Determine which system is attacked by Windows commands

In this two-part tutorial, the first part will cover five useful command-line tools in Windows to perform such an analysis.

Network administration - Windows computers are the most hacked computers. That's why Microsoft has built a lot of tools in Windows operating systems for administrators and some users to analyze to determine if their computers are compromised. In this two-part tutorial, the first part will cover five useful command-line tools in Windows to perform such an analysis.

1. WMIC

Windows Management Instrumentation Command-line (WMIC) is not merely a command but has many other features. This tool has a command line interface for Windows Management Instrumentation API inside Windows. WMIC allows users to access detailed information on Windows computers, including detailed properties of thousands of settings and objects. WMIC is built into Windows XP Professional, Windows 2003 and Windows Vista. To use it, the user must launch the program by running the WMIC command, followed by the part that the user is interested in (often referred to as aliases within the system). For example, to know the processes running on the computer, the user can run the command:

C:> wmic process

The output part of this command seems to be quite difficult to read because the format is not specified. However with WMIC, the output that this tool provides is a completely different format, in which the " **full list** " section displays detailed information for each area of ??interest to the user, while the " **list brief** " section. "will provide an output stream for each report item under the list of items, such as running processes, autostart programs and existing shares.

For example, we can observe every process running on the computer by running the command:

C:> wmic process list brief

The above command will display the name, process ID and priority of each running process as well as other attributes. To get more detailed information, run the command:

C:> wmic process list full

This command displays all the details, including the executable file path associated with its process and command-line invocation. When studying whether a computer is infected or not, the administrator needs to consider each process to determine if the processes are valid on the computer, then investigate the strange process or not. Expect using search engines.

In addition to the process aliases, users can replace the startup to get a list of automatically launched programs on the computer, including programs that launch when the system starts or the user logs in, this is programs defined by an *auto-start registry key* or directory:

C:> wmic startup list full

A lot of malware can automatically run on your computer by adding an auto-start item next to other valid entries inside antivirus tools or system tray programs. Users can view other settings on the computer with WMIC by replacing " **startup** " with " **QFE** " (the initials for Quick Fix Engineering) to see the patch level of a system, by " **share** "to see a list of shared files on Windows or" **useraccount** "to see the user's detailed account settings.

Another option within WMIC is the ability to run a command to gather information on a certain cycle by using the syntax **"/ every: [N]** " after the rest of the WMIC command. [N] here is an integer, indicating that WMIC will run the above command every [N] seconds. That way, users can search for changes in system settings over time, allowing a thorough survey of the output. Using this function to pull all information about the process every 5 seconds, the user can run:

C:> wmic process list brief / every: 1

Press **CTRL + C** will stop the cycle.

2. Net command

As mentioned above, WMIC is a relatively new command, but there are some other commands that are not new but quite useful, the " **net** " command. Administrators can use this command to display all useful information.

For example, the " **net user** " command will display all internal user accounts defined on the computer. The " **net localgroup** " command displays groups, " **net localgroup administrators** " displays members of the

administrator group and the " **net start** " command displays running services.

Hackers often bring users into a system or put their accounts into an administrator group, so we must always check the output of these commands to see if the hacker has modified the accounts on the computer. count or not. In addition, some hackers can create bad services on computers, so users should be careful with them.

3. Openfiles

Many Windows administrators are not used to using strong **openfiles** commands available in Windows. However, as the name implies, this command displays all the files opened in the computer, indicating the process name interacting with each file. It is built in new versions of Windows, from XP Pro to Vista. Like the popular **lsof** command for Linux and Unix, it also shows administrators all the files that are open on the computer, provides an overly-named name and a complete path for each file. However, unlike **lsof** , it does not provide much detailed information, such as the process ID number, user number or other information.

Considering the volume of information it gathers, it should come as no surprise that the **openfiles** command really consumes a lot of performance. Therefore, normally the processes related to **openfiles** are turned off by default, meaning that users cannot drag any data from this command until it is turned on. This function can be activated by running the command:

C:> openfiles / local on

Users will need to reboot and when the system resumes, they will be able to run the **openfiles** command as follows:

C:> openfiles / query / v

This command displays the output in detail, including the user account that the process for an open file is running inside. From there, you can see what malware has been installed, or what attack might be being done on the computer, users should search for abnormal files or unexpected files, especially those associated. related to unwanted internal users on the computer.

When finished with the **openfiles** command, its calculation function can be turned off and the system will return to normal performance status by running the following command and restarting the computer:

C:> openfiles / local off

4. Netstat

The **netstat** command in Windows can display network behavior, focusing on default TCP and UDP. Because malware typically communicates throughout the network, users can search for unusual connections in netstat's output, running the following command:

C:> netstat -nao

The **-n** option will tell **netstat to** display the numbers in its output, except for the host name and protocol that will instead display IP and TCP addresses or UDP port numbers. **–A** indicates to show all connections and listening ports. The **-o** option notifies **netstat to** display the processID number of each program interacting with TCP or UDP ports. If instead of TCP and UDP, you only care about ICMP, then you can run the netstat command as follows:

C:> netstat –s –p icmp

The above command indicates that it will return statistics (-s) of the ICMP protocol. Although it doesn't show much detail with TCP and UDP, users can see whether the computer is sending unexpected ICMP traffic on the network. However, some backdoors and some other malware can communicate using the payload of ICMP Echo messages.

Like WMIC, the **netstat** command also allows us to run it in a repetitive cycle. However, instead of using the " **/ every: [N]** " syntax like WMIC, users only need to add after the call to **netstat** spaces and integers. Thus, to list the currently used TCP and UDP ports on the computer every 2 seconds, the user can run:

C:> netstat –na 2

5. Find

Most of the commands that I have introduced so far show a lot of output on the screen, which sometimes makes it difficult for users to take a full view to find something that they are interested in. But Windows has another tool that can help you fix this. Users can search the entire output of each command using the **findstr** and **find** command in Windows. The **find** command will search for simple strings, while **findstr** will support regular

words, a more complex way to distinguish search patterns. Because regular words are supported by **findstr** beyond the scope of this article, we only focus on the **find** command. By default the **find** command will distinguish between uppercase and lowercase letters, but by using the **/ i** option you can lose this distinction.
The **find** command is also capable of counting. Invoked with the **/ c** command, it will count the line number of the output containing the given string. If the user wants to count the number of lines in the output of the command to know the number of running processes, the number of startup items present, or a series of other actions on the machine. To count the number of output lines, users can lead their output via **find / c / v "".** This command will count (**/ c**) the number of lines except the blank line.

Now, with the find command, users can observe the output of each of the commands that we have introduced here to find interesting things. For example, to see the information every second about **cmd.exe** processes running on your computer, type:

C:> wmic process list brief / every: 1 | find "cmd.exe"

To count the number of files opened on the computer when openfiles are activated, you only need to type:

C:> openfiles / query / v | find / c / v ""

No matter how you count items, remember to subtract the number of lines associated with the column header. For example, to see with every second accuracy when TCP port 2222 starts to be used on the computer, along with the process ID being used on the port, run:

C:> netstat -nao 1 | find "2222"

Research output

With these 5 tools, users can handle configuration information and security status of each Windows computer. However, in order to use each command to identify compromises, users need to compare the current settings of the suspected computer to the normal computer.

There are three ways to establish this comparison. First, if the user is an experienced "hunter" of malware, he can be aware of what is right and what is wrong with the computer, recognizing unusual problems based on experience. Secondly, this comparison can be done with an uninfected machine if available. Without a 'clean' computer, users can rely on a third option - search for specific files, process names, file names and port numbers recognized by these commands and search for them online to determine if they are regular files for the computer and software that it has installed or are associated with some kind of malware.

In this section, I have shown you five very powerful commands in Windows. In the next part of this series, we will introduce you to 5 other useful commands from the command line.