10 Windows 7 commands every administrator should know

Brien
Posey

Holiday rerun: An oldie-but-a-goodie, these command-line basics topped the list of popular troubleshooter posts last year.

PC troubleshooting is becoming less common in larger organizations, but consultants and techs in smaller shops still have to get their hands dirty identifying and fixing desktop problems. Oftentimes, troubleshooting Windows 7 means delving into the command line. Here are 10 fundamental Windows 7 commands you might find helpful.

Before I begin...

This article is intended solely as an introduction to some useful troubleshooting commands. Many of them offer numerous optional switches, which I won't cover here due to space limitations. You can find out more about each command by checking out TechNet's command-line reference.

1: System File Checker

Malicious software will often attempt to replace core system files with modified versions in an effort to take control of the system. The System File Checker can be used to verify the integrity of the Windows system files. If any of the files are found to be missing or corrupt, they will be replaced. You can run the System File Checker by using this command:

```
sfc /scannow
```

2: File Signature Verification

One way to verify the integrity of a system is to make sure that all the system files are digitally signed. You can accomplish this with the File Signature Verification tool. This tool is launched from the command line but uses a GUI interface. It will tell you which system files are signed and which aren't. As a rule, all the system files should be digitally signed, although some hardware vendors don't sign driver files. The command used to launch the File Signature Verification tool is:

```
sigverif
```

3: Driverquery

Incorrect device drivers can lead to any number of system problems. If you want to see which drivers are installed on a Windows 7 system, you can do so by running the driverquery tool. This simple command-line tool provides information about each driver that is being used. The command is:

```
driverquery
```

If you need a bit more information, you can append the -v switch. Another option is to append the -si switch, which causes the tool to display signature information for the drivers. Here's how they look:

```
driverquery -v
```

```
driverquery -si
```

4: Nslookup

The nslookup tool can help you to verify that DNS name resolution is working correctly. When you run nslookup against a host name, the tool will show you how the name was resolved, as well as which DNS server was used during the lookup. This tool can be extremely helpful when troubleshooting problems related to legacy DNS records that still exist but that are no longer correct.

To use this tool, just enter the nslookup command, followed by the name of the host you want to resolve. For example:

```
nslookup dc1.contoso.com
```

5: Ping

Ping is probably the simplest of all diagnostic commands. It's used to verify basic TCP/IP connectivity to a network host. To use it, simply enter the command, followed by the name or IP address of the host you want to test. For example:

```
ping 192.168.1.1
```

Keep in mind that this command will work only if Internet Control Message Protocol (ICMP) traffic is allowed to pass between the two machines. If at any point a firewall is blocking ICMP traffic, the ping will fail.

6: Pathping

Ping does a good job of telling you whether two machines can communicate with one another over TCP/IP, but if a ping does fail, you won't receive any information regarding the nature of the failure. This is where the pathping utility comes in.

Pathping is designed for environments in which one or more routers exist between hosts. It sends a series of packets to each router that's in the path to the destination host in an effort to determine whether the router is performing slowly or dropping packets. At its simplest, the syntax for pathping is identical to that of the ping command (although there are some optional switches you can use). The command looks like this:

```
pathping 192.168.1.1
```

7: Ipconfig

The ipconfig command is used to view or modify a computer's IP addresses. For example, if you wanted to view a Windows 7 system's full IP configuration, you could use the following command:

```
ipconfig /all
```

Assuming that the system has acquired its IP address from a DHCP server, you can use the ipconfig command to release and then renew the IP address. Doing so involves using the following commands:

```
ipconfig /release
```

```
ipconfig /renew
```

Another handy thing you can do with ipconfig is flush the DNS resolver cache. This can be helpful when a system is resolving DNS addresses incorrectly. You can flush the DNS cache by using this command:

```
ipconfig /flushdns
```

8: Repair-bde

If a drive that is encrypted with BitLocker has problems, you can sometimes recover the data using a utility called repair-bde. To use this command, you will need a destination drive to which the recovered data can be written, as

well as your BitLocker recovery key or recovery password. The basic syntax for this command is:

```
repair-bde <source> <destination> -rk | rp <source>
```

You must specify the source drive, the destination drive, and either the rk (recovery key) or the rp (recovery password) switch, along with the path to the recovery key or the recovery password. Here are two examples of how to use this utility:

```
repair-bde c: d: -rk e:\recovery.bek
```

```
repair-bde c: d: -rp 111111-111111-111111-111111-111111-111111
```

9: Tasklist

The tasklist command is designed to provide information about the tasks that are running on a Windows 7 system. At its most basic, you can enter the following command:

```
tasklist
```

The tasklist command has numerous optional switches, but there are a couple I want to mention. One is the -m switch, which causes tasklist to display all the DLL modules associated with a task. The other is the -svc switch, which lists the services that support each task. Here's how they look:

```
tasklist -m
```

```
tasklist -svc
```

10: Taskkill

The taskkill command terminates a task, either by name (which is referred to as the *image name*) or by process ID. The syntax for this command is simple. You must follow the taskkill command with -pid (process ID) or -im (image name) and the name or process ID of the task that you want to terminate. Here are two examples of how this command works:

```
taskkill -pid 4104
```

```
taskkill -im iexplore.exe
```