This subnetting scheme uses a difficult mask because one of the octets is not a 0 or a 255. The fourth octet is “interesting” in this case. The key part of the trick to get the right answers is to calculate the magic number, which is 256 – 192 = 64 in this case (256 – mask’s value in the interesting octet). The subnet number’s value in the interesting octet (inside the box) is the multiple of the magic number that is not higher than the original IP address’s value in the interesting octet. In this case, 192 is the multiple of 64 that is closest to 200 but not higher than 200. So, the fourth octet of the subnet number is 192.

The second part of this process calculates the subnet broadcast address, with the tricky part, as usual, in the “interesting” octet. Take the subnet number’s value in the interesting octet, add the magic number, and subtract 1. That is the broadcast address’s value in the interesting octet. In this case, it is 192 + 64 – 1 = 255.

You can easily forget that the subnet part of this address, when using this mask, actually covers all the third octet as well as 2 bits of the fourth octet. For example, the valid subnet numbers in order are listed here:

172.16.0.0 (zero subnet)

172.16.0.64

172.16.0.128

172.16.0.192

172.16.1.0

172.16.1.64

172.16.1.128

172.16.1.192

172.16.2.0

172.16.2.64

172.16.2.128

172.16.2.192

172.16.3.0

172.16.3.64

172.16.3.128

172.16.3.192

And so on.

Answer to Problem 25

Congratulations! You made it through the extra practice in this appendix! Here is an easy one to complete your review—one with no subnetting at all.

Table F-74 Question 25: Size of Network, Subnet, Host, Number of Subnets, and Number of Hosts

Item	Example	Rules to Remember
Address	10.1.1.1	—
Mask	255.0.0.0	—
Number of network bits	8	Always defined by Class A, B, C
Number of host bits	24	Always defined as number of binary 0s in mask
Number of subnet bits	0	32 – (network size + host size)
Number of subnets	0	$2^{\text{number-of-subnet-bits}}$
Number of hosts	$2^{24} - 2 = 16{,}777{,}214$	$2^{\text{number-of-host-bits}} - 2$

Table F-75 contains the important binary calculations for finding the subnet number and subnet broadcast address. To calculate the subnet number, perform a Boolean AND on the address and mask. To find the broadcast address for this subnet, change all the host bits to binary 1s in the subnet number. The host bits are in **bold** print in the table.

Table F-75 Question 25: Binary Calculation of Subnet and Broadcast Addresses

Address	10.1.1.1	00001010 **00000001 00000001 00000001**
Mask	255.0.0.0	11111111 **00000000 00000000 00000000**
AND result (subnet number)	10.0.0.0	00001010 **00000000 00000000 00000000**
Change host to 1s (broadcast address)	10.255.255.255	00001010 **11111111 11111111 11111111**

Just add 1 to the subnet number to get the first valid IP address; just subtract 1 from the broadcast address to get the last valid IP address. In this case:

10.0.0.1 through 10.255.255.254

Alternatively, you can use the processes that only use decimal math to find the subnet and broadcast address. Table F-76 shows the work for this problem.

Table F-76 Question 25: Subnet, Broadcast, and First and Last Addresses Calculated Using the Subnet Chart

	Octet 1	Octet 2	Octet 3	Octet 4
Mask	255	0	0	0
Address	10	1	1	1
Network Number	10	0	0	0
First Valid Address	10	0	0	1
Last Valid Address	10	255	255	254
Broadcast	10	255	255	255

APPENDIX G

Practice for Chapter 22: Fundamentals of IP Version 6

This appendix provides extra practice problems for two topics discussed in Chapter 22, "Fundamentals of IP Version 6," of the book. The first problems let you convert from a full 32-digit IPv6 address to its abbreviated form, or to do the reverse. The second set of problems begins with IPv6 addresses and prefix lengths, asking you to determine the IPv6 prefix (subnet).

Address Abbreviating and Expanding Problems

Chapter 22 discusses some reasons why you may need to be able to mentally convert from the full 32-digit IPv6 address to the abbreviated form, or vice versa. The practice problems in this section simply provide more opportunities to practice.

Table G-1 lists some practice problems, with the full 32-digit IPv6 address on the left and the best abbreviation on the right. The table gives you either the expanded or abbreviated address, and you need to supply the opposite value. The answers sit at the end of the appendix, in the section "Answers to Address Abbreviating and Expanding Problems."

Table G-1 IPv6 Address Abbreviation and Expansion Practice

	Full	Abbreviation
1	2987:BA11:B011:B00A:1000:0001:F001:F003	
2		3100::1010:D00D:D000:D00B:B00D
3	FD00:0001:0001:0001:0200:00FF:FE00:0001	
4		FDDF:8080:880:1001:0:FF:FE01:507
5	32CC:0000:0000:000D:210F:0000:0000:0000	
6		2100:E:E0::E00
7	3A11:CA00:0000:0000:0000:00FF:FECC:000C	
8		3799:9F9F:F000:0:FFFF::1
9	2A2A:0000:0000:0000:0000:0000:0000:2A2A	
10		3194::1:0:0:101
11	2001:0DB8:0000:0000:0001:0000:0002:0100	
12		2001:DB8::10:A000
13	3330:0000:0000:0100:0000:0002:0000:0003	
14		FD00::1000:2000:0:1:20
15	FD11:1000:0100:0010:0001:0000:1000:0100	
16		2000::2

Calculating the IPv6 Prefix Problems

Routers take the interface IPv6 address configuration and add a connected IPv6 route to the IPv6 routing table, for the IPv6 prefix (subnet) connected to that interface. This section provides some practice problems so that you can do the same math and predict the prefix value that the router will add to the routing table.

Table G-2 lists practice problems that all use the same prefix length (/64), which is the most common prefix length you see. Table G-3 that follows lists additional practice problems, with prefix lengths other than /64.

Table G-2 Finding the IPv6 Prefix When Using a /64 Prefix Length

	Address (Assume a /64 Prefix Length)	Prefix (Subnet)
1	2987:BA11:B011:B00A:1000:0001:F001:F003	
2	3100:0000:0000:1010:D00D:D000:D00B:B00D	
3	FD00:0001:0001:0001:0200:00FF:FE00:0001	
4	FDDF:8080:0880:1001:0000:00FF:FE01:0507	
5	32CC:0000:0000:000D:210F:0000:0000:0000	
6	2100:000E:00E0:0000:0000:0000:0000:0E00	
7	3A11:CA00:0000:0000:0000:00FF:FECC:000C	
8	3799:9F9F:F000:0000:FFFF:0000:0000:0001	
9	2A2A:0000:0000:0000:0000:0000:0000:2A2A	
10	3194:0000:0000:0000:0001:0000:0000:0101	
11	2001:0DB8:0000:0000:0001:0000:0002:0100	
12	2001:0DB8:0000:0000:0000:0000:0010:A000	
13	3330:0000:0000:0100:0000:0002:0000:0003	
14	FD00:0000:0000:1000:2000:0000:0001:0020	
15	FD11:1000:0100:0010:0001:0000:1000:0100	
16	2000:0000:0000:0000:0000:0000:0000:0002	

Table G-3 Finding the IPv6 Prefix Using a Prefix Length Other Than /64

	Address	Prefix (Subnet)
1	2987:BA11:B011:B00A:1000:0001:F001:F003 /60	
2	3100:0000:0000:1010:D00D:D000:D00B:B00D /56	
3	FD00:0001:0001:0001:0200:00FF:FE00:0001 /52	
4	FDDF:8080:0880:1001:0000:00FF:FE01:0507 /48	
5	32CC:0000:0000:000D:210F:0000:0000:0000 /44	
6	2100:000E:00E0:0000:0000:0000:0000:0E00 /60	
7	3A11:CA00:0000:0000:0000:00FF:FECC:000C /56	
8	3799:9F9F:F000:0000:FFFF:0000:0000:0001 /52	
9	2A2A:0000:0000:0000:0000:0000:0000:2A2A /48	
10	3194:0000:0000:0000:0001:0000:0000:0101 /44	

Answers to Address Abbreviating and Expanding Problems

Table G-4 lists the answers to the problems listed earlier in Table G-1.

Table G-4 Answers: IPv6 Address Abbreviation and Expansion Practice

	Full	Abbreviation
1	2987:BA11:B011:B00A:1000:0001:F001:F003	2987:BA11:B011:B00A:1000:1:F001:F003
2	3100:0000:0000:1010:D00D:D000:D00B:B00D	3100::1010:D00D:D000:D00B:B00D
3	FD00:0001:0001:0001:0200:00FF:FE00:0001	FD00:1:1:1:200:FF:FE00:1
4	FDDF:8080:0880:1001:0000:00FF:FE01:0507	FDDF:8080:880:1001:0:FF:FE01:507
5	32CC:0000:0000:000D:210F:0000:0000:0000	32CC:0:0:D:210F::
6	2100:000E:00E0:0000:0000:0000:0000:0E00	2100:E:E0::E00
7	3A11:CA00:0000:0000:0000:00FF:FECC:000C	3A11:CA00::FF:FECC:C
8	3799:9F9F:F000:0000:FFFF:0000:0000:0001	3799:9F9F:F000:0:FFFF::1
9	2A2A:0000:0000:0000:0000:0000:0000:2A2A	2A2A::2A2A
10	3194:0000:0000:0000:0001:0000:0000:0101	3194::1:0:0:101
11	2001:0DB8:0000:0000:0001:0000:0002:0100	2001:DB8::1:0:2:100
12	2001:0DB8:0000:0000:0000:0000:0010:A000	2001:DB8::10:A000
13	3330:0000:0000:0100:0000:0002:0000:0003	3330::100:0:2:0:3
14	FD00:0000:0000:1000:2000:0000:0001:0020	FD00::1000:2000:0:1:20
15	FD11:1000:0100:0010:0001:0000:1000:0100	FD11:1000:100:10:1:0:1000:100
16	2000:0000:0000:0000:0000:0000:0000:0002	2000::2

Answers to Calculating IPv6 Prefix Problems

Tables G-5 and G-6 list the answers to the problems listed earlier in Tables G-2 and G-3.

Table G-5 Answers: Finding the IPv6 Prefix, with a /64 Prefix Length

	Address (Assume a /64 Prefix Length)	Prefix (Subnet)
1	2987:BA11:B011:B00A:1000:0001:F001:F003	2987:BA11:B011:B00A::/64
2	3100:0000:0000:1010:D00D:D000:D00B:B00D	3100:0:0:1010::/64
3	FD00:0001:0001:0001:0200:00FF:FE00:0001	FD00:1:1:1::/64
4	FDDF:8080:0880:1001:0000:00FF:FE01:0507	FDDF:8080:880:1001::/64
5	32CC:0000:0000:000D:210F:0000:0000:0000	32CC:0:0:D::/64
6	2100:000E:00E0:0000:0000:0000:0000:0E00	2100:E:E0::/64
7	3A11:CA00:0000:0000:0000:00FF:FECC:000C	3A11:CA00::/64
8	3799:9F9F:F000:0000:FFFF:0000:0000:0001	3799:9F9F:F000::/64
9	2A2A:0000:0000:0000:0000:0000:0000:2A2A	2A2A::/64
10	3194:0000:0000:0000:0001:0000:0000:0101	3194::/64
11	2001:0DB8:0000:0000:0001:0000:0002:0100	2001:DB8::/64
12	2001:0DB8:0000:0000:0000:0000:0010:A000	2001:DB8::/64
13	3330:0000:0000:0100:0000:0002:0000:0003	3330:0:0:100::/64
14	FD00:0000:0000:1000:2000:0000:0001:0020	FD00:0:0:1000::/64
15	FD11:1000:0100:0010:0001:0000:1000:0100	FD11:1000:100:10::/64
16	2000:0000:0000:0000:0000:0000:0000:0002	2000::/64

Table G-6 Answers: Finding the IPv6 Prefix, with Other Prefix Lengths

	Address	Prefix (Subnet)
1	2987:BA11:B011:B00A:1000:0001:F001:F003 /60	2987:BA11:B011:B000::/60
2	3100:0000:0000:1010:D00D:D000:D00B:B00D /56	3100:0:0:1000::/56
3	FD00:0001:0001:0001:0200:00FF:FE00:0001 /52	FD00:1:1::/52
4	FDDF:8080:0880:1001:0000:00FF:FE01:0507 /48	FDDF:8080:880::/48
5	32CC:0000:0000:000D:210F:0000:0000:0000 /44	32CC::/44
6	2100:000E:00E0:0000:0000:0000:0000:0E00 /60	2100:E:E0::/60
7	3A11:CA00:0000:0000:0000:00FF:FECC:000C /56	3A11:CA00::/56
8	3799:9F9F:F000:0000:FFFF:0000:0000:0001 /52	3799:9F9F:F000::/52
9	2A2A:0000:0000:0000:0000:0000:0000:2A2A /48	2A2A::/48
10	3194:0000:0000:0000:0001:0000:0000:0101 /44	3194::/44

APPENDIX H

Practice for Chapter 24: Implementing IPv6 Addressing on Routers

This appendix provides practice problems for two types of addresses: unicast addresses formed with the EUI-64 feature and solicited node multicast addresses. With EUI-64, you take the 64-bit (16 hex digit) prefix and a MAC address, manipulate the MAC address into a 64-bit value, and use those 64 bits as the interface ID. Solicited node multicast addresses are formed from a standard 26 hex digit prefix, combined with the same last 6 hex digits as the unicast address.

EUI-64 and Solicited Node Multicast Problems

Table H-1 lists some practice problems. Each problem lists a prefix and a MAC address. Then, in Table H-2, record your answers for the unicast IPv6 address, assuming that EUI-64 rules are used. Also in Table H-2, list the solicited node multicast address associated with your calculated unicast address.

For each answer, use the best abbreviation, instead of a full 32-digit address.

The answers sit at the end of the appendix, in Table H-3.

Table H-1 IPv6 EUI-64 Unicast and Solicited Node Multicast Problems

	Prefix	MAC Address
1	2987:BA11:B011:B00A::/64	0000.1234.5678
2	3100:0000:0000:1010::/64	1234.5678.9ABC
3	FD00:0001:0001:0001::/64	0400.AAAA.0001
4	FDDF:8080:0880:1001::/64	0611.BABA.DADA
5	32CC:0000:0000:000D::/64	0000.0000.0001
6	2100:000E:00E0:0000::/64	0505.0505.0707
7	3A11:CA00:0000:0000::/64	0A0A.B0B0.0C0C
8	3799:9F9F:F000:0000::/64	F00F.0005.0041
9	2A2A:0000:0000:0000::/64	0200.0101.0101
10	3194:0000:0000:0000::/64	0C0C.000C.00CC

Table H-2 Blank Answer Table for Problems in Table H-1

	Unicast Address Using EUI-64	Solicited Node Multicast Address
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

Answers to EUI-64 and Solicited Node Multicast Problems

Table H-3 lists the answers to the problems listed earlier in Table H-1.

Table H-3 Answers to Problems in Table H-1

	Unicast Address Using EUI-64	Solicited Node Multicast Address
1	2987:BA11:B011:B00A:200:12FF:FE34:5678	FF02::01:FF34.5678
2	3100::1010:1034:56FF:FE78:9ABC	FF02::01:FF78.9ABC
3	FD00:1:1:1:600:AAFF:FEAA:1	FF02::01:FFAA:1
4	FDDF:8080:880:1001:411:BAFF:FEBA:DADA	FF02::01:FFBA:DADA
5	32CC::D:200:FF:FE00:1	FF02::01:FF00:1
6	2100:E:E0:0:705:5FF:FE05:707	FF02::01:FF05:707
7	3A11:CA00::80A:B0FF:FEB0:C0C	FF02::01:FFB0:C0C
8	3799:9F9F:F000:0:F20F:FF:FE05:41	FF02::01:FF05:41
9	2A2A::1FF:FE01:101	FF02::01:FF01:101
10	3194::E0C:FF:FE0C:CC	FF02::01:FF0C:CC

Appendix I

Study Planner

Practice Test	Reading	Task

Element	Task	Goal Date	First Date Completed	Second Date Completed (Optional)	Notes
Introduction	Read Introduction				
Your Study Plan	Read Your Study Plan				
1. Introduction to TCP/IP Networking	Read Foundation Topics				
1. Introduction to TCP/IP Networking	Review Key Topics using the book or companion website				
1. Introduction to TCP/IP Networking	Define Key Terms using the book or companion website				
1. Introduction to TCP/IP Networking	Repeat DIKTA questions using the book or PTP exam engine				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
2. Fundamentals of Ethernet LANs	Read Foundation Topics				
2. Fundamentals of Ethernet LANs	Review Key Topics using the book or companion website				
2. Fundamentals of Ethernet LANs	Define Key Terms using the book or companion website				
2. Fundamentals of Ethernet LANs	Repeat DIKTA questions using the book or PTP exam engine				
2. Fundamentals of Ethernet LANs	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
3. Fundamentals of WANs and IP Routing	Read Foundation Topics				
3. Fundamentals of WANs and IP Routing	Review Key Topics using the book or companion website				

3. Fundamentals of WANs and IP Routing	Define Key Terms using the book or companion website				
3. Fundamentals of WANs and IP Routing	Repeat DIKTA questions using the book or PTP exam engine				
3. Fundamentals of WANs and IP Routing	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part I. Introduction to Networking	Complete all exercises in Part I Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
4. Using the Command-Line Interface	Read Foundation Topics				
4. Using the Command-Line Interface	Review Key Topics using the book or companion website				
4. Using the Command-Line Interface	Define Key Terms using the book or companion website				
4. Using the Command-Line Interface	Repeat DIKTA questions using the book or PTP exam engine				
4. Using the Command-Line Interface	Review the command tables				
4. Using the Command-Line Interface	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this chapter				
5. Analyzing Ethernet LAN Switching	Read Foundation Topics				
5. Analyzing Ethernet LAN Switching	Review Key Topics using the book or companion website				
5. Analyzing Ethernet LAN Switching	Define Key Terms using the book or companion website				
5. Analyzing Ethernet LAN Switching	Repeat DIKTA questions using the book or PTP exam engine				
5. Analyzing Ethernet LAN Switching	Do labs listed for this chapter using the Sim Lite app				
5. Analyzing Ethernet LAN Switching	Review the command tables				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this chapter				
6. Configuring Basic Switch Management	Read Foundation Topics				

6. Configuring Basic Switch Management	Review Key Topics using the book or companion website				
6. Configuring Basic Switch Management	Define Key Terms using the book or companion website				
6. Configuring Basic Switch Management	Repeat DIKTA questions using the book or PTP exam engine				
6. Configuring Basic Switch Management	Complete config checklists in this chapter using the companion website				
6. Configuring Basic Switch Management	Do labs listed for this chapter using the Sim Lite app				
6. Configuring Basic Switch Management	Review command tables for this chapter				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
7. Configuring and Verifying Switch Interfaces	Read Foundation Topics				
7. Configuring and Verifying Switch Interfaces	Review Key Topics using the book or companion website				
7. Configuring and Verifying Switch Interfaces	Define Key Terms using the book or companion website				
7. Configuring and Verifying Switch Interfaces	Repeat DIKTA questions using the book or PTP exam engine				
7. Configuring and Verifying Switch Interfaces	Review command tables for this chapter				
7. Configuring and Verifying Switch Interfaces	Complete all memory tables in this chapter using the companion website				
7. Configuring and Verifying Switch Interfaces	Do labs listed for this chapter using the Sim Lite app				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part II. Implementing Ethernet LANs	Complete all exercises in Part II Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
8. Implementing Ethernet Virtual LANs	Read Foundation Topics				
8. Implementing Ethernet Virtual LANs	Review Key Topics using the book or companion website				
8. Implementing Ethernet Virtual LANs	Define Key Terms using the book or companion website				
8. Implementing Ethernet Virtual LANs	Repeat DIKTA questions using the book or PTP exam engine				

8. Implementing Ethernet Virtual LANs	Complete config checklists in this chapter using the companion website				
8. Implementing Ethernet Virtual LANs	Review command tables for this chapter				
8. Implementing Ethernet Virtual LANs	Complete all memory tables in this chapter using the companion website				
8. Implementing Ethernet Virtual LANs	Do labs listed for this chapter using the Sim Lite app				
8. Implementing Ethernet Virtual LANs	Watch video for this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
9. Spanning Tree Protocol Concepts	Read Foundation Topics				
9. Spanning Tree Protocol Concepts	Review Key Topics using the book or companion website				
9. Spanning Tree Protocol Concepts	Define Key Terms using the book or companion website				
9. Spanning Tree Protocol Concepts	Repeat DIKTA questions using the book or PTP exam engine				
9. Spanning Tree Protocol Concepts	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
10. RSTP and EtherChannel Configuration	Read Foundation Topics				
10. RSTP and EtherChannel Configuration	Review Key Topics using the book or companion website				
10. RSTP and EtherChannel Configuration	Define Key Terms using the book or companion website				
10. RSTP and EtherChannel Configuration	Repeat DIKTA questions using the book or PTP exam engine				
10. RSTP and EtherChannel Configuration	Complete config checklists in this chapter using the companion website				
10. RSTP and EtherChannel Configuration	Review command tables for this chapter				
10. RSTP and EtherChannel Configuration	Complete all memory tables in this chapter using the companion website				
10. RSTP and EtherChannel Configuration	Do labs listed for this chapter using the Sim Lite app				

Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part III. Implementing VLANs and STP	Complete all exercises in Part III Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
11. Perspectives on IPv4 Subnetting	Read Foundation Topics				
11. Perspectives on IPv4 Subnetting	Review Key Topics using the book or companion website				
11. Perspectives on IPv4 Subnetting	Define Key Terms using the book or companion website				
11. Perspectives on IPv4 Subnetting	Repeat DIKTA questions using the book or PTP exam engine				
11. Perspectives on IPv4 Subnetting	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
12. Analyzing Classful IPv4 Networks	Read Foundation Topics				
12. Analyzing Classful IPv4 Networks	Review Key Topics using the book or companion website				
12. Analyzing Classful IPv4 Networks	Define Key Terms using the book or companion website				
12. Analyzing Classful IPv4 Networks	Repeat DIKTA questions using the book or PTP exam engine				
12. Analyzing Classful IPv4 Networks	Complete all memory tables in this chapter using the companion website				
12. Analyzing Classful IPv4 Networks	Practice analyzing classful IPv4 networks using Appendix D on the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
13. Analyzing Subnet Masks	Read Foundation Topics				
13. Analyzing Subnet Masks	Review Key Topics using the book or companion website				
13. Analyzing Subnet Masks	Define Key Terms using the book or companion website				
13. Analyzing Subnet Masks	Repeat DIKTA questions using the book or PTP exam engine				

13. Analyzing Subnet Masks	Complete all memory tables in this chapter using the companion website				
13. Analyzing Subnet Masks	Practice analyzing subnet masks using Appendix E on the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
14. Analyzing Existing Subnets	Read Foundation Topics				
14. Analyzing Existing Subnets	Review Key Topics using the book or companion website				
14. Analyzing Existing Subnets	Define Key Terms using the book or companion website				
14. Analyzing Existing Subnets	Repeat DIKTA questions using the book or PTP exam engine				
14. Analyzing Existing Subnets	Complete all memory tables in this chapter using the companion website				
14. Analyzing Existing Subnets	Practice mask analysis using Appendix F on the companion website				
14. Analyzing Existing Subnets	Practice analyzing existing subnets using Appendix F on the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part IV. IPv4 Addressing	Complete all exercises in Part IV Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
15. Operating Cisco Routers	Read Foundation Topics				
15. Operating Cisco Routers	Review Key Topics using the book or companion website				
15. Operating Cisco Routers	Define Key Terms using the book or companion website				
15. Operating Cisco Routers	Repeat DIKTA questions using the book or PTP exam engine				
15. Operating Cisco Routers	Review command tables for this chapter				
15. Operating Cisco Routers	Complete all memory tables in this chapter using the companion website				
15. Operating Cisco Routers	Do labs listed for this chapter using the Sim Lite app				
15. Operating Cisco Routers	Watch video for this chapter using the companion website				

Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
16. Configuring IPv4 Addresses and Static Routes	Read Foundation Topics				
16. Configuring IPv4 Addresses and Static Routes	Review Key Topics using the book or companion website				
16. Configuring IPv4 Addresses and Static Routes	Define Key Terms using the book or companion website				
16. Configuring IPv4 Addresses and Static Routes	Repeat DIKTA questions using the book or PTP exam engine				
16. Configuring IPv4 Addresses and Static Routes	Review command tables for this chapter				
16. Configuring IPv4 Addresses and Static Routes	Do labs listed for this chapter using the Sim Lite app				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
17. IP Routing in the LAN	Read Foundation Topics				
17. IP Routing in the LAN	Review Key Topics using the book or companion website				
17. IP Routing in the LAN	Define Key Terms using the book or companion website				
17. IP Routing in the LAN	Repeat DIKTA questions using the book or PTP exam engine				
17. IP Routing in the LAN	Complete config checklists in this chapter using the companion website				
17. IP Routing in the LAN	Review command tables for this chapter				
17. IP Routing in the LAN	Do labs listed for this chapter using the Sim Lite app				
17. IP Routing in the LAN	Watch video for this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
18. Troubleshooting IPv4 Routing	Read Foundation Topics				
18. Troubleshooting IPv4 Routing	Review Key Topics using the book or companion website				
18. Troubleshooting IPv4 Routing	Define Key Terms using the book or companion website				
18. Troubleshooting IPv4 Routing	Watch video for this chapter using the companion website				

Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part V. IPv4 Routing	Complete all exercises in Part V Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
19. Understanding OSPF Concepts	Read Foundation Topics				
19. Understanding OSPF Concepts	Review Key Topics using the book or companion website				
19. Understanding OSPF Concepts	Define Key Terms using the book or companion website				
19. Understanding OSPF Concepts	Repeat DIKTA questions using the book or PTP exam engine				
19. Understanding OSPF Concepts	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
20. Implementing OSPF	Read Foundation Topics				
20. Implementing OSPF	Review Key Topics using the book or companion website				
20. Implementing OSPF	Define Key Terms using the book or companion website				
20. Implementing OSPF	Repeat DIKTA questions using the book or PTP exam engine				
20. Implementing OSPF	Complete config checklists in this chapter using the companion website				
20. Implementing OSPF	Review command tables for this chapter				
20. Implementing OSPF	Do labs listed for this chapter using the Sim Lite app				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
21. OSPF Network Types and Neighbors	Read Foundation Topics				
21. OSPF Network Types and Neighbors	Review Key Topics using the book or companion website				
21. OSPF Network Types and Neighbors	Repeat DIKTA questions using the book or PTP exam engine				

21. OSPF Network Types and Neighbors	Complete all memory tables in this chapter using the companion website				
21. OSPF Network Types and Neighbors	Watch video for this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part VI. OSPF	Complete all exercises in Part VI Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
22. Fundamentals of IP Version 6	Read Foundation Topics				
22. Fundamentals of IP Version 6	Review Key Topics using the book or companion website				
22. Fundamentals of IP Version 6	Define Key Terms using the book or companion website				
22. Fundamentals of IP Version 6	Repeat DIKTA questions using the book or PTP exam engine				
22. Fundamentals of IP Version 6	Review command tables for this chapter				
22. Fundamentals of IP Version 6	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
23. IPv6 Addressing and Subnetting	Read Foundation Topics				
23. IPv6 Addressing and Subnetting	Review Key Topics using the book or companion website				
23. IPv6 Addressing and Subnetting	Define Key Terms using the book or companion website				
23. IPv6 Addressing and Subnetting	Repeat DIKTA questions using the book or PTP exam engine				
23. IPv6 Addressing and Subnetting	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
24. Implementing IPv6 Addressing on Routers	Read Foundation Topics				
24. Implementing IPv6 Addressing on Routers	Review Key Topics using the book or companion website				
24. Implementing IPv6 Addressing on Routers	Define Key Terms using the book or companion website				

24. Implementing IPv6 Addressing on Routers	Repeat DIKTA questions using the book or PTP exam engine				
24. Implementing IPv6 Addressing on Routers	Review command tables for this chapter				
24. Implementing IPv6 Addressing on Routers	Complete all memory tables in this chapter using the companion website				
24. Implementing IPv6 Addressing on Routers	Do labs listed for this chapter using the Sim Lite app				
24. Implementing IPv6 Addressing on Routers	Watch video for this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
25. Implementing IPv6 Routing	Read Foundation Topics				
25. Implementing IPv6 Routing	Review Key Topics using the book or companion website				
25. Implementing IPv6 Routing	Define Key Terms using the book or companion website				
25. Implementing IPv6 Routing	Repeat DIKTA questions using the book or PTP exam engine				
25. Implementing IPv6 Routing	Review command tables for this chapter				
25. Implementing IPv6 Routing	Complete all memory tables in this chapter using the companion website				
25. Implementing IPv6 Routing	Do labs listed for this chapter using the author's blog site				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part VII. IP Version 6	Complete all exercises in Part VII Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				
26. Fundamentals of Wireless Networks	Read Foundation Topics				
26. Fundamentals of Wireless Networks	Review Key Topics using the book or companion website				
26. Fundamentals of Wireless Networks	Define Key Terms using the book or companion website				
26. Fundamentals of Wireless Networks	Repeat DIKTA questions using the book or PTP exam engine				
26. Fundamentals of Wireless Networks	Complete all memory tables in this chapter using the companion website				

Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
27. Analyzing Cisco Wireless Architectures	Read Foundation Topics				
27. Analyzing Cisco Wireless Architectures	Review Key Topics using the book or companion website				
27. Analyzing Cisco Wireless Architectures	Define Key Terms using the book or companion website				
27. Analyzing Cisco Wireless Architectures	Repeat DIKTA questions using the book or PTP exam engine				
27. Analyzing Cisco Wireless Architectures	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
28. Securing Wireless Networks	Read Foundation Topics				
28. Securing Wireless Networks	Review Key Topics using the book or companion website				
28. Securing Wireless Networks	Define Key Terms using the book or companion website				
28. Securing Wireless Networks	Repeat DIKTA questions using the book or PTP exam engine				
28. Securing Wireless Networks	Complete all memory tables in this chapter using the companion website				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
29. Building a Wireless LAN	Read Foundation Topics				
29. Building a Wireless LAN	Review Key Topics using the book or companion website				
29. Building a Wireless LAN	Define Key Terms using the book or companion website				
29. Building a Wireless LAN	Repeat DIKTA questions using the book or PTP exam engine				
Practice Test	Take practice test in study mode using DIKTA exam in practice test software for this chapter				
Part VIII. Wireless LANs	Complete all exercises in Part VII Review				
Practice Test	Take practice test in study mode using Part Review exam in practice test software for this part				

Final Review	Take practice test in study mode for all Book Questions in practice test software				
Final Review	Review all Key Topics in all chapters or in the Key Topics App using the companion website				
Final Review	Review all Key Terms in all chapters or using the Key Terms Flashcards on the companion website				
Final Review	Complete all memory tables for all chapters using the companion website				
Final Review	Take practice test in practice exam mode using Exam Bank #1 questions for all chapters				
Final Review	Take practice test in practice exam mode using Exam Bank #2 questions for all chapters				

Topics from Previous Editions

Cisco changes the exams, renaming the exams on occasion, and changing the exam numbers every time it changes the exam with a new blueprint, even with a few name changes over the years. As a result, the current CCNA 200-301 exam serves as the eighth separate version of CCNA in its 20-plus year history. At every change to the exams, we create new editions of the books to match the new exam.

We base the books' contents on Cisco's exam topics; that is, the book attempts to cover the topics Cisco lists as exam topics. However, the book authoring process does create some challenges, particularly with the balance of what to include in the books and what to leave out.

For instance, when comparing a new exam to the old, I found Cisco had removed some topics—and I might want to keep the content in the book. There are a few reasons why. Sometimes I just expect that some readers will still want to read about that technology. Also, more than a few schools use these books as textbooks, and keeping some of the older-but-still-relevant topics can be a help. And keeping the old material available on each book's companion website takes only a little extra work, so we do just that.

Some of the older topics that I choose to keep on the companion website are small, so I collect them into this appendix. Other topics happen to have been an entire chapter in a previous edition of the books, so we include those topics each as a separate appendix. Regardless, the material exists here in this appendix, and in the appendices that follow, for your use if you have a need. But do not feel like you have to read this appendix for the current exam.

The topics in this appendix are as follows:

- IPv4 Address Types
- Bandwidth and Clock Rate on Serial Interfaces
- Using traceroute to Isolate the Problem to Two Routers
- Troubleshooting Static IPv6 Routes
- Default Routes with SLAAC on Router Interfaces

NOTE The content under the heading "IPv4 Address Types" was most recently published for the 100-105 Exam in 2016, in Chapter 20 of the *Cisco CCNA ICND1 100-105 Official Cert Guide*.

IPv4 Address Types

The IPv4 address space includes three major categories of addresses: unicast, broadcast, and multicast. For the current exam, Cisco lists one exam topic that asks you to compare and contrast these address types. To help you make those comparisons, this section explains multicast addressing, while pulling together the key ideas about unicast and broadcast IP addresses that have already been introduced, to pull the ideas together.

You may be wondering why this topic about IPv4 address types sits at the end of a chapter about DHCP and IP networking on hosts. Honestly, I could have put this topic in several chapters. The main reason it is here is that you have already seen the IP broadcast addresses in action, including the 255.255.255.255 local broadcast as shown in this chapter.

Review of Unicast (Class A, B, and C) IP Addresses

Unicast IP addresses are those Class A, B, and C IP addresses assigned to hosts, router interfaces, and other networking devices. Because most discussions about IP addressing refer to unicast IP addresses, most of us just refer to them as IP addresses, and leave out the word *unicast*.

Just to be complete and define the concept, unicast addresses identify one interface on one device to IP. Just like your postal address gives the post office an address to use to send letters to your one specific house or apartment, a unicast IP address gives the IP network an address to use to send packets to one specific host. However, with IP, instead of addressing the device, unicast addresses identify individual interfaces. For example:

- A router with four LAN interfaces, and two WAN interfaces, has six unicast addresses, each in a different subnet, one for each interface.
- A PC with both an Ethernet network interface card (NIC) and a wireless NIC would have two unicast IPv4 addresses, one for each interface.

IP Broadcast Addresses

Broadcast IPv4 addresses give IP a way to send one packet that the network delivers to multiple hosts. IPv4 defines several types of broadcast addresses, with each type being used to reach a different set of hosts. These different broadcast IP addresses give different overhead protocols like DHCP the ability to efficiently reach all hosts in a specific part of the network. The following list reviews the three IP broadcast address types:

Key Topic

Local broadcast address: 255.255.255.255. Used to send a packet on a local subnet, knowing that routers will not forward the packet as is. Also called a *limited broadcast*.

Subnet broadcast address: One reserved address for each subnet, namely the numerically highest number in the subnet, as discussed in Chapter 13, "Analyzing Subnet Masks." A packet sent to a subnet broadcast address can be routed to the router connected to that subnet, and then sent as a data-link broadcast to all hosts in that one subnet. Also called an *all-hosts broadcast* to emphasize that all hosts in a subnet are reached, and also called a *directed broadcast*.

Network broadcast address: One reserved address for each classful network, namely the numerically highest number in the network. Used to send one packet to all hosts in that one network. Also called an *all-subnets broadcast*, referring to the fact that the packet reaches all subnets in a network.

This chapter has already shown how a local broadcast works, sending the message over the same subnet in which it was first transmitted, but no further. However, the other two types are a little more interesting.

Subnet and network broadcasts provide a way to send packets to all hosts in a subnet or network (respectively) while reducing waste. For instance, with a subnet broadcast, routers forward the packet just like any other IP packet going to that subnet. When that packet arrives at the router connected to that subnet, the last router then encapsulates the packet in a LAN broadcast, so that all hosts receive a copy. Figure J-1 shows the idea.

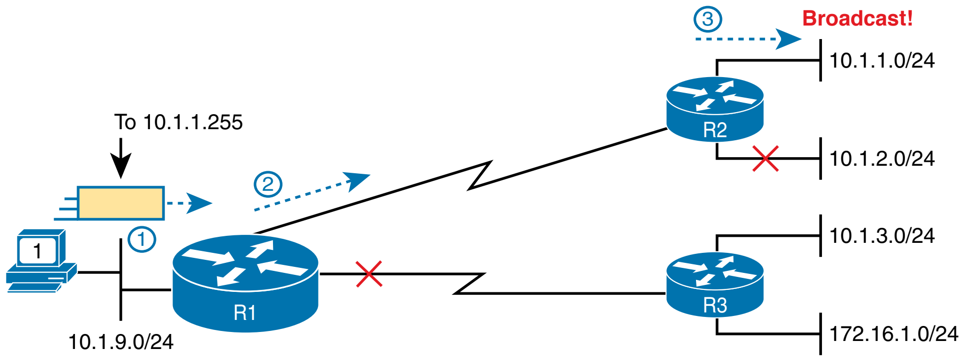

Figure J-1 *Example of a Subnet Broadcast to 10.1.1.255*

The figure shows two key points. R1 does not flood or broadcast the frame to all other routers, instead routing it to the next router (R2 in this case) so that the packet reaches subnet 10.1.1.0/24. R2, connected to subnet 10.1.1.0/24, forwards the packet onto the LAN, but encapsulates the packet in an Ethernet broadcast frame, so that it reaches all hosts in the subnet.

The figure shows the intended use of the subnet broadcast address; however, it presents a security issue today. Many attacks start with a ping to subnet broadcast addresses, hoping to get many hosts to reply. Cisco changed the IOS default many years ago to disable the forwarding of subnet broadcasts onto a connected subnet (that is, it disables Step 3 in Figure J-1). That default setting is based on the **no ip directed-broadcast** interface subcommand.

A network broadcast packet (a packet with a network broadcast address as the destination) works in a similar way. To reach all subnets, however, the routers create copies of the packet and flood it so it reaches all subnets inside the classful network. On any LAN interfaces, the packet is forwarded in a LAN broadcast, just as shown in Step 3 of Figure J-1.

IPv4 Multicast Addresses (Class D Addresses)

Multicast IP addresses and the related protocols help solve a similar problem as compared to broadcast addresses, but mainly for applications, and without the same security issues experienced by broadcast addresses. To see how it works, consider this example. A video application may be designed to show live video feeds. If 10 people at the same remote site in the same subnet want to watch the same video at the same time, the application could be designed so that the application sent the same video data 10 times, once to each client in the same subnet. An application designed to use Class D multicast addresses could send 1 packet, which the routers would route across the WAN, and then deliver a copy to all 10 hosts in the destination subnet.

When using multicast, all the hosts still use their individual unicast IP address for their normal traffic, while also using the same multicast IPv4 address for the multicast application. Any server or client that happens to use an application designed to take advantage of IP multicast then also uses the Class D multicast addresses that the application chooses to use. You can think of a Class D address more as a multicast group—in fact, it is often called that—because hosts join the group so that they can receive the packets sent by the multicast application.

Class D addresses begin with a first octet of between 224 and 239, with some ranges reserved for various purposes. Much of the Class D address space is set aside for a company to deploy one of these multicast applications, and then pick an address from the Class D range, and configure it to be used by a multicast application.

As an example, imagine the video application uses Class D address 226.1.1.1. Figure J-2 illustrates the process by which the application at the server on the left sends one multicast packet with destination address 226.1.1.1. Note that for this process to work, the hosts with * beside them registered with their local routers to notify the routers that the host wants to receive packets destined to multicast address 226.1.1.1. When the action in this figure begins, the routers collectively know which subnets have hosts that want a copy of multicasts sent to 226.1.1.1, and which subnets do not.

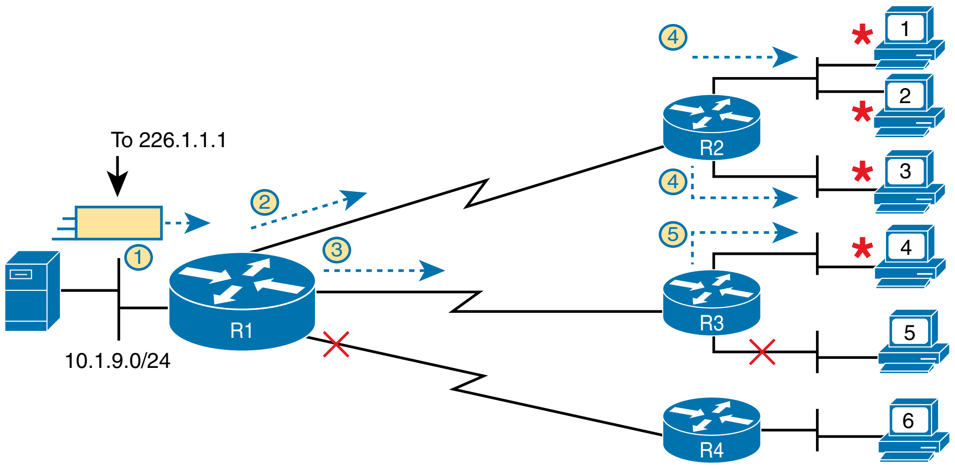

Figure J-2 *Example of a Multicast Packet Flow for Three Registered Hosts*

Following the steps in the figure:

1. The server on the left generates and sends a multicast packet.
2. Router R1 replicates the packet to send a copy to both R2…
3. …and to R3. R1 does not replicate and send a copy to R4, because there are no hosts near R4 listening for packets sent to 226.1.1.1.
4. R2 processes the multicast packet received from R1, and because of the earlier host registration process, R2 knows that at least one host off both its LAN interfaces are listening for packets sent to 226.1.1.1. R2 therefore forwards a copy of the packet out each of its LAN interfaces.
5. R3 receives the multicast packet from R1, and uses the same kind of logic as R2. However, R3 knows from the earlier host registration process that only one of its LAN interfaces connects to a subnet with hosts listening for packets sent to 226.1.1.1, so R3 forwards a copy of the packet out that one interface only.

As you can see from this example, the server sent one packet and the routers replicated the packet so it reached all the correct locations in the network.

As another comparison between unicast and multicast addresses, note that multicast addresses may be used as destination IP addresses only, whereas unicast addresses may be used as both the destination and source address. For instance, consider the packets in the example shown in Figure J-2. All those packets flow from one host, so the packet uses a unicast IP address of that host's unicast IP address.

Finally, to complete one more comparison between unicast IP addressing and multicast IP addressing, think about that last hop router in the example shown in Figure J-1. If a router such as R2 or R3 had forwarded a unicast IP packet, the router would look in its ARP cache to find the unicast IP address for the destination in that connected subnets, and the associated unicast MAC address. That will not work when forwarding a multicast packet with a multicast (Class D) destination IP address.

To encapsulate a multicast IP packet over an Ethernet LAN, IP multicast calculates the destination MAC address with a simple process. The process copies the last 23 bits of the IP address behind a reserved 25-bit prefix to form the 48-bit destination MAC address. The resulting MAC address, called a multicast MAC address, begins with hex 01005E. So, the multicast IP packet, encapsulated in the multicast Ethernet frame, is forwarded out the router interface onto the LAN. At that point, the switches take one of the following approaches to forwarding the frame so that all hosts who want a copy of the frame get a copy:

- Flood the multicast frame as if it were a broadcast
- Use other Ethernet multicast features that flood the frame only to those same devices that registered to receive a copy

If you feel like these few pages probably left out some detail; indeed, several books have been written about IP multicast all to itself. The topic is indeed large. For this book's purposes, know the main comparison points with unicast addressing. Multicast addressing gives applications that need to communicate the same data at the same time to multiple hosts a much more efficient way to do that. If the application is written to make use of IP multicast, the application can consume much less traffic in the network, as compared to using unicast IP addresses and sending every host a copy of the packet.

Comparing and Contrasting IP Address Types

The last few pages reviewed unicast and broadcast addresses, and explained the core concepts behind IP multicast addresses. Table J-1 summarizes the key comparison points mentioned throughout this section for convenient study.

Table J-1 Comparisons of Unicast, Broadcast, and Multicast IP Addresses

	Unicast	Broadcast	Multicast
Primarily used for data sent by the most common user apps (web, email, chat, and so on)	Yes	No	No
Assigned to hosts with DHCP	Yes	No	No
Uses Class A, B, and C addresses	Yes	No	No
Primarily used by overhead protocols (DHCP, ARP) to send one message to more than one device	No	Yes	No
Used as destination IP address only	No	Yes	Yes
Primarily used by applications that send the same data at the same time to multiple clients	No	No	Yes
Uses Class D addresses	No	No	Yes

NOTE The content under the heading "Bandwidth and Clock Rate on Serial Interfaces" was most recently published for the 100-105 Exam in 2016, in Chapter 17 of the *CCENT/CCNA ICND1 100-105 Official Cert Guide*.

Bandwidth and Clock Rate on Serial Interfaces

WAN serial links can run at a wide variety of speeds. To deal with the wide range of speeds, routers physically slave themselves to the speed as dictated by the CSU/DSU through a process called *clocking*. As a result, routers can use serial links without the need for additional configuration or autonegotiation to sense the serial link's speed. The CSU/DSU knows the speed, the CSU/DSU sends clock pulses over the cable to the router, and the router reacts to the clocking signal.

To build a serial link in a home lab, the routers can use serial interface cards that normally use an external CSU/DSU, and make a serial link, without requiring the expense of two CSU/DSUs. Figure J-3 shows the concept. To make it work, the link uses two serial cables—one a DTE cable and the other a DCE cable—which swap the transmit and receive pair on the cables.

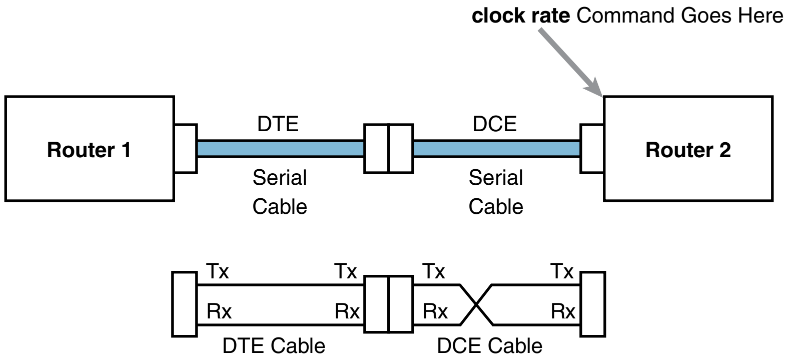

Figure J-3 *Serial Link in Lab*

Using the correct cabling works, as long as you add one command: the **clock rate** interface subcommand. This command tells that router the speed at which to transmit bits on a serial link like the one shown in Figure J-3. The **clock rate** command is not needed on real serial links, because the CSU/DSU provides the clocking. When you create a serial link in the lab using cables, without any real CSU/DSUs on the link, the router with the DCE cable must supply that clocking function, and the **clock rate** command tells the router to provide it.

NOTE Newer router IOS versions automatically add a default **clock rate 2000000** command on serial interfaces that have a DCE cable connected to them. While helpful, this speed might be too high for some types of back-to-back serial cables, so consider using a lower speed in lab.

Example J-1 shows the configuration of the **clock rate** command. The end of the example verifies that this router can use the **clock rate** command with the **show controllers** command. This command confirms that R1 has a V.35 DCE cable connected.

Example J-1 *Router R1 Configuration with the* **clock rate** *Command*

```
R1# show running-config
! lines omitted for brevity
interface Serial0/0/0
   ip address 172.16.4.1 255.255.255.0
   clock rate 2000000
!
interface Serial0/0/1
   ip address 172.16.5.1 255.255.255.0
   clock rate 128000

! lines omitted for brevity

R1# show controllers serial 0/0/1
Interface Serial0
Hardware is PowerQUICC MPC860
DCE V.35, clock rate 128000
idb at 0x8169BB20, driver data structure at 0x816A35E4
! Lines omitted for brevity
```

NOTE The **clock rate** command does not allow just any speed to be configured. However, the list of speeds does vary from router to router.

Some people confuse the router **bandwidth** command with the **clock rate** command. The **clock rate** command sets the actual Layer 1 speed used on the link, if no CSU/DSU is used, as just described. Conversely, every router interface has a bandwidth setting, either by default or configured. The bandwidth of the interface is the documented speed of the interface, which does not have to match the actual Layer 1 speed used on the interface.

That bandwidth setting does not impact how fast the interface transmits data. Instead, routers use the interface bandwidth setting as both documentation and as input to some other processes. For instance, the Open Shortest Path First (OSPF) and Enhanced Interior Gateway Routing Protocol (EIGRP) routing protocols base their routing protocol metrics on the bandwidth by default.

Example J-2 highlights the bandwidth setting on Router R1's S0/0/1 interface, as configured in the previous example. In that previous example, the **clock rate 128000** command sets the clock rate to 128 kbps, but it leaves the **bandwidth** command unset. As a result, IOS uses the default serial bandwidth setting of 1544, which means 1544 kbps—which is the speed of a T1 serial link.

Example J-2 *Router Bandwidth Settings*

```
R1# show interfaces s0/0/1
Serial0/0/1 is up, line protocol is up
   Hardware is WIC MBRD Serial
   Description: link to R3
   Internet address is 10.1.13.1/24
   MTU 1500 bytes, BW 1544 Kbit/sec, DLY 20000 usec,
      reliability 255/255, txload 1/255, rxload 1/255
   Encapsulation HDLC, loopback not set
```

The common mistake people make is to know about clock rate, but mistakenly think that the bandwidth setting is just another term for "clock rate." It is not. Follow these rules to find these two interface settings:

To see the clock rate, look for the **clock rate** interface subcommand in the configuration, or use the **show controllers serial** *number* command (as shown in Example J-1.)

To see the bandwidth setting on an interface, look for the **bandwidth** interface subcommand in the configuration, or use the **show interfaces** [*type number*] command (as shown in Example J-2).

Note that using default bandwidth settings on most router interfaces makes sense, with the exception of serial interfaces. IOS defaults to a bandwidth of 1544 (meaning 1544 kbps, or 1.544 Mbps) for serial interfaces, regardless of the speed dictated by the provider or by a **clock rate** command in the lab. Most engineers set the bandwidth to match the actual speed, for example, using the **bandwidth 128** interface subcommand on a link running at 128 kbps. On Ethernet 10/100 or 10/100/1000 interfaces, the router knows the speed used, and dynamically sets the Ethernet interface's bandwidth to match.

NOTE The content under the heading "Using traceroute to Isolate the Problem to Two Routers" was most recently published for the 100-105 Exam in 2016, in Chapter 23 of the *Cisco CCNA ICND1 100-105 Official Cert Guide*.

Using traceroute to Isolate the Problem to Two Routers

One of the best features of the **traceroute** command, as compared to ping, is that when it does not complete it gives an immediate clue as to where to look next. With ping, when the ping fails, the next step is usually to use more **ping** commands. With traceroute, it tells you what router to try to connect and look at the routes and in which direction.

NOTE As a reminder, this book uses the term *forward route* for routes that send the packets sent by the **ping** or **traceroute** command, and *reverse route* for the packets sent back.

When a problem exists, a **traceroute** command results in a partial list of routers. Then the command either finishes with an incomplete list or it runs until the user must stop the command. In either case, the output does not list all routers in the end-to-end route, because of the underlying problem.

NOTE In addition, the **traceroute** command may not finish even though the network has no problems. Routers and firewalls may filter the messages sent by the **traceroute** command, or the TTL Exceeded messages, which would prevent the display of portions or all or part of the path.

The last router listed in the output of a **traceroute** command's output tells us where to look next to isolate the problem, as follows:

- Connect to the CLI of the last router listed, to look at forward route issues.
- Connect to the CLI of the next router that should have been listed, to look for reverse route issues.

To see why, consider an example based on the internetwork in Figure J-4. In this case, R1 uses an extended traceroute to host 5.5.5.5, with source IP address 1.1.1.1. This command's output lists router 2.2.2.2, then 3.3.3.3, and then the command cannot complete.

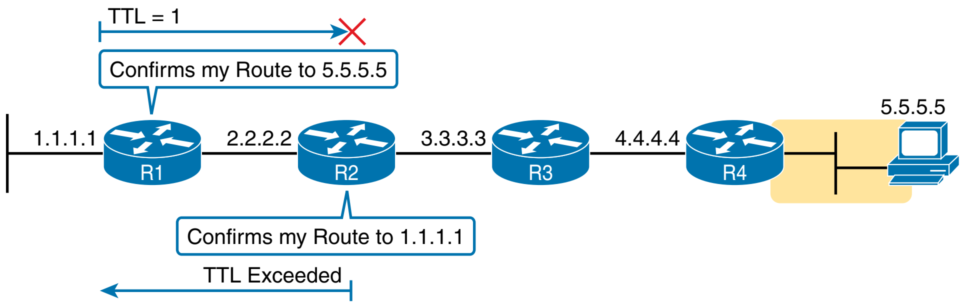

Figure J-4 *Messages That Cause the* **traceroute** *Command to List 2.2.2.2*

First, Figure J-4 focuses on the first line of output: the line that lists first-hop router 2.2.2.2.

The figure shows the TTL=1 message at the top and the TTL Exceeded message back on the bottom. This first pair of messages in the figure must have worked, because without them, the **traceroute** command on R1 cannot have learned about a router with address 2.2.2.2. The first (top) message required R1 to have a route for 5.5.5.5, which sent the packets to R2 next. The TTL Exceeded message required that R2 have a route that matched address 1.1.1.1, to send the packets back to R1's LAN IP address.

Next, Figure J-5 focuses on the messages that allow the second line of output on R1's sample **traceroute** command: the line that correctly lists 3.3.3.3 as the next router in the route.

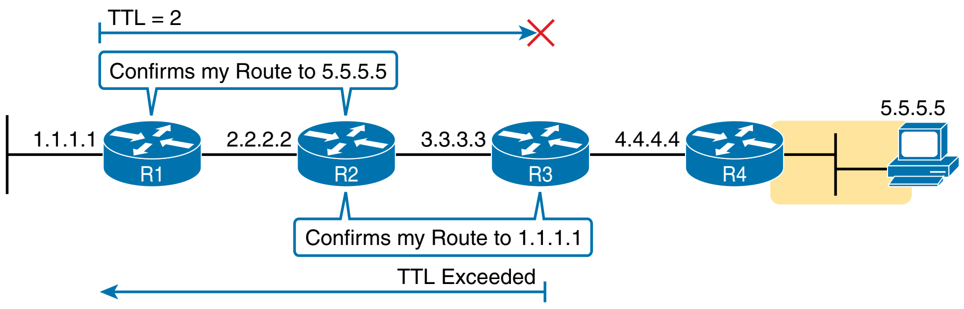

Figure J-5 *Messages That Cause the* **traceroute** *Command to List 3.3.3.3*

Following the same logic, the traceroute output lists 3.3.3.3 because the messages in Figure J-5 must have worked. For these messages to flow, the routes listed in Figure J-4 must exist, plus new routes listed in 18-15. Specifically, the TTL=2 packet at the top requires R2 to have a route for 5.5.5.5, which sends the packets to R3 next. The TTL Exceeded message requires that R3 have a route that matches address 1.1.1.1, to send the packets back toward R1's LAN IP address.

In this example, the **traceroute 5.5.5.5** command does not list any routers beyond 2.2.2.2 and 3.3.3.3 However, based on the figures, it is clear that 4.4.4.4 should be the next IP address listed. To help isolate the problem further, why might the next messages—the message with TTL=3 and the response—fail?

Figure J-6 points out the routing issues that can cause this command to not be able to list 4.4.4.4 as the next router. First, R3 must have a forward route matching destination 5.5.5.5 and forwarding the packet to Router R4. The return message requires a reverse route matching destination 1.1.1.1 and forwarding the packet back to Router R3.

In conclusion, for this example, if a routing problem prevents the **traceroute** command from working, the problem exists in one of two places: the forward route to 5.5.5.5 on Router R3, or the reverse route to 1.1.1.1 on R4.

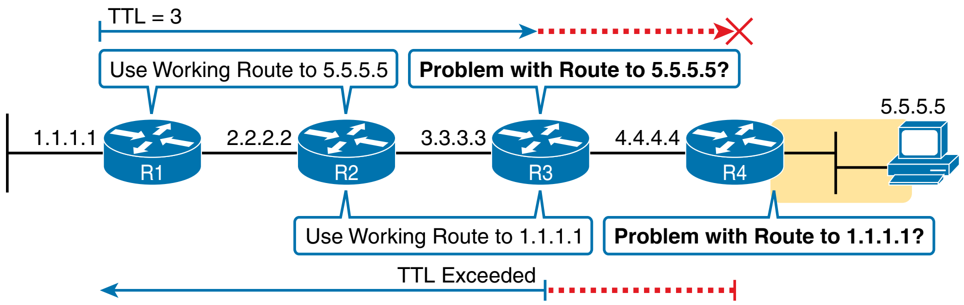
TTL = 3
Use Working Route to 5.5.5.5
Problem with Route to 5.5.5.5?
5.5.5.5
1.1.1.1
2.2.2.2
3.3.3.3
4.4.4.4
R1
R2
R3
R4
Use Working Route to 1.1.1.1
Problem with Route to 1.1.1.1?
TTL Exceeded

Figure J-6 *Issues That Could Prevent* traceroute *from Listing 4.4.4.4*

NOTE The content under the heading “Troubleshooting Static IPv6 Routes” was most recently published in Chapter 32 of the *Cisco CCNA ICND1 100-105 Official Cert Guide*.

Troubleshooting Static IPv6 Routes

This last part of the chapter looks at troubleshooting IPv6 static routes, reviewing many of the same troubleshooting rules applied to IPv4 static routes, while focusing on the details specific to IPv6.

This topic breaks static route troubleshooting into two perspectives: the route is in the routing table but is incorrect and cases in which the route is not in the routing table.

Troubleshooting Incorrect Static Routes That Appear in the IPv6 Routing Table

A static route is only as good as the input typed into the **ipv6 route** command. IOS checks the syntax of the command, of course. However, IOS cannot tell if you choose the incorrect outgoing interface, incorrect next-hop address, or incorrect prefix/prefix-length in a static route. If the parameters pass the syntax checks, IOS places the **ipv6 route** command into the running-config file. Then, if no other problem exists (as discussed at the next heading), IOS puts the route into the IP routing table—even though the route may not work because of the poorly chosen parameters.

For instance, an exam question might show a figure with Router R1 having an address of 2001:1:1:1::1 and neighboring Router R2 with an address of 2001:1:1:1::2. If R1 lists a static route with the command **ipv6 route 3333::/64 2001:1:1:1::1**, the command would be accepted by IOS with correct syntax, but it would not be effective as a route. R1 cannot use its own IPv6 address as a next-hop address. IOS does not prevent the configuration of the command, however; it allows the command and adds the route to the IPv6 routing table, but the route cannot possibly forward packets correctly.

When you see an exam question that has static routes, and you see them in the output of **show ipv6 route**, remember that the routes may have incorrect parameters. Check for these types of mistakes:

Step 1. Prefix/Length: Does the **ipv6 route** command reference the correct subnet ID (prefix) and mask (prefix length)?

Step 2. If using a next-hop IPv6 address that is a link-local address:

  A. Is the link-local address an address on the correct neighboring router? (It should be an address on another router on a shared link.)

  B. Does the **ipv6 route** command also refer to the correct outgoing interface on the local router?

Step 3. If using a next-hop IPv6 address that is a global unicast or unique local address, is the address the correct unicast address of the neighboring router?

Step 4. If referencing an outgoing interface, does the **ipv6 route** command reference the interface on the local router (that is, the same router where the static route is configured)?

This troubleshooting checklist works through the various cases in which IOS would accept the configuration of the static IPv6 route, but the route would not work because of the incorrect parameters in context. It helps to see a few examples. Figure J-7 shows a sample network to use for the examples; all the examples focus on routes added to Router R1, for the subnet on the far right.

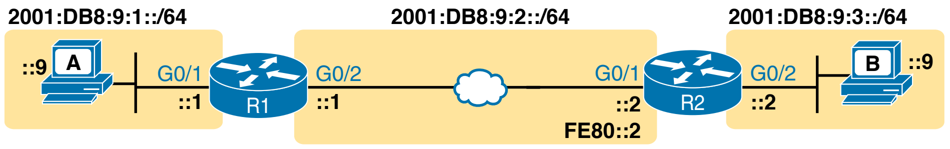

Figure J-7 *Sample Topology for Incorrect IPv6 Route Examples*

Example J-3 shows five **ipv6 route** commands. All have correct syntax, but all have one incorrect value; that is, the route will not work because of the types of problems in the troubleshooting checklist. Look for the short comment at the end of each configuration command to see why each is incorrect.

Example J-3 **ipv6 route** *Commands with Correct Syntax but Incorrect Ideas*

```
ipv6 route 2001:DB8:9:33::/64 2001:DB8:9:2::2 ! Step 1: Wrong prefix
ipv6 route 2001:DB8:9:3::/64 G0/2 FE80::AAA9 ! Step 2A: Wrong neighbor link local
ipv6 route 2001:DB8:9:3::/64 FE80::2 ! Step 2B: Missing outgoing interface
ipv6 route 2001:DB8:9:3::/64 2001:DB8:9:2::1 ! Step 3: Wrong neighbor address
ipv6 route 2001:DB8:9:3::/64 G0/1 FE80::2 ! Step 4: Wrong interface on R1
```

All these incorrect examples have correct syntax and would be added to R1's IPv6 routing table if configured on R1. However, all have flaws. Working through the examples in order:

Step 1. The prefix (2001:DB8:9:33::) has a typo in the fourth quartet (33 instead of 3).

Step 2A. The figure shows R2's G0/1 with link-local address FE80::2, but the command uses FE80::AAA9.

Step 2B. The command uses the correct link-local address on R2's address on the common link (FE80::2 per the figure), but it omits the outgoing interface of R1's G0/2 interface. (See the next example for more detail.)

Step 3. The figure shows the subnet in the center as 2001:DB8:9:2::/64, with R1 using the ::1 address and R2 using ::2. For the fourth command, R1's command should use R2's address 2001:DB8:9:2::2, but it uses R1's own 2001:DB8:9:2::1 address instead.

Step 4. As a command on R1, the outgoing interface references R1's own interfaces. R1's G0/1 is the interface on the left, whereas R1 should use its G0/2 interface on the right when forwarding packets to subnet 2001:DB8:9:3::/64.

The key takeaway for this section is to know that a route in the IPv6 routing table may be incorrect due to poor choices for the parameters. The parameters should always include the neighboring router's IPv6 addresses, but the local router's interface type/number, and in all cases, the correct prefix/length. The fact that a route is in the IPv6 routing table, particularly a static route, does not mean it is a correct route.

Note that of the five example commands in Example J-3, IOS would accept all of them except the third one. IOS can notice the case of omitting the outgoing interface if the next-hop address is a link-local address. Example J-4 shows a sample of the error message from IOS.

Example J-4 *IOS Rejects the* **ipv6 route** *Command with Link-Local and No Outgoing Interface*

```
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# ipv6 route 2001:DB8:9:3::/64 FE80::2
% Interface has to be specified for a link-local nexthop
R1(config)# ^Z
R1#
R1# show running-config | include ipv6 route
R1#
```

The Static Route Does Not Appear in the IPv6 Routing Table

The preceding few pages focused on IPv6 static routes that show up in the IPv6 routing table but unfortunately have incorrect parameters. The next page looks at IPv6 routes that have correct parameters, but IOS does not place them into the IPv6 routing table.

When you add an **ipv6 route** command to the configuration, and the syntax is correct, IOS considers that route to be added to the IPv6 routing table. IOS makes the following checks before adding the route; note that IOS uses this same kind of logic for IPv4 static routes:

Key Topic

- For **ipv6 route** commands that list an outgoing interface, that interface must be in an up/up state.
- For **ipv6 route** commands that list a global unicast or unique local next-hop IP address (that is, not a link-local address), the local router must have a route to reach that next-hop address.
- If another IPv6 route exists for that exact same prefix/prefix-length, the static route must have a better (lower) administrative distance.

For example, Router R1, again from Figure J-7, has been configured with IPv6 addresses. Example J-5 shows the addition of an **ipv6 route** command for remote subnet 2001:DB8:9:3::/64, but with incorrect next-hop address 2001:DB8:9:3::2. That address is on R2, but it is the address on the far side of R2, on R2's G0/2 interface.

Example J-5 *No Route for Next-Hop IPv6 Address in Static Route*

```
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# ipv6 route 2001:DB8:9:3::/64 2001:DB8:9:3::2
R1(config)# ^Z
R1# show ipv6 route
IPv6 Routing Table - default - 5 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, HA - Home Agent, MR - Mobile Router, R - RIP
       H - NHRP, I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
      IS - ISIS summary, D - EIGRP, EX - EIGRP external, NM - NEMO
      ND - ND Default, NDp - ND Prefix, DCE - Destination, NDr - Redirect
      RL - RPL, O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1
      OE2 - OSPF ext 2, ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
      la - LISP alt, lr - LISP site-registrations, ld - LISP dyn-eid
       a - Application
C 2001:DB8:9:1::/64 [0/0]
     via GigabitEthernet0/1, directly connected
L 2001:DB8:9:1::1/128 [0/0]
     via GigabitEthernet0/1, receive
C 2001:DB8:9:2::/64 [0/0]
     via GigabitEthernet0/2, directly connected
L 2001:DB8:9:2::1/128 [0/0]
     via GigabitEthernet0/2, receive
L FF00::/8 [0/0]
     via Null0, receive
```

NOTE The content under the heading "Default Routes with SLAAC on Router Interfaces" was most recently published in Chapter 32 of the *Cisco CCNA ICND1 100-105 Official Cert Guide*.

Default Routes with SLAAC on Router Interfaces

Routers can use DHCP on their own interface and learn their IP address, mask, and even a default IPv4 route. In particular, that process can be useful on a router that connects to the Internet. The enterprise router uses DHCP as a client, learning its own IPv4 address with DHCP and adding a default route pointing to the ISP's router as the next-hop IPv4 address.

Routers can accomplish the same goals with IPv6, just with a few different protocols and methods. As with IPv4, the IPv6 enterprise router can dynamically learn its IPv6 address and dynamically create a default IPv6 route to the ISP's router. This section shows the details, with the enterprise router using SLAAC to learn its address and the information needed to create a default route.

First, the enterprise router that connects to the ISP, like Router R1 in Figure J-8, requires the configuration of the interface subcommand **ipv6 address autoconfig default**. This command tells the router that, on that interface, use SLAAC to build its own IPv6 address. R1 would act like any host that uses SLAAC, as shown in Step 2 of the figure, and send an NDP RS message over the link. As noted at Step 3, the ISP router would send back an RA message, announcing router ISP1's IPv6 address and the IPv6 prefix used on the link.

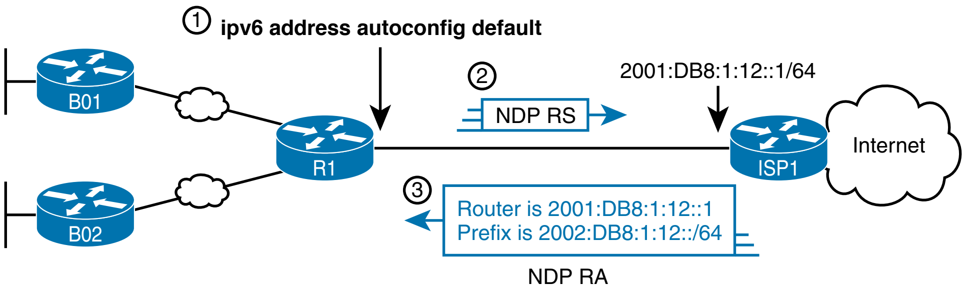
1
ipv6 address autoconfig default
B01
B02
R1
2
NDP RS
2001:DB8:1:12::1/64
ISP1
Internet
3
Router is 2001:DB8:1:12::1
Prefix is 2002:DB8:1:12::/64
NDP RA

Figure J-8 *Enterprise Router Using SLAAC to Build IPv6 Address and Default IPv6 Route*

When R1 receives the NDP RA message, it does the following:

Interface address: Builds its own interface IPv6 address using the SLAAC process, based on the prefix in the RA.

Local /128 Route: Adds a local (/128) IPv6 route for the address, as it would for any interface IPv6 address.

Connected Route for Prefix: Adds a connected (/64) route for the prefix learned in the NDP RA message.

Default route: R1 adds a default route, to destination ::/0, with the next-hop address of ISP's link-local address, as learned in the RA sent by router ISP1.

Note that the router can be configured to add this default route or not. As shown in the figure, the router builds a default route. Using the **ipv6 address autoconfig** subcommand without the **default** keyword causes the router to build its address with SLAAC but not add a default route.

Example J-6 shows the three IPv6 routes on Router R1 just mentioned in the list. In particular, note the codes for the connected route and the default route; both codes begin with ND, meaning the route was learned with NDP. In particular, as highlighted in the legend part of the output, *ND* refers to an NDP-learned default route, and *NDp* refers to an NDP-learned prefix (as listed in the NDP RA message in Figure J-9 in this case). Note also that these same two routes have an administrative distance of 2, which is the default administrative distance of IPv6 routes learned with NDP.

Example J-6 *Learning an Address and Default Static Route with DHCP*

```
R1# show ipv6 route
IPv6 Routing Table - default - 4 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, HA - Home Agent, MR - Mobile Router, R - RIP
       H - NHRP, I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       IS - ISIS summary, D - EIGRP, EX - EIGRP external, NM - NEMO
       ND - ND Default, NDp - ND Prefix, DCE - Destination, NDr - Redirect
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2, la - LISP alt
       lr - LISP site-registrations, ld - LISP dyn-eid, a - Application
ND ::/0 [2/0]
     via FE80::22FF:FE22:2222, Serial0/0/0
NDp 2001:DB8:1:12::/64 [2/0]
     via Serial0/0/0, directly connected
L 2001:DB8:1:12:32F7:DFF:FE29:8560/128 [0/0]
    via Serial0/0/0, receive
! lines omitted for brevity
```

Analyzing Ethernet LAN Designs

NOTE This appendix contains an entire chapter that was published as a chapter in one of the past editions of this book or a related book. The author includes this appendix with the current edition as extra reading for anyone interested in learning more. However, note that the content in this appendix has not been edited since it was published in the earlier edition, so references to exams and exam topics, and to other chapters, will be outdated. This appendix was previously published as Chapter 10 of the book *CCENT/CCNA ICND1 100-105 Official Cert Guide*, published in 2016.

Ethernet defines what happens on each Ethernet link, but the more interesting and more detailed work happens on the devices connected to those links: the network interface cards (NIC) inside devices and the LAN switches. This chapter takes the Ethernet LAN basics introduced in Chapter 2, "Fundamentals of Ethernet LANs," and dives deeply into many aspects of a modern Ethernet LAN, while focusing on the primary device used to create these LANs: LAN switches.

This chapter breaks down the discussion of Ethernet and LAN switching into two sections. The first major section looks at the logic used by LAN switches when forwarding Ethernet frames, along with the related terminology. The second section considers design and implementation issues, as if you were building a new Ethernet LAN in a building or campus. This second section considers design issues, including using switches for different purposes, when to choose different types of Ethernet links, and how to take advantage of Ethernet autonegotiation.

Foundation Topics

Analyzing Collision Domains and Broadcast Domains

Ethernet devices, and the logic they use, have a big impact on why engineers design modern LANs in a certain way. Some of the terms used to describe key design features come from far back in the history of Ethernet, and because of their age, the meaning of each term may or may not be so obvious to someone learning Ethernet today. This first section of the chapter looks at two of these older terms in particular: collision domain and broadcast domain. And to understand these terms and apply them to modern Ethernet LANs, this section needs to work back through the history of Ethernet a bit, to put some perspective on the meaning behind these terms.

Ethernet Collision Domains

The term *collision domain* comes from the far back history of Ethernet LANs. To be honest, sometimes people new to Ethernet can get a little confused about what this term really means in the context of a modern Ethernet LAN, in part because modern Ethernet LANs, done properly, can completely prevent collisions. So to fully understand collision domains, we must first start with a bit of Ethernet history. This next section of the chapter looks at a few of the historical Ethernet devices, for the purpose of defining a collision domain, and then closing with some comments about how the term applies in a modern Ethernet LAN that uses switches.

10BASE-T with Hub

10BASE-T, introduced in 1990, significantly changed the design of Ethernet LANs, more like the designs seen today. 10BASE-T introduced the cabling model similar to today's Ethernet LANs, with each device connecting to a centralized device using an unshielded twisted-pair (UTP) cable. However, 10BASE-T did not originally use LAN switches; instead, the early 10BASE-T networks used a device called an *Ethernet hub*. (The technology required to build even a basic LAN switch was not yet available at that time.)

Although both a hub and a switch use the same cabling star topology, an Ethernet hub does not forward traffic like a switch. Ethernet hubs use physical layer processing to forward data. A hub does not interpret the incoming electrical signal as an Ethernet frame, look at the source and destination MAC address, and so on. Basically, a hub acts like a repeater, just with lots of ports. When a repeater receives an incoming electrical signal, it immediately *forwards a regenerated signal out all the other ports except the incoming port*. Physically, the hub just sends out a cleaner version of the same incoming electrical signal, as shown in Figure K-1, with Larry's signal being repeated out the two ports on the right.

Key Topic

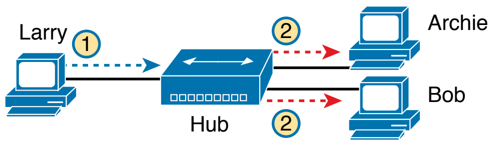

Figure K-1 *10BASE-T (with a Hub): The Hub Repeats Out All Other Ports*

Because of the physical layer operation used by the hub, the devices attached to the network must use carrier sense multiple access with collision detection (CSMA/CD) to take turns (as introduced at the end of Chapter 2). Note that the hub itself does not use CSMA/CD logic; the hub always receives an electrical signal and starts repeating a (regenerated) signal out all other ports, with no thought of CSMA/CD. So, although a hub's logic works well to make sure all devices get a copy of the original frame, that same logic causes frames to collide. Figure K-2 demonstrates that effect, when the two devices on the right side of the figure send a frame at the same time, and the hub physically transmits both electrical signals out the port to the left (toward Larry).

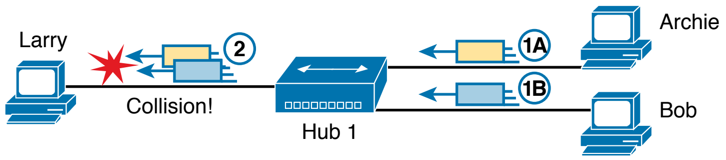

Figure K-2 *Hub Operation Causing a Collision*

Because a hub makes no attempt to prevent collisions, the devices connected to it all sit within the same collision domain. A *collision domain* is the set of NICs and device ports for which if they sent a frame at the same time, the frames would collide. In Figures K-1 and K-2, all three PCs are in the same collision domain, as well as the hub. Summarizing the key points about hubs:

Key Topic

- The hub acts a multiport repeater, blindly regenerating and repeating any incoming electrical signal out all other ports, even ignoring CSMA/CD rules.
- When two or more devices send at the same time, the hub's actions cause an electrical collision, making both signals corrupt.
- The connected devices must take turns by using carrier sense multiple access with collision detection (CSMA/CD) logic, so the devices share the bandwidth.
- Hubs create a physical star topology.

Ethernet Transparent Bridges

From a design perspective, the introduction of 10BASE-T was a great improvement over the earlier types of Ethernet. It reduced cabling costs and cable installation costs, and improved the availability percentages of the network. But sitting here today, thinking of a LAN in which all devices basically have to wait their turn may seem like a performance issue, and it was. If Ethernet could be improved to allow multiple devices to send at the same time without causing a collision, Ethernet performance could be improved.

The first method to allow multiple devices to send at the same time was Ethernet transparent bridges. Ethernet *transparent bridges*, or simply *bridges*, made these improvements:

- Bridges sat between hubs and divided the network into multiple *collision domains*.
- Bridges increase the capacity of the entire Ethernet, because each collision domain is basically a separate instance of CSMA/CD, so each collision domain can have one sender at a time.

Figure K-3 shows the effect of building a LAN with two hubs, each separated by a bridge. The resulting two collision domains each support at most 10 Mbps of traffic each, compared to at most 10 Mbps if a single hub were used.

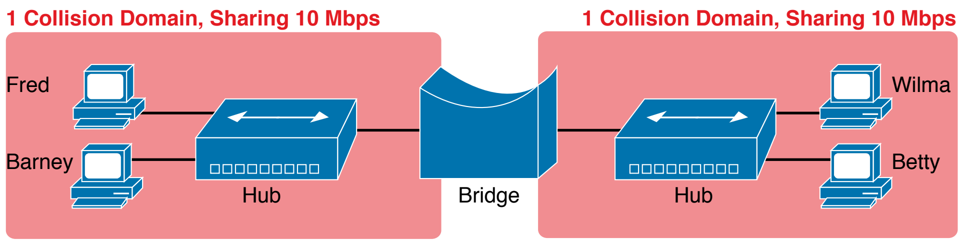
1 Collision Domain, Sharing 10 Mbps
1 Collision Domain, Sharing 10 Mbps
Fred
Barney
Hub
Bridge
Hub
Wilma
Betty

Figure K-3 *Bridge Creates Two Collision Domains and Two Shared Ethernets*

Bridges create multiple collision domains as a side effect of their forwarding logic. A bridge makes forwarding decisions just like a modern LAN switch; in fact, bridges were the predecessors of the modern LAN switch. Like switches, bridges hold Ethernet frames in memory, waiting to send out the outgoing interface based on CSMA/CD rules. In other cases, the bridge does not even need to forward the frame. For instance, if Fred sends a frame destined to Barney's MAC address, then the bridge would never forward frames from the left to the right.

Ethernet Switches and Collision Domains

LAN switches perform the same basic core functions as bridges but at much faster speeds and with many enhanced features. Like bridges, switches segment a LAN into separate collision domains, each with its own capacity. And if the network does not have a hub, each single link in a modern LAN is considered its own collision domain, even if no collisions can actually occur in that case.

For example, Figure K-4 shows a simple LAN with a switch and four PCs. The switch creates four collision domains, with the ability to send at 100 Mbps in this case on each of the four links. And with no hubs, each link can run at full duplex, doubling the capacity of each link.

Key Topic

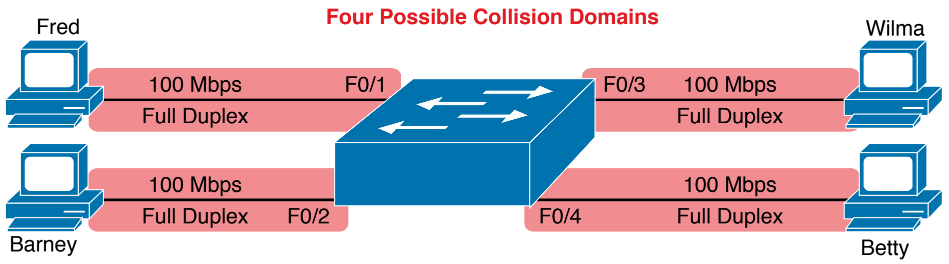

Figure K-4 *Switch Creates Four Collision Domains and Four Ethernet Segments*

Now take a step back for a moment and think about some facts about modern Ethernet LANs. Today, you build Ethernet LANs with Ethernet switches, not with Ethernet hubs or bridges. The switches connect to each other. And every single link is a separate collision domain.

As strange as it sounds, each of those collision domains in a modern LAN may also never have a collision. Any link that uses full duplex—that is, both devices on the link use full duplex—does not have collisions. In fact, running with full duplex is basically this idea: No collisions can occur between a switch and a single device, so we can turn off CSMA/CD by running full duplex.

NOTE The routers in a network design also create separate collision domains, because frames entering or exiting one router LAN interface do not collide with frames on another of the router's LAN interfaces.

The Impact of Collisions on LAN Design

So, what is the useful takeaway from this discussion about collision domains? A long time ago, collisions were normal in Ethernet, so analyzing an Ethernet design to determine where the collision domains were was useful. On the other end of the spectrum, a modern campus LAN that uses only switches (and no hubs or transparent bridges), and full duplex on all links, has no collisions at all. So does the collision domain term still matter today? And do we need to think about collisions even still today?

In a word, the term collision domain still matters, and collisions still matter, in that network engineers need to be ready to understand and troubleshoot exceptions. Whenever a port that could use full duplex (therefore avoiding collisions) happens to use half duplex—by incorrect configuration, by the result of autonegotiation, or any other reason—collisions can now occur. In those cases, engineers need to be able identify the collision domain.

Summarizing the key points about collision domains:

Key Topic

- LAN switches place each separate interface into a separate collision domain.
- LAN bridges, which use the same logic as switches, placed each interface into a separate collision domain.
- Routers place each LAN interface into a separate collision domain. (The term collision domain does not apply to WAN interfaces.)
- LAN hubs do not place each interface into a separate collision domain.

- A modern LAN, with all LAN switches and routers, with full duplex on each link, would not have collisions at all.
- In a modern LAN with all switches and routers, even though full duplex removes collisions, think of each Ethernet link as a separate collision domain when the need to troubleshoot arises.

Figure K-5 shows an example with a design that includes hubs, bridges, switches, and routers—a design that you would not use today, but it makes a good backdrop to remind us about which devices create separate collision domains.

Key Topic

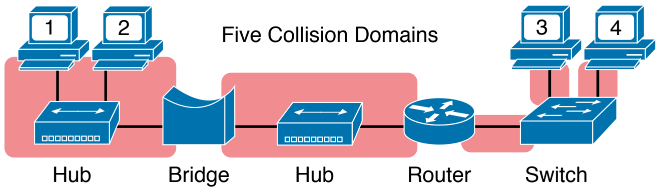

Figure K-5 *Example of a Hub Not Creating Multiple Collision Domains, While Others Do*

Ethernet Broadcast Domains

Take any Ethernet LAN, and pick any device. Then think of that device sending an Ethernet broadcast. An Ethernet *broadcast domain* is the set of devices to which that broadcast is delivered.

To begin, think about a modern LAN for a moment, and where a broadcast frame flows. Imagine that all the switches still used the switch default to put each interface into VLAN 1. As a result, a broadcast sent by any one device would be flooded to all devices connected to all switches (except for the device that sent the original frame). For instance, in Figure K-6, under the assumption that all ports are still assigned to VLAN 1, a broadcast would flow to all the devices shown in the figure.

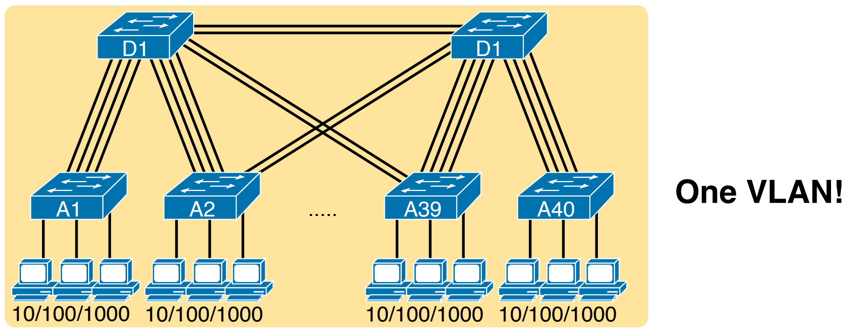

Figure K-6 *A Single Large Broadcast Domain*

Of all the common networking devices discussed in this book, only a router does not forward a LAN broadcast. Hubs of course forward broadcasts, because hubs do not even think about the electrical signal as an Ethernet frame. Bridges and switches use the same forwarding logic, flooding LAN broadcasts. Routers, as a side effect of their routing logic, do not forward Ethernet broadcast frames, so they separate a network into separate broadcast domains. Figure K-7 collects those thoughts into a single example.

Key Topic

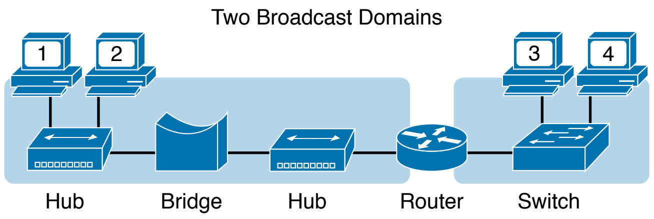

Figure K-7 *Broadcast Domains Separated by a Router*

By definition, broadcasts sent by a device in one broadcast domain are not forwarded to devices in another broadcast domain. In this example, there are two broadcast domains. The router does not forward a LAN broadcast sent by a PC on the left to the network segment on the right.

Virtual LANs

Routers create multiple broadcast domains mostly as a side effect of how IP routing works. While a network designer might set about to use more router interfaces for the purpose of making a larger number of smaller broadcast domains, that plan quickly consumes router interfaces. But a better tool exists, one that is integrated into LAN switches and consumes no additional ports: virtual LANs (VLAN).

By far, VLANs give the network designer the best tool for designing the right number of broadcast domains, of the right size, with the right devices in each. To appreciate how VLANs do that, you must first think about one specific definition of what a LAN is:

A LAN consists of all devices in the same broadcast domain.

With VLANs, a switch configuration places each port into a specific VLAN. The switches create multiple broadcast domains by putting some interfaces into one VLAN and other interfaces into other VLANs. The switch forwarding logic does not forward frames from a port in one VLAN out a port into another VLAN—so the switch separates the LAN into separate broadcast domains. Instead, routers must forward packets between the VLANs by using routing logic. So, instead of all ports on a switch forming a single broadcast domain, the switch separates them into many, based on configuration.

For perspective, think about how you would create two different broadcast domains with switches if the switches had no concept of VLANs. Without any knowledge of VLANs, a switch would receive a frame on one port and flood it out all the rest of its ports. Therefore, to make two broadcast domains, two switches would be used—one for each broadcast domain, as shown in Figure K-8.

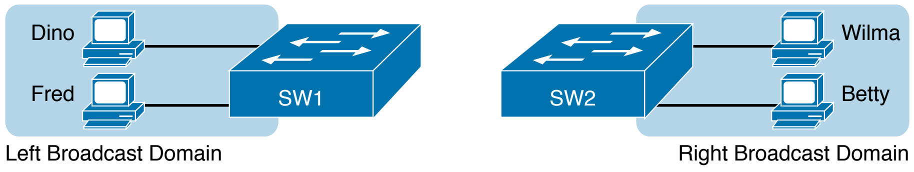

Figure K-8 *Sample Network with Two Broadcast Domains and No VLANs*

Alternatively, with a switch that understands VLANs, you can create multiple broadcast domains using a single switch. All you do is put some ports in one VLAN and some in the other. (The Cisco Catalyst switch interface subcommand to do so is **switchport access vlan 2**, for instance, to place a port into VLAN 2.) Figure K-9 shows the same two broadcast domains as in Figure K-8, now implemented as two different VLANs on a single switch.

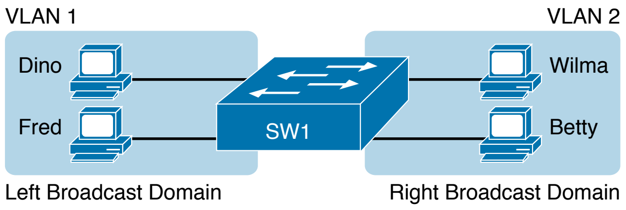

Figure K-9 *Sample Network with Two VLANs Using One Switch*

This section briefly introduces the concept of VLANs, but Chapter 11, "Implementing Ethernet Virtual LANs," discusses VLANs in more depth, including the details of how to configure VLANs in campus LANs.

The Impact of Broadcast Domains on LAN Design

Modern LAN designs try to avoid collisions, because collisions make performance worse. There is no benefit to keeping collisions in the network. However, a LAN design cannot remove broadcasts, because broadcast frames play an important role in many protocols. So when thinking about broadcast domains, the choices are more about tradeoffs rather than designing to remove broadcasts.

For just one perspective, just think about the size of a broadcast domain—that is, the number of devices in the same broadcast domain. A small number of large broadcast domains can lead to poor performance for the devices in that broadcast domain. However, moving in the opposite direction, to making a large number of broadcast domains each with just a few devices, leads to other problems.

Consider the idea of a too-large broadcast domain for a moment. When a host receives a broadcast, the host must process the received frame. All hosts need to send some broadcasts to function properly, so when a broadcast arrives, the NIC must interrupt the computer's CPU to give the incoming message to the CPU. The CPU must spend time thinking about the received broadcast frame. (For example, IP Address Resolution Protocol [ARP] messages are LAN broadcasts, as mentioned in Chapter 4, "Fundamentals of IPv4 Addressing and Routing.") So, broadcasts happen, which is good, but broadcasts do require all the hosts to spend time processing each broadcast frame. The more devices in the same broadcast domain, the more unnecessary interruptions of each device's CPU.

This section of the book does not try to give a sweeping review of all VLAN design tradeoffs. Instead, you can see that the size of a VLAN should be considered, but many other factors come in to play as well. How big are the VLANs? How are the devices grouped? Do VLANs span across all switches or just a few? Is there any apparent consistency to the VLAN design, or is it somewhat haphazard? Answering these questions helps reveal what the designer was thinking, as well as what the realities of operating a network may have required.

NOTE If you would like more detail about Cisco recommendations about what to put in what VLAN, which impacts the size of VLANs, read the most recent Cisco document, "Campus LAN validated design" by searching on that phrase at Cisco.com.

Summarizing the main points about broadcast domains:

Key Topic

- Broadcasts exists, so be ready to analyze a design to define each broadcast domain, that is, each set of devices whose broadcasts reach the other devices in that domain.
- VLANs by definition are broadcast domains created though configuration.
- Routers, because they do not forward LAN broadcasts, create separate broadcast domains off their separate Ethernet interfaces.

Analyzing Campus LAN Topologies

The term *campus LAN* refers to the LAN created to support the devices in a building or in multiple buildings in somewhat close proximity to one another. For example, a company might lease office space in several buildings in the same office park. The network engineers can then build a campus LAN that includes switches in each building, plus Ethernet links between the switches in the buildings, to create a larger campus LAN.

When planning and designing a campus LAN, the engineers must consider the types of Ethernet available and the cabling lengths supported by each type. The engineers also need to choose the speeds required for each Ethernet segment. In addition, some thought needs to be given to the idea that some switches should be used to connect directly to end-user devices, whereas other switches might need to simply connect to a large number of these end-user switches. Finally, most projects require that the engineer consider the type of equipment that is already installed and whether an increase in speed on some segments is worth the cost of buying new equipment.

This second of three major sections of the chapter discusses the topology of a campus LAN design. Network designers do not just plug in devices to any port and connect switches to each other in an arbitrary way, like you might do with a few devices on the same table in a lab. Instead, there are known better ways to design the topology of a campus LAN, and this section introduces some of the key points and terms. The last major section of the chapter then looks at how to choose which Ethernet standard to use for each link in that campus LAN design, and why you might choose one versus another.

Two-Tier Campus Design (Collapsed Core)

To sift through all the requirements for a campus LAN, and then have a reasonable conversation about it with peers, most Cisco-oriented LAN designs use some common terminology to refer to the design. For this book's purposes, you should be aware of some of the key campus LAN design terminology.

The Two-Tier Campus Design

Figure K-10 shows a typical design of a large campus LAN, with the terminology included in the figure. This LAN has around 1000 PCs connected to switches that support around 25 ports each. Explanations of the terminology follow the figure.

Key Topic

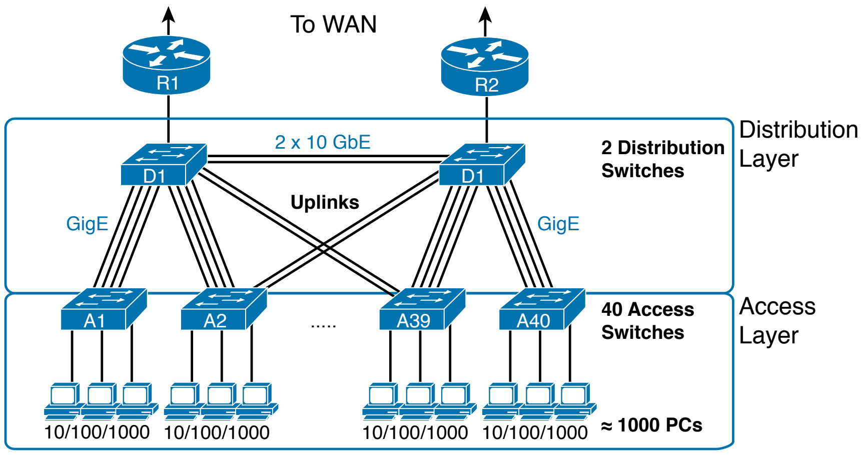

Figure K-10 *Campus LAN with Design Terminology Listed*

Cisco uses three terms to describe the role of each switch in a campus design: *access*, *distribution*, and *core*. The roles differ based on whether the switch forwards traffic from user devices and the rest of the LAN (access), or whether the switch forwards traffic between other LAN switches (distribution and core).

Access switches connect directly to end users, providing user device access to the LAN. Access switches normally send traffic to and from the end-user devices to which they are connected and sit at the edge of the LAN.

Distribution switches provide a path through which the access switches can forward traffic to each other. By design, each of the access switches connects to at least one distribution switch, typically to two distribution switches for redundancy. The distribution switches provide the service of forwarding traffic to other parts of the LAN. Note that most designs use at least two uplinks to two different distribution switches (as shown in Figure K-10) for redundancy.

The figure shows a two-tier design, with the tiers being the access tier (or layer) and the distribution tier (or layer). A two-tier design solves two major design needs:

- Provides a place to connect end-user devices (the access layer, with access switches)
- Connects the switches with a reasonable number of cables and switch ports by connecting all 40 access switches to two distribution switches

Topology Terminology Seen Within a Two-Tier Design

The exam topics happen to list a couple of terms about LAN and WAN topology and design, so this is a good place to pause to discuss those terms for a moment.

First, consider these more formal definitions of four topology terms:

Key Topic

Star: A design in which one central device connects to several others, so that if you drew the links out in all directions, the design would look like a star with light shining in all directions.

Full mesh: For any set of network nodes, a design that connects a link between each pair of nodes.

Partial mesh: For any set of network nodes, a design that connects a link between some pairs of nodes, but not all. In other words, a mesh that is not a full mesh.

Hybrid: A design that combines topology design concepts into a larger (typically more complex) design.

Armed with those formal definitions, note that the two-tier design is indeed a hybrid design that uses both a star topology at the access layer and a partial mesh at the distribution layer. To see why, consider Figure K-11. It redraws a typical access layer switch, but instead of putting the PCs all below the switch, it spreads them around the switch. Then on the right, a similar version of the same drawing shows why the term star might be used—the topology looks a little like a child's drawing of a star.

Key Topic

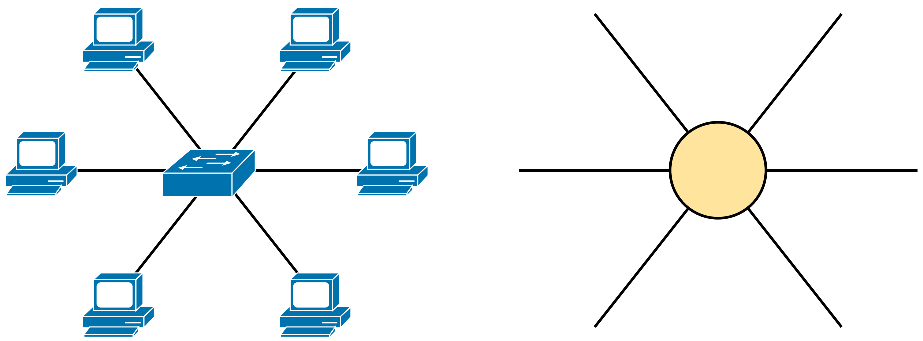

Figure K-11 *The Star Topology Design Concept in Networking*

The distribution layer creates a partial mesh. If you view the access and distribution switches as nodes in a design, some nodes have a link between them, and some do not. Just refer to Figure K-10 and note that, by design, none of the access layer switches connect to each other.

Finally, a design could use a full mesh. However, for a variety of reasons beyond the scope of the design discussion here, a campus design typically does not need to use the number of links and ports required by a full mesh design. However, just to make the point, first consider how many links and switch ports would be required for a single link between nodes in a full mesh, with six nodes, as shown in Figure K-12.

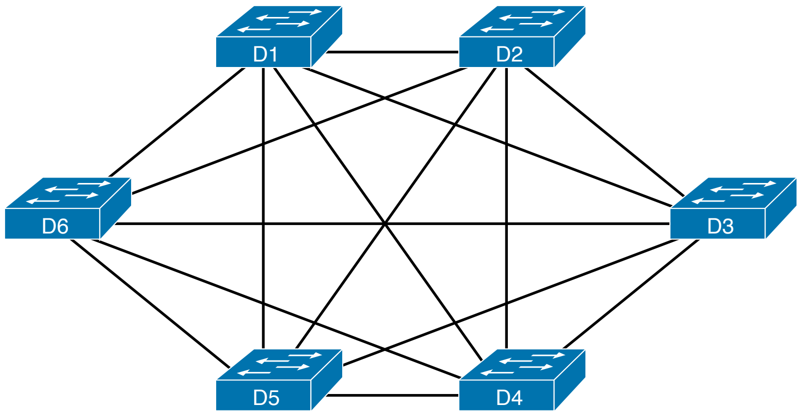

Figure K-12 *Using a Full Mesh at the Distribution Layer, 6 Switches, 15 Links*

Even with only six switches, a full mesh would consume 15 links (and 30 switch ports—two per link).

Now think about a full mesh at the distribution layer for a design like Figure K-10, with 40 access switches and two distribution switches. Rather than drawing it and counting it, the number of links is calculated with this old math formula from high school: $N(N - 1) / 2$, or in this case, 42 * 41 / 2 = 861 links, and 1722 switch ports consumed among all switches.

For comparison's sake, the partial mesh design of Figure K-10, with a pair of links from each access switch to each distribution switch, requires only 160 links and a total of 320 ports among all switches.

Three-Tier Campus Design (Core)

The two-tier design of Figure K-10, with a partial mesh of links at the distribution layer, happens to be the most common campus LAN design. It also goes by two common names: a two-tier design (for obvious reasons), and a collapsed core (for less obvious reasons). The term *collapsed core* refers to the fact that the two-tier design does not have a third tier, the core tier. This next topic examines a three-tier design that does have a core, for perspective.

Imagine your campus has just two or three buildings. Each building has a two-tier design inside the building, with a pair of distribution switches in each building and access switches spread around the building as needed. How would you connect the LANs in each building? Well, with just a few buildings, it makes sense to simply cable the distribution switches together, as shown in Figure K-13.

Key Topic

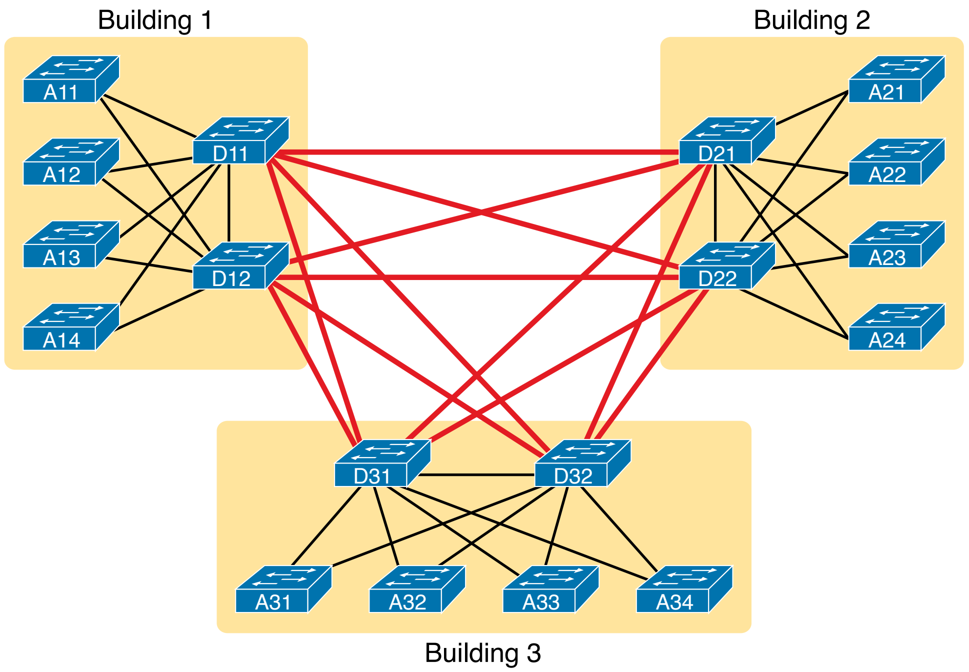

Figure K-13 *Two-Tier Building Design, No Core, Three Buildings*

The design in Figure K-13 works well, and many companies use this design. Sometimes the center of the network uses a full mesh, sometimes a partial mesh, depending on the availability of cables between the buildings.

However, a design with a third tier (a core tier) saves on switch ports and on cables in larger designs. And note that with the links between buildings, the cables run outside, are often more expensive to install, are almost always fiber cabling with more expensive switch ports, so conserving the number of cables used between buildings can help reduce costs.

A three-tier core design, unsurprisingly at this point, adds a few more switches (core switches), which provide one function: to connect the distribution switches. Figure K-14 shows the migration of the Figure K-13 collapsed core (that is, a design without a core) to a three-tier core design.

Key Topic

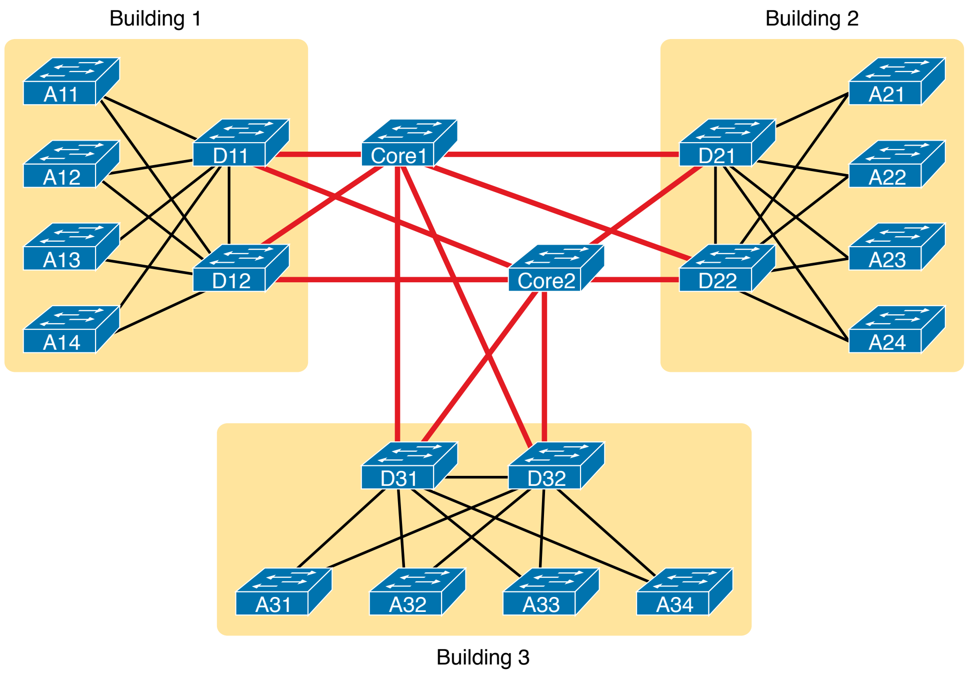

Figure K-14 *Three-Tier Building Design (Core Design), Three Buildings*

NOTE The core switches sit in the middle of the figure. In the physical world, they often sit in the same room as one of the distribution switches, rather than in some purpose-built room in the middle of the office park. The figure focuses more on the topology rather than the physical location.

By using a core design, with a partial mesh of links in the core, you still provide connectivity to all parts of the LAN, and to the routers that send packets over the WAN, just with fewer links between buildings.

The following list summarizes the terms that describe the roles of campus switches:

- **Access:** Provides a connection point (access) for end-user devices. Does not forward frames between two other access switches under normal circumstances.
- **Distribution:** Provides an aggregation point for access switches, providing connectivity to the rest of the devices in the LAN, forwarding frames between switches, but not connecting directly to end-user devices.
- **Core:** Aggregates distribution switches in very large campus LANs, providing very high forwarding rates for the larger volume of traffic due to the size of the network.

Topology Design Terminology

The ICND1 and CCNA exam topics specifically mention several network design terms related to topology. This next topic summarizes those key terms to connect the terms to the matching ideas.

First, consider Figure K-15, which shows a few of the terms. First, on the left, drawings often show access switches with a series of cables, parallel to each other. However, an access switch and its access links is often called a *star topology*. Why? Look at the redrawn access switch in the center of the figure, with the cables radiating out from the center. It does not look like a real star, but it looks a little like a child's drawing of a star, hence the term star topology.

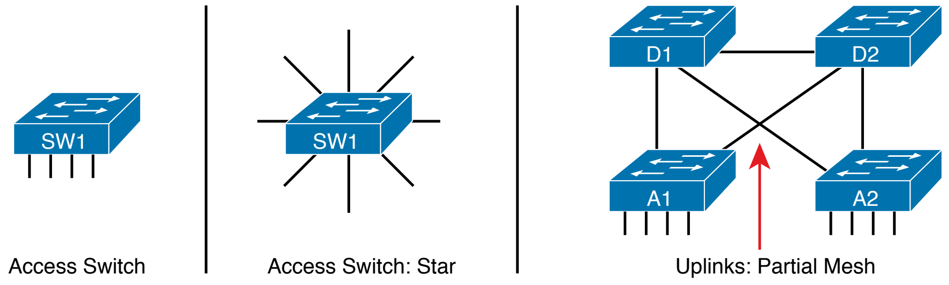

Figure K-15 *LAN Design Terminology*

The right side of the figure repeats a typical two-tier design, focusing on the mesh of links between the access and distribution switches. Any group of nodes that connect with more links than a star topology is typically called a *mesh*. In this case, the mesh is a *partial mesh*, because not all nodes have a direct link between each other. A design that connects all nodes with a link would be a *full mesh*.

Real networks make use of these topology ideas, but often a network combines the ideas together. For instance, the right side of Figure K-14 combines the star topology of the access layer with the partial mesh of the distribution layer. So you might hear these designs that combine concepts called a *hybrid design*.

Analyzing LAN Physical Standard Choices

When you look at the design of a network designed by someone else, you can look at all the different types of cabling used, the different types of switch ports, and the Ethernet standards used in each case. Then ask yourself: Why did they choose a particular type of Ethernet link for each link in the network? Asking that question, and investigating the answer, starts to reveal much about building the physical campus LAN.

The IEEE has done an amazing job developing Ethernet standards that give network designers many options. Two themes in particular have helped Ethernet grow over the long term:

Key Topic

- The IEEE has developed many additional 802.3 standards for different types of cabling, different cable lengths, and for faster speeds.
- All the physical standards rely on the same consistent data-link details, with the same standard frame formats. That means that one Ethernet LAN can use many types of physical links to meet distance, budget, and cabling needs.

For example, think about the access layer of the generic design drawings, but now think about cabling and Ethernet standards. In practice, access layer switches sit in a locked wiring closet somewhere on the same floor as the end user devices. Electricians have installed unshielded twisted-pair (UTP) cabling used at the access layer, running from that wiring closet to each wall plate at each office, cubicle, or any place where an Ethernet device might need to connect to the LAN. The type and quality of the cabling installed between the wiring closet and each Ethernet outlet dictate what Ethernet standards can be supported. Certainly, whoever designed the LAN at the time the cabling was installed thought about what type of cabling was needed to support the types of Ethernet physical standards that were going to be used in that LAN.

Ethernet Standards

Over time, the IEEE has continued to develop and release new Ethernet standards, for new faster speeds and to support new and different cabling types and cable lengths. Figure K-16 shows some insight into Ethernet speed improvements over the years. The early standards up through the early 1990s ran at 10 Mbps, with steadily improving cabling and topologies. Then, with the introduction of Fast Ethernet (100 Mbps) in 1995, the IEEE began ramping up the speeds steadily over the next few decades, continuing even until today.

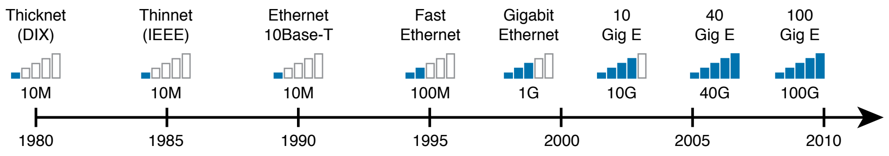

Figure K-16 *Ethernet Standards Timeline*

NOTE Often, the IEEE first introduces support for the next higher speed using some forms of fiber optic cabling, and later, sometimes many years later, the IEEE completes the work to develop standards to support the same speed on UTP cabling. Figure K-16 shows the earliest standards for each speed, no matter what cabling.

When the IEEE introduces support for a new type of cabling, or a faster speed, they create a new standard as part of 802.3. These new standards have a few letters behind the name. So, when speaking of the standards, sometimes you might refer to the standard name (with letters). For instance, the IEEE standardized Gigabit Ethernet support using inexpensive UTP cabling in standard 802.3ab. However, more often, engineers refer to that same standard as 1000BASE-T or simply Gigabit Ethernet. Table K-1 lists some of the IEEE 802.3 physical layer standards and related names for perspective.

Table K-1 IEEE Physical Layer Standards

Original IEEE Standard	Shorthand Name	Informal Names	Speed	Typical Cabling
802.3i	10BASE-T	Ethernet	10 Mbps	UTP
802.3u	100BASE-T	Fast Ethernet	100 Mbps	UTP
802.3z	1000BASE-X	Gigabit Ethernet, GigE	1000 Mbps (1 Gbps)	Fiber
802.3ab	1000BASE-T	Gigabit Ethernet, GigE	1000 Mbps (1 Gbps)	UTP
802.3ae	10GBASE-X	10 GigE	10 Gbps	Fiber
802.3an	10GBASE-T	10 GigE	10 Gbps	UTP
802.3ba	40GBASE-X	40 GigE	40 Gbps	Fiber
802.3ba	100GBASE-X	100 GigE	100 Gbps	Fiber

Choosing the Right Ethernet Standard for Each Link

When designing an Ethernet LAN, you can and should think about the topology, with an access layer, a distribution layer, and possibly a core layer. But thinking about the topology does not tell you which specific standards to follow for each link. Ultimately, you need to pick which Ethernet standard to use for each link, based on the following kinds of facts about each physical standard:

- The speed
- The maximum distance allowed between devices when using that standard/cabling
- The cost of the cabling and switch hardware
- The availability of that type of cabling already installed at your facilities

Consider the three most common types of Ethernet today (10BASE-T, 100BASE-T, and 1000BASE-T). They all have the same 100-meter UTP cable length restriction. They all use UTP cabling. However, not all UTP cabling meets the same quality standard, and as it turns out, the faster the Ethernet standard, the higher the required cable quality category needed to support that standard. As a result, some buildings might have better cabling that supports speeds up through Gigabit Ethernet, whereas some buildings may support only Fast Ethernet.

The Telecommunications Industry Association (TIA; tiaonline.org) defines Ethernet cabling quality standards. Each Ethernet UTP standard lists a TIA cabling quality (called a *category*) as the minimum category that the standard supports. For example, 10BASE-T allows for Category 3 (CAT3) cabling or better. 100BASE-T requires higher-quality CAT5 cabling, and 1000BASE-T requires even higher-quality CAT5e cabling. (The TIA standards follow a general "higher number is better cabling" in their numbering.) For instance, if an older facility had only CAT5 cabling installed between the wiring closets and each cubicle, the engineers would have to consider upgrading the cabling to fully support Gigabit Ethernet. Table K-2 lists the more common types of Ethernet and their cable types and length limitations.

Table K-2 Ethernet Types, Media, and Segment Lengths (Per IEEE)

Ethernet Type	Media	Maximum Segment Length
10BASE-T	TIA CAT3 or better, 2 pairs	100 m (328 feet)
100BASE-T	TIA CAT5 UTP or better, 2 pairs	100 m (328 feet)
1000BASE-T	TIA CAT5e UTP or better, 4 pairs	100 m (328 feet)
10GBASE-T	TIA CAT6a UTP or better, 4 pairs	100 m (328 feet)
10GBASE-T[1]	TIA CAT6 UTP or better, 4 pairs	38–55 m (127–180 feet)
1000BASE-SX	Multimode fiber	550 m (1800 feet)
1000BASE-LX	Multimode fiber	550 m (1800 feet)
1000BASE-LX	9-micron single-mode fiber	5 km (3.1 miles)

[1] The option for 10GBASE-T with slightly less quality CAT6 cabling, but at shorter distances, is an attempt to support 10Gig Ethernet for some installations with CAT6 installed cabling.

Ethernet defines standards for using fiber optic cables as well. Fiber optic cables include ultrathin strands of glass through which light can pass. To send bits, the switches can alternate between sending brighter and dimmer light to encode 0s and 1s on the cable.

Generally comparing optical cabling versus UTP cabling Ethernet standards, two obvious points stand out. Optical standards allow much longer cabling, while generally costing more for the cable and the switch hardware components. Optical cables experience much less interference from outside sources compared to copper cables, which allows for longer distances.

When considering optical Ethernet links, many standards exist, but with two general categories. Comparing the two, the cheaper options generally support distances into the hundreds of meters, using less expensive light-emitting diodes (LED) to transmit data. Other optical standards support much longer distances into multiple kilometers, using more expensive cabling and using lasers to transmit the data. The trade-off is basic: For a given link, how long does the cable need to run, what standards support that distance, and which is the least expensive to meet that need?

In reality, most engineers remember only the general facts from tables like Table K-2: 100 meters for UTP, about 500 meters for multimode fiber, and about 5000 meters for some single mode fiber Ethernet standards. When it is time to get serious about designing the details of each link, the engineer must get into the details, calculating the length of each cable based on its path through the building, and so on.

Wireless LANs Combined with Wired Ethernet

Modern campus LANs include a large variety of wireless devices that connect to the access layer of the LAN. As it turns out, Cisco organizes wireless LANs into a separate certification track—CCNA, CCNP, and CCIE Wireless—so the CCNA R&S track has traditionally had only a little wireless LAN coverage. The current version of the exams are no different, with this one exam CCNA R&S topic mentioning wireless LANs:

> Describe the impact of infrastructure components in an enterprise network: Access points and wireless controllers

Do not let that small mention of wireless technology make you think that wireless is less important than Ethernet. In fact, there may be more wireless devices than wired at the access layer of today's enterprise networks. Both are important; Cisco just happens to keep the educational material for wireless in a separate certification track.

This last topic in the chapter examines that one exam topic that mentions two wireless terms.

Home Office Wireless LANs

First, the IEEE defines both Ethernet LANs and Wireless LANs. In case it was not obvious yet, all Ethernet standards use cables—that is, Ethernet defines wired LANs. The IEEE 802.11 working group defines Wireless LANs, also called Wi-Fi per a trademarked term from the Wi-Fi Alliance (wi-fi.org), a consortium that helps to encourage wireless LAN development in the marketplace.

Most of you have used Wi-Fi, and may use it daily. Some of you may have set it up at home, with a basic setup as shown in Figure K-17. In a home, you probably used a single consumer device called a *wireless router*. One side of the device connects to the Internet, while the other side connects to the devices in the home. In the home, the devices can connect either with Wi-Fi or with a wired Ethernet cable.

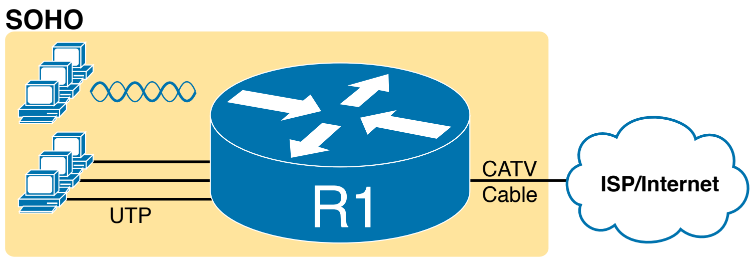

Figure K-17 *A Typical Home Wired and Wireless LAN*

While the figure shows the hardware as a single router icon, internally, that one wireless router acts like three separate devices you would find in an enterprise campus:

- An Ethernet switch, for the wired Ethernet connections
- A wireless access point (AP), to communicate with the wireless devices and forward the frames to/from the wired network
- A router, to route IP packets to/from the LAN and WAN (Internet) interfaces

Figure K-18 repeats the previous figure, breaking out the internal components as if they were separate physical devices, just to make the point that a single consumer wireless router acts like several different devices.

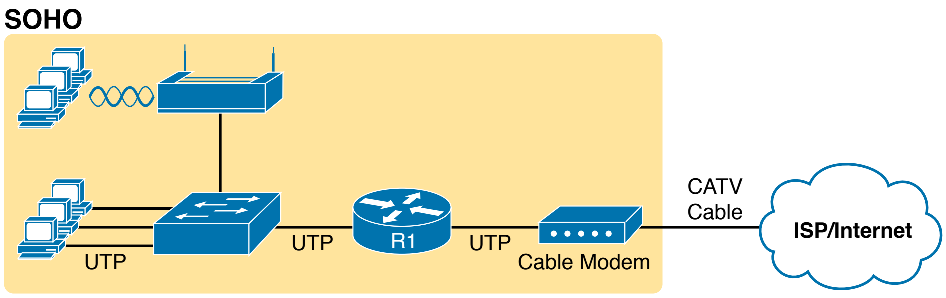

Figure K-18 *A Representation of the Functions Inside a Consumer Wireless Routing Product*

In a small office/home office (SOHO) wireless LAN, the wireless AP acts autonomously, doing all the work required to create and control the wireless LAN (WLAN). (In most enterprise WLANs, the AP does not act autonomously.) In other words, the autonomous AP communicates with the various wireless devices using 802.11 protocols and radio waves. It uses Ethernet protocols on the wired side. It converts between the differences in header formats between 802.11 and 802.3 frames before forwarding to/from 802.3 Ethernet and 802.11 wireless frames.

Beyond those basic forwarding actions, the autonomous AP must perform a variety of control and management functions. The AP authenticates new devices, defines the name of the WLAN (called a service set ID, or SSID), and other details.

Enterprise Wireless LANs and Wireless LAN Controllers

If you connect to your WLAN at home from your tablet, phone, or laptop, and then walk down the street with that same device, you expect to lose your Wi-Fi connection at some point. You do not expect to somehow automatically connect to a neighbor's Wi-Fi network, particularly if they did the right thing and set up security functions on their AP to prevent others from accessing their home Wi-Fi network. The neighborhood does not create one WLAN supported by the devices in all the houses and apartments; instead, it has lots of little autonomous WLANs.

However, in an enterprise, the opposite needs to happen. We want people to be able to roam around the building and office campus and keep connected to the Wi-Fi network. This requires many APs, which work together rather than autonomously to create one wireless LAN.

First, think about the number of APs an enterprise might need. Each AP can cover only a certain amount of space, depending on a large number of conditions and the wireless standard. (The size varies, but the distances sit in the 100 to 200 feet range.) At the same time, you might have the opposite problem; you may just need lots of APs in a small space, just to add capacity to the WLAN. Much of the time spent designing WLANs revolves around deciding how many APs to place in each space, and of what types, to handle the traffic.

NOTE If you have not paid attention before, start looking around the ceilings of any new buildings you enter, even retail stores, and look for their wireless APs.

Each AP must then connect to the wired LAN, because most of the destinations that wireless users need to communicate with sit in the wired part of the network. In fact, the APs typically sit close to where users sit, for obvious reasons, so the APs connect to the same access switches as the end users, as shown in Figure K-19.

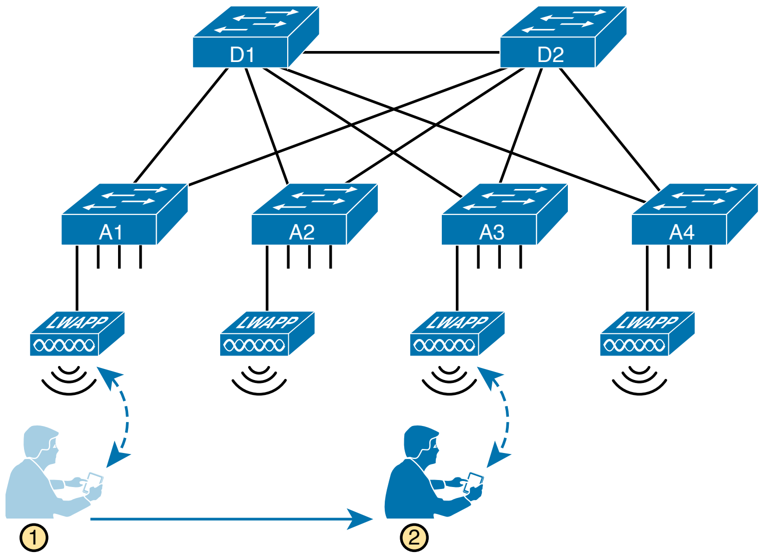

Figure K-19 *Campus LAN, Multiple Lightweight APs, with Roaming*

Now imagine that is you at the bottom of the figure. Your smartphone has Wi-Fi enabled, so that when you walk into work, your phone automatically connects to the company WLAN. You roam around all day, going to meetings, lunch, and so on. All day long you stay connected to the company WLAN, but your phone connects to and uses many different APs.

Supporting roaming and other enterprise WLAN features by using autonomous APs can be difficult at best. You could imagine that if you had a dozen APs per floor, you might have hundreds of APs in a campus—all of which need to know about that one WLAN.

The solution: remove all the control and management features from the APs, and put them in one centralized place, called a Wireless Controller, or Wireless LAN Controller (WLC). The APs no longer act autonomously, but instead act as lightweight APs (LWAPs), just forwarding data between the wireless LAN and the WLC. All the logic to deal with roaming, defining WLANs (SSIDs), authentication, and so on happens in the centralized WLC rather than on each AP. Summarizing:

Wireless LAN controller: Controls and manages all AP functions (for example, roaming, defining WLANs, authentication)

Lightweight AP (LWAP): Forwards data between the wired and wireless LAN, and specifically forwarding data through the WLC using a protocol like Control And Provisioning of Wireless Access Points (CAPWAP)

With the WLC and LWAP design, the combined LWAPs and WLC can create one big wireless network, rather than creating a multitude of disjointed wireless networks. The key to making it all work is that all wireless traffic flows through the WLC, as shown in Figure K-20. (The LWAPs commonly use a protocol called CAPWAP, by the way.)

Key Topic

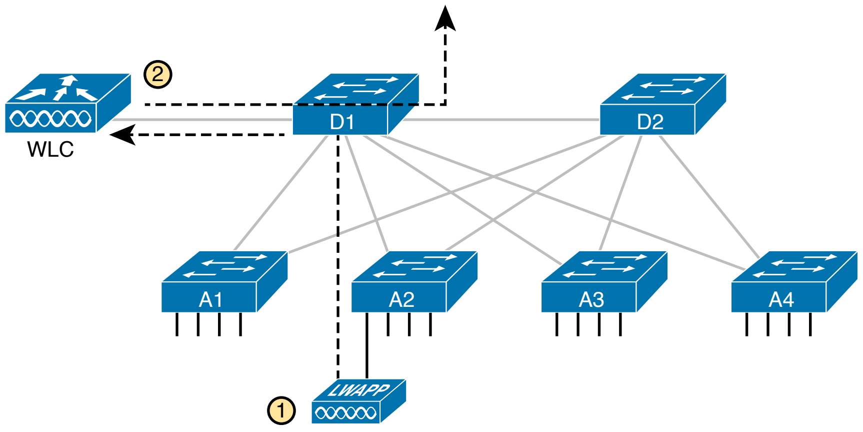

Figure K-20 *Campus LAN, Multiple Lightweight APs, with Roaming*

By forwarding all the traffic through the WLC, the WLC can make the right decisions across the enterprise. For example, you might create a marketing WLAN, an engineering WLAN, and so on, and all the APs know about and support those multiple different WLANs. Users that connect to the engineering WLAN should use the same authentication rules regardless of which AP they use—and the WLC makes that possible. Or consider roaming for a moment. If at one instant a packet arrives for your phone, and you are associated with AP1, and when the next packet arrives over the wired network you are now connected to AP4, how could that packet be delivered through the network? Well, it always goes to the WLC, and because the WLC keeps in contact with the APs and knows that your phone just roamed to another AP, the WLC knows where to forward the packet.

Subnet Design

NOTE This appendix contains an entire chapter that was published as a chapter in one of the past editions of this book or a related book. The author includes this appendix with the current edition as extra reading for anyone interested in learning more. However, note that the content in this appendix has not been edited since it was published in the earlier edition, so references to exams and exam topics, and to other chapters, will be outdated. This appendix was previously published as Chapter 21 of the book *CCENT/CCNA ICND1 100-105 Official Cert Guide*, published in 2016.

So far in this book, most of the discussion about IPv4 used examples with the addresses and masks already given. This book has shown many examples already, but the examples so far do not ask you to pick the IP address or pick the mask. Instead, as discussed back in Chapter 11, "Perspectives on IPv4 Subnetting," this book so far has assumed that someone else designed the IP addressing and subnetting plan, and this book shows how to implement it.

This chapter turns that model around. It goes back to the progression of building and implementing IPv4, as discussed in Chapter 11, as shown in Figure L-1. This chapter picks up the story right after some network engineer has chosen a Class A, B, or C network to use for the enterprise's IPv4 network. And then this chapter discusses the design choices related to picking one subnet mask to use for all subnets (the first major section) and what subnet IDs that choice creates (the second major section).

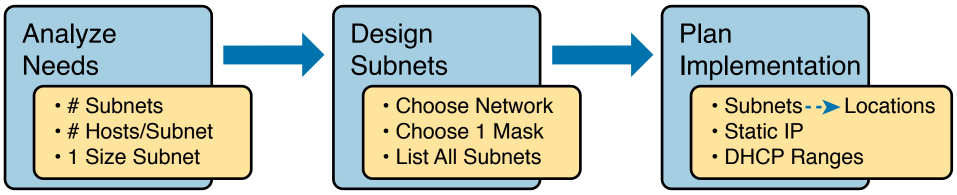

Figure L-1 *Subnet Design and Implementation Process from Chapter 11*

Foundation Topics

Choosing the Mask(s) to Meet Requirements

This first major section examines how to find all the masks that meet the stated requirements for the number of subnets and the number of hosts per subnet. To that end, the text assumes that the designer has already determined these requirements and has chosen the network number to be subnetted. The designer has also made the choice to use a single subnet mask value throughout the classful network.

Armed with the information in this chapter, you can answer questions such as the following, a question that matters both for real engineering jobs and the Cisco exams:

> You are using Class B network 172.16.0.0. You need 200 subnets and 200 hosts/subnet. Which of the following subnet mask(s) meet the requirements? (This question is then followed by several answers that list different subnet masks.)

To begin, this section reviews the concepts in Chapter 13's section "Choose the Mask." That section introduced the main concepts about how an engineer, when designing subnet conventions, must choose the mask based on the requirements.

After reviewing the related concepts from Chapter 13, this section examines this topic in more depth. In particular, this chapter looks at three general cases:

- No masks meet the requirements.
- One and only one mask meets the requirements.
- Multiple masks meet the requirements.

For this last case, the text discusses how to determine all masks that meet the requirements and the trade-offs related to choosing which one mask to use.

Review: Choosing the Minimum Number of Subnet and Host Bits

The network designer must examine the requirements for the number of subnets and number of hosts/subnet, and then choose a mask. As discussed in detail in Chapter 15, "Analyzing Subnet Masks," a classful view of IP addresses defines the three-part structure of an IP address: network, subnet, and host. The network designer must choose the mask so that the number of subnet and host bits (S and H, respectively, in Figure L-2) meet the requirements.

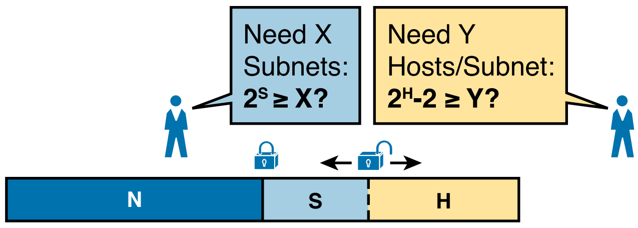
Need X Subnets: $2^S \geq X$?
Need Y Hosts/Subnet: $2^H-2 \geq Y$?
N
S
H

Figure L-2 *Choosing the Number of Subnet and Host Bits*

Basically, the designer must choose S subnet bits so that the number of subnets that can be uniquely numbered with S bits (2^S) is at least as large as the required number of subnets. The designer applies similar logic to the number of host bits H, while noting that the formula is $2^H - 2$, because of the two reserved numbers in each subnet. So, keeping the powers of 2 handy, as shown in Table L-1, will be useful when working through these problems.

Table L-1 Powers of 2 Reference for Designing Masks

Number of Bits	2^X	Number of Bits	2^X	Number of Bits	2^X	Number of Bits	2^X
1	2	5	32	9	512	13	8192
2	4	6	64	10	1024	14	16,384
3	8	7	128	11	2048	15	32,768
4	16	8	256	12	4096	16	65,536

More formally, the process must determine the minimum values for both S and H that meet the requirements. The following list summarizes the initial steps to choose the mask:

Step 1. Determine the number of network bits (N) based on the class.

Step 2. Determine the smallest value of S, so that 2^S => X, where X represents the required number of subnets.

Step 3. Determine the smallest value of H, so that $2^H - 2$ => Y, where Y represents the required number of hosts/subnet.

The next three sections examine how to use these initial steps to choose a subnet mask.

No Masks Meet Requirements

After you determine the required number of subnet and host bits, those bits might not fit into a 32-bit IPv4 subnet mask. Remember, the mask always has a total of 32 bits, with binary 1s in the network and subnet parts and binary 0s in the host part. For the exam, a question might provide a set of requirements that simply cannot be met with 32 total bits.

For example, consider the following sample exam question:

> A network engineer is planning a subnet design. The engineer plans to use Class B network 172.16.0.0. The network has a need for 300 subnets and 280 hosts per subnet. Which of the following masks could the engineer choose?

The three-step process shown in the previous section shows that these requirements mean that a total of 34 bits will be needed, so no mask meets the requirements. First, as a Class B network, 16 network bits exist, with 16 host bits from which to create the subnet part and to leave enough host bits to number the hosts in each subnet. For the number of subnet bits, S=8 does not work, because $2^8 = 256 < 300$. However, S=9 works, because $2^9 = 512$ => 300. Similarly, because $2^8 - 2 = 254$, which is less than 300, 8 host bits are not enough but 9 host bits ($2^9 - 2 = 510$) are just enough.

These requirements do not leave enough space to number all the hosts and subnet, because the network, subnet, and host parts add up to more than 32:

> N=16, because as a Class B network, 16 network bits exist.
>
> The minimum S=9, because S=8 provides too few subnets ($2^8 = 256 < 300$) but S=9 provides $2^9 = 512$ subnets.

The minimum H=9, because H=8 provides too few hosts ($2^8 - 2 = 254 < 280$) but H=9 provides $2^9 - 2 = 510$ hosts/subnet.

Figure L-3 shows the resulting format for the IP addresses in this subnet, after the engineer has allocated 9 subnet bits on paper. Only 7 host bits remain, but the engineer needs 9 host bits.

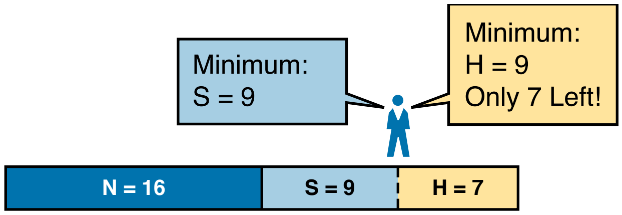

Figure L-3 *Too Few Bits for the Host Part, Given the Requirements*

One Mask Meets Requirements

The process discussed in this chapter in part focuses on finding the smallest number of subnet bits and the smallest number of host bits to meet the requirements. If the engineer tries to use these minimum values, and the combined network, subnet, and host parts add up to exactly 32 bits, exactly one mask meets the requirements.

For example, consider a revised version of the example in the previous section, with smaller numbers of subnet and hosts, as follows:

A network engineer is planning a subnet design. The engineer plans to use Class B network 172.16.0.0. The network has a need for 200 subnets and 180 hosts per subnet. Which of the following masks could the engineer choose?

The three-step process to determine the numbers of network, minimum subnet, and minimum host bits results in a need for 16, 8, and 8 bits, respectively. As before, with a Class B network, 16 network bits exist. With a need for only 200 subnets, S=8 does work, because $2^8 = 256 => 200$; 7 subnet bits would not supply enough subnets ($2^7 = 128$). Similarly, because $2^8 - 2 = 254 => 180$, 8 host bits meet the requirements; 7 host bits (for 126 total hosts/subnet) would not be enough.

Figure L-4 shows the resulting format for the IP addresses in this subnet.

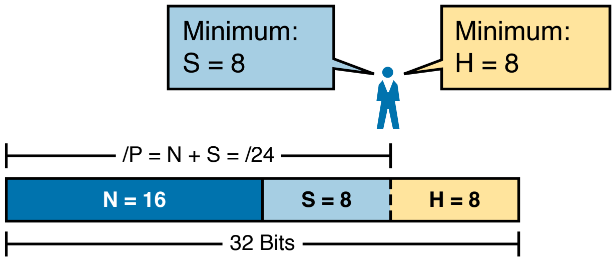

Figure L-4 *One Mask That Meets Requirements*

Figure L-4 shows the mask conceptually. To find the actual mask value, simply record the mask in prefix format (/P), where P = N + S or, in this case, /24.

Multiple Masks Meet Requirements

Depending on the requirements and choice of network, several masks might meet the requirements for the numbers of subnets and hosts/subnet. In these cases, you need to find all the masks that could be used. Then, you have a choice, but what should you consider when choosing one mask among all those that meet your requirements? This section shows how to find all the masks, as well as the facts to consider when choosing one mask from the list.

Finding All the Masks: Concepts

To help you better understand how to find all the subnet masks in binary, this section uses two major steps. In the first major step, you build the 32-bit binary subnet mask on paper. You write down binary 1s for the network bits, binary 1s for the subnet bits, and binary 0s for the host bits, just as always. However, you will use the minimum values for S and H. And when you write down these bits, you will not have 32 bits yet!

For example, consider the following problem, similar to the earlier examples in this chapter but with some changes in the requirements:

> A network engineer is planning a subnet design. The engineer plans to use Class B network 172.16.0.0. The network has a need for 50 subnets and 180 hosts per subnet. Which of the following masks could the engineer choose?

This example is similar to an earlier example, except that only 50 subnets are needed in this case. Again, the engineer is using private IP network 172.16.0.0, meaning 16 network bits. The design requires only 6 subnet bits in this case, because $2^6 = 64 => 50$, and with only 5 subnet bits, $2^5 = 32 < 50$. The design then requires a minimum of 8 host bits.

One way to discuss the concepts and find all the masks that meet these requirements is to write down the bits in the subnet mask: binary 1s for the network and subnet parts and binary 0s for the host part. However, think of the 32-bit mask as 32-bit positions, and when writing the binary 0s, *write them on the far right*. Figure L-5 shows the general idea.

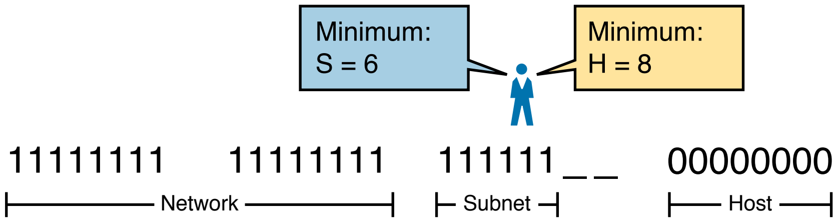

Figure L-5 *Incomplete Mask with N=16, S=6, and H=8*

Figure L-5 shows 30 bits of the mask, but the mask must have 32 bits. The 2 remaining bits might become subnet bits, being set to binary 1. Alternatively, these 2 bits could be made host bits, being set to binary 0. The engineer simply needs to choose based on whether he would like more subnet bits, to number more subnets, or more host bits, to number more hosts/subnet.

Regardless of the requirements, when choosing any IPv4 subnet mask, you must always follow this rule:

Key Topic

A subnet mask begins with all binary 1s, followed by all binary 0s, with no interleaving of 1s and 0s.

With the example shown in Figure L-5, with 2 open bits, one value (binary 01) breaks this rule. However, the other three combinations of 2 bits (00, 10, and 11) do not break the rule. As a result, three masks meet the requirements in this example, as shown in Figure L-6.

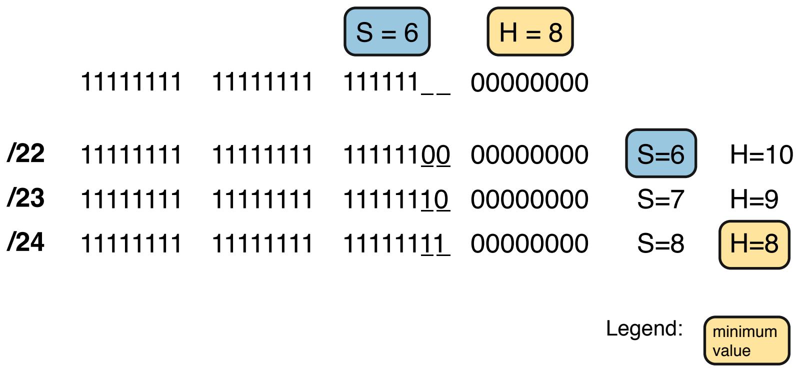
S = 6
H = 8
11111111 11111111 111111__ 00000000
/22 11111111 11111111 11111100 00000000 S=6 H=10
/23 11111111 11111111 11111110 00000000 S=7 H=9
/24 11111111 11111111 11111111 00000000 S=8 H=8
Legend:
minimum value

Figure L-6 *Three Masks That Meet the Requirements*

In the three masks, the first has the least number of subnet bits among the three masks, but therefore has the most number of host bits. So, the first mask maximizes the number of hosts/subnet. The last mask uses the minimum value for the number of host bits, therefore using the most number of subnet bits allowed while still meeting the requirements. As a result, the last mask maximizes the number of subnets allowed.

Finding All the Masks: Math

Although the concepts related to the example shown in Figures L-5 and L-6 are important, you can find the range of masks that meets the requirements more easily just using some simple math. The process to find the masks just requires a few steps, after you know N and the minimum values of S and H. The process finds the value of /P when using the least number of subnet bits, and when using the least number of host bits, as follows:

Key Topic

Step 1. Calculate the shortest prefix mask (/P) based on the *minimum value of S*, where P = N + S.

Step 2. Calculate the longest prefix mask (/P) based on the *minimum value of H*, where P = 32 – H.

Step 3. The range of valid masks includes all /P values between the two values calculated in the previous steps.

For example, in the example shown in Figure L-6, N = 16, the minimum S = 6, and the minimum H = 8. The first step identifies the shortest prefix mask (the /P with the smallest value of P) of /22 by adding N and S (16 + 6). The second step identifies the longest prefix mask that meets the requirements by subtracting the smallest possible value for H (8, in this case) from 32, for a mask of /24. The third step reminds us that the range is from /22 to /24, meaning that /23 is also an option.

Choosing the Best Mask

When multiple possible masks meet the stated requirements, the engineer has a choice of masks. That, of course, begs some questions: Which mask should you choose? Why would one mask be better than the other? The reasons can be summarized into three main options:

Key Topic

- **To maximize the number of hosts/subnet:** To make this choice, use the shortest prefix mask (that is, the mask with the smallest /P value), because this mask has the largest host part.
- **To maximize the number of subnets:** To make this choice, use the longest prefix mask (that is, the mask with the largest /P value), because this mask has the largest subnet part.
- **To increase both the numbers of supported subnets and hosts:** To make this choice, choose a mask in the middle of the range, which gives you both more subnet bits and more host bits.

For example, in Figure L-6, the range of masks that meet the requirements is /22 – /24. The shortest mask, /22, has the least subnet bits but the largest number of host bits (10) of the three answers, maximizing the number of hosts/subnet. The longest mask, /24, maximizes the number of subnet bits (8), maximizing the number of subnets, at least among the options that meet the original requirements. The mask in the middle, /23, provides some growth in both subnets and hosts/subnet.

The Formal Process

Although this chapter has explained various steps in finding a subnet mask to meet the design requirements, it has not yet collected these concepts into a list for the entire process. The following list collects all these steps into one place for reference. Note that this list does not introduce any new concepts compared to the rest of this chapter; it just puts all the ideas in one place.

Key Topic

Step 1. Find the number of network bits (N) per class rules.

Step 2. Calculate the minimum number of subnet bits (S) so that 2^S => the number of required subnets.

Step 3. Calculate the minimum number of host bits (H) so that $2^H - 2$ => the number of required hosts/subnet.

Step 4. If N + S + H > 32, no mask meets the need.

Step 5. If N + S + H = 32, one mask meets the need. Calculate the mask as /P, where P = N + S.

Step 6. If N + S + H < 32, multiple masks meet the need:

A. Calculate mask /P based on the minimum value of S, where P = N + S. This mask maximizes the number of hosts/subnet.

B. Calculate mask /P based on the minimum value of H, where P = 32 – H. This mask maximizes the number of possible subnets.

C. Note that the complete range of masks includes all prefix lengths between the two values calculated in Steps 6A and 6B.

Practice Choosing Subnet Masks

Take the usual two-phase approach to learning new subnetting math and processes. Take the time now to practice to make sure you understand the fundamentals, using the book and notes as needed. Then, sometime before taking the exam, practice until you can reach the goals in the right column of Table L-2.

Table L-2 Keep-Reading and Take-Exam Goals for Choosing a Subnet Mask

Time Frame	Before Moving to the Next Chapter	Before Taking the Exam
Focus On	Learning how	Being correct and fast
Tools Allowed	All	Your brain and a notepad
Goal: Accuracy	90% correct	100% correct
Goal: Speed	Any speed	15 seconds

Practice Problems for Choosing a Subnet Mask

The following list shows three separate problems, each with a classful network number and a required number of subnets and hosts/subnet. For each problem, determine the minimum number of subnet and host bits that meet the requirements. If more than one mask exists, note which mask maximizes the number of hosts/subnet and which maximizes the number of subnets. If only one mask meets the requirements, simply list that mask. List the masks in prefix format:

1. Network 10.0.0.0, need 1500 subnets, need 300 hosts/subnet
2. Network 172.25.0.0, need 130 subnets, need 127 hosts/subnet
3. Network 192.168.83.0, need 8 subnets, need 8 hosts/subnet

Table L-7, found in the later section "Answers to Earlier Practice Problems," lists the answers.

Finding All Subnet IDs

After the person designing the IP subnetting plan has chosen the one mask to use throughout the Class A, B, or C network, he will soon need to start assigning specific subnet IDs for use in specific VLANs, serial links, and other places in the internetwork that need a subnet. But what are those subnet IDs? As it turns out, after the network ID and one subnet mask for all subnets have been chosen, finding all the subnet IDs just requires doing a little math. This second major section of this chapter focuses on that math, which focuses on a single question:

> Given a single Class A, B, or C network, and the single subnet mask to use for all subnets, what are all the subnet IDs?

When learning how to answer this question, you can think about the problem in either binary or decimal. This chapter approaches the problem using decimal. Although the process itself requires only simple math, the process requires practice before most people can confidently answer this question.

The decimal process begins by identifying the first, or numerically lowest, subnet ID. After that, the process identifies a pattern in all subnet IDs for a given subnet mask so that you can find each successive subnet ID through simple addition. This section examines the key ideas behind this process first; then you are given a formal definition of the process.

NOTE Some videos included on the companion website describe the same fundamental processes to find all subnet IDs. You can view those videos before or after reading this section, or even instead of reading this section, as long as you learn how to independently find all subnet IDs. The process step numbering in the videos might not match the steps shown in this edition of the book.

First Subnet ID: The Zero Subnet

The first step in finding all subnet IDs of one network is incredibly simple: Copy the network ID. That is, take the Class A, B, or C network ID—in other words, the classful network ID—and write it down as the first subnet ID. No matter what Class A, B, or C network you use, and no matter what subnet mask you use, the first (numerically lowest) subnet ID is equal to the network ID.

For example, if you begin with classful network 172.20.0.0, no matter what the mask is, the first subnet ID is 172.20.0.0.

This first subnet ID in each network goes by two special names: either *subnet zero* or the *zero subnet*. The origin of these names is related to the fact that a network's zero subnet, when viewed in binary, has a subnet part of all binary 0s. In decimal, the zero subnet can be easily identified, because the zero subnet always has the exact same numeric value as the network ID itself.

In the past, engineers avoided using zero subnets because of the ambiguity with one number that could represent the entire classful network or it could represent one subnet inside the classful network. To help control that, IOS has a global command that can be set one of two ways:

ip subnet-zero, which allows the configuration of addresses in the zero subnet.

no ip subnet-zero, which prevents the configuration of addresses in the zero subnet.

Although most sites use the default setting to allow zero subnets, you can use the **no ip subnet-zero** command to prevent configuring addresses that are part of a zero subnet. Example L-1 shows how a router rejects an **ip address** command after changing to use **no ip subnet-zero**. Note that the error message does not mention the zero subnet, instead simply stating "bad mask."

Example L-1 *Effects of* **[no] ip subnet-zero** *on a Local Router*

```
R1# configure terminal
Enter configuration commands, one per line.   End with CNTL/Z.
R1(config)# no ip subnet-zero
R1(config)# interface g0/1
R1(config-if)# ip address 10.0.0.1 255.255.255.0
Bad mask /24 for address 10.0.0.1
```

Note that the **no ip subnet-zero** command affects the local router's **ip address** commands, as well as the local router's **ip route** commands (which define static routes). However, it does not affect the local router's routes as learned with a routing protocol.

Finding the Pattern Using the Magic Number

Subnet IDs follow a predictable pattern, at least when using our assumption of a single subnet mask for all subnets of a network. The pattern uses the *magic number*, as discussed in Chapter 14, "Analyzing Existing Subnets." To review, the magic number is 256, minus the mask's decimal value, in a particular octet that this book refers to as the *interesting octet*.

Figure L-7 shows four examples of these patterns with four different masks. For example, just look at the top of the figure to start. It lists mask 255.255.128.0 on the left. The third octet is the interesting octet, with a mask value other than 0 or 255 in that octet. The left side shows a magic number calculated as 256 – 128 = 128. So, the pattern of subnet IDs is shown in the highlighted number line; that is, the subnet IDs when using this mask will have either a 0 or 128 in the third octet. For example, if using network 172.16.0.0, the subnet IDs would be 172.16.0.0 and 172.16.128.0.

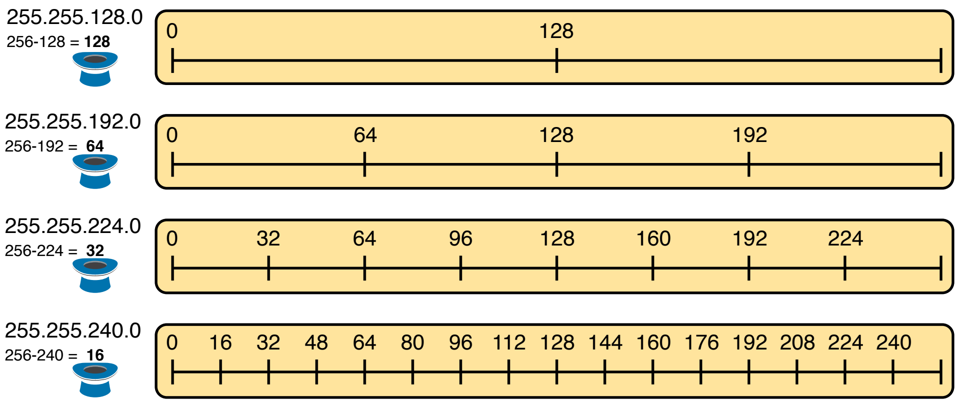

Figure L-7 *Patterns with Magic Numbers for Masks /17 – /20*

Now focus on the second row, with another example, with mask 255.255.192.0. This row shows a magic number of 64 (256 – 192 = 64), so the subnet IDs will use a value of 0, 64, 128, or 192 (multiples of 64) in the third octet. For example, if used with network 172.16.0.0, the subnet IDs would be 172.16.0.0, 172.16.64.0, 172.16.128.0, and 172.16.192.0.

Looking at the third row/example, the mask is 255.255.224.0, with a magic number of 256 – 224 = 32. So, as shown in the center of the figure, the subnet ID values will be multiples of 32. For example, if used with network 172.16.0.0 again, this mask would tell us that the subnet IDs are 172.16.0.0, 172.16.32.0, 172.16.64.0, 172.16.96.0, and so on.

Finally, for the bottom example, mask 255.255.240.0 makes the magic number, in the third octet, be 16. So, all the subnet IDs will be a multiple of 16 in the third octet, with those values shown in the middle of the figure.

A Formal Process with Less Than 8 Subnet Bits

Although it can be easy to see the patterns in Figure L-7, it might not be as obvious exactly how to apply those concepts to find all the subnet IDs in every case. This section outlines a specific process to find all the subnet IDs.

To simplify the explanations, this section assumes that less than 8 subnet bits exist. Later, the section “Finding All Subnets with More Than 8 Subnet Bits,” describes the full process that can be used in all cases.

First, to organize your thoughts, you might want to organize the data into a table like Table L-3. This book refers to this chart as the list-all-subnets chart.

Table L-3 Generic List-All-Subnets Chart

Octet	1	2	3	4
Mask				
Magic Number				
Network Number/Zero Subnet				
Next Subnet				
Next Subnet				
Next Subnet				
Broadcast Subnet				
Out of Range—Used by Process				

A formal process to find all subnet IDs, given a network and a single subnet mask, is as follows:

Key Topic

Step 1. Write down the subnet mask, in decimal, in the first empty row of the table.

Step 2. Identify the interesting octet, which is the one octet of the mask with a value other than 255 or 0. Draw a rectangle around the column of the interesting octet.

Step 3. Calculate and write down the magic number by subtracting the *subnet mask’s interesting octet* from 256.

Step 4. Write down the classful network number, which is the same number as the zero subnet, in the next empty row of the list-all-subnets chart.

Step 5. To find each successive subnet number:

A. For the three uninteresting octets, copy the previous subnet number’s values.

B. For the interesting octet, add the magic number to the previous subnet number’s interesting octet.

Step 6. When the sum calculated in Step 5B reaches 256, stop the process. The number with the 256 in it is out of range, and the previous subnet number is the broadcast subnet.

Although the written process is long, with practice, most people can find the answers much more quickly with this decimal-based process than by using binary math. As usual, most people learn this process best by seeing it in action, exercising it, and then practicing it. To that end, review the two following examples and watch any videos that came with this book that show additional examples.

Example 1: Network 172.16.0.0, Mask 255.255.240.0

To begin this example, focus on the first four of the six steps, when subnetting network 172.16.0.0 using mask 255.255.240.0. Figure L-8 shows the results of these first four steps:

Step 1. Record mask 255.255.240.0, which was given as part of the problem statement. (Figure L-8 also shows the network ID, 172.16.0.0, for easy reference.)

Step 2. The mask's third octet is neither 0 nor 255, which makes the third octet interesting.

Step 3. Because the mask's value in the third octet is 240, the magic number = 256 – 240 = 16.

Step 4. Because the network ID is 172.16.0.0, the first subnet ID, the zero subnet, is also 172.16.0.0.

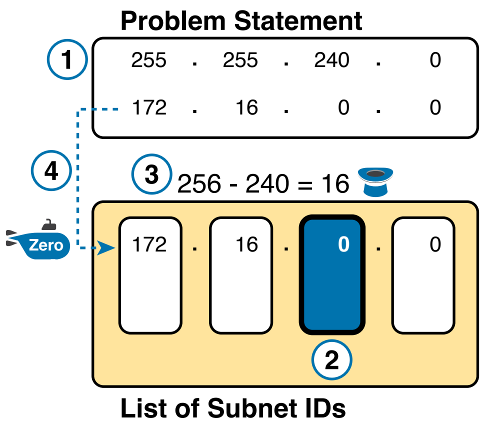

Figure L-8 *Results of First Four Steps: 172.16.0.0, 255.255.240.0.*

These first four steps discover the first subnet (the zero subnet) and get you ready to do the remaining steps by identifying the interesting octet and the magic number. Step 5 in the process tells you to copy the three boring octets and add the magic number (16, in this case) in the interesting octet (octet 3, in this case). Keep repeating this step until the interesting octet value equals 256 (per Step 6). When the total is 256, you have listed all the subnet IDs, and the line with 256 on it is not a correct subnet ID. Figure L-9 shows the results of the Step 5 actions.

Key Topic

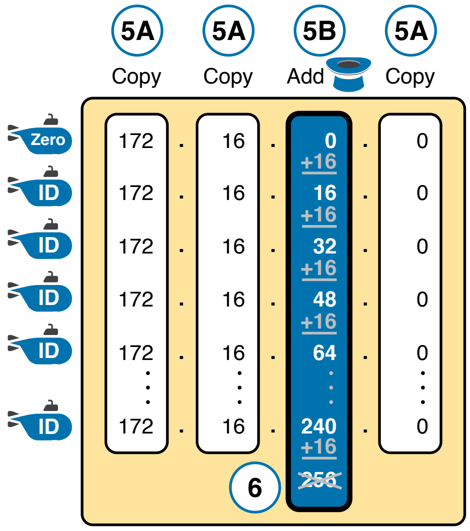

Figure L-9 *List of Subnet IDs: 172.16.0.0, 255.255.240.0*

NOTE In any list of all the subnet IDs of a network, the numerically highest subnet ID is called the *broadcast subnet*. Decades ago, engineers avoided using the broadcast subnet. However, using the broadcast subnet causes no problems. The term *broadcast subnet* has its origins in the fact that if you determine the subnet broadcast address inside the broadcast subnet, it has the same numeric value as the network-wide broadcast address.

NOTE People sometimes confuse the terms *broadcast subnet* and *subnet broadcast address*. The *broadcast subnet* is one subnet, namely the numerically highest subnet; only one such subnet exists per network. The term *subnet broadcast address* refers to the one number in each and every subnet that is the numerically highest number in that subnet.

Example 2: Network 192.168.1.0, Mask 255.255.255.224

With a Class C network and a mask of 255.255.255.224, this example makes the fourth octet the interesting octet. However, the process works the same, with the same logic, just with the interesting logic applied in a different octet. As with the previous example, the following list outlines the first four steps, with Figure L-10 showing the results of the first four steps:

Step 1. Record mask 255.255.255.224, which was given as part of the problem statement, and optionally record the network number (192.168.1.0).

Step 2. The mask's fourth octet is neither 0 nor 255, which makes the fourth octet interesting.

Step 3. Because the mask's value in the fourth octet is 224, the magic number = 256 – 224 = 32.

Step 4. Because the network ID is 192.168.1.0, the first subnet ID, the zero subnet, is also 192.168.1.0.

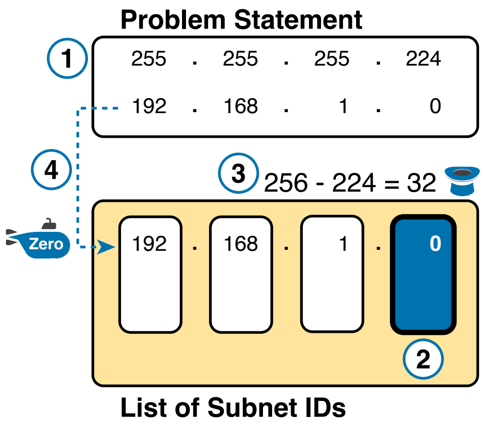

Figure L-10 *Results of First Four Steps: 192.168.1.0, 255.255.255.224*

From this point, Step 5 in the process tells you to copy the values in the first three octets and then add the magic number (32, in this case) in the interesting octet (octet 4, in this case). Keep doing so until the interesting octet value equals 256 (per Step 6). When the total is 256, you have listed all the subnet IDs, and the line with 256 on it is not a correct subnet ID. Figure L-11 shows the results of these steps.

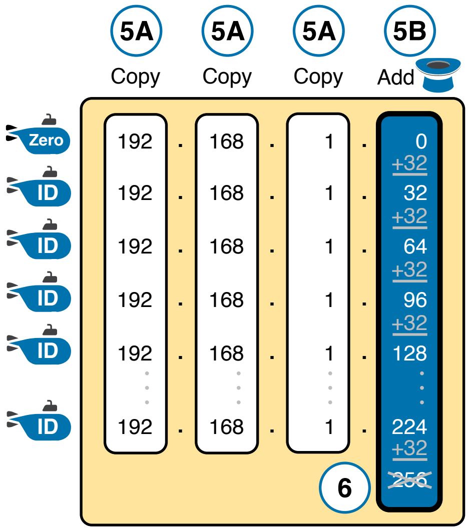

Figure L-11 *List of Subnet IDs: 192.168.1.0, 255.255.255.224*

Finding All Subnets with Exactly 8 Subnet Bits

The formal process in the earlier section "A Formal Process with Less Than 8 Subnet Bits" identified the interesting octet as the octet whose mask value is neither a 255 nor a 0. If the mask defines exactly 8 subnet bits, you must use a different logic to identify the interesting octet; otherwise, the same process can be used. In fact, the actual subnet IDs can be a little more intuitive.

Only two cases exist with exactly 8 subnet bits:

A Class A network with mask 255.255.0.0; the entire second octet contains subnet bits.

A Class B network with mask 255.255.255.0; the entire third octet contains subnet bits.

In each case, use the same process as with less than 8 subnet bits, but identify the interesting octet as the one octet that contains subnet bits. Also, because the mask's value is 255, the magic number will be 256 – 255 = 1, so the subnet IDs are each 1 larger than the previous subnet ID.

For example, for 172.16.0.0, mask 255.255.255.0, the third octet is the interesting octet and the magic number is 256 – 255 = 1. You start with the zero subnet, equal in value to network number 172.16.0.0, and then add 1 in the third octet. For example, the first four subnets are as follows:

172.16.0.0 (zero subnet)

172.16.1.0

172.16.2.0

172.16.3.0

Finding All Subnets with More Than 8 Subnet Bits

Earlier, the section "A Formal Process with Less Than 8 Subnet Bits" assumed less than 8 subnet bits for the purpose of simplifying the discussions while you learn. In real life, you need to be able to find all subnet IDs with any valid mask, so you cannot assume less than 8 subnet bits.

The examples that have at least 9 subnet bits have a minimum of 512 subnet IDs, so writing down such a list would take a lot of time. To conserve space, the examples will use shorthand rather than list hundreds or thousands of subnet IDs.

The process with less than 8 subnet bits told you to count in increments of the magic number in one octet. With more than 8 subnet bits, the new expanded process must tell you how to count in multiple octets. So, this section breaks down two general cases: (a) when 9–16 subnet bits exist, which means that the subnet field exists in only two octets, and (b) cases with 17 or more subnet bits, which means that the subnet field exists in three octets.

Process with 9–16 Subnet Bits

To understand the process, you need to know a few terms that the process will use. Figure L-12 shows the details, with an example that uses Class B network 130.4.0.0 and mask 255.255.255.192. The lower part of the figure details the structure of the addresses per the mask: a network part of two octets because it is a Class B address, a 10-bit subnet part per the mask (/26), and 6 host bits.

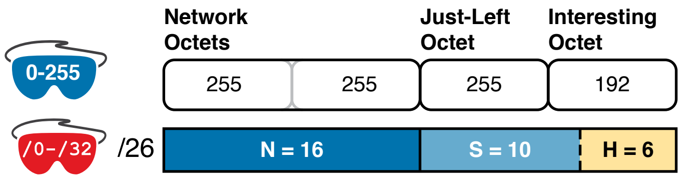

Figure L-12 *Fundamental Concepts and Terms for the >8 Subnet Bit Process*

In this case, subnet bits exist in two octets: octets 3 and 4. For the purposes of the process, the rightmost of these octets is the interesting octet, and the octet just to the left is the cleverly named *just-left* octet.

The updated process, which makes adjustments for cases in which the subnet field is longer than 1 octet, tells you to count in increments of the magic number in the interesting octet, but count by 1s in the just-left octet. Formally:

Key Topic

Step 1. Calculate subnet IDs using the 8-subnet-bits-or-less process. However, when the total adds up to 256, move to the next step; consider the subnet IDs listed so far as a *subnet block*.

Step 2. Copy the previous subnet block, but add 1 to the just-left octet in all subnet IDs in the new block.

Step 3. Repeat Step 2 until you create the block with a just-left octet of 255, but go no further.

To be honest, the formal concept can cause you problems until you work through some examples, so even if the process remains a bit unclear in your mind, you should work through the following examples instead of rereading the formal process.

First, consider an example based on Figure L-12, with network 130.4.0.0 and mask 255.255.255.192. Figure L-12 already showed the structure, and Figure L-13 shows the subnet ID block created at Step 1.

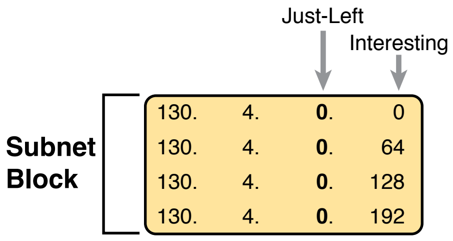

Figure L-13 *Step 1: Listing the First Subnet ID Block*

The logic at Step 1, to create this subnet ID block of four subnet IDs, follows the same magic number process seen before. The first subnet ID, 130.4.0.0, is the zero subnet. The next three subnet IDs are each 64 bigger, because the magic number, in this case, is 256 – 192 = 64.

Steps 2 and 3 from the formal process tell you how to create 256 subnet blocks, and by doing so, you will list all 1024 subnet IDs. To do so, create 256 total subnet blocks: one with a 0 in the just-left octet, one with a 1 in the just-left octet, and another with a 2 in the just-left octet, up through 255. The process continues through the step at which you create the subnet block with 255 in the just-left octet (third octet, in this case). Figure L-14 shows the idea, with the addition of the first few subnet blocks.

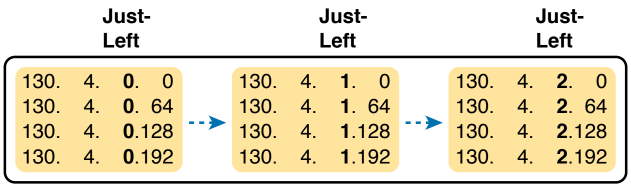

Figure L-14 *Step 2: Replicating the Subnet Block with +1 in the Just-Left Octet*

This example, with 10 total subnet bits, creates 256 blocks of four subnets each, for a total of 1024 subnets. This math matches the usual method of counting subnets, because 2^{10} = 1024.

Process with 17 or More Subnet Bits

To create a subnet design that allows 17 or more subnet bits to exist, the design must use a Class A network. In addition, the subnet part will consist of the entire second and third octets, plus part of the fourth octet. That means a lot of subnet IDs: at least 2^{17} (or 131,072) subnets. Figure L-15 shows an example of just such a structure, with a Class A network and a /26 mask.

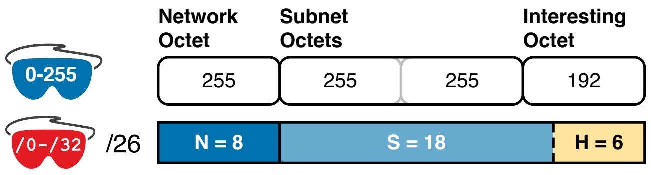

Figure L-15 *Address Structure with 18 Subnet Bits*

To find all the subnet IDs in this example, you use the same general process as with 9–16 subnet bits, but with many more subnet blocks to create. In effect, you have to create a subnet block for all combinations of values (0–255, inclusive) in both the second and third octet. Figure L-16 shows the general idea. Note that with only 2 subnet bits in the fourth octet in this example, the subnet blocks will have four subnets each.

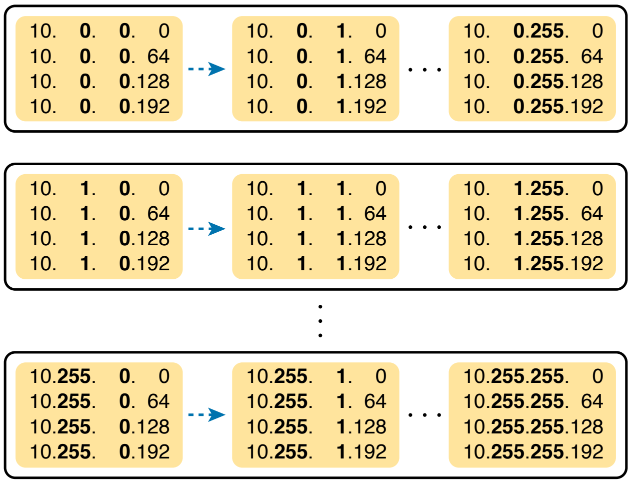

Figure L-16 *256 Times 256 Subnet Blocks of Four Subnets*

Practice Finding All Subnet IDs

Before moving to the next chapter, practice until you get the right answer most of the time—but use any tools you want and take all the time you need. Then, you can move on with your reading. *Before taking the exam*, practice until you reach the goals in the right column of Table L-4, which summarizes the key concepts and suggestions for this two-phase approach.

Table L-4 Keep-Reading and Take-Exam Goals for This Chapter's Topics

Time Frame	Before Moving to the Next Chapter	Before Taking the Exam
Focus On	Learning how	Being correct and fast
Tools Allowed	All	Your brain and a notepad
Goal: Accuracy	90% correct	100% correct
Goal: Speed	Any speed	45 seconds

Practice Problems for Finding All Subnet IDs

The following list shows three separate problems, each with a classful network number and prefix-style mask. Find all subnet IDs for each problem:

1. 192.168.9.0/27
2. 172.30.0.0/20
3. 10.0.0.0/17

The section "Answers to Earlier Practice Problems," later in this chapter, lists the answers.

Answers to Earlier Practice Problems

Answers to Practice Choosing Subnet Masks

The earlier section "Practice Choosing Subnet Masks" listed three practice problems. The answers are listed here so that the answers are nearby but not visible from the list of problems. Table L-5 lists the answers, with notes related to each problem following the table.

Table L-5 Practice Problems: Find the Masks That Meet Requirements

Problem	Class	Minimum Subnet Bits	Minimum Host Bits	Prefix Range	Prefix to Maximize Subnets	Prefix to Maximize Hosts
1	A	11	9	/19 – /23	/23	/19
2	B	8	8	/24	—	—
3	C	3	4	/27 – /28	/28	/27

1. N=8, because the problem lists Class A network 10.0.0.0. With a need for 1500 subnets, 10 subnet bits supply only 1024 subnets (per Table L-1), but 11 subnet bits (S) would provide 2048 subnets—more than the required 1500. Similarly, the smallest number of host bits would be 9, because $2^8 - 2 = 254$, and the design requires 300

hosts/subnet. The shortest prefix mask would then be /19, found by adding N (8) and the smallest usable number of subnet bits S (11). Similarly, with a minimum H value of 9, the longest prefix mask, maximizing the number of subnets, is 32 – H = /23.

2. N=16, because the problem lists Class B network 172.25.0.0. With a need for 130 subnets, 7 subnet bits supply only 128 subnets (per Table L-1), but 8 subnet bits (S) would provide 256 subnets—more than the required 130. Similarly, the smallest number of host bits would be 8, because $2^7 - 2 = 126$—close to the required 127, but not quite enough, making H = 8 the smallest number of host bits that meets requirements. Note that the network, minimum subnet bits, and minimum host bits add up to 32, so only one mask meets the requirements, namely /24, found by adding the number of network bits (16) to the minimum number of subnet bits (8).
3. N=24, because the problem lists Class C network 192.168.83.0. With a need for eight subnets, 3 subnet bits supply enough, but just barely. The smallest number of host bits would be 4, because $2^3 - 2 = 6$, and the design requires 8 hosts/subnet. The shortest prefix mask would then be /27, found by adding N (24) and the smallest usable number of subnet bits S (3). Similarly, with a minimum H value of 4, the longest prefix mask, maximizing the number of subnets, is 32 – H = /28.

Answers to Practice Finding All Subnet IDs

The earlier section "Practice Finding All Subnet IDs" listed three practice problems. The answers are listed here so that they are not visible from the same page as the list of problems.

Answer, Practice Problem 1

Problem 1 lists network 192.168.9.0, mask /27. The mask converts to DDN mask 255.255.255.224. When used with a Class C network, which has 24 network bits, only 3 subnet bits exist, and they all sit in the fourth octet. So, this problem is a case of less than 8 subnet bits, with the fourth octet as the interesting octet.

To get started listing subnets, first write down the zero subnet and then start adding the magic number in the interesting octet. The zero subnet equals the network ID (192.168.9.0, in this case). The magic number, calculated as 256 – 224 = 32, should be added to the previous subnet ID's interesting octet. Table L-6 lists the results.

Table L-6 List-All-Subnets Chart: 192.168.9.0/27

Octet	1	2	3	4
Mask	255	255	255	224
Magic Number	—	—	—	32
Classful Network/Subnet Zero	192	168	9	0
First Nonzero Subnet	192	168	9	32
Next Subnet	192	168	9	64
Next Subnet	192	168	9	96
Next Subnet	192	168	9	128
Next Subnet	192	168	9	160

Octet	1	2	3	4
Next Subnet	192	168	9	192
Broadcast Subnet	192	168	9	224
Invalid—Used by Process	192	168	9	256

Answer, Practice Problem 2

Problem 2 lists network 172.30.0.0, mask /20. The mask converts to DDN mask 255.255.240.0. When used with a Class B network, which has 16 network bits, only 4 subnet bits exist, and they all sit in the third octet. So, this problem is a case of less than 8 subnet bits, with the third octet as the interesting octet.

To get started listing subnets, first write down the zero subnet and then start adding the magic number in the interesting octet. The zero subnet equals the network ID (or 172.30.0.0, in this case). The magic number, calculated as 256 – 240 = 16, should be added to the previous subnet ID's interesting octet. Table L-7 lists the results.

Table L-7 List-All-Subnets Chart: 172.30.0.0/20

Octet	1	2	3	4
Mask	255	255	240	0
Magic Number	—	—	16	—
Classful Network/Subnet Zero	172	30	0	0
First Nonzero Subnet	172	30	16	0
Next Subnet	172	30	32	0
Next Subnet	172	30	Skipping…	0
Next Subnet	172	30	224	0
Broadcast Subnet	172	30	240	0
Invalid—Used by Process	172	30	256	0

Answer, Practice Problem 3

Problem 3 lists network 10.0.0.0, mask /17. The mask converts to DDN mask 255.255.128.0. When used with a Class A network, which has 8 network bits, 9 subnet bits exist. Using the terms unique to this chapter, octet 3 is the interesting octet, with only 1 subnet bit in that octet, and octet 2 is the just-left octet, with 8 subnet bits.

In this case, begin by finding the first subnet block. The magic number is 256 – 128 = 128. The first subnet (zero subnet) equals the network ID. So, the first subnet ID block includes the following:

10.0.0.0

10.0.128.0

Then, you create a subnet block for all 256 possible values in the just-left octet, or octet 2 in this case. The following list shows the first three subnet ID blocks, plus the last subnet ID block, rather than listing page upon page of subnet IDs:

10.0.0.0 (zero subnet)

10.0.128.0

10.1.0.0

10.1.128.0

10.2.0.0

10.2.128.0

...

10.255.0.0

10.255.128.0 (broadcast subnet)

This page intentionally left blank

Practice for Appendix L: Subnet Design

NOTE This appendix contains an entire chapter that was published as a chapter in one of the past editions of this book or a related book. The author includes this appendix with the current edition as extra reading for anyone interested in learning more. However, note that the content in this appendix has not been edited since it was published in the earlier edition, so references to exams and exam topics, and to other chapters, will be outdated. This appendix was previously published as Appendix G of the book *CCENT/CCNA ICND1 100-105 Official Cert Guide*, published in 2016.

This appendix exists as two halves to match the two major sections of the chapter. The first half lists mask design problems, and then the answers to those problems. The second half lists problems where you need to find the subnet ID, but with less than 8 subnet bits and with more than 8 subnet bits.

To solve these problems, use the processes explained in Appendix L of *CCNA 200-301 Official Cert Guide, Volume 1*, the current edition of this book you are reading.

Mask Design Practice Problems

This section lists problems with a short set of requirements regarding how a particular classful network should be subnetted. The requirements include the classful network, the number of subnets the design must support, and the number of hosts in each subnet. For each problem, supply the following information:

- The minimum number of subnet and host bits needed in the mask to support the design requirements
- The dotted-decimal format mask(s) that meet the requirements
- The mask you would choose if the problem said to maximize the number of subnets
- The mask you would choose if the problem said to maximize the number of hosts per subnet

Also note that you should assume that the two special subnets in each network—the zero subnet and broadcast subnet—are allowed to be used for these questions.

When doing the problems, the information in Table M-1 can be helpful. Note that Appendix A, “Numeric Reference Tables,” in the printed book, also includes this table.

Table M-1 Powers of 2

Number of Bits	2^X	Number of Bits	2^X	Number of Bits	2^X	Number of Bits	2^X
1	2	5	32	9	512	13	8192
2	4	6	64	10	1024	14	16,384
3	8	7	128	11	2048	15	32,768
4	16	8	256	12	4096	16	65,536

Find the key facts for these sets of requirements:

1. Network 10.0.0.0, need 50 subnets, need 200 hosts/subnet
2. Network 172.32.0.0, need 125 subnets, need 125 hosts/subnet
3. Network 192.168.44.0, need 15 subnets, need 6 hosts/subnet
4. Network 10.0.0.0, need 300 subnets, need 500 hosts/subnet
5. Network 172.32.0.0, need 500 subnets, need 15 hosts/subnet
6. Network 172.16.0.0, need 2000 subnets, need 2 hosts/subnet

Mask Design Answers

This section includes the answers to the six problems listed in this appendix. The answer section for each problem explains how to use the process outlined in Appendix L, "Subnet Design," to find the answers.

Answer to Mask Design Problem 1

Problem 1 shows a Class A network, with 8 network bits, with a minimum of 6 subnet bits and 8 host bits to meet the required number of subnets and hosts/subnet. The following masks all meet the requirements in this problem, with the masks that maximize the number of hosts/subnet and the number of subnets noted:

- 255.252.0.0 (maximizes the number of hosts per subnet)
- 255.254.0.0
- 255.255.0.0
- 255.255.128.0
- 255.255.192.0
- 255.255.224.0
- 255.255.240.0
- 255.255.248.0
- 255.255.252.0
- 255.255.254.0
- 255.255.255.0 (maximizes the number of subnets)

As for the process to find the answers, the following list explains the details:

NOTE The following explanation uses step numbers that match the process listed in Appendix L, but only the steps from that process that apply to this problem. As a result, the step numbers in the explanation are not sequential.

Step 1. The question lists Class A network 10.0.0.0, so there are 8 network bits.

Step 2. The question states that 50 subnets are needed. A mask with 5 subnet bits supplies only 2^5 (32) subnets, but a mask with 6 subnet bits supplies 2^6 (64) subnets. So, the mask needs at least 6 subnet bits.

Step 3. The question states that 200 hosts are needed per subnet. A mask with 7 host bits supplies only $2^7 - 2$ (126) hosts per subnet, but a mask with 8 host bits supplies $2^8 - 2$ (254) hosts per subnet. So, the mask needs at least 8 host bits.

Step 6A. With N=8, a minimum S=6, and a minimum H=8, multiple masks exist. The first mask, with the minimum number of subnet bits, is /14, found by adding N (8) to the minimum value of S (6). This mask maximizes the number of host bits and therefore maximizes the number of hosts/subnet.

Step 6B. The minimum value of H, the number of host bits, is 8. So, the mask with the fewest H bits, maximizing the number of subnets, is 32 – H = 32 – 8 = /24.

Step 6C. All masks between /14 and /24 also meet the requirements.

Answer to Mask Design Problem 2

Problem 2 shows a Class B network, with 16 network bits, with a minimum of 7 subnet bits and 7 host bits to meet the required number of subnets and hosts/subnet. The following masks all meet the requirements in this problem, with the masks that maximize the number of hosts/subnet and the number of subnets noted:

- 255.255.254.0 (maximizes the number of hosts/subnet)
- 255.255.255.0
- 255.255.255.128 (maximizes the number of subnets)

As for the process to find the answers, the following list explains the details:

Step 1. The question lists Class B network 172.32.0.0, so there are 16 network bits.

Step 2. The question states that 125 subnets are needed. A mask with 6 subnet bits supplies only 2^6 (64) subnets, but a mask with 7 subnet bits supplies 2^7 (128) subnets. So, the mask needs at least 7 subnet bits.

Step 3. The question states that 125 hosts are needed per subnet. A mask with 6 host bits supplies only $2^6 - 2$ (62) hosts per subnet, but a mask with 7 host bits supplies $2^7 - 2$ (126) hosts per subnet. So, the mask needs at least 7 host bits.

Step 6A. With N=16, a minimum S=7, and a minimum H=7, multiple masks exist. The first mask, with the minimum number of subnet bits, is /23, found by adding N (16) to the minimum value of S (7). This mask maximizes the number of host bits and therefore maximizes the number of hosts/subnet.

Step 6B. The minimum value of H, the number of host bits, is 7. So, the mask with the fewest H bits, maximizing the number of subnets, is 32 – H = 32 – 7 = /25.

Step 6C. All masks between /23 and /25 also meet the requirements (/23, /24, and /25).

Answer to Mask Design Problem 3

Problem 3 shows a Class C network, with 24 network bits, with a minimum of 4 subnet bits and 3 host bits to meet the required number of subnets and hosts/subnet. The following masks all meet the requirements in this problem, with the masks that maximize the number of hosts/subnet and the number of subnets noted:

- 255.255.255.240 (maximizes the number of hosts/subnet)
- 255.255.255.248 (maximizes the number of subnets)

As for the process to find the answers, the following list explains the details:

Step 1. The question lists Class C network 192.168.44.0, so there are 24 network bits.

Step 2. The question states that 15 subnets are needed. A mask with 3 subnet bits supplies only 2^3 (8) subnets, but a mask with 4 subnet bits supplies 2^4 (16) subnets. So, the mask needs at least 4 subnet bits.

Step 3. The question states that 6 hosts are needed per subnet. A mask with 2 host bits supplies only $2^2 - 2$ (2) hosts per subnet, but a mask with 3 host bits supplies $2^3 - 2$ (6) hosts per subnet. So, the mask needs at least 3 host bits.

Step 6A. With N=24, a minimum S=4, and a minimum H=3, multiple masks exist. The first mask, with the minimum number of subnet bits, is /28, found by adding N (24) to the minimum value of S (4). This mask maximizes the number of host bits and therefore maximizes the number of hosts/subnet.

Step 6B. The minimum value of H, the number of host bits, is 3. So, the mask with the fewest H bits, maximizing the number of subnets, is 32 – H = 32 – 3 = /29.

Step 6C. Only masks /28 and /29 meet the requirements.

Answer to Mask Design Problem 4

Problem 4 shows a Class A network, with 8 network bits, with a minimum of 9 subnet bits and 9 host bits to meet the required number of subnets and hosts/subnet. The following masks all meet the requirements in this problem, with the masks that maximize the number of hosts/subnet and the number of subnets noted:

- 255.255.128.0 (maximizes the number of hosts/subnet)
- 255.255.192.0
- 255.255.224.0
- 255.255.240.0
- 255.255.248.0
- 255.255.252.0
- 255.255.254.0 (maximizes the number of subnets)

As for the process to find the answers, the following list explains the details:

Step 1. The question lists Class A network 10.0.0.0, so there are 8 network bits.

Step 2. The question states that 300 subnets are needed. A mask with 8 subnet bits supplies only 2^8 (256) subnets, but a mask with 9 subnet bits supplies 2^9 (512) subnets. So, the mask needs at least 9 subnet bits.

Step 3. The question states that 500 hosts are needed per subnet. A mask with 8 host bits supplies only $2^8 - 2$ (254) hosts per subnet, but a mask with 9 host bits supplies $2^9 - 2$ (510) hosts per subnet. So, the mask needs at least 9 host bits.

Step 6A. With N=8, a minimum S=9, and a minimum H=9, multiple masks exist. The first mask, with the minimum number of subnet bits, is /17, found by adding N (8) to the minimum value of S (9). This mask maximizes the number of host bits and therefore maximizes the number of hosts/subnet.

Step 6B. The minimum value of H, the number of host bits, is 9. So, the mask with the fewest H bits, maximizing the number of subnets, is 32 – H = 32 – 9 = /23.

Step 6C. All masks between /17 and /23 also meet the requirements (/17, /18, /19, /20, /21, /22, /23).

Answer to Mask Design Problem 5

Problem 5 shows a Class B network, with 16 network bits, with a minimum of 9 subnet bits and 5 host bits to meet the required number of subnets and hosts/subnet. The following masks all meet the requirements in this problem, with the masks that maximize the number of hosts/subnet and the number of subnets noted:

- 255.255.255.128 (maximizes the number of hosts/subnet)
- 255.255.255.192
- 255.255.255.224 (maximizes the number of subnets)

As for the process to find the answers, the following list explains the details:

Step 1. The question lists Class B network 172.32.0.0, so there are 16 network bits.

Step 2. The question states that 500 subnets are needed. A mask with 8 subnet bits supplies only 2^8 (256) subnets, but a mask with 9 subnet bits supplies 2^9 (512) subnets. So, the mask needs at least 9 subnet bits.

Step 3. The question states that 15 hosts are needed per subnet. A mask with 4 host bits supplies only $2^4 - 2$ (14) hosts per subnet, but a mask with 5 host bits supplies $2^5 - 2$ (30) hosts per subnet. So, the mask needs at least 5 host bits.

Step 6A. With N=16, a minimum S=9, and a minimum H=5, multiple masks exist. The first mask, with the minimum number of subnet bits, is /25, found by adding N (16) to the minimum value of S (9). This mask maximizes the number of host bits and therefore maximizes the number of hosts/subnet.

Step 6B. The minimum value of H, the number of host bits, is 5. So, the mask with the fewest H bits, maximizing the number of subnets, is 32 – H = 32 – 5 = /27.

Step 6C. All masks between /25 and /27 also meet the requirements (/25, /26, /27).

Answer to Mask Design Problem 6

Problem 6 shows a Class B network, with 16 network bits, with a minimum of 11 subnet bits and 2 host bits to meet the required number of subnets and hosts/subnet. The following masks all meet the requirements in this problem, with the masks that maximize the number of hosts/subnet and the number of subnets noted:

- 255.255.255.224 (maximizes the number of hosts/subnet)
- 255.255.255.240
- 255.255.255.248
- 255.255.255.252 (maximizes the number of subnets)

As for the process to find the answers, the following list explains the details:

Step 1. The question lists Class B network 172.16.0.0, so there are 16 network bits.

Step 2. The question states that 2000 subnets are needed. A mask with 10 subnet bits supplies only 2^{10} (1024) subnets, but a mask with 11 subnet bits supplies 2^{11} (2048) subnets. So, the mask needs at least 11 subnet bits.

Step 3. The question states that 2 hosts are needed per subnet. A mask with 2 host bits supplies $2^2 - 2$ (2) hosts per subnet. So, the mask needs at least 2 host bits.

Step 6A. With N=16, a minimum S=11, and a minimum H=2, multiple masks exist. The first mask, with the minimum number of subnet bits, is /27, found by adding N (16) to the minimum value of S (11). This mask maximizes the number of host bits and therefore maximizes the number of hosts/subnet.

Step 6B. The minimum value of H, the number of host bits, is 2. So, the mask with the fewest H bits, maximizing the number of subnets, is 32 – H = 32 – 2 = /30.

Step 6C. All masks between /27 and /30 also meet the requirements (/27, /28, /29, /30).

Practice Finding All Subnet IDs

The remainder of this appendix lists two sets of problems. Both problem sets list an IP network and mask; your job is to list all the subnet IDs for each network/mask combination. The first problem set includes problems that happen to have 8 or fewer subnet bits, and the second problem set includes problems that happen to have more than 8 subnet bits. In particular, for each problem, find the following:

- All subnet numbers
- The subnet that is the zero subnet
- The subnet that is the broadcast subnet

To find this information, you can use the processes explained in Appendix L.

Find Subnet IDs, Problem Set 1: 8 or Fewer Subnet Bits

The problems, which consist of a classful network and static-length mask, are as follows:

1. 172.32.0.0/22
2. 200.1.2.0/28

3. 10.0.0.0/15
4. 172.20.0.0/24

Find Subnet IDs, Problem Set 2: More Than 8 Subnet Bits

The problems, which consist of a classful network and static-length mask, are as follows:

1. 172.32.0.0/25
2. 10.0.0.0/21

Answers to Find Subnet IDs, Problem Set 1

This section includes the answers to the four problems listed in Problem Set 1.

Problem Set 1, Answer 1: 172.32.0.0/22

The answer is as follows:

- 172.32.0.0 (zero subnet)
- 172.32.4.0
- 172.32.8.0
- 172.32.12.0
- 172.32.16.0
- 172.32.20.0
- 172.32.24.0

  (Skipping many subnets; each new subnet is the same as the previous subnet, after adding 4 to the third octet.)
- 172.32.248.0
- 172.32.252.0 (broadcast subnet)

The process to find all subnets depends on three key pieces of information:

- The mask has fewer than 8 subnet bits (6 bits), because the network is a Class B network (16 network bits), and the mask has 22 binary 1s in it—implying 10 host bits and leaving 6 subnet bits.
- The mask in dotted-decimal format is 255.255.252.0. The interesting octet is the third octet because the subnet bits are all in the third octet.
- Each successive subnet number is 4 higher than the previous subnet number, in the interesting octet, because the magic number is 256 – 252 = 4.

As a result, in this case, all the subnets begin with 172.32, have a multiple of 4 in the third octet, and end in 0.

Table M-2 shows the results of the various steps of the process, as outlined in Appendix L.

Table M-2 8 or Fewer Subnet Bits, Question 1: Answer Table

	Octet 1	Octet 2	Octet 3	Octet 4
Subnet Mask (Step 1)	255	255	252	0
Magic Number (Step 3)			256 – 252 = 4	
Zero Subnet Number (Step 4)	172	32	0	0
Next Subnet (Step 5)	172	32	4	0
Next Subnet (Step 5)	172	32	8	0
Next Subnet (Step 5)	172	32	12	0
Next Subnet (Step 5)	172	32	16	0
(You might need many more such rows.)	172	32	X	0
Next Subnet	172	32	244	0
Next Subnet (Step 5)	172	32	248	0
Broadcast Subnet (Step 6)	172	32	252	0
Out of Range—Stop Process (Step 6)			256	

Problem Set 1, Answer 2: 200.1.2.0/28

The answer is as follows:

- 200.1.2.0 (zero subnet)
- 200.1.2.16
- 200.1.2.32
- 200.1.2.48
- 200.1.2.64
- 200.1.2.80

  (Skipping many subnets; each new subnet is the same as the previous subnet, after adding 16 to the fourth octet.)
- 200.1.2.224
- 200.1.2.240 (broadcast subnet)

The process to find all subnets depends on three key pieces of information, as follows:

- The mask has fewer than 8 subnet bits (4 bits), because the network is a Class C network (24 network bits), and the mask has 28 binary 1s in it, which implies 4 host bits and leaves 4 subnet bits.
- The mask in dotted-decimal format is 255.255.255.240. The interesting octet is the fourth octet, because all the subnet bits are in the fourth octet.
- Each successive subnet number is 16 higher than the previous subnet number, in the interesting octet, because the magic number is 256 – 240 = 16.

As a result, in this case, all the subnets begin with 200.1.2 and have a multiple of 16 in the fourth octet.

Table M-3 shows the results of the various steps of the process, as outlined in Appendix L.

Table M-3 Problem Set 1, Question 2: Answer Table

	Octet 1	Octet 2	Octet 3	Octet 4
Subnet Mask (Step 1)	255	255	255	240
Magic Number (Step 3)				256 – 240 = 16
Zero Subnet Number (Step 4)	200	1	2	0
Next Subnet (Step 5)	200	1	2	16
Next Subnet (Step 5)	200	1	2	32
Next Subnet (Step 5)	200	1	2	48
(You might need many more such rows.) (Step 5)	200	1	2	X
Next Subnet (Step 5)	200	1	2	224
Broadcast Subnet (Step 6)	200	1	2	240
Out of Range—Stop Process (Step 6)				256

Problem Set 1, Answer 3: 10.0.0.0/15

The answer is as follows:

- 10.0.0.0 (zero subnet)
- 10.2.0.0
- 10.4.0.0
- 10.6.0.0

  (Skipping many subnets; each new subnet is the same as the previous subnet, after adding 2 to the second octet.)
- 10.252.0.0
- 10.254.0.0 (broadcast subnet)

The process to find all subnets depends on three key pieces of information:

- The mask has fewer than 8 subnet bits (7 subnet bits), because the network is a Class A network (8 network bits), and the mask has 15 binary 1s in it, which implies 17 host bits and leaves 7 subnet bits.
- The mask in dotted-decimal format is 255.254.0.0. The interesting octet is the second octet, because all the subnet bits exist in the second octet.
- Each successive subnet number is 2 higher than the previous subnet number, in the interesting octet, because the magic number is 256 – 254 = 2.

As a result, in this case, all the subnets begin with 10, have a multiple of 2 in the second octet, and end in 0.0.

Table M-4 shows the results of the various steps of the process, as outlined in Appendix L.

Table M-4 Problem Set 1, Question 3: Answer Table

	Octet 1	Octet 2	Octet 3	Octet 4
Subnet Mask (Step 1)	255	254	0	0
Magic Number (Step 3)		256 – 254 = 2		
Zero Subnet Number (Step 4)	10	0	0	0
Next Subnet (Step 5)	10	2	0	0
Next Subnet (Step 5)	10	4	0	0
Next Subnet (Step 5)	10	6	0	0
(You might need many more such rows.) (Step 5)	10	X	0	0
Next Subnet (Step 5)	10	252	0	0
Broadcast Subnet (Step 6)	10	254	0	0
Out of Range—Stop Process (Step 6)		256		

Problem Set 1, Answer 4: 172.20.0.0/24

This problem has an 8-bit subnet field, meaning that 2^8, or 256, possible subnets exist. The following list shows some of the subnets, which should be enough to see the trends in how to find all subnet numbers:

- 172.20.0.0 (zero subnet)
- 172.20.1.0
- 172.20.2.0
- 172.20.3.0
- 172.20.4.0

  (Skipping many subnets; each new subnet is the same as the previous subnet, after adding 1 to the third octet.)
- 172.20.252.0
- 172.20.253.0
- 172.20.254.0
- 172.20.255.0 (broadcast subnet)

The process to find all subnets depends on three key pieces of information:

- The mask has exactly 8 subnet bits, specifically all bits in the third octet, making the third octet the interesting octet.
- The magic number is 256 – 255 = 1, because the mask's value in the interesting (third) octet is 255.
- Beginning with the network number of 172.20.0.0, which is the same value as the zero subnet, just add the magic number (1) in the interesting octet.

Essentially, you just count by 1 in the third octet until you reach the highest legal number (255). The first subnet, 172.20.0.0, is the zero subnet, and the last subnet, 172.20.255.0, is the broadcast subnet.

Answers to Find Subnet IDs, Problem Set 2

Problem Set 2, Answer 1: 172.32.0.0/25

This problem has a 9-bit subnet field, meaning that 2^9, or 512, possible subnets exist. The following list shows some of the subnets, which should be enough to see the trends in how to find all subnet numbers:

- 172.32.0.0 (zero subnet)
- 172.32.0.128
- 172.32.1.0
- 172.32.1.128
- 172.32.2.0
- 172.32.2.128
- 172.32.3.0
- 172.32.3.128

  (Skipping many subnets; the subnets occur in blocks of two, with either 0 or 128 in the fourth octet, with each successive block being one greater in the third octet.)
- 172.32.254.0
- 172.32.254.128
- 172.32.255.0
- 172.32.255.128 (broadcast subnet)

The process to find all subnets depends on three key pieces of information, as follows:

- The mask has more than 8 subnet bits (9 bits), because the network is a Class B network (16 network bits), and the mask has 25 binary 1s in it, which implies 7 host bits and leaves 9 subnet bits.
- Using the terminology in Appendix L, octet 4 is the *interesting* octet, where the counting occurs based on the magic number. Octet 3 is the "just left" octet, in which the process counts by 1, from 0 to 255.
- The magic number, which will be used to calculate each successive subnet number, is 256 – 128 = 128.

To calculate the first subnet block, use the same six-step process as used in the simpler problems that have 8 or fewer subnet bits. In this case, with only 1 subnet bit in octet 4, only two subnets exist in each subnet block. Table M-5 shows the steps as compared to the six-step process to find the subnets in a subnet block.

Table M-5 Creating the First Subnet Block

	Octet 1	Octet 2	Octet 3	Octet 4
Subnet Mask (Step 1)	255	255	255	128
Magic Number (Step 3)				256 – 128 = 128
Zero Subnet Number (Step 4)	172	32	0	0
Next Subnet (Step 5)	172	32	0	128
Step 6 Needs to Be Used Here (Sum of 256 in the 4th Octet)	172	32	0	256

The table represents the logic, but to make sure that the answer is clear, the first subnet block includes the following:

172.32.0.0

172.32.0.128

The next major task—to create subnet blocks for all possible values in the "just left" octet—completes the process. Essentially, create 256 blocks like the previous list. The first has a value of 0, in the "just left" octet; the next has a value of 1; the next, a value of 2; and so on, through a block that begins with 172.30.255. Figure M-1 shows the concept.

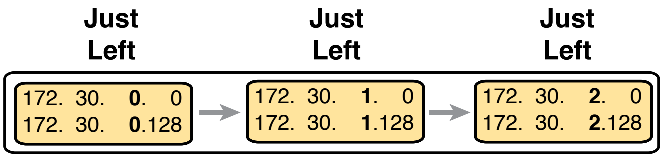

Figure M-1 *Creating Subnet Blocks by Adding 1 in the "Just Left" Octet*

Problem Set 2, Answer 2: 10.0.0.0/21

This problem has a 13-bit subnet field, meaning that 2^{13}, or 8192, possible subnets exist. The following list shows some of the subnets, which should be enough to see the trends in how to find all subnet numbers:

- 10.0.0.0 (zero subnet)
- 10.0.8.0
- 10.0.16.0
- 10.0.24.0

  (Skipping several subnets)
- 10.0.248.0
- 10.1.0.0
- 10.1.8.0
- 10.1.16.0

  (Skipping several subnets)
- 10.1.248.0

- 10.2.0.0
- 10.2.8.0
- 10.2.16.0

  (Skipping several subnets)
- 10.255.232.0
- 10.255.240.0
- 10.255.248.0 (broadcast subnet)

The process to find all subnets depends on three key pieces of information, as follows:

- The mask has more than 8 subnet bits (13 bits), because the network is a Class A network (8 network bits), and the mask has 21 binary 1s in it, which implies 11 host bits and leaves 13 subnet bits.
- Using the terminology in Appendix L, octet 3 is the interesting octet, where the counting occurs based on the magic number. Octet 2 is the "just left" octet, in which the process counts by 1, from 0 to 255.
- The magic number, which will be used to calculate each successive subnet number, is 256 – 248 = 8.

To calculate the first subnet block, use the same six-step process as used in the simpler problems that have 8 or fewer subnet bits. In this case, with 5 subnet bits in octet 3, 32 subnets exist in each subnet block. Table M-6 shows the steps as compared to the six-step process to find the subnets in a subnet block.

Table M-6 Creating the First Subnet Block

	Octet 1	Octet 2	Octet 3	Octet 4
Subnet Mask (Step 1)	255	255	248	0
Magic Number (Step 3)			256 – 248 = 8	
Zero Subnet Number (Step 4)	10	0	0	0
Next Subnet (Step 5)	10	0	8	0
(Skipping several subnets)	10	0	X	0
Next Subnet (Step 5)	10	0	248	0
Step 6 Needs to Be Used Here (Sum of 256 in the 3rd Octet)	10	0	256	0

The table represents the logic, but to make sure that the answer is clear, the first subnet block includes the following:

10.0.0.0

10.0.8.0

10.0.16.0

10.0.24.0

10.0.32.0

10.0.40.0

10.0.48.0

10.0.56.0

10.0.64.0

And so on…

10.0.248.0

The next major task—to create subnet blocks for all possible values in the "just left" octet—completes the process. Essentially, create 256 blocks like the previous list. The first has a value of 0, in the "just left" octet; the next has a value of 1; the next, a value of 2; and so on, through a block that begins with 10.255. Figure M-2 shows the concept.

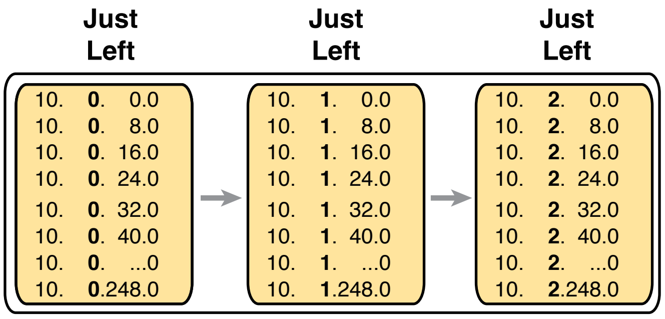

Figure M-2 *Creating Subnet Blocks by Adding 1 in the "Just Left" Octet*

APPENDIX N

Variable-Length Subnet Masks

NOTE This appendix contains an entire chapter that was published as a chapter in one of the past editions of this book or a related book. The author includes this appendix with the current edition as extra reading for anyone interested in learning more. However, note that the content in this appendix has not been edited since it was published in the earlier edition, so references to exams and exam topics, and to other chapters, will be outdated. This appendix was previously published as Chapter 22 of the book *CCENT/CCNA ICND1 100-105 Official Cert Guide*, published in 2016.

IPv4 addressing and subnetting use a lot of terms, a lot of small math steps, and a lot of concepts that fit together. While learning those concepts, it helps to keep things as simple as possible. One way this book has kept the discussion simpler so far was to show examples that use one mask only inside a single Class A, B, or C network.

This chapter removes that restriction by introducing variable-length subnet masks (VLSM). VLSM simply means that the subnet design uses more than one mask in the same classful network. VLSM has some advantages and disadvantages, but when learning, the main challenge is that a subnetting design that uses VLSM requires more math, and it requires that you think about some other issues as well. This chapter walks you through the concepts, the issues, and the math.

Foundation Topics

VLSM Concepts and Configuration

VLSM occurs when an internetwork uses more than one mask for different subnets of a single Class A, B, or C network. Figure N-1 shows an example of VLSM used in Class A network 10.0.0.0.

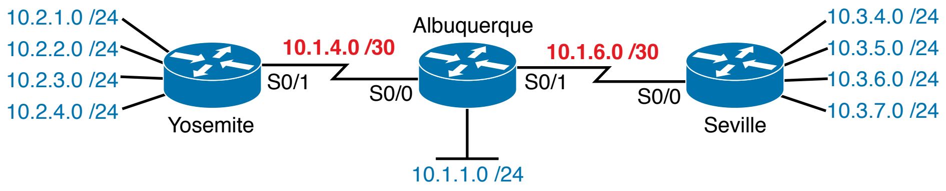

Figure N-1 *VLSM in Network 10.0.0.0: Masks /24 and /30*

Figure N-1 shows a typical choice of using a /30 prefix (mask 255.255.255.252) on point-to-point serial links, with mask /24 (255.255.255.0) on the LAN subnets. All subnets are of Class A network 10.0.0.0, with two masks being used, therefore meeting the definition of VLSM.

Oddly enough, a common mistake occurs when people think that VLSM means "using more than one mask in some internetwork" rather than "using more than one mask *in a single classful network*." For example, if in one internetwork diagram, all subnets of network 10.0.0.0 use a 255.255.240.0 mask, and all subnets of network 11.0.0.0 use a 255.255.255.0 mask, the design uses two different masks. However, Class A network 10.0.0.0 uses only one mask, and Class A network 11.0.0.0 uses only one mask. In that case, the design does not use VLSM.

VLSM provides many benefits for real networks, mainly related to how you allocate and use your IP address space. Because a mask defines the size of the subnet (the number of host addresses in the subnet), VLSM allows engineers to better match the need for addresses with the size of the subnet. For example, for subnets that need fewer addresses, the engineer uses a mask with fewer host bits, so the subnet has fewer host IP addresses. This flexibility reduces the number of wasted IP addresses in each subnet. By wasting fewer addresses, more space remains to allocate more subnets.

VLSM can be helpful for both public and private IP addresses, but the benefits are more dramatic with public networks. With public networks, the address savings help engineers avoid having to obtain another registered IP network number from regional IP address assignment authorities. With private networks, as defined in RFC 1918, running out of addresses is not as big a negative, because you can always grab another private network from RFC 1918 if you run out.

Classless and Classful Routing Protocols

Before you can deploy a VLSM design, you must first use a routing protocol that supports VLSM. To support VLSM, the routing protocol must advertise the mask along with each subnet. Without mask information, the router receiving the update would be confused.

For example, if a router learned a route for 10.1.8.0, but with no mask information, what does that mean? Is that subnet 10.1.8.0/24? 10.1.8.0/23? 10.1.8.0/30? The dotted-decimal number 10.1.8.0 happens to be a valid subnet number with a variety of masks, and because multiple masks can be used with VLSM, the router has no good way to make an educated guess. To effectively support VLSM, the routing protocol needs to advertise the correct mask along with each subnet so that the receiving router knows the exact subnet that is being advertised.

By definition, *classless routing protocols* advertise the mask with each advertised route, and *classful routing protocols* do not. The classless routing protocols, as noted in Table N-1, are the newer, more advanced routing protocols. Not only do these more advanced classless routing protocols support VLSM, but they also support manual route summarization, which allows a routing protocol to advertise one route for a larger subnet instead of multiple routes for smaller subnets.

Key Topic

Table N-1 Classless and Classful Interior IP Routing Protocols

Routing Protocol	Is It Classless?	Sends Mask in Updates?	Supports VLSM?	Supports Manual Route Summarization?
RIPv1	No	No	No	No
RIPv2	Yes	Yes	Yes	Yes
EIGRP	Yes	Yes	Yes	Yes
OSPF	Yes	Yes	Yes	Yes

Beyond VLSM itself, the routing protocols do not have to be configured to support VLSM or to be classless. There is no command to enable or disable the fact that classless routing protocols include the mask with each route. The only configuration choice you must make is to use a classless routing protocol.

VLSM Configuration and Verification

Cisco routers do not configure VLSM, enable or disable it, or need any configuration to use it. From a configuration perspective, VLSM is simply a side effect of using the **ip address** interface subcommand. Routers collectively configure VLSM by virtue of having IP addresses in the same classful network but with different masks.

For example, Example N-1 shows two of the interfaces from router Yosemite from Figure N-1. The example shows the IP address assignments on two interfaces, one with a /24 mask and one with a /30 mask, both with IP addresses in Class A network 10.0.0.0.

Example N-1 *Configuring Two Interfaces on Yosemite, Resulting in VLSM*

```
Yosemite# configure terminal
Yosemite(config)# interface Fa0/0
Yosemite(config-if)# ip address 10.2.1.1 255.255.255.0
Yosemite(config-if)# interface S0/1
Yosemite(config-if)# ip address 10.1.4.1 255.255.255.252
```

The use of VLSM can also be detected by a detailed look at the output of the **show ip route** command. This command lists routes in groups, by classful network, so that you see all the subnets of a single Class A, B, or C network all in a row. Just look down the list, and look to see, if any, how many different masks are listed. For example, Example N-2 lists the routing table on Albuquerque from Figure N-1; Albuquerque uses masks /24 and /30 inside network 10.0.0.0, as noted in the highlighted line in the example.

Example N-2 *Albuquerque Routing Table with VLSM*

```
Albuquerque# show ip route
! Legend omitted for brevity

     10.0.0.0/8 is variably subnetted, 14 subnets, 3 masks
D       10.2.1.0/24 [90/2172416] via 10.1.4.1, 00:00:34, Serial0/0
D       10.2.2.0/24 [90/2172416] via 10.1.4.1, 00:00:34, Serial0/0
D       10.2.3.0/24 [90/2172416] via 10.1.4.1, 00:00:34, Serial0/0
D       10.2.4.0/24 [90/2172416] via 10.1.4.1, 00:00:34, Serial0/0
D       10.3.4.0/24 [90/2172416] via 10.1.6.2, 00:00:56, Serial0/1
D       10.3.5.0/24 [90/2172416] via 10.1.6.2, 00:00:56, Serial0/1
D       10.3.6.0/24 [90/2172416] via 10.1.6.2, 00:00:56, Serial0/1
D       10.3.7.0/24 [90/2172416] via 10.1.6.2, 00:00:56, Serial0/1
C       10.1.1.0/24 is directly connected, FastEthernet0/0
L       10.1.1.1/32 is directly connected, FastEthernet0/0
C       10.1.6.0/30 is directly connected, Serial0/1
L       10.1.6.1/32 is directly connected, Serial0/1
C       10.1.4.0/30 is directly connected, Serial0/0
L       10.1.4.1/32 is directly connected, Serial0/0
```

NOTE For the purposes of understanding whether a design uses VLSM, ignore the /32 "local" routes that a router automatically creates for its own interface IP addresses.

So ends the discussion of VLSM as an end to itself. This chapter is devoted to VLSM, but it took a mere three to four pages to fully describe it. Why the entire VLSM chapter? Well, to work with VLSM, to find problems with it, to add subnets to an existing design, and to design using VLSM from scratch—in other words, to apply VLSM to real networks—takes skill and practice. To do these same tasks on the exam requires skill and practice. The rest of this chapter examines the skills to apply VLSM and provides some practice for these two key areas:

- Finding VLSM overlaps
- Adding new VLSM subnets without overlaps

Finding VLSM Overlaps

Key Topic

Regardless of whether a design uses VLSM, the subnets used in any IP internetwork design should not overlap their address ranges. When subnets in different locations overlap their addresses, a router's routing table entries overlap. As a result, hosts in different locations can be assigned the same IP address. Routers clearly cannot route packets correctly in these cases. In short, a design that uses overlapping subnets is considered to be an incorrect design and should not be used.