really fascinating stuff. The install launches a WScript (pretty normal), which calls the install component. It would seem that on some installations (I have no idea which), the WScript process begins idle-ing at 0-15% CPU usage. This can last a real long time.

So what is it doing? Reading my hard drive.

I ran FileMon and found out that the process was scanning files all over my hard disk, including my temporary Internet files and files located in "My Documents" (obviously personal stuff). I couldn't track down what the script was trying to do with all the information it was accumulating.

I also found that it was repeatedly scanning invalid locations in my registry (used RegMon). Kind of interesting. I also saw some packets go out to Microsoft (used NetMon). Hmm! The packets had an interesting signature, they said that Microsoft was sending non-personal information to themselves in order to better help me and that I could turn it off in some setting on my computer. But they were reading files in "My Documents" including my source code.

I'm not saying that Microsoft is doing anything here. I didn't see anything go out the wire that shouldn't have. I'm just reporting what I saw. I'll run some offline test to see what else I can figure out. Very well could be some feature went wild and a lot of cowinkidink (coincidence).

I'd like to hear from Roger Wolter and Tammara Combs on this one, so I CCed them.

--

Randy Charles Morin

The post went unanswered, which is not normal. Roger Wolter answers pretty much every reasonable post in the newsgroups and Tammara Combs has indicated to me on previous occasions that I can best get an answer by addressing these questions directly to her.

Two weeks passed and no answer in the newsgroup, neither did I get a direct answer from Roger or Tammara. I had given them enough time to respond. It was time for me to do a bit more research into this anomaly.

FileMon Revisited

This time I ran the test again, but with only two applications running side-by-side, FileMon and the SOAP Toolkit installer. I needed to find out exactly what the installer was reading.

Master File Table

I had an NTFS hard drive, so most of the file reads were from a file called c:\\$mft.

This file is the NTFS master file table. Similar to a File Allocation Table (FAT), but different ☺

My discovery was quite distressing. It seems that the installer was tree scanning every single folder on my computer. For what reason, I do not know. I revisited the FileMon logs over and over and couldn't make anything of them. Surely Microsoft isn't interested in your directory structure? Scanning folders is one thing, but unless you scan the contents of files, then you aren't really acquiring anything except filenames and directory structures.

I took another look. Another distressing fact. The installer is tree scanning the entire directory structure on my computer three times over. Wow! During the first scan, the installer seems to be searching for a file called MSSOAP1.DLL. During the second scan, the installer seems to be searching for a file called MSSOAPR.DLL. During the third scan, the installer seems to be searching for a file called WISC10.DLL.

RegMon

I followed the FileMon test with a comprehensive RegMon test. Running only the two applications side-by-side. This test revealed nothing out of the ordinary. A few accesses here and there in the Registry, but nothing I wouldn't expect from the install package.

NetMon

The last test would be a comprehensive NetMon test. Running again, just the two applications, NetMon and the SOAP installer side-by-side. Nothing. There was actually no network activity at all.

If you run the installer directly from Microsoft's site, then you get a lot of initial activity as the install package is downloaded over the Internet. But if you download the package first and then run it from a local folder, the network traffic is nil.

Conclusion

So, what is the conclusion? Definitely, no direct evidence that Microsoft is sending any information from my computer to any destination. But why would the installer be scanning all the folders on my hard drive? Three times over?

The answer is most likely a case of a really bad installation routine.

I actually ran these tests on a variety of computers. Your typical server or freshly installed desktop machine would delay only a few seconds between the time the installation was started and the installation wizard appeared. This was because the amount of folders on the machine was limited to a few dozen. But with a mature

installation of Windows 2000, you might have a few hundred folders. And as I am one of the biggest geeks, I have a few hundred too many folders. I imagine the anonymous Yufu is in the same boat as I.