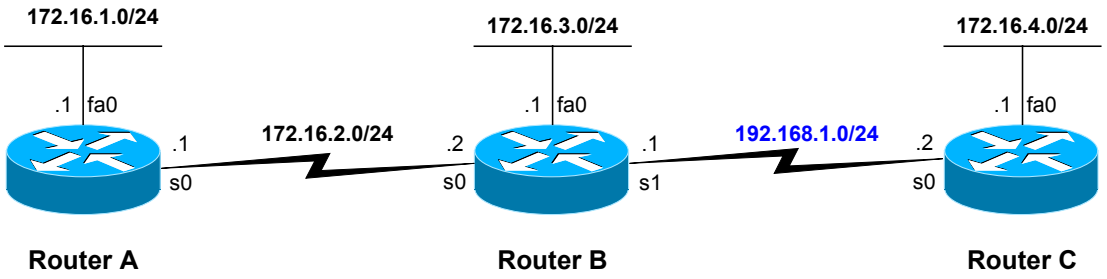
The Routing Table

Part I Understanding the Routing Table Structure

By Rick Graziani

This document is the first of two parts dealing with the routing table. Part I discusses the structure of the routing table, how routes are created. Part II discusses the routing table lookup process.

The network we will be using is a simple three router network. RouterA and Router B share a common 172.16.0.0/24 network. RouterB and RouterC are connected by the 192.168.1.0/24 network. You will notice that RouterC also has a 172.16.4.0/24 subnet which is disconnected, or discontiguous, from the 172.16.0.0 network that RouterA and RouterB shares. This was done on purpose, and will be significant in Part II – The Routing Table Lookup Process.



One last note before we begin. In order to keep this document simple and brief (and for those of you who know me, that is not easy for me to do!), I have left out some of the detail and exact terminology. For those of you who are interested in reading more about this subject and the inner-workings of the Cisco IOS as it pertains to routing, I highly recommend the book, *Cisco IP Routing, by Alex Zinin (ISBN 0-201-60473-6)*. However, this is not a book for beginners and does contain some pseudocode which can be skipped if you wish.

Part I – Adding Routes to the Routing Table

We will begin by assuming the following configurations have already been done to the three routers. If you are using the document as a lab, configure the routers with these configurations. You will notice that we have not added any static routes or dynamic routing protocols.

The DCE cable is attached the serial interfaces of RouterA and RouterC. The command **exec-timeout 0 0** is an optional command that keeps the router from exiting privileged mode when the idle timer expires. The other optional command is **logging synchronous**. This command eliminates debug and other output from becoming intermixed with the router prompt and input commands.

RouterA

```
hostname RouterA
!
interface FastEthernet0
 ip address 172.16.1.1 255.255.255.0
!
interface Serial0
 ip address 172.16.2.1 255.255.255.0
 clockrate 64000
!
line con 0
 exec-timeout 0 0
 logging synchronous
```

RouterB

```
hostname RouterB
!
line con 0
 exec-timeout 0 0
 logging synchronous
```

RouterC

```
hostname RouterC
!
interface FastEthernet0
 ip address 172.16.4.1 255.255.255.0
!
interface Serial0
 ip address 192.168.1.2 255.255.255.0
 clockrate 64000
!
line con 0
 exec-timeout 0 0
 logging synchronous
```

Creating an Ultimate Route

Since we have not configured any interfaces for RouterB, the routing table for RouterB does not currently contain any routes (Figure 1).

Figure 1

```
RouterB#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

RouterB#
```

Before we configure the interfaces, we will use the **debug ip routing** command (Figure 2), which allows us to view the routing table process adding, deleting, and modifying entries. As usual, don't forget to use the **undebug ip routing** or **undebug all** command to turn off debugging.

Figure 2

```
RouterB#debug ip routing
```

Now, let's begin by configuring RouterB's serial 1 interface (Figure 3). The output from the **debug ip routing** command is highlighted in Figure 3. Other IOS output has been omitted.

Figure 3

```
RouterB(config)#interface s 1
RouterB(config-if)#ip add 192.168.1.1 255.255.255.0
RouterB(config-if)#no shutdown
00:59:48: %LINK-3-UPDOWN: Interface Serial1, changed state to up
00:59:48: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1, changed state
 to up
00:59:48: RT: add 192.168.1.0/24 via 0.0.0.0, connected metric [0/0]
00:59:48: RT: interface Serial1 added to routing table
RouterB(config-if)#end
RouterB#undebug all
All possible debugging has been turned off
RouterB#
```

When an interface is configured with the line protocol and status in the “up” state, the network, or subnet, that the interface belongs to is added to the routing table with an administrative distance of 0. (Figure 4)

Figure 4

```
RouterB#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
<text omitted>

Gateway of last resort is not set

C    192.168.1.0/24 is directly connected, Serial1
RouterB#
```

Lets take a look at this entry in more detail. Some of this may seem obvious, but there are some very important items that need to be addressed. First of all, this type of routing table entry is known as an **ultimate network route**.

To be an ultimate network route, the route must have the following characteristics:

- The network route is a classful (major) network or a supernet/default route. In our case **192.168.1.0** is a major, Class C network.
- The route can be resolved using an exit interface. Directly connected networks use the interface on which the interface was configured as the exit interface. We will see later that this may not necessarily be the case for static or dynamically learned routes. In our directly connected network example, the exit interface is **Serial1**.

Besides the code for how the route was learned (**C** for 192.168.1.0) and the classful or supernet network, ultimate network routes in the routing table will also contain:

- The subnet mask, which is either the classful mask or a supernet mask. We will discuss supernet masks later in the Part II. In our example, **/24** is shown as the subnet part of the ultimate route.
- The exit interface, which is once again, **Serial1**.

Notes:

- There is another type of ultimate route, the child/ultimate route which is discussed in the next section, Creating a Parent/Child Route.
- In Part II The Routing Table Lookup Process, we will examine how the subnet mask is used to determine the longest-bit match when determining the best route.

It is the exit interface that ultimately (thus “ultimate route”) determines which interface the routing table will use to forward the packets. An exit interface must exist for the packet to be routed. Later we will see where the routing table must do recursive or multiple lookups in order to find an ultimate route in the routing table with an exit interface.

To summarize, an ultimate network route is created in the routing table when the network is a classful (major) network and the route contains an exit interface. An ultimate network route can also be a supernet or default route which will be discussed later in Part II. This is regardless of how the route entered the routing table, either directly connected, statically configured or dynamically learned. Other situations will be discussed in the next section, Creating a Parent/Child Route.

Creating a Parent/Child Route

Now lets see what happens when a route is created that is a subnet of a major or classful network. Figure 5 shows the configuration of RouterB's fastethernet interface with an IP address of 172.16.3.1 and a subnet mask of 255.255.255.0 (/24) and the output from the **debug ip routing** command.

Figure 5

```
RouterB(config)#interface fastethernet 0
RouterB(config-if)#ip add 172.16.3.1 255.255.255.0
RouterB(config-if)#no shutdown
01:21:11: RT: add 172.16.3.0/24 via 0.0.0.0, connected metric [0/0]
01:21:11: RT: interface FastEthernet0 added to routing table
RouterB(config-if)#end
RouterB#undebug all
All possible debugging has been turned off
RouterB#
```

Looking at the routing table in Figure 6, we notice that this creates two entries in the routing table. Because the interface was configured with the IP address and subnet mask of 172.16.3.1 255.255.255.0, the routing table process recognizes that this interface is part of the 172.16.3.0/24 network and creates an entry in the routing table to reflect this directly connected network "C 172.16.3.0 is directly connected, FastEthernet0."

Figure 6

```
RouterB#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
<text omitted>

Gateway of last resort is not set

     172.16.0.0/24 is subnetted, 1 subnets
C       172.16.3.0 is directly connected, FastEthernet0
C    192.168.1.0/24 is directly connected, Serial1
RouterB#
```

But we notice a second entry has also been created, "172.16.0.0/24 is subnetted, 1 subnets." This is the major classful network that the 172.16.3.0 subnet belongs to, i.e. the Class B network of 172.16.0.0.

The top entry is known as the **Parent Route**, and the bottom route is known as the **Child Route**.

```
Parent Route:        172.16.0.0/24 is subnetted, 1 subnets
Child Route:      C    172.16.3.0 is directly connected, FastEthernet0
```

It is the child route that contains the actual route for the directly connected 172.16.3.0/24 network. How was the parent route created and why? Whenever a route enters the routing table that contains a subnetted network address, a child route, a parent network route is automatically created. If there is already a parent network route in the routing table, the child route is placed below the parent. *All subnetted routing entries will be installed within the parent (classful) route to guarantee proper route lookup.*

What do we mean by a network that contains a subnetted network address? We mean a network address that has a subnet mask larger than the default classful mask. In our example we configured fastethernet0 with the IP address and subnet mask of 172.16.3.1 255.255.255.0. The /24 subnet mask is larger than the default Class B subnet mask of /16, so besides the child route being entered into the routing table, a parent route of 172.16.0.0 was also created.

What differences do you notice between the parent network route, the child network route, and the ultimate network route that was created in previously?

```
Parent Route:        172.16.0.0/24 is subnetted, 1 subnets
Child Route:      C    172.16.3.0 is directly connected, FastEthernet0
Ultimate Route:   C  192.168.1.0/24 is directly connected, Serial1
```

Whereas an ultimate network route contains the network, the subnet mask and the exit interface in one entry, this is not true for the parent/child network routes.

The child route does not contain the subnet mask. Instead, the subnet mask is part of the parent network route, **/24** in our example. (Note: The Classful mask of /16 is not displayed.) Besides the subnet mask, the parent network route also contains the major or classful network address, **172.16.0.0**, and the number of subnets (child routes) known by this routing table for this classful network, **is subnetted, 1 subnets**. (Okay, so the grammar is poor. What do you want, good grammar or good routing?)

Note: Later in the section, VLSM and Routing Tables, we will see that using a variable length subnet mask will display slightly different information for the parent and child routes.

Even though the parent route does not look like it contains any pertinent routing information for routing packets, later in Part II, we will see the important role these parent routes play in the routing table lookup process. (I bet you can hardly wait now!)

One last note about our child route, 192.168.1.0/24. You will notice that it also contains an exit interface, FastEthernet0. If the child route contains an exit interface, it is also considered an ultimate route. This is because it can be used to forward packets out the exit interface. We will see examples of child routes that are not ultimate routes in the section, Static Routes and Recursive Table Lookups.

In summary, a parent route is automatically created any time a route with a subnet mask that is greater than the classful subnet mask is installed in the routing table. Like an ultimate route, a child route can enter the routing table from a directly connected interface, a statically configured route, or a dynamically learned route.
The parent route displays information about the number of subnets or child routes, and the subnet mask for those child routes (except for VLSM networks which is discussed later). The child routes may also be ultimate routes if they also contain an exit interface. All subnetted routing entries will be installed within the parent (classful) route.

Creating a second child route

To solidify what we just learned, lets configure the serial0 interface on RouterB, which is part of the same classful network as RouterB's fastethernet0 interface. (Figure 7) This will create a second child route of **172.16.2.0** within the same, existing parent route of **172.16.0.0**.

Figure 7

```
RouterB(config)#inter s 0
RouterB(config-if)#ip add 172.16.2.2 255.255.255.0
RouterB(config-if)#end

RouterB#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
<text omitted>

Gateway of last resort is not set

     172.16.0.0/24 is subnetted, 2 subnets
C       172.16.2.0 is directly connected, Serial0
C       172.16.3.0 is directly connected, FastEthernet0
C    192.168.1.0/24 is directly connected, Serial1
RouterB#
```

Besides the new, second child route being added under a common classful parent, the number of subnets (child routes) in the parent route incremented to 2

Same Structure: Directly Connected, Static or Dynamic

It doesn't matter how the route enters the routing table, from being directly connected, a manually configured static route, or a dynamically learned route, the same ultimate and parent/child structure exists. We will see later why this is important when we discuss classful routing behavior (no ip classless) versus classless routing behavior (ip classless). By the way, this is not the same thing as classful routing protocols and classless routing protocols.

Here is an example of a routing table with directly connect, static and dynamically learned routes. This example is used to illustrate the fact that the ultimate and parent/child routing table structure applies to all routing table entries, no matter how the routes were installed the routing table.

Using the routing table below, see if you can determine how each route entered the routing table.

Figure 8

```
RouterB#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
<text omitted>

Gateway of last resort is not set

     172.16.0.0/24 is subnetted, 4 subnets
S       172.16.4.0 is directly connected, Serial1
R       172.16.1.0 [120/1] via 172.16.2.1, 00:00:08, Serial0
C       172.16.2.0 is directly connected, Serial0
C       172.16.3.0 is directly connected, FastEthernet0
     10.0.0.0/16 is subnetted, 1 subnets
S       10.1.0.0 is directly connected, Serial1
C    192.168.1.0/24 is directly connected, Serial1
S    192.168.100.0/24 is directly connected, Serial1
RouterB#
```

Here is the configuration for RouterA (Figure 9). Note: There are no configuration changes for RouterC, as this section is meant only to display the routing table information of RouterB and not meant to create an actual routable network.

Figure 9

```
hostname RouterA
!
interface FastEthernet0
 ip address 172.16.1.1 255.255.255.0
!
interface Serial0
 ip address 172.16.2.1 255.255.255.0
 clockrate 64000
!
router rip
 network 172.16.0.0
```

Here the configuration for RouterB (Figure 10).

Figure 10

```
hostname RouterB
!
interface Serial0
 ip address 172.16.2.2 255.255.255.0
!
interface Serial1
 ip address 192.168.1.1 255.255.255.0
!
interface FastEthernet0
 ip address 172.16.3.1 255.255.255.0
!
router rip
 network 172.16.0.0
 network 192.168.1.0
!
ip route 10.1.0.0 255.255.0.0 Serial1
ip route 172.16.4.0 255.255.255.0 Serial1
ip route 192.168.100.0 255.255.255.0 Serial1
```

Ultimate routes are created when the route is a classful or major network and there is also an exit interface. The two ultimate routes, that are not child routes are:

```
(Ultimate)  C    192.168.1.0/24 is directly connected, Serial1
(Ultimate)  S    192.168.100.0/24 is directly connected, Serial1
```

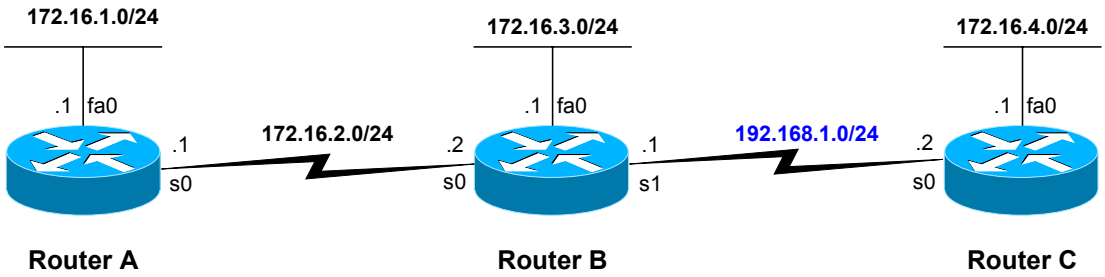
Notice that when a route is created which is a subnet of a major or classful network, it becomes a child route, and the parent route is automatically created. Looking at the routing table you will see that directly connected, static, and dynamic routing table entries will share a common parent entry as long as they belong to the same classful network.

```
(Parent)     172.16.0.0/24 is subnetted, 4 subnets
(Child)  S       172.16.4.0 is directly connected, Serial1
(Child)  R       172.16.1.0 [120/1] via 172.16.2.1, 00:00:08, Serial0
(Child)  C       172.16.2.0 is directly connected, Serial0
(Child)  C       172.16.3.0 is directly connected, FastEthernet0
```

The child route in many cases is also an ultimate route because it also contains an exit interface. In the next section we will see an example of a child route that is not an ultimate route. Examples of the parent/child routes are:

```
(Parent)      172.16.0.0/24 is subnetted, 4 subnets
(Child)  S       172.16.4.0 is directly connected, Serial1
(Child)  R       172.16.1.0 [120/1] via 172.16.2.1, 00:00:08, Serial0
(Child)  C       172.16.2.0 is directly connected, Serial0
(Child)  C       172.16.3.0 is directly connected, FastEthernet0

(Parent)     10.0.0.0/16 is subnetted, 1 subnets
(Child)  S       10.1.0.0 is directly connected, Serial1
```

Static Routes and Recursive Table Lookups



Now lets examine what happens when a child route is installed in the routing table that is not an ultimate route. We will also discuss how to create static routes that optimize the routing table lookup process.

Our current router configurations are as follows.

RouterA

```
hostname RouterA
!
interface FastEthernet0
 ip address 172.16.1.1 255.255.255.0
!
interface Serial0
 ip address 172.16.2.1 255.255.255.0
 clockrate 64000
!
line con 0
 exec-timeout 0 0
 logging synchronous
```

RouterB

```
hostname RouterB
!
interface Serial0
 ip address 172.16.2.2 255.255.255.0
!
interface Serial1
 ip address 192.168.1.1 255.255.255.0
!
interface FastEthernet0
 ip address 172.16.3.1 255.255.255.0
!
line con 0
 exec-timeout 0 0
 logging synchronous
```

RouterC

```
hostname RouterC
!
interface FastEthernet0
 ip address 172.16.4.1 255.255.255.0
!
interface Serial0
 ip address 192.168.1.2 255.255.255.0
 clockrate 64000
!
line con 0
 exec-timeout 0 0
 logging synchronous
```

Using an intermediate address

We will now create static routes on both RouterA and RouterB to reach each others LANs (Ethernet interfaces), i.e. the 172.16.1.0/24 and 172.16.3.0/24 networks. We will use intermediate addresses as part of the static route.

Here is the routing table on RouterA before we created the static route. (Figure 11)

Figure 11

```
RouterA#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
<text omitted>

Gateway of last resort is not set

     172.16.0.0/24 is subnetted, 2 subnets
C       172.16.1.0 is directly connected, FastEthernet0
C       172.16.2.0 is directly connected, Serial0
RouterA#
```

Now lets create a static route on RouterA, using an intermediate, i.e. next-hop, address of RouterB's adjacent serial interface. (Figure 12) Figure 12 also shows the output from the **debug ip routing** command.

Figure 12

```
RouterA(config)#ip route 172.16.3.0 255.255.255.0 172.16.2.2
03:12:45: RT: add 172.16.3.0/24 via 172.16.2.2, static metric [1/0]
RouterA(config)#
```

The static route that we created uses an *intermediate IP address* of 172.16.2.2 (Figure 12). Sometimes this address is called the *next-hop IP address*, but the IP address does not have to be the physical next-hop. This is beyond the scope of this discussion, but as long as the intermediate IP address can be found in the routing table, it does not have to be the actual next-hop router's interface. Ultimately, the static network route, 172.16.3.0 in our example, must finally be resolved to a route in the routing table that has an exit interface. We will see this in a moment.

Lets examine the routing table of RouterA and see how this route was installed. (Figure 13)

Figure 13

```
RouterA#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
<text omitted>

Gateway of last resort is not set

     172.16.0.0/24 is subnetted, 3 subnets
C       172.16.1.0 is directly connected, FastEthernet0
C       172.16.2.0 is directly connected, Serial0
S       172.16.3.0 [1/0] via 172.16.2.2
RouterA#
```

The first thing to notice is that this static route becomes a child route within an existing parent route of 172.16.0.0. The parent route contains the subnet information for this static route, **/24**, which we will see the significance of in Part II. The parent route has been updated to include the third child route. The number of subnets has been incremented to, **3 subnets**.

Take a close look at our installed static route in the routing table. Do you notice anything different?

Notice that it does not contain an exit interface like the other routing table entries that we have discussed. Instead, the static routing table entry contains the intermediate address that we used when configuring the static route of **172.16.2.2**. This is an example of a child network route that is not an ultimate route, because it does not contain an exit interface.

So, how does a packet destined for the 172.16.3.0/24 network get resolved in the routing table?

This is called a recursive lookup. What happens is the routing table process finds the static entry for the **172.16.3.0** network (exact details coming in Part II).

```
S       172.16.3.0 [1/0] via 172.16.2.2
```

Because this routing table entry does not contain an exit interface, but instead an intermediate address, **via 172.16.2.2**, it cannot use the entry alone to forward the packets. Instead, the routing table process must take the intermediate address of 172.16.2.2 and do another, recursive, lookup in the routing table to find a route for this 172.16.2.0 network.

The routing table process finds the directly connected network entry for 172.16.2.0.

```
C       172.16.2.0 is directly connected, Serial0
```

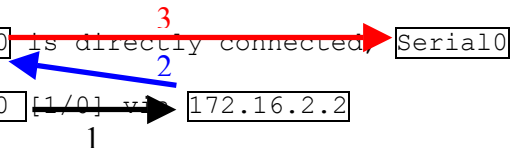
Because this entry does have an exit interface, **Serial0**, the routing table process can use this route to forward the packets for 172.16.3.0/24. But you have noticed that it took two routing table lookups, the first one for the 172.16.3.0 network, the destination IP address of the packet, and a second lookup for the intermediate address for intermediate address used in the entry.

```
(Second lookup of 172.16.2.0) C       172.16.2.0 is directly connected, Serial0
(First lookup of 172.16.3.0)  S       172.16.3.0 [1/0] via 172.16.2.2
```

Figure 14 shows the steps in of this recursive lookup process.

Figure 14

```
RouterA#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
<text omitted>

Gateway of last resort is not set

     172.16.0.0/24 is subnetted, 3 subnets

C       172.16.1.0 is directly connected, FastEthernet0

C       172.16.2.0 is directly connected, Serial0

S       172.16.3.0 [1/0] via 172.16.2.2

RouterA#
```



Using an exit interface

Are there a ways to avoid recursive table lookups? Yes there are and we will see how in just a moment. However, there may be times when recursive table lookups are preferred and configured by the network administrator. Again, this is beyond the scope of this discussion, but there can be instances where the network administrator only wants certain routes installed in the routing table if a specific, intermediate route is available.

Now, lets see how to avoid recursive table lookups. Most of the time, static routes over serial point-to-point networks can easily avoid recursive route lookups by using an exit interface instead of the intermediate or next-hop address. Figure 15 shows an example of creating a static route on RouterB for RouterA's LAN, using an exit interface instead of an intermediate address.

Figure 15

```
RouterB(config)#ip route 172.16.1.0 255.255.255.0 serial 0
05:05:30: RT: add 172.16.1.0/24 via 0.0.0.0, static metric [1/0]
RouterB(config)#
```

Lets take a look at how this route was installed in the routing table. (Figure 16)

Figure 16

```
RouterB#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
<text omitted>

Gateway of last resort is not set

     172.16.0.0/24 is subnetted, 3 subnets
S       172.16.1.0 is directly connected, Serial0
C       172.16.2.0 is directly connected, Serial0
C       172.16.3.0 is directly connected, FastEthernet0
C    192.168.1.0/24 is directly connected, Serial1
RouterB#
```

As with the static route using the intermediate address, this static route becomes a child network route of the **172.16.0.0** parent route. What is different is the static child route. Instead of using an intermediate address, this route is resolved with the exit interface, **Serial0** that we configured with the static route command.

Because this child route has an exit interface, it is considered an ultimate child route. The routing process, using this single entry, can forward any packets destined for the 172.16.3.0/24 network. No recursive route lookups needed. Because only a single routing table lookup is needed, instead of multiple, recursive lookups, this type of static route will have increase performance of the routing table process.

You might have also noticed that the routing table states that this static route is "directly connected."

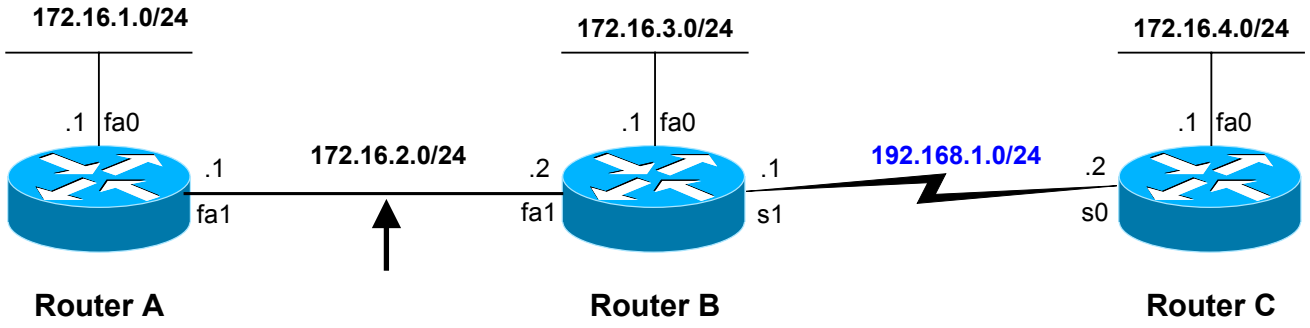
```
S       172.16.1.0 is directly connected, Serial0
```

In the case of a static route, directly connected means that the static route was configured with an exit interface. This does not mean that it is a directly connected network like an interface on the router. Like all static routes, the default administrative distance of this static route is still "1". Only directly connected interfaces, (which have a code of "C" in the routing table) can have an administrative distance of "0". Although it is possible to modify the administrative distance of static and dynamic routes, they cannot be given the administrative distance of "0". Likewise, directly connected interfaces can only have an administrative distance of "0".

Configuring a Static route over Ethernet interface

What about configuring static routes over multi-access networks like Ethernet? Using an exit-interface instead of an intermediate address causes a problem. Because the network is multi-access, there most likely are multiple devices, receivers, on sharing this network.

Lets assume that the network between RouterA and RouterB is a multi-access, Fast Ethernet link.



One solution would be to use an intermediate address, however this will cause a recursive routing table lookup. Figure 17 shows both the static route command and its installment in the routing table. When packets need to be routed for 172.16.1.0/14, the recursive route lookup happens first for the 172.16.1.0 network, and then for the intermediate address, with the lookup of 172.16.2.0 network.

Figure 17

```
RouterB(config)#ip route 172.16.1.0 255.255.255.0 172.16.2.1
RouterB(config)#end

RouterB#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
<text omitted>

Gateway of last resort is not set

     172.16.0.0/24 is subnetted, 3 subnets
S       172.16.1.0/24 [1/0] via 172.16.2.1
C       172.16.2.0 is directly connected, Serial0
C       172.16.3.0 is directly connected, FastEthernet0
C    192.168.1.0/24 is directly connected, Serial1
RouterB#
```

In order to avoid the recursive route lookup, the solution is to use both an intermediate address and an exit interface. Figure 18 shows the recommended way to configure a static route in this case, and the routing table entry. When using both the intermediate address and the exit-interface, only a single lookup is needed in the routing table lookup process.

Figure 18

```
RouterB(config)#ip route 172.16.1.0 255.255.255.0 fastethernet 1 172.16.2.1
RouterB(config)#end

RouterB#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
<text omitted>

Gateway of last resort is not set

     172.16.0.0/24 is subnetted, 3 subnets
S       172.16.1.0/24 [1/0] via 172.16.2.1 FastEthernet1
C       172.16.2.0 is directly connected, Serial0
C       172.16.3.0 is directly connected, FastEthernet0
C    192.168.1.0/24 is directly connected, Serial1
RouterB#
```

The standard rule-of-thumb when configuring static routes is to use an exit-interface over point-to-point and exit-interfaces with the intermediate address over mult-access networks. This will avoid the recursive route lookups caused by static routing entries that only contain an intermediate address.

Using VLSM

I won't go into what VLSM (Variable Length Subnet Mask) is, as that is explained in other parts of the Cisco curriculum. In this section we will focus only on the structure of the routing table when VLSM is used.

Lets use the diagram below, RouterX, to see the affect VLSM has on the routing table. We only need to use directly connected networks to learn what these affects are. The changes in the routing table (Figure 19) will occur automatically as soon as there are more than two or more child routes with different subnet masks. In other words, as whenever there are two or more child routes with different subnet masks, the routing table will present a different, "variably subnetted," look.

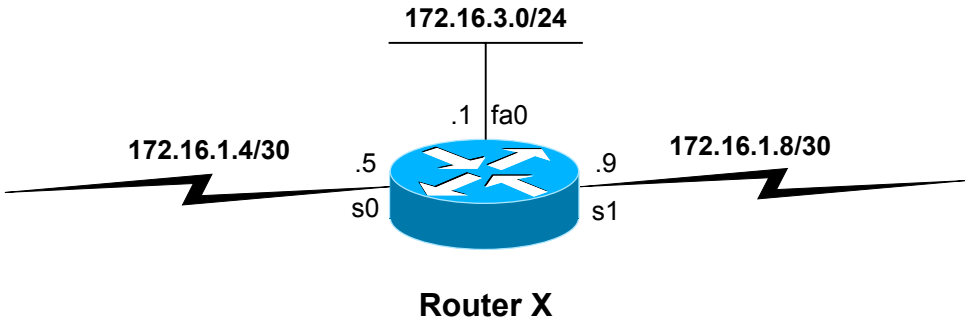


Figure 19

```
RouterX#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
<text omitted>

Gateway of last resort is not set

     172.16.0.0/16 is variably subnetted, 3 subnets, 2 masks
C       172.16.1.4/30 is directly connected, Serial0
C       172.16.1.8/30 is directly connected, Serial1
C       172.16.3.0/24 is directly connected, FastEthernet0
RouterX#
```

The first thing you may notice is that the parent network contains different information.

```
     172.16.0.0/16 is variably subnetted, 3 subnets, 2 masks
```

The parent route displays the classful network mask, **/16**, is instead of the subnet masks used by the child routes. This is because with VLSM, we will have more than one subnet mask for these different subnets. We no longer have one subnet mask that is the same for all of the child routes within this parent route.

The parent route will also show us that this classful network, **172.16.0.0**, is **variably subnetted**, with **3 subnets**, with **2 masks**.

There is also a difference in the display of the child routes.

```
C       172.16.1.4/30 is directly connected, Serial0
C       172.16.1.8/30 is directly connected, Serial1
C       172.16.3.0/24 is directly connected, FastEthernet0
```

The child routes now include the subnet mask for the child route. Since we have subnets with different subnet masks, the parent route can no longer display a single subnet mask for all the child routes. The mask must be displayed with the child route itself.

Next: Part II The Routing Table Lookup Process