



Global Knowledge®

Expert Reference Series of White Papers

Routing Essentials

Routing Essentials

Dheeraj (Raj) Tolani, Global Knowledge Instructor

Introduction

Every time we teach a class to get students started with Cisco and networking technologies, there are always some common things that stop students from grasping some basic topics. We instructors are always asked if there is a basic to the basic course they are taking. That's the purpose of this white paper—to help students with some of those “basic of the basic concepts.” This white paper won't answer all your questions, but it will definitely help with some of those concepts to make your class experience more enjoyable and productive. After reading this white paper, you should be ready for more complex concepts found in ICND1, followed by ICND2 (or the CCNA Boot Camp).

Router Basics

A router is a device that allows you to move packets between networks. This forwarding of packets occurs using the best path. What is the best path? The best path depends on the routing protocols you're using on your routers.

I like to think of best path determination as asking multiple people how to go from point A to point B. Some might give me an option to start walking from A to B, while others might ask me to use the public bus service.

Keep in mind that there might be multiple public buses, so if I choose the public bus option, I must then pick the best bus line to get from point A to point B. Initially, I have to choose the person to trust: the one asking me to walk or the one giving me the bus line information. I can listen to all possible directions, and then focus only on the directions from the individual I trust.

Routers do the same kind of thing. Multiple routing protocols could be running on routers, especially when you are migrating from a non-Cisco environment to Cisco-only environments, or for other political reasons in your organization. We need a method to pick the best routing protocol from all the routing protocols running. And, since that best protocol might have multiple paths, we then have to pick the best one among all the paths.

The parameters these protocols use to determine their best path will vary. For example, some protocols will only consider the number of hops between two points, as with Routing Information Protocol (RIP). Other protocols will consider multiple items including bandwidth, as with Cisco's proprietary protocol Enhanced Interior Gateway Routing Protocol (EIGRP) or the industry-standard Open Shortest Path First (OSPF). The things that are considered for path determination by these routing protocols are known as metrics. If there is a change in the

metrics, then these paths will be recalculated. Therefore, we call these **dynamic routing protocols** since they have the capability to adapt with the changes in your network.

Some routing protocols, called **distance vector protocols**, periodically exchange routing protocol tables with other routers running the same protocols. While other protocols, called **link state routing protocols**, only send the path information when something changes and don't have a periodic exchange of routes (other than some "hellos" to ensure that their other neighbor routers running the same routing protocols are still out there).

So, as we said earlier, we pick these routing protocols based on the criteria/metric (hops or bandwidth, for example) they use, and then only the best path (i.e., the one with the lower hop count or the better bandwidth) from the best routing protocol goes in the **routing table**, which is used for forwarding these IP packets.

Cisco routers have a method of picking the preferred protocol using **administrative distance**. As explained on Cisco's web site,

"...administrative distance defines the reliability of a routing protocol. Each routing protocol is prioritized in order of most to least reliable (believable) with the help of an administrative distance value."

Cisco assigns a number to the routing protocols on a scale of 0-255, where the protocol with the lower number is preferred. (And, yes, you can change those numbers.) Once we pick the best routing protocol, then we need to see all the different paths available using that routing protocol. At that point, the consideration is the best path among all possibilities using the lowest metric for that protocol (as in our example of picking the best bus line from all other buses).

This table of all the best routes is kept in the router's Random Access Memory (RAM). RAM is a volatile part of the router, meaning that, if you lose power, you lose the contents of RAM. So, if you lost power, you'd lose this routing table, and you would have to relearn the routes.

How did we get the contents in the routing table? Remember that, depending on the type of routing protocol, we were either learning the contents periodically (distance vector routing protocols) or learning the contents when something changed in the network, such as an interface coming up or going down (link state routing protocols). This volatile table not only keeps the routing table entries, but it also keeps other information such as the configuration that is currently running on your router or switch (called **running-config**).

This table also keeps various caches, such as ARP, cache a device's operating system once it's decompressed, and many other things.

So, is there a component that is not volatile? I'm glad you asked. There is another component called Non-Volatile RAM (NVRAM). Since it's not volatile, if you lose power, you will not lose the contents of NVRAM. NVRAM is

where you keep the **startup-config** file - the configuration that your router or switch comes up with and loads into RAM. If you want your changes to be permanent, then you must be sure that your changes are in NVRAM.

Commands and Configurations Basics

Now that you are familiar with the basics of the routers, it makes sense to get a router up and running with basic recommended configurations. Let's go through various basic commands, including the commands you need to configure the box and the commands to verify what you did.

We will assume at this point that you have an unconfigured router with a console cable plugged into the console port of your Cisco router, and you are using a HyperTerminal application with these basic Com port settings: 9600, 8, N, 1 with no Flow Control.

When you turn on an unconfigured Cisco router, it will prompt you to determine if you wish to start the basic configuration script. This script gives you the option to configure a box without knowing any Cisco commands.

This is a simple YES/NO script that prompts you with basic questions such as whether you want to configure IP addresses, what interfaces to use, and the addresses for those appropriate interfaces.

If you're just getting started, it might seem okay for you to simply go through the script and configure your first router. But, just imagine if you have to make a change to one of the parameters that you configured using the setup script. Then you'd have a problem. Do you really want to go through 20-30 questions to change one parameter? It might not be the best use of your time. (I had a student once who said going through multiple questions in the script was okay with him since he got paid by the hour. But I'm assuming that's not the case with you, and that you want to bring the network up very fast.) So, if speed and efficiency are your goal, then you should be configuring your devices manually using the commands available in the Cisco operating system.

Let's check out some of the basic commands. Before we do that, though, we have to say NO to the setup script prompt. Once you say NO to the setup script, your interfaces are all SHUTDOWN. So, once we configure the interfaces we want to use, we'll need to undo the shutdown. Ready? Here we go.

```
Router>
```

This prompt lets you know you're in user EXEC mode - also known as privilege level 1 - a level with minimal permissions. In other words, you have permission to do almost nothing. For example, you can't see anything important like configurations that contain passwords. To be able to see things like that, you have to go to the next level, privilege EXEC mode. To do that, use the **enable** command.

```
Router>enable
Router#
```

The router's prompt now includes the # at the end, and you're in privilege EXEC mode, also known as enable mode or privilege level 15. (**Note:** Having a # sign doesn't mean that you are in the enable mode. It could just

be that someone is playing a trick on you by changing your router name to include the # as part of the name.) This privilege level gives you all the power you need to configure or destroy your router. If you ever wish to see what privilege level you have, you can use the command **show privilege**.

```
Router# show privilege
Current privilege level is 15
```

To be able to configure anything on the router, you have to be in global configuration mode. The command to get there is:

```
Router#configure terminal
```

This means you are configuring this particular router from the terminal you are sitting on and entering one command at a time. This command changes your prompt to:

```
Router(config)#
```

So we've made it to global configuration mode. Commands executed at this prompt have a global impact on the router. Let's consider the kinds of things that have global impact. The hostname of the router? Certainly. The IP address on our Ethernet 0 interface? No. IP on an interface is, well, on an interface. It's not global.

```
Router(config)#hostname Backbone_NYC
Backbone_NYC(config)#
```

It always makes sense to give your router a meaningful name instead of using the default name "Router." You should plan the router name in your organization. The name should have some meaning to you. Of course, you'll hear different schools of thought on that. Some people argue that your router should be able to identify the purpose of the box. Others will argue that your router shouldn't do that, because it allows hackers to know exactly what box they are hitting. They don't have to guess the purpose of the router. In my experience, every client is different, so check your client's security policy recommendations before you assign your router a hostname.

As you can see from the example above, the router name changes the moment you hit the enter key after entering the name using the **hostname** command. Keep in mind that this is still only in the **running-config** (discussed previously). You'll need to save this change to your router's name, and the command to do so will be discussed later.

Let's pretend that, at this point, you wish, to put an IP address on your Ethernet interface. Remember that the assumption here is that you have an Ethernet interface on the router you are using. Today, you might have an Ethernet interface (10 mbps link), or a FastEthernet interface (100 mbps link), or maybe even a Gigabit interface (1000 mbps link). In our next few configuration steps, we are going to assume you have a standard Ethernet interface, but remember changing the IP address or to enable the interface is exactly the same, regardless of the type of interface you have.

Also, note that today routers and switches are mostly modular devices, meaning you can swap the interfaces that are plugged in for another type of interface. So, you'll have to know the type of module that is plugged in

and what slot it's plugged into. For our purposes, we will use a simple Ethernet interface; Port # 0 (not a modular device).

Here is how to do that:

```
Backbone_NYC(config)#interface Ethernet 0
Backbone_NYC(config-if)#ip address 10.1.1.1 255.255.255.0
Backbone_NYC(config-if)#no shutdown
```

As you can see, the prompt changed to **Backbone_NYC(config-if)#**. This is known as interface configuration mode.

A common complaint I get from students is that you cannot tell what interface you are configuring by looking at the prompt. That means you'd better know what command brought you here. In our case, the command was **interface Ethernet 0**. This means the commands at this prompt will affect Ethernet 0. And, this IP that we just entered followed by the subnet mask is for interface Ethernet 0. The last command (**no shutdown**) ensures that the interface is NOT SHUTDOWN (meaning it hasn't been shutdown/disabled by the administrator). Now, since we said NO to setup script, we want to make sure that the interface is UP, since saying no to the script did shutdown all our interfaces, including this Ethernet 0.

The next recommendation is to be prompted for a password when someone tries to connect to the router. Let's look at the commands first, and then discuss them individually.

```
1 Backbone_NYC(config-if)#exit
2 Backbone_NYC(config)#line console 0
3 Backbone_NYC(config-line)#login
4 Backbone_NYC(config-line)#password cisco
5 Backbone_NYC(config-line)#exit
6 Backbone_NYC(config)#line vty 0 4
7 Backbone_NYC(config-line)#login
8 Backbone_NYC(config-line)#password cisco
9 Backbone_NYC(config-line)#exit
10 Backbone_NYC(config)#
```

For easy reference, I have entered line numbers to the left of the commands.

In the 1st command, we entered the **exit** command to go back one step. Remember, we just finished configuring the Ethernet 0 interface, so we were in the interface configuration mode for the Ethernet interface. In this case, the **exit** command takes us back to global configuration mode (shown in step 2).

Note: You don't really have to go back to Global Configuration mode to type the command on line #2, but that command will also work in the interface configuration mode. If you wanted any syntax help then, the syntax help is only available if you are in the mode where the command belongs (You can get syntax help by hitting the ?). If you are very familiar with the syntax of the command and the parameters required, then you can type the command at the interface mode, and it will still work. After typing the command, when you hit the **ENTER**, key the system will take you to the line configuration mode for the console prompt.

I would recommend that you should always be very comfortable with the CCNA commands so you don't need any help with those basic commands.

In the 2nd command, we want to configure the console port. This takes us to line configuration mode (shown in step 3) affecting the console port.

In the 3rd command, we are basically specifying that we want people to be able to login in the console port.

In the 4th command, we are entering the acceptable password when these people do login to the console port.

In the 5th command, we are again using **exit** to take us back one step (same as we did in the 1st command). One step back is the global configuration mode. Remember from the note above that we don't have to do the **exit** command, we could do the command in Line # 6 without going back to Global Configuration mode by just staying at the line configuration mode for console.

In the 6th command, we are going to the **vty** mode. For us, this is for telnet purposes. This will allow users to be able to telnet into our router. Now we are allowing five simultaneous telnet sessions in our router (line vty 0 4; 0 is the first allowed session, and 4 is the last session - the fifth session, so we are specifying a range here from 0 to 4). Just as before, we are in the line configuration mode after we type the command (shown in step 7 – same as the prompt for line 3). Again, by looking at the prompt, we cannot tell which line mode we are configuring (same problem that we had for the interface configuration mode).

Lines 7, 8, and 9 are the same as Lines 3, 4, and 5, but this time for the vty mode where we are specifying telnet connectivity. VTY lines can be configured for multiple things. In CCNA-level curriculum, we use the insecure method telnet and the secure method SSH (Secure Shell). For our white paper purposes, we are using the simple telnet in this example.

So far, we have given the router a hostname, assigned an IP address on the Ethernet 0 address, enabled console access with a password, and allowed users to be able to telnet into the router with a password.

In the last step, we are in the global configuration mode because the exit from the 9th step brought us here.

Once here, we should enforce a password that people have to enter while going from user mode to privilege mode. The command that allows us to do that is:

```
Backbone_NYC(config)#enable secret san-fran
Backbone_NYC(config)#end
Backbone_NYC#
```

Now every time we type the enable command to go from user mode to privilege mode, we will be prompted for this password (san-fran in our example).

At the end, we should consider saving our work to the NVRAM. This way, if we lose power, our work is saved. The command to do that is:

```
Backbone_NYC#copy running-config startup-config
```

Our configuration is now saved in startup-config in NVRAM. This is the configuration that will be loaded after our router comes up. It may be helpful if you think of this command as:

```
Backbone_NYC#copy <source> <destination>
```

So if you are ever asked to save the configuration to the TFTP server, you can easily just specify the command:

```
Backbone_NYC#copy running-config tftp
```

Great, now we have saved our configuration to the TFTP also - providing you had connectivity to the TFTP - if you have the TFTP server sitting in your local segment. If your TFTP server is not on your local segment, you will need to enable routing so you can get to those remote subnets and then save your **running-config to the remote TFTP server**.

When you type the above command **copy running-config tftp** and hit the ENTER key, the system will prompt you for all the required parameters like the IP address of the TFTP server and the name of the file you will be creating on the TFTP server; just follow the prompts. I typically like to ping the IP address of the TFTP server before I start copying things to ensure that I have IP reachability. It would also make sense to check with the TFTP server admin to make sure that the TFTP server software is running, and you have the appropriate permissions.

Lets see what we have configured. The way you see what is configured is by using the **Show running-config** command. Of course, you could abbreviate the command if you wish, as long as your command is not ambiguous.

Here is a sample partial output from **Show running-config** command

```
<output omitted>
!
line con 0
  exec-timeout 0 0
  password cisco
```

```
login
line aux 0
line vty 0 4
password cisco
login
!
<output omitted>
```

As you can see in this partial output, the passwords for console and VTY are showing up in your configuration clear text. One could argue that in order to be able to do **show running-config** command you have to be in privileged mode, which is protected with a enable password (in our example that password was san-fran)

If seeing the password in clear text bugs you then you could type the **service password-encryption** command in Global configuration mode. Now, after typing the command, if you do the **Show running-config** command, you can see that the passwords are not clear text in your configuration but instead are "hashed."

Here is a sample partial output from **Show running-config** command after we typed the **Service password-encryption** command in the global configuration mode:

```
<output omitted>

!
line con 0
exec-timeout 0 0
password 7 121A0C041104
login
line aux 0
line vty 0 4
password 7 094F471A1A0A
login
!
<output omitted>
```

I am NOT saying that this is the most secure method, because you can search the internet for a utility called **getpass.exe**, and, unfortunately, this utility can be used to decrypt the password you had typed. You can hope that the person you gave your configs to doesn't know about this utility or many other similar utilities, but that is just wishful thinking). So, there has to be a better way to protect your access to the devices in your company

using the console or the VTY lines. Well, we need to see you in the class at Global Knowledge so we can show you more of these cool things.

Note: Please take precautions when using any utilities you download from the internet. You should check to make sure you trust the source, and these files are free from any viruses, Trojans, or any such malicious and evil things. Download and use these tools at your own risk.

Summary

This takes care of some of the basics you would typically be concerned about with the basic router setups. I hope this white paper has gotten you thinking about configuring some of the common elements in the routers and possibly excited about other white papers for some more involved items.

Glossary of Key Router Terms

Routing Protocol: Dynamic protocols that exchange path information with other routers so IP packets could be moved from one network to another.

Distance Vector: A type of routing protocol that exchanges routing table information periodically with other routers that are running the same protocol.

Link State: A type of routing protocol that exchanges information when the state of the link changes, such as when an interface comes up or goes down. In the name "link state", link is the interface, and state refers to being up or down.

Metric: Path calculation parameter used by the routing protocols. Metrics vary from protocol to protocol. Routing Information Protocol (RIP) uses hop count as its metric. The lower the number, the most preferred the path. So, in the case of RIP, the lowest hop count path will be selected if you had multiple paths available.

Administrative Distance: Number assigned to the routing protocols by Cisco. This is used to pick the routing protocol to use if you have multiple routing protocols running on your router. The lower the number, the most preferred the routing protocol.

RAM: Volatile component of the router where entries are kept until you can maintain power to the box. Your running configuration is kept here.

NVRAM: Non-volatile component of the router where entries are kept and saved even after you lose power. Your startup configuration is kept here.

Learn More

Learn more about how you can improve productivity, enhance efficiency, and sharpen your competitive edge. Check out the following Global Knowledge courses:

[Understanding Networking Fundamentals](#)

[ICND1—Interconnecting Cisco Network Devices 1](#)

[ICND2—Interconnecting Cisco Network Devices 2](#)

[CCNA Boot Camp v2.0](#)

For more information or to register, visit www.globalknowledge.com or call **1-800-COURSES** to speak with a sales representative.

Our courses and enhanced, hands-on labs and exercises offer practical skills and tips that you can immediately put to use. Our expert instructors draw upon their experiences to help you understand key concepts and how to apply them to your specific work situation. Choose from our more than 1,200 courses, delivered through Classrooms, e-Learning, and On-site sessions, to meet your IT and business training needs.

About the Author

Dheeraj (Raj) Tolani has been working with Global Knowledge as a contract instructor teaching various networking courses including CCNA, CCDA, CCNP, CCDP, CCIP, CCVP tracks. He has been in the industry for over 17 years working with various technologies and multiple vendors including Cisco, Banyan Vines, Microsoft, Comptia, and Novell. Dheeraj has worked as a consultant for various medical, financial, legal, government, and publishing companies. He runs a consulting company based out of New York City, which provides IP integration solutions. You can visit his web site at www.rajtolani.com.