CHAPTER 2

VLSM

Variable-length subnet masking (VLSM) is the more realistic way of subnetting a network to make for the most efficient use of all of the bits.

Remember that when you perform classful (or what I sometimes call classical) subnetting, all subnets have the same number of hosts because they all use the same subnet mask. This leads to inefficiencies. For example, if you borrow 4 bits on a Class C network, you end up with 14 valid subnets of 14 valid hosts. A serial link to another router only needs 2 hosts, but with classical subnetting you end up wasting 12 of those hosts. Even with the ability to use NAT and private addresses, where you should never run out of addresses in a network design, you still want to ensure that the IP plan that you create is as efficient as possible. This is where VLSM comes in to play.

VLSM is the process of "subnetting a subnet" and using different subnet masks for different networks in your IP plan. What you have to remember is that you need to make sure that there is no overlap in any of the addresses.

IP Subnet Zero

When you work with classical subnetting, you always have to eliminate the subnets that contain either all zeros or all ones in the subnet portion. Hence, you always used the formula $\mathbf{2^N - 2}$ to define the number of valid subnets created. However, Cisco devices can use those subnets, as long as the command **ip subnet-zero** is in the configuration. This command is on by default in Cisco IOS Software Release 12.0 and later; if it was turned off for some reason, however, you can re-enable it by using the following command:

```
Router(config)#ip subnet-zero
```

Now you can use the formula $\mathbf{2^N}$ rather than $\mathbf{2^N - 2}$.

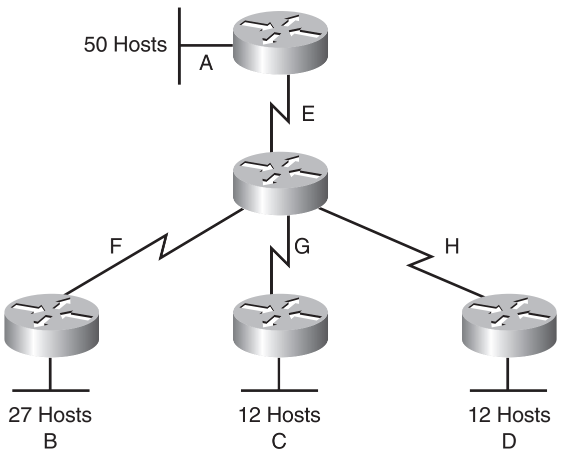
2^N	Number of total subnets created	
~~$2^N - 2$~~	~~Number of valid subnets created~~	No longer needed because you have the **ip subnet-zero** command enabled
2^H	Number of total hosts per subnet	
$2^H - 2$	Number of valid hosts per subnet	

VLSM Example

You follow the same steps in performing VLSM as you did when performing classical subnetting.

Consider Figure 2-1 as you work through an example.

Figure 2-1 Sample Network Needing a VLSM Address Plan



A Class C network—192.168.100.0/24—is assigned. You need to create an IP plan for this network using VLSM.

Once again, you cannot use the N bits—192.168.100. You can use only the H bits. Therefore, ignore the N bits, because they cannot change!

The steps to create an IP plan using VLSM for the network illustrated in Figure 2-1 are as follows:

Step 1 Determine how many H bits will be needed to satisfy the *largest* network.

Step 2 Pick a subnet for the largest network to use.

Step 3 Pick the next largest network to work with.

Step 4 Pick the third largest network to work with.

Step 5 Determine network numbers for serial links.

The remainder of the chapter details what is involved with each step of the process.

Step 1 Determine How Many H Bits Will Be Needed to Satisfy the *Largest* Network

A is the largest network with 50 hosts. Therefore, you need to know how many H bits will be needed:

If $2^H - 2$ = Number of valid hosts per subnet
Then $2^H - 2 \geq 50$
Therefore H = 6 (6 is the smallest valid value for H)

You need 6 H bits to satisfy the requirements of Network A.

If you need 6 H bits and you started with 8 N bits, you are left with 8 – 6 = 2 N bits to create subnets:

Started with: NNNNNNNN (these are the 8 bits in the fourth octet)
Now have: NNHHHHHH

All subnetting will now have to start at this reference point, to satisfy the requirements of Network A.

Step 2 Pick a Subnet for the Largest Network to Use

You have 2 N bits to work with, leaving you with 2^N or 2^2 or 4 subnets to work with:

NN = 00HHHHHH (The Hs = The 6 H bits you need for Network A)
01HHHHHH
10HHHHHH
11HHHHHH

If you add all zeros to the H bits, you are left with the network numbers for the four subnets:

00**000000** = .0
01**000000** = .64
10**000000** = .128
11**000000** = .192

All of these subnets will have the same subnet mask, just like in classful subnetting.

Two borrowed H bits means a subnet mask of:

11111111.11111111.11111111.11000000

or

255.255.255.192

or

/26

The /x notation represents how to show different subnet masks when using VLSM.

/8 means that the first 8 bits of the address are network, the remaining 24 bits are H bits.

/24 means that the first 24 bits are network, the last 8 are host—this is either a traditional default Class C address, or a traditional Class A network that has borrowed 16 bits, or even a traditional Class B network that has borrowed 8 bits!

Pick *one* of these subnets to use for Network A. The rest of the networks will have to use the other three subnets.

For purposes of this example, pick the .64 network.

00**000000** =	.0	
01**000000** =	.64	Network A
10**000000** =	.128	
11**000000** =	.192	

Step 3 Pick the Next Largest Network to Work With

Network B = 27 hosts

Determine the number of H bits needed for this network:

$2^H - 2 \geq 27$
$H = 5$

You need 5 H bits to satisfy the requirements of Network B.

You started with a pattern of 2 N bits and 6 H bits for Network A. You have to maintain that pattern.

Pick one of the remaining /26 networks to work with Network B.

For purposes of this example, select the .128/26 network:

10**000000**

But you need only 5 H bits, not 6. Therefore, you are left with:

10**N00000**

where:

10 represents the original pattern of subnetting.
N represents the extra bit.
00000 represents the 5 H bits you need for Network B.

Because you have this extra bit, you can create two smaller subnets from the original subnet:

10**000000**
10**100000**

Converted to decimal, these subnets are as follows:

10**000000** =.128
10**100000** =.160

You have now subnetted a subnet! This is the basis of VLSM.

Each of these sub-subnets will have a new subnet mask. The original subnet mask of /24 was changed into /26 for Network A. You then take one of these /26 networks and break it into two /27 networks:

10**000000** and 10**100000** both have 3 N bits and 5 H bits.

The mask now equals:

11111111.11111111.11111111.11100000

or

255.255.255.224

or

/27

Pick one of these new sub-subnets for Network B:

10**000000** /27 = Network B

Use the remaining sub-subnet for future growth, or you can break it down further if needed.

You want to make sure the addresses are not overlapping with each other. So go back to the original table.

00**000000** =	.0/26	
01**000000** =	.64/26	Network A
10**000000** =	.128/26	
11**000000** =	.192/26	

You can now break the .128/26 network into two smaller /27 networks and assign Network B.

00**000000** =	.0/26	
01**000000** =	.64/26	Network A
10**000000** =	.128/26	Cannot use because it has been subnetted
10**000000** =	.128/27	Network B
10**100000** =	.160/27	
11**000000** =	.192/26	

The remaining networks are still available to be assigned to networks, or subnetted further for better efficiency.

Step 4 Pick the Third Largest Network to Work With

Networks C and Network D = 12 hosts each

Determine the number of H bits needed for these networks:

$2^H - 2 \geq 12$
$H = 4$

You need 4 H bits to satisfy the requirements of Network C and Network D.

You started with a pattern of 2 N bits and 6 H bits for Network A. You have to maintain that pattern.

You now have a choice as to where to put these networks. You could go to a different /26 network, or you could go to a /27 network and try to fit them into there.

For the purposes of this example, select the other /27 network—.160/27:

101**00000** (The 1 in the third bit place is no longer bold, because it is part of the N bits.)

But you only need 4 H bits, not 5. Therefore you are left with:

101**N0000**

where:

10 represents the original pattern of subnetting.
N represents the extra bit you have.
00000 represents the 5 H bits you need for Network B.

Because you have this extra bit, you can create two smaller subnets from the original subnet:

101**00000**
101**10000**

Converted to decimal, these subnets are as follows:

101**00000** = .160
101**10000** = .176

These new sub-subnets will now have new subnet masks. Each sub-subnet now has 4 N bits and 4 H bits, so their new masks will be:

11111111.11111111.11111111.11110000

or

255.255.255.240

or

/28

Pick one of these new sub-subnets for Network C and one for Network D.

00**000000** =	.0/26	
01**000000** =	.64/26	Network A
10**000000** =	.128/26	Cannot use because it has been subnetted

10**000000 =**	.128/27	Network B
10**100000 =**	.160/27	Cannot use because it has been subnetted
101**00000**	.160/28	Network C
101**10000**	.176/28	Network D
11**000000 =**	.192/26	

You have now used two of the original four subnets to satisfy the requirements of four networks. Now all you need to do is determine the network numbers for the serial links between the routers.

Step 5 Determine Network Numbers for Serial Links

Serial links between routers all have the same property in that they only need two addresses in a network—one for each router interface.

Determine the number of H bits needed for these networks:

$2^H - 2 \geq 2$
$H = 2$

You need 2 H bits to satisfy the requirements of Networks E, F, G, and H.

You have two of the original subnets left to work with.

For purposes of this example, select the .0/26 network:

00**000000**

But you need only 2 H bits, not 6. Therefore, you are left with:

00**NNNN00**

where:

00 represents the original pattern of subnetting.
NNNN represents the extra bits you have.
00 represents the 2 H bits you need for the serial links.

Because you have 4 **N** bits, you can create 16 sub-subnets from the original subnet:

00**000000** = .0/30
00**000100** = .4/30
00**001000** = .8/30
00**001100** = .12/30
00**010000** = .16/30
.
.
.
00**111000** = .56/30

00**111100** = .60/30

You need only four of them. You can hold the rest for future expansion, or recombine them for a new, larger subnet:

00**010000** = .16/30
.
.
.
00**111000** = .56/30
00**111100** = .60/30

These can all be recombined into the following:

00**010000** = .16/28

Going back to the original table, you now have the following:

00**000000** =	.0/26	Cannot use because it has been subnetted
00**000000** =	.0/30	Network E
00**000100** =	.4/30	Network F
00**001000** =	.8/30	Network G
00**001100** =	.12/30	Network H
00**010000** =	.16/28	Future growth
01**000000** =	.64/26	Network A
10**000000** =	.128/26	Cannot use because it has been subnetted
10**000000** =	.128/27	Network B
10**100000** =	160/27	Cannot use because it has been subnetted
101**00000**	160/28	Network C
101**10000**	176/28	Network D
11**000000** =	.192/26	Future growth

Looking at the plan, you can see that no number is used twice. You have now created an IP plan for the network, and have made the plan as efficient as possible, wasting no addresses in the serial links and leaving room for future growth. This is the power of VLSM!