5장. IPv4 주소

2025년 1학기
단국대학교 컴퓨터공학과
박태근

Contents

[from TCP/IP Protocol Suite Fourth Edition]

5.1 개요 (Introduction)

✓ Internet에 연결되어 있는 **각 장치를 식별 (identify)**하기 위해 TCP/IP 프로토콜 모음의 IP 계층에서 사용하는 **식별자 (identifier)**를 **인터넷 주소 (Internet address) 또는 IP 주소 (IP address)**라고 함

✓ **IPv4 주소**는 **32비트 주소**로서 인터넷에 있는 호스트나 라우터의 연결을 **유일 (unique)하고 보편적 (universal)**으로 나타냄

- IP 주소는 **인터페이스의 주소** (**the address of the interface**)임

IPv4 주소는 32비트 주소임

IP 주소는 유일 (unique)하고 보편적 (universal)임

IPv4의 주소 공간은 2^{32}개 또는 4,294,967,296개임

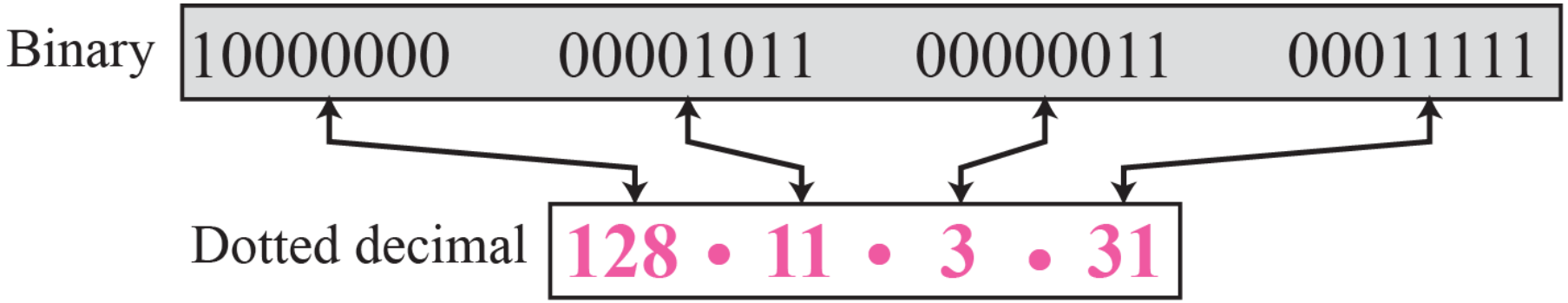
[from TCP/IP Protocol Suite Fourth Edition]

5.1 개요 (Introduction) - Topics

1) 표기법 (Notation)
2) 주소의 범위 (Range of Addresses)
3) 연산 (Operations)

[from TCP/IP Protocol Suite Fourth Edition]

5.1 개요 (Introduction) – 표기법 (Notation)



[from TCP/IP Protocol Suite Fourth Edition]

5.1 개요 (Introduction) – 표기법 (Notation) – Example 5.1

✓ 다음의 **2진 표기법** (**binary notation**) IPv4 주소를 **점-10진 표기법** (**dotted-decimal notation**)으로 변환하라.

a. 10000001 00001011 00001011 11101111

b. 11000001 10000011 00011011 11111111

c. 11100111 11011011 10001011 01101111

d. 11111001 10011011 11111011 00001111

Solution

✓ 각 **8비트 그룹**을 대응하는 **10진수**로 변환 (부록 B 참조)하고 **각 그룹을 점으로 구분**하였음

a. 129.11.11.239

b. 193.131.27.255

c. 231.219.139.111

d. 249.155.251.15

[from TCP/IP Protocol Suite Fourth Edition]

5.1 개요 (Introduction) – 표기법 (Notation) – Example 5.2

✓ 다음의 **점-10진 표기법** (**dotted-decimal notation**) IPv4 주소를 **2진 표기법** (**binary notation**) 으로 변환하라.

a. 111.56.45.78

b. 221.34.7.82

c. 241.8.56.12

d. 75.45.34.78

Solution

✓ 각 **10진수**를 대응하는 **2진수**로 변환함

a. 01101111 00111000 00101101 01001110

b. 11011101 00100010 00000111 01010010

c. 11110001 00001000 00111000 00001100

d. 01001011 00101101 00100010 01001110

[from TCP/IP Protocol Suite Fourth Edition]

5.1 개요 (Introduction) – 표기법 (Notation) – Example 5.3

✓ 다음의 IPv4 주소 표기법에서 **잘못된 점을 찾아보라**.

a. 111.56.045.78

b. 221.34.7.8.20

c. 75.45.301.14

d. 11100010.23.14.67

Solution

a. 점-10진 표기법에서 **0이 맨 앞에 나와서는 안됨** (045)

b. IPv4 주소에서는 **4바이트보다 많으면 안됨**

c. 점-10진 표기법에서 각 숫자는 **255보다 같거나 작아야 함** (301)

d. 2진 표기법과 점-10진 표기법을 **혼합 (mixture)해서 사용**하는 것은 **허용되지 않음**

[from TCP/IP Protocol Suite Fourth Edition]

5.1 개요 (Introduction) – 표기법 (Notation) – Example 5.4

✓ 다음의 2진 표기법 (**binary notation**) IPv4 주소를 16진 표기법 (**hexadecimal notation**)으로 **변환**하라.

a. 10000001 00001011 00001011 11101111

b. 11000001 10000011 00011011 11111111

Solution

✓ 각 4비트 그룹을 16진수로 변환함

✓ 주소의 **처음 부분**에 **0X (또는 0x)를 덧붙이**거나 주소의 **끝 부분**에 **아래 첨자** (**subscript**)로 **16을 표시**하여야 함

a. **0X**810B0BEF or $810B0BEF_{16}$

b. **0X**C1831BFF or $C1831BFF_{16}$

[from TCP/IP Protocol Suite Fourth Edition]

5.1 개요 (Introduction) – 주소의 범위 (Range of Addresses) – Example 5.5

✓ **처음 주소 (first address)**가 **146.102.29.0**이고 **마지막 주소 (last address)**가 **146.102.32.255**인 경우, 이 범위 내의 **주소의 수를 찾아보라**.

Solution

✓ **기수 256 (base 256)**으로, **마지막 주소에서 처음 주소를 뺌**

✓ 결과는 **0.0.3.255**임

✓ 이 범위 내의 **주소 수를 찾기** 위해, 이 수를 **기수 10의 수 (base 10)로 변환**하고, 그 **결과에 1을 더함**

$$\text{Number of addresses} = (0 \times 256^3 + 0 \times 256^2 + 3 \times 256^1 + 255 \times 256^0) + 1 = 1024$$

[from TCP/IP Protocol Suite Fourth Edition]

5.1 개요 (Introduction) – 주소의 범위 (Range of Addresses) – Example 5.6

✓ 주소 범위의 **첫 주소 (first address)**는 **14.11.45.96**임
✓ 만약 범위 내의 주소의 수가 **32**라면, **마지막 주소**는 무엇인가?

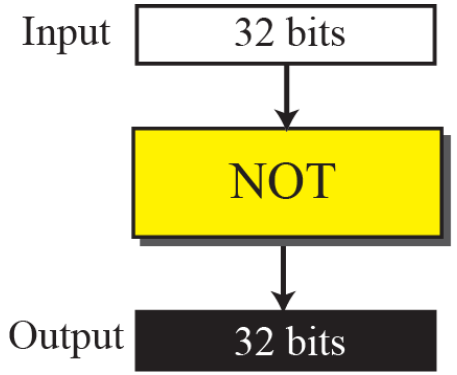
Solution

✓ **주소의 수에서 1을 뺀** 후, 그 결과를 **기수 256의 수 (base 256)로 변환**하면 **0.0.0.31**이 됨
✓ 이 값을 첫 주소에 **더하여 마지막 주소를 얻음** (**덧셈은 기수 256으로 수행**됨)

$$\text{Last address} = (14.11.45.96 + 0.0.0.31)_{256} = 14.11.45.127$$

[from TCP/IP Protocol Suite Fourth Edition]

5.1 개요 (Introduction) – 연산 (Operations) – Example 5.7



NOT operation

Input	Output
0	1
1	0

Operation for each bit

그림 5.2 비트단위의 NOT 연산 (bitwise NOT operation)

The following shows how we can apply the NOT operation on a 32-bit number in binary.

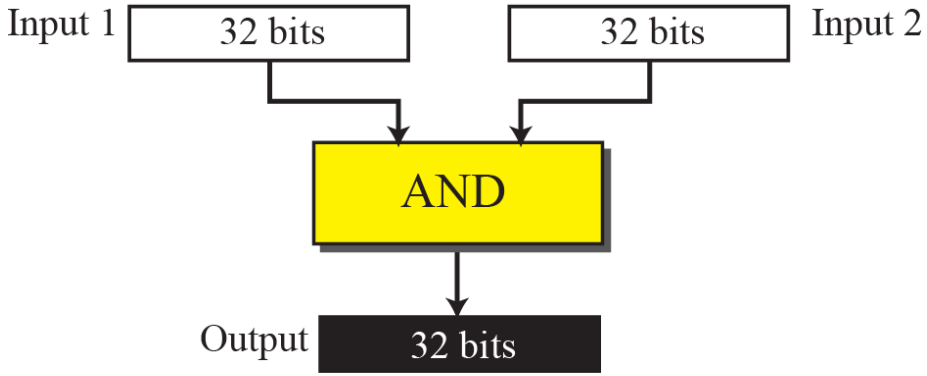
Original number:	00010001	01111001	00001110	00100011
Complement:	11101110	10000110	11110001	11011100

We can use the same operation using the dotted-decimal representation and the short cut.

Original number:	17	.	121	.	14	.	35
Complement:	238	.	134	.	241	.	220
	(+) 255		(+) 255		(+) 255		(+) 255

[from TCP/IP Protocol Suite Fourth Edition]

5.1 개요 (Introduction) – 연산 (Operations) – Example 5.8 (1)



AND

Input 1	Input 2	Output
0	0	0
0	1	0
1	0	0
1	1	1

Operation for each bit

그림 5.3 비트단위의 AND 연산 (bitwise AND operation)

다음은 **이진법**으로 표기된 **두 개의 32비트 수**에 **AND 연산을 적용**하는 방법을 보여줌

First number:	00010001	01111001	00001110	00100011
Second number:	11111111	11111111	10001100	00000000
Result	00010001	01111001	00001100	00000000

[from TCP/IP Protocol Suite Fourth Edition]

5.1 개요 (Introduction) – 연산 (Operations) – Example 5.8 (2)

We can use the same operation using the dotted-decimal representation and the short cut.

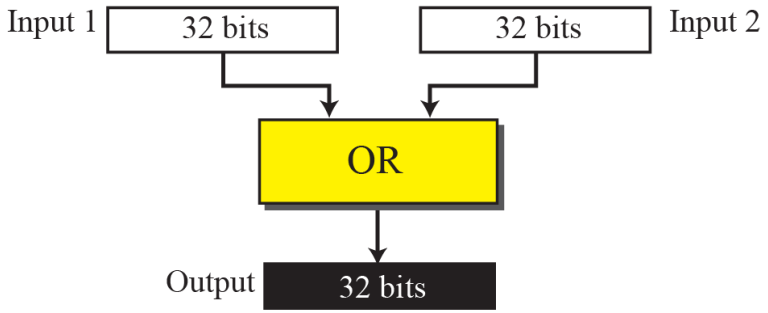
First number:	17	.	121	.	14	.	35
Second number:	255	.	255	.	140	.	0
Result:	17	.	121	.	12	.	0

We have applied the first short cut on the first, second, and the fourth byte; we have applied the second short cut on the third byte. We have written 14 and 140 as the sum of terms and selected the smaller term in each pair as shown below.

Powers	2^7		2^6		2^5		2^4		2^3		2^2		2^1		2^0
Byte (14)	0	+	0	+	0	+	0	+	8	+	4	+	2	+	0
Byte (140)	128	+	0	+	0	+	0	+	8	+	4	+	0	+	0
Result (12)	0	+	0	+	0	+	0	+	8	+	4	+	0	+	0

[from TCP/IP Protocol Suite Fourth Edition]

5.1 개요 (Introduction) – 연산 (Operations) – Example 5.9



OR

Input 1	Input 2	Output
0	0	0
0	1	1
1	0	1
1	1	1

Operation for each bit

그림 5.4 비트단위의 OR 연산 (bitwise OR operation)

The following shows how we can apply the OR operation on two 32-bit numbers in binary.

First number:	00010001	01111001	00001110	00100011
Second number:	11111111	11111111	10001100	00000000
Result	11111111	11111111	10001110	00100011

We can use the same operation using the dotted-decimal representation and the short cut.

First number:	17	.	121	.	14	.	35
Second number:	255	.	255	.	140	.	0
Result:	255	.	255	.	142	.	35

We have used the first short cut for ~~the first and second bytes~~ the first, second, and fourth byte and the second short cut for the third byte.

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 (Classful Addressing)

- ✓ **수십 년 전의 시작 시점**에, IP 주소는 **클래스 개념을 사용**하였음
  - ➢ 이러한 **구조 (architecture)**를 **클래스기반 주소지정 (classful addressing)**이라고 부름
- ✓ **1990년대 중반**, **클래스없는 주소지정** (**classless addressing**)이라 불리는 **새로운 아키텍처**가 도입되었으며, 이것이 원래의 아키텍처를 대체할 것이라 소개됨

- ✓ **클래스없는 주소지정** (**classless addressing**)으로 대체되어야 하는 이유를 이해하기 위하여, 본 절에서, **클래스기반 주소지정 (classful addressing)**의 개념을 먼저 살펴봄
- ✓ 다음 절에서, **클래스없는 주소지정** (**classless addressing**)에 대하여 살펴봄

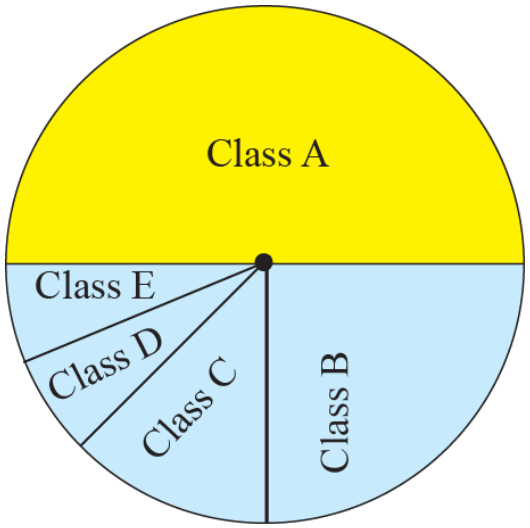
[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – Topics

1) 클래스 (Classes)
2) 클래스와 블록 (Classes and Blocks)
3) 2계층 주소지정 (Two-Level Addressing)
4) 3계층 주소지정: 서브네팅 (Three-Level Addressing: Subnetting)
5) 슈퍼네팅 (Supernetting)

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 클래스



Class A: 2^{31} = 2,147,483,648 addresses, 50%

Class B: 2^{30} = 1,073,741,824 addresses, 25%

Class C: 2^{29} = 536,870,912 addresses, 12.5%

Class D: 2^{28} = 268,435,456 addresses, 6.25%

Class E: 2^{28} = 268,435,456 addresses, 6.25%

그림 5.5 주소 공간의 점유 (Occupation of address space)

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 클래스

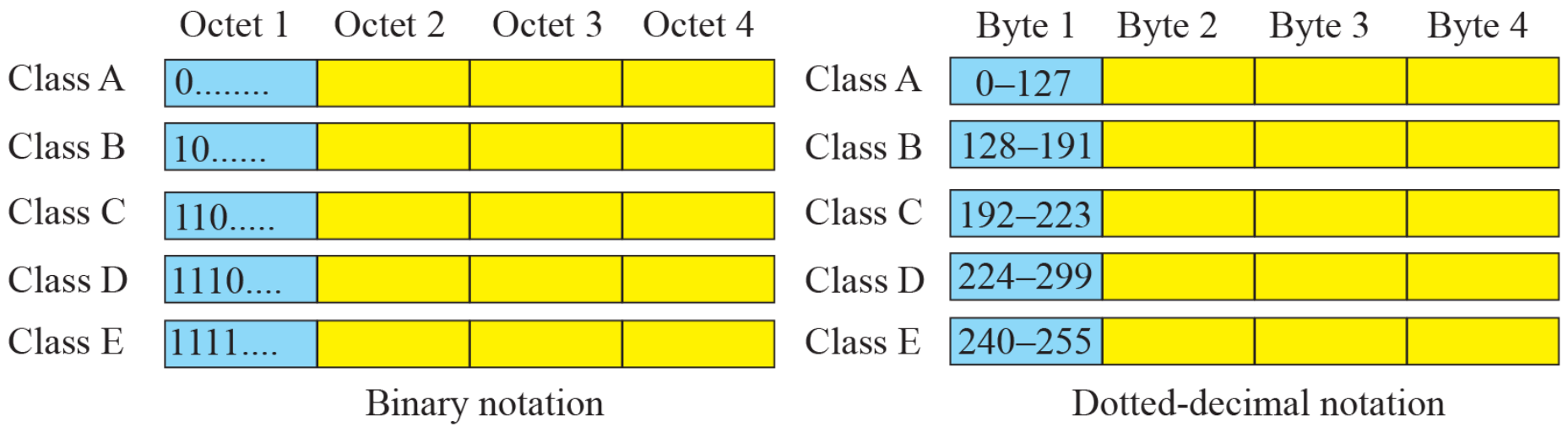


그림 5.6 주소의 클래스 찾기 (Finding the class of an address)

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 클래스

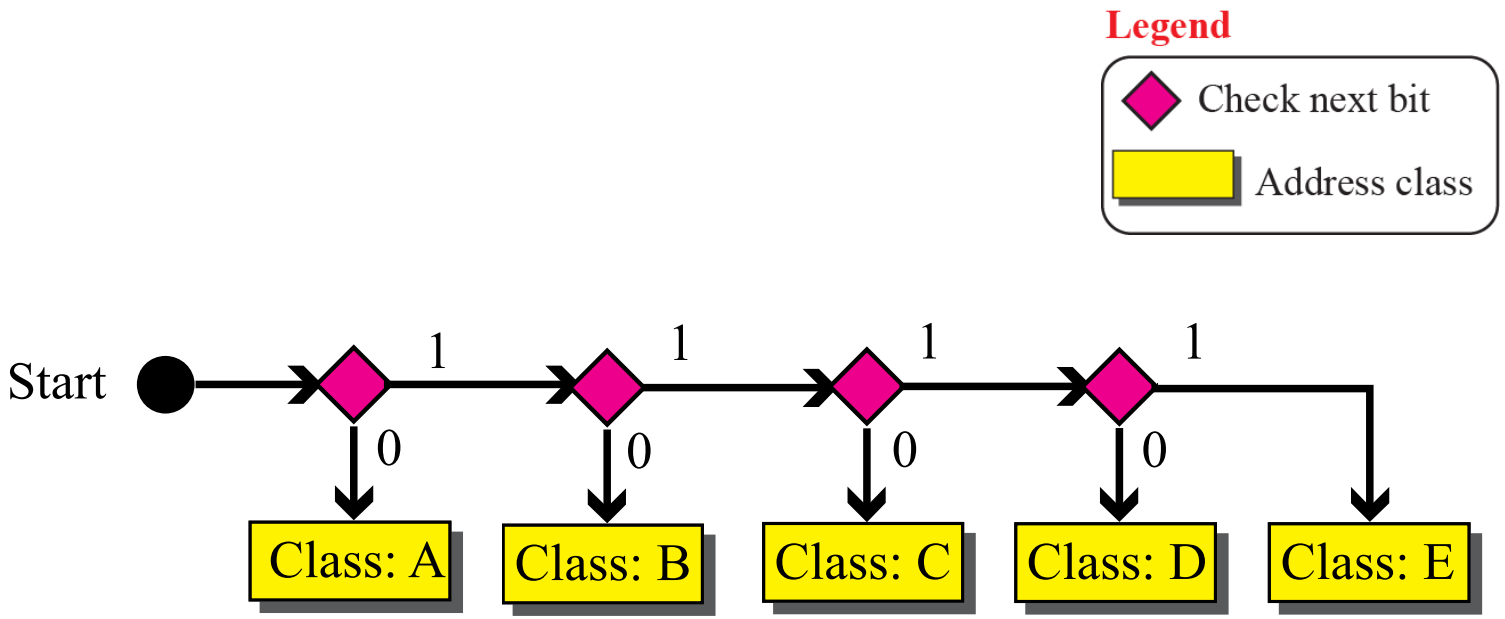


그림 5.7 연속적 검증을 사용한 주소 클래스 찾기 (Finding the address class using continuous checking)

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 클래스 – Example 5.10

✓ 각 주소의 클래스를 나타내어라.

a. 00000001 00001011 00001011 11101111
b. 11000001 10000011 00011011 11111111
c. 10100111 11011011 10001011 01101111
d. 11110011 10011011 11111011 00001111

Solution

✓ 그림 5.7의 처리과정을 참고하라.

a. 첫 번째 비트가 0이므로, 클래스 A 주소이다.
b. 처음 2비트가 1이고, 세 번째 비트가 0이므로, 클래스 C 주소이다.
c. 첫 번째 비트가 1이고, 두 번째 비트가 0이므로, 클래스 B 주소이다.
d. 처음 4비트가 1이므로, 클래스 E 주소이다.

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 클래스 – Example 5.11

✓ 각 주소의 클래스를 나타내어라.

a. 227.12.14.87

b. 193.14.56.22

c. 14.23.120.8

d. 252.5.15.111

Solution

a. 첫 번째 바이트가 227 (224와 239 사이)이므로, 클래스는 D이다.

b. 첫 번째 바이트가 193 (192와 223 사이)이므로, 클래스는 C이다.

c. 첫 번째 바이트가 14 (0과 127 사이)이므로, 클래스는 A이다.

d. 첫 번째 바이트가 252 (240과 255사이)이므로, 클래스는 E이다.

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 클래스와 블록

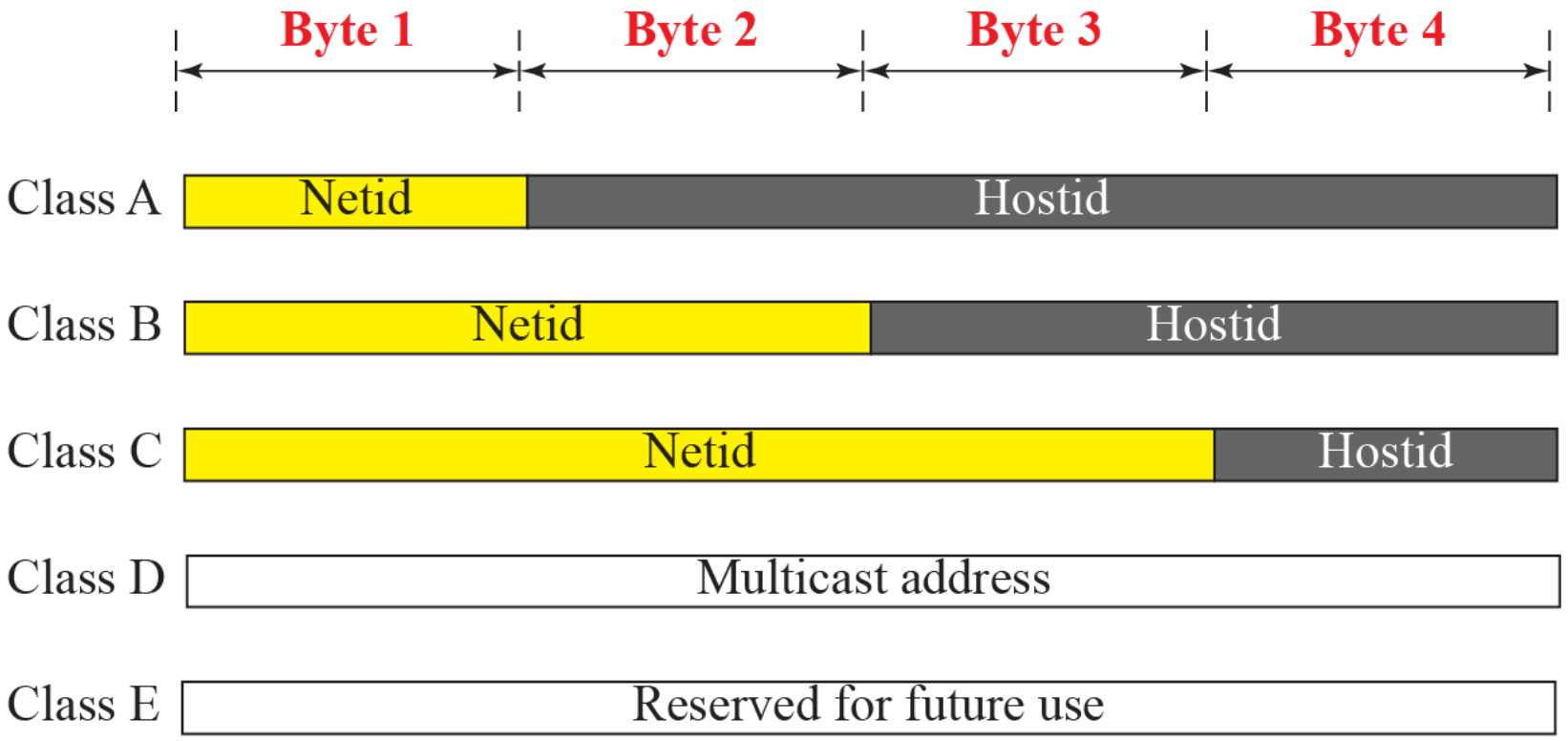


그림 5.8 Netid와 Hostid

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 클래스와 블록

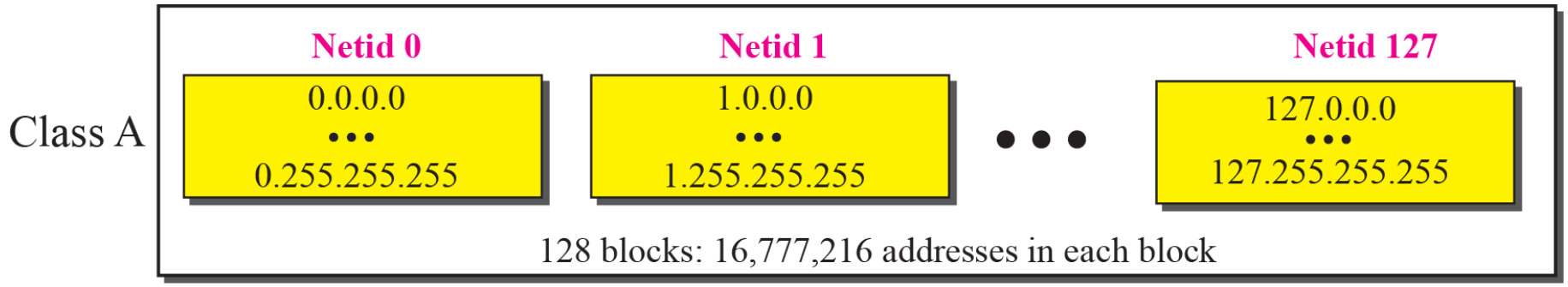


그림 5.9 클래스 A의 블록 (Blocks in Class A)

수백만의 클래스 **A** 주소는 낭비된다.

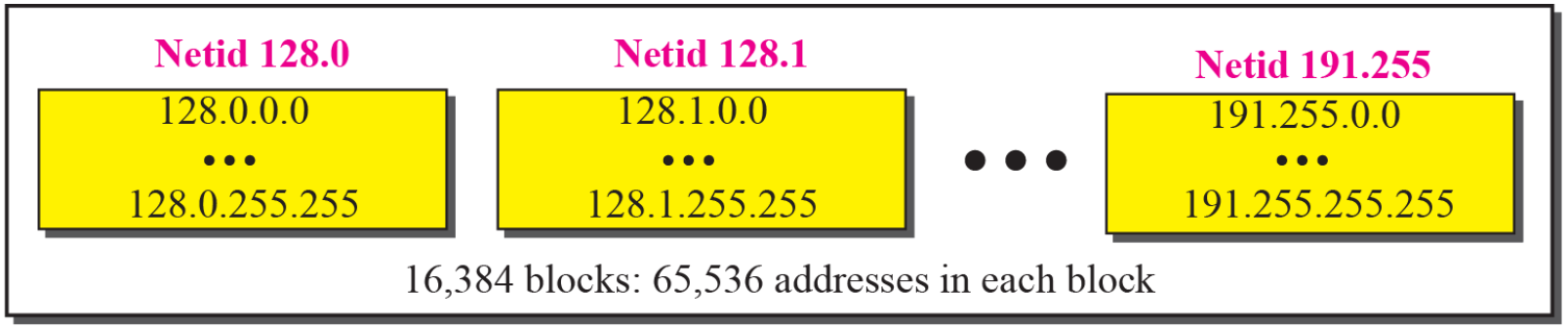


그림 5.10 클래스 B의 블록 (Blocks in Class B)

많은 클래스 **B** 주소가 낭비된다.

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 클래스와 블록

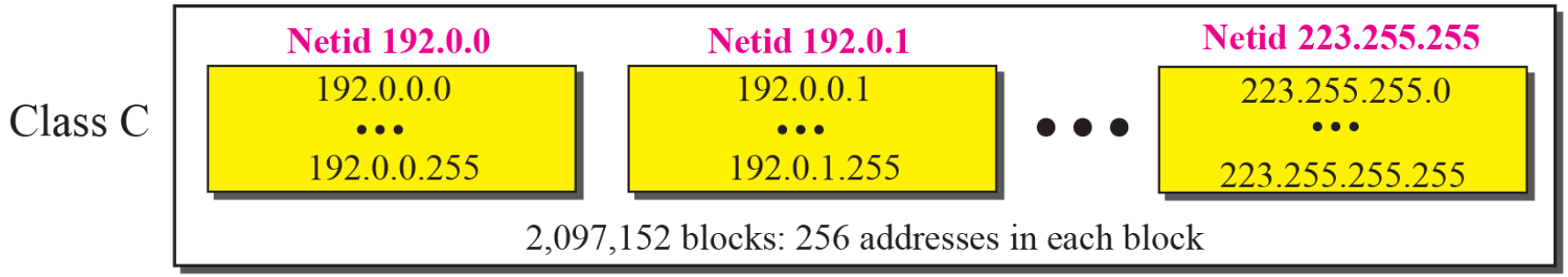


그림 5.11 클래스 C의 블록 (Blocks in Class C)

C 클래스 블록의 크기가 충분한 작은 크기의 기관이 많지 않다.

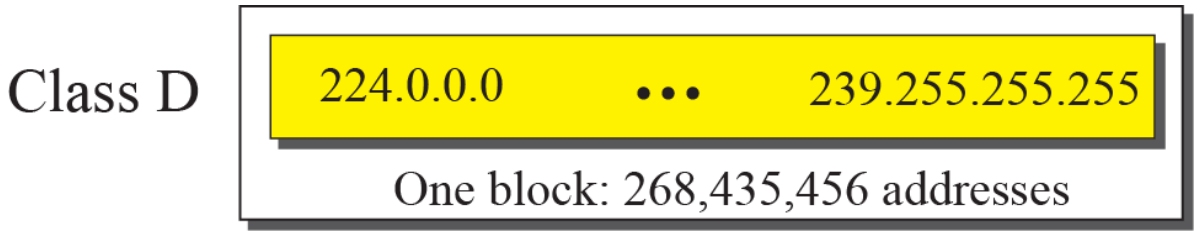


그림 5.12 클래스 D의 단일 블록 (The single block in Class D)

클래스 **D** 주소는 하나의 블록만을 가지며, 멀티캐스팅을 위해 사용된다.

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 클래스와 블록

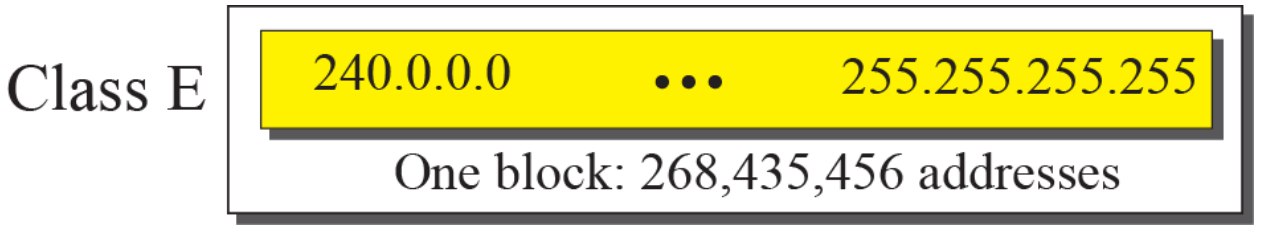


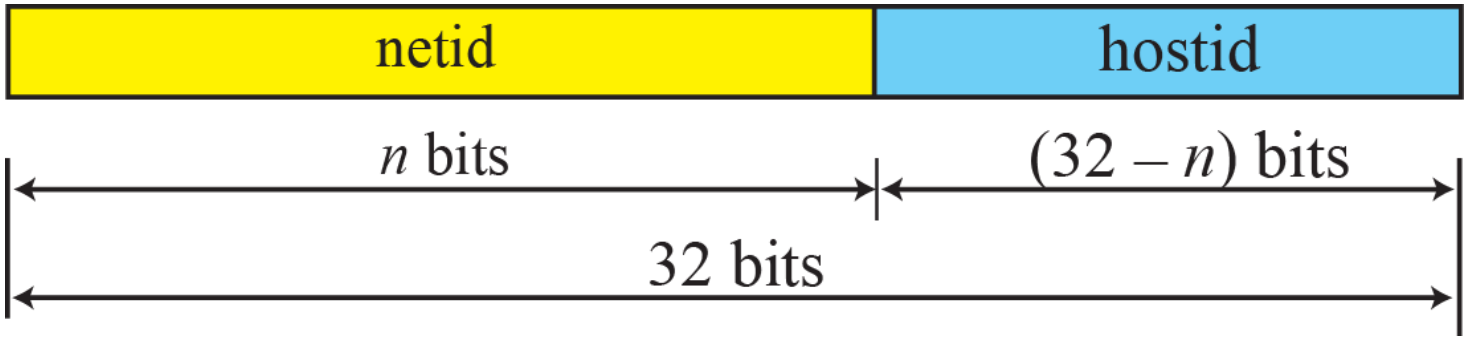
그림 5.13 클래스 E의 단일 블록 (The single block in Class E)

클래스 **E** 주소는 하나의 블록만을 가지며, 미래 사용을 위해 예약되어 있다.

클래스기반 주소지정 **(classful addressing)**에서 조직에 할당되는 주소 범위는 **A, B, C** 클래스에 속하는 주소블록이다.

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 2계층 주소지정



Class A: $n = 8$

Class B: $n = 16$

Class C: $n = 24$

그림 5.14 클래스기반 주소지정에서의 2계층 주소지정
(Two-level addressing in classful addressing)

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 2계층 주소지정 – Example 5.12

Two-level addressing can be found in other communication systems. For example, a telephone system inside the United States can be thought of as two parts: area code and local part. The area code defines the area, the local part defines a particular telephone subscriber in that area.

(626) 3581301

The area code, 626, can be compared with the netid, the local part, 3581301, can be compared to the hostid.

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 2계층 주소지정

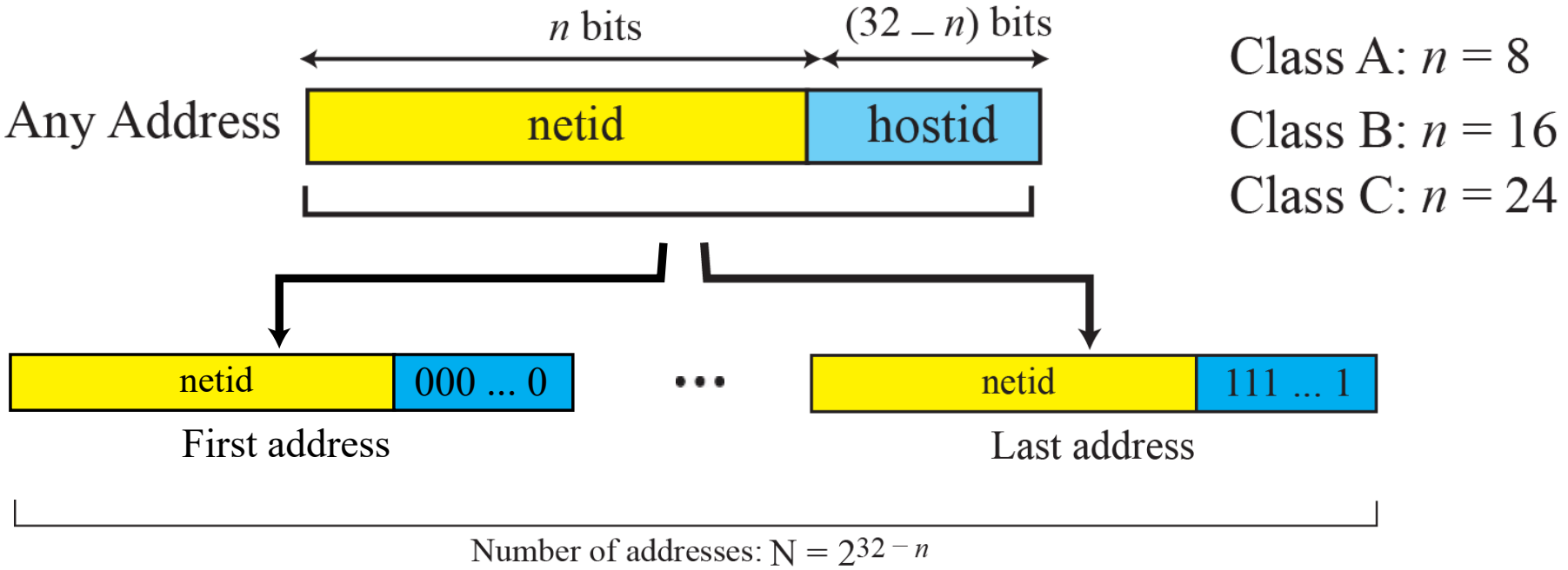


그림 5.15 클래스기반 주소지정에서의 정보 추출
(Information extraction in classful addressing)

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 2계층 주소지정 – Example 5.13 (1)

✓ 블록 내에 속한 하나의 (어떤) 주소가 **73.22.17.25**이라 하자.
✓ 해당 블록의 (1) 주소 수, (2) 첫 주소, (3) 마지막 주소를 구하라.

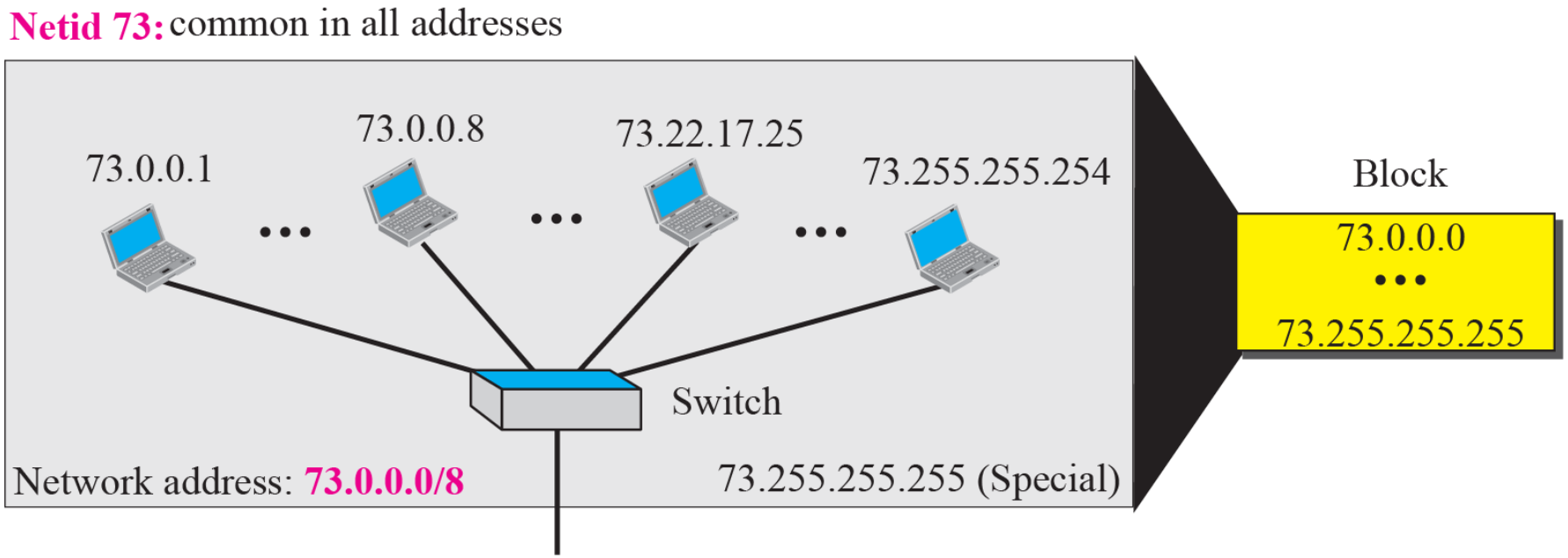
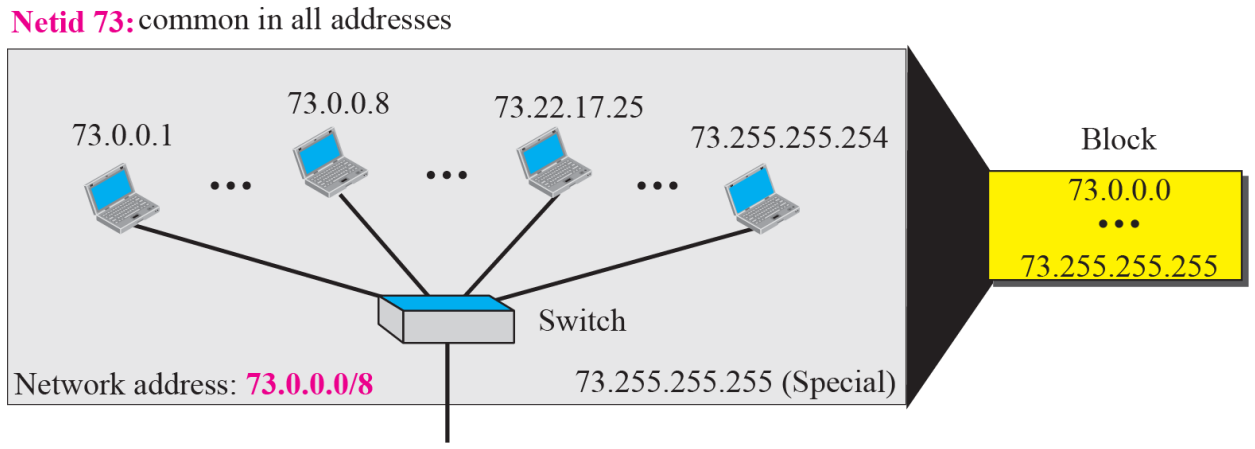


그림 5.16 예제 5.13에 대한 정답

(상세 설명은 다음 페이지)

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 2계층 주소지정 – Example 5.13 (2)



해답 (Solution)

✓ 그림 5.16은 해당 블록을 사용하는 네트워크에서 "하나의 가능한 구성"을 보여준다.

1. 해당 블록의 주소 수는

   $N = 2^{32-n} = 2^{24} = 16{,}777{,}216.$

2. 첫 주소를 찾기 위하여, 왼쪽의 8비트를 유지하고, 오른쪽의 24비트를 0으로 만든다.
   첫 번째 주소는 73.0.0.0/8이고, 8은 n의 값이다.
3. 마지막 주소를 찾기 위하여, 왼쪽의 8비트를 유지하고, 오른쪽의 24비트를 1로 만든다.
   마지막 주소는 73.255.255.255이다.

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 2계층 주소지정 – Example 5.14 (1)

✓ 블록 내에 속한 하나의 (어떤) 주소가 **180.8.17.9**이라 하자.
✓ 해당 블록의 (1) 주소 수, (2) 첫 주소, (3) 마지막 주소를 구하라.

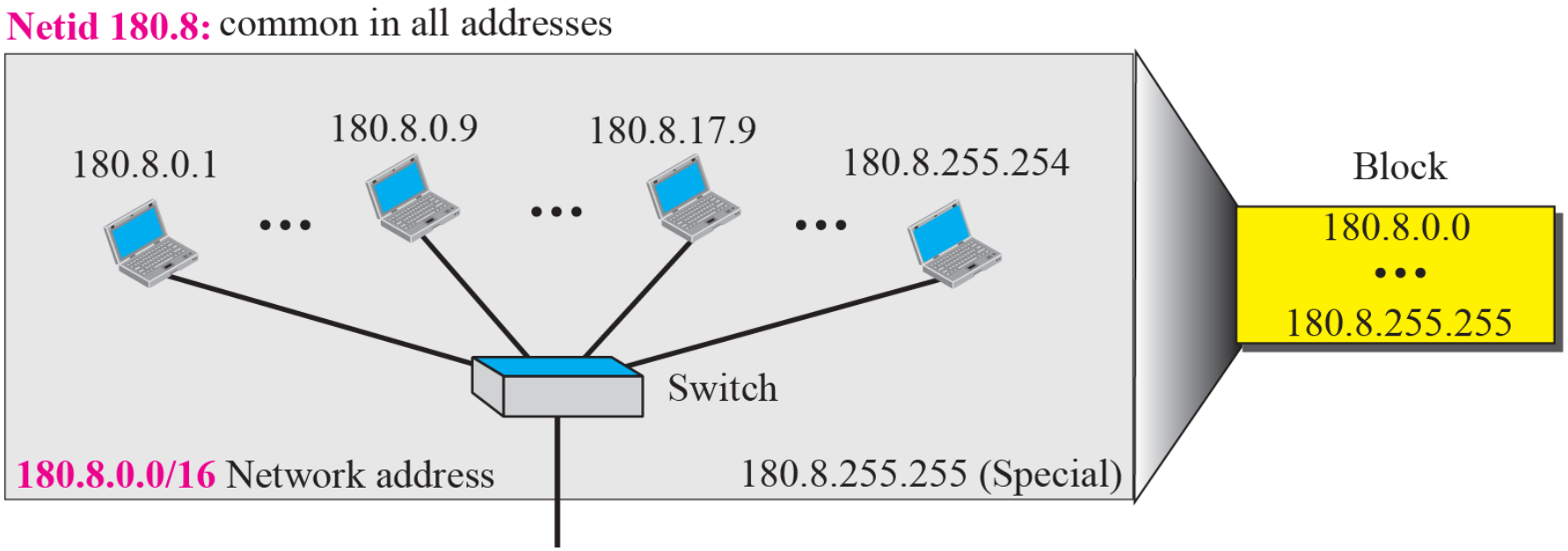
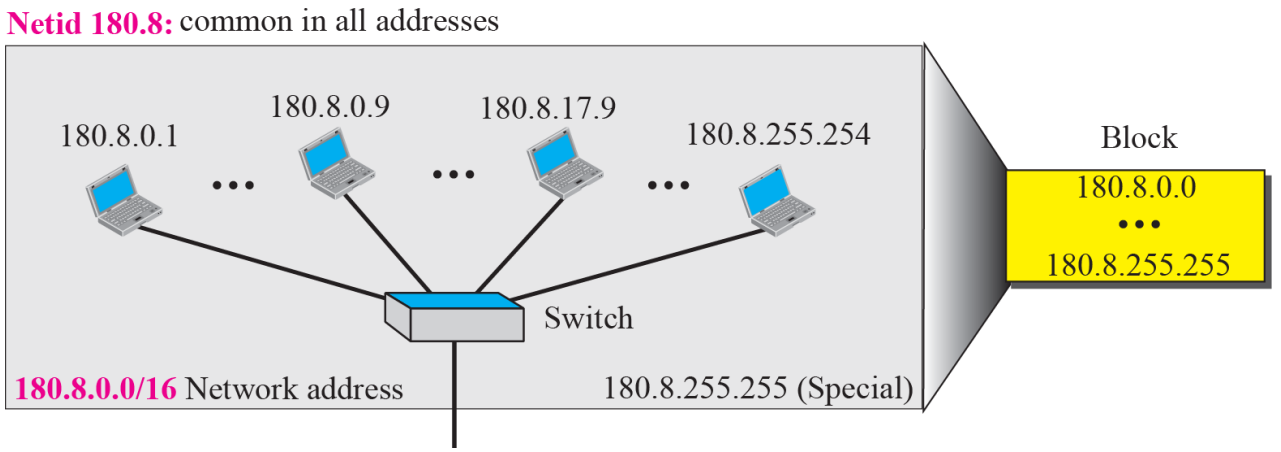


그림 5.17 예제 5.14에 대한 해답

(상세 설명은 다음 페이지)

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 2계층 주소지정 – Example 5.14 (2)



해답 (Solution)

✓ 그림 5.17은 해당 블록을 사용하는 네트워크에서 "하나의 가능한 구성"을 보여준다.

1. 해당 블록의 주소 수는

   $N = 2^{32-n} = 2^{16} = 65{,}536.$

2. 첫 주소를 찾기 위하여, 왼쪽의 16비트를 유지하고, 오른쪽의 16비트를 0으로 만든다.

   첫 번째 주소는 180.8.0.0/16이고, 16은 n의 값이다.

3. 마지막 주소를 찾기 위하여, 왼쪽의 16비트를 유지하고, 오른쪽의 16비트를 1로 만든다.

   마지막 주소는 180.8.255.255이다.

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 2계층 주소지정 – Example 5.15 (1)

✓ 블록 내에 속한 하나의 (어떤) 주소가 **200.11.8.45**이라 하자.
✓ 해당 블록의 (1) 주소 수, (2) 첫 주소, (3) 마지막 주소를 구하라.

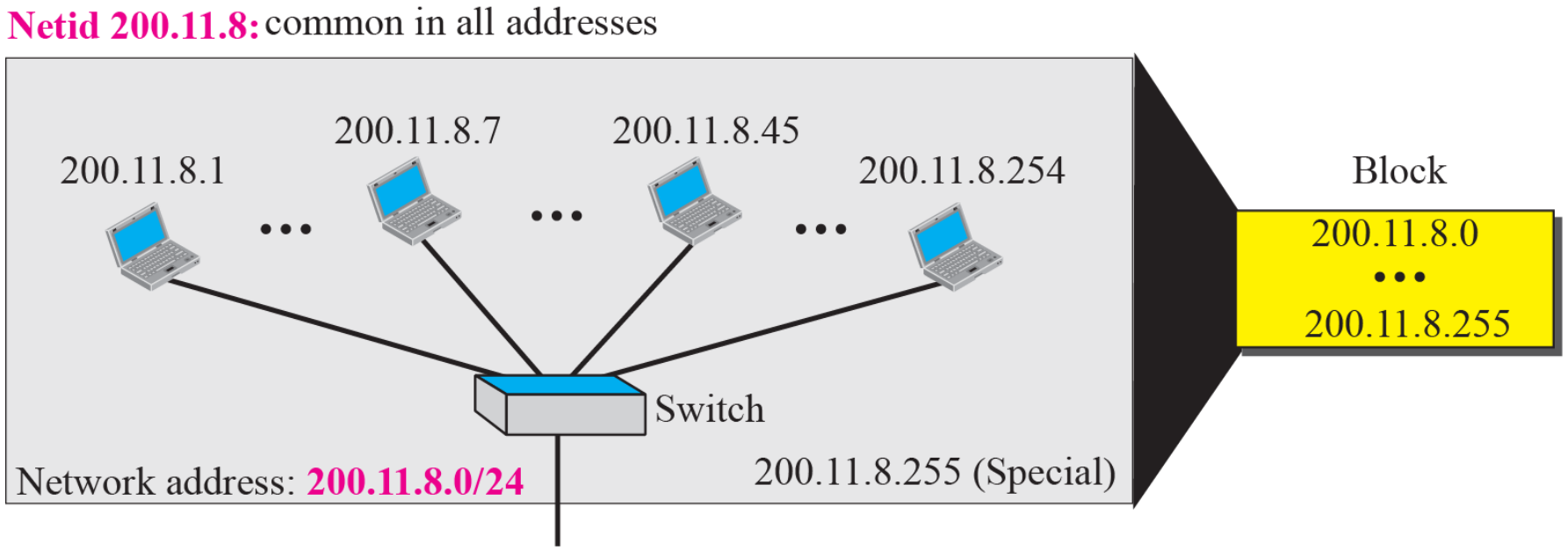
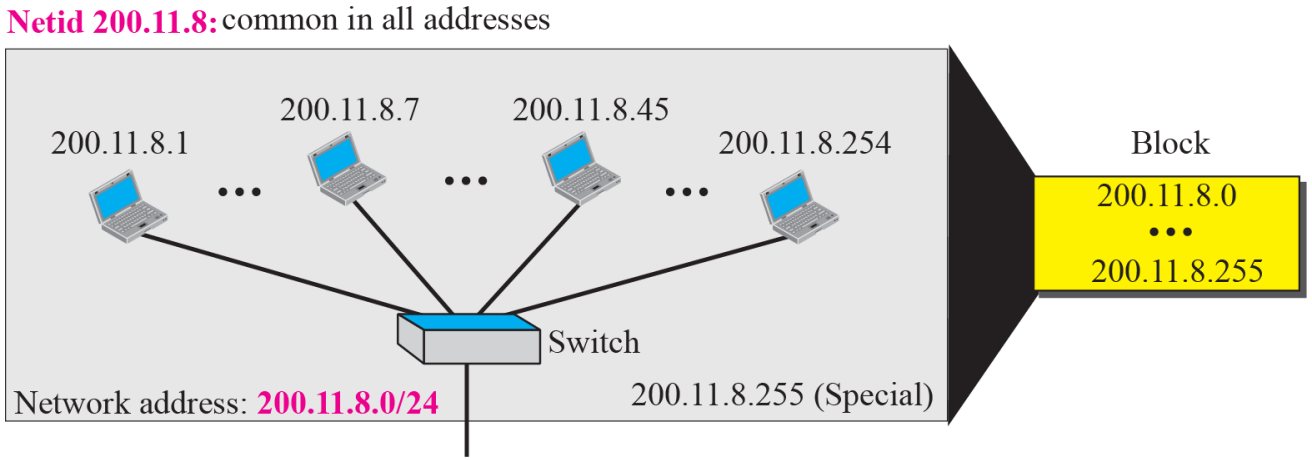


그림 5.18 예제 5.15에 대한 정답

(상세 설명은 다음 페이지)

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 2계층 주소지정 – Example 5.15 (2)



해답 (Solution)

✓ 그림 5.18은 해당 블록을 사용하는 네트워크에서 "하나의 가능한 구성"을 보여준다.

1. 해당 블록의 주소 수는

   $N = 2^{32-n} = 2^8 = 256$.

2. 첫 주소를 찾기 위하여, 왼쪽의 24비트를 유지하고, 오른쪽의 8비트를 0으로 만든다.

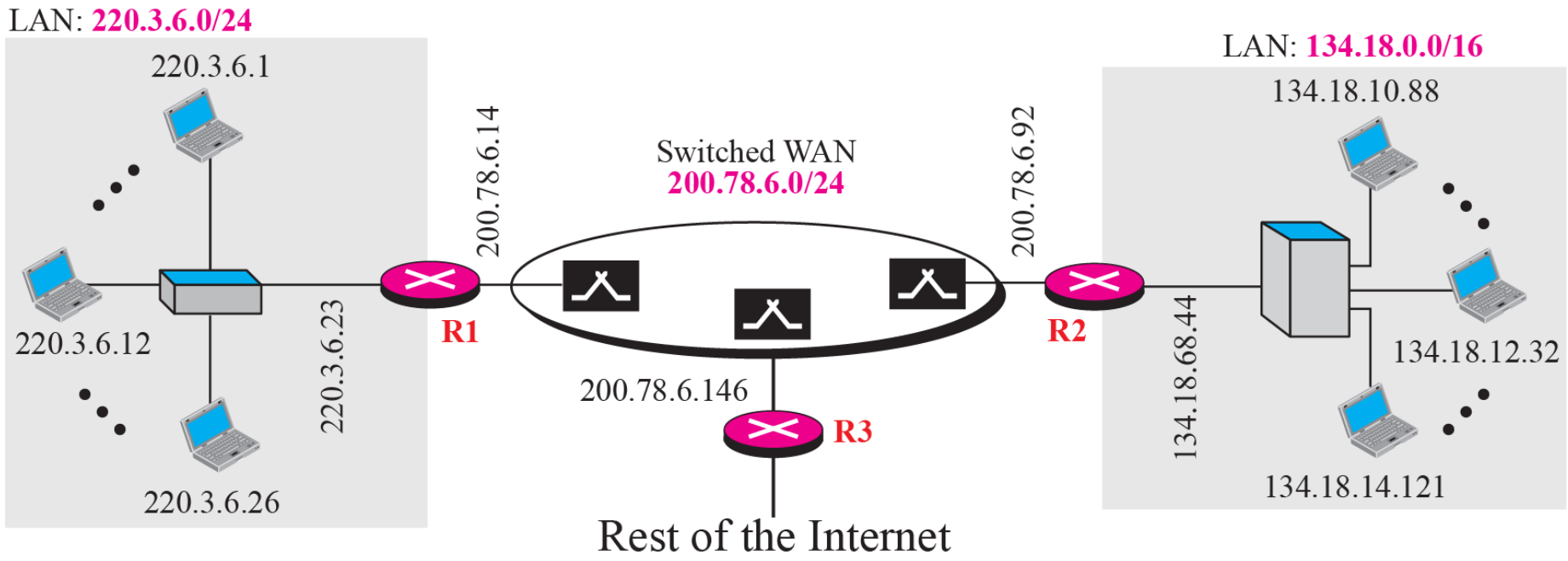
   첫 번째 주소는 200.11.8.0/24이고, 24은 n의 값이다.

3. 마지막 주소를 찾기 위하여, 왼쪽의 24비트를 유지하고, 오른쪽의 8비트를 1로 만든다.

   마지막 주소는 200.11.8.255이다.

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 2계층 주소지정

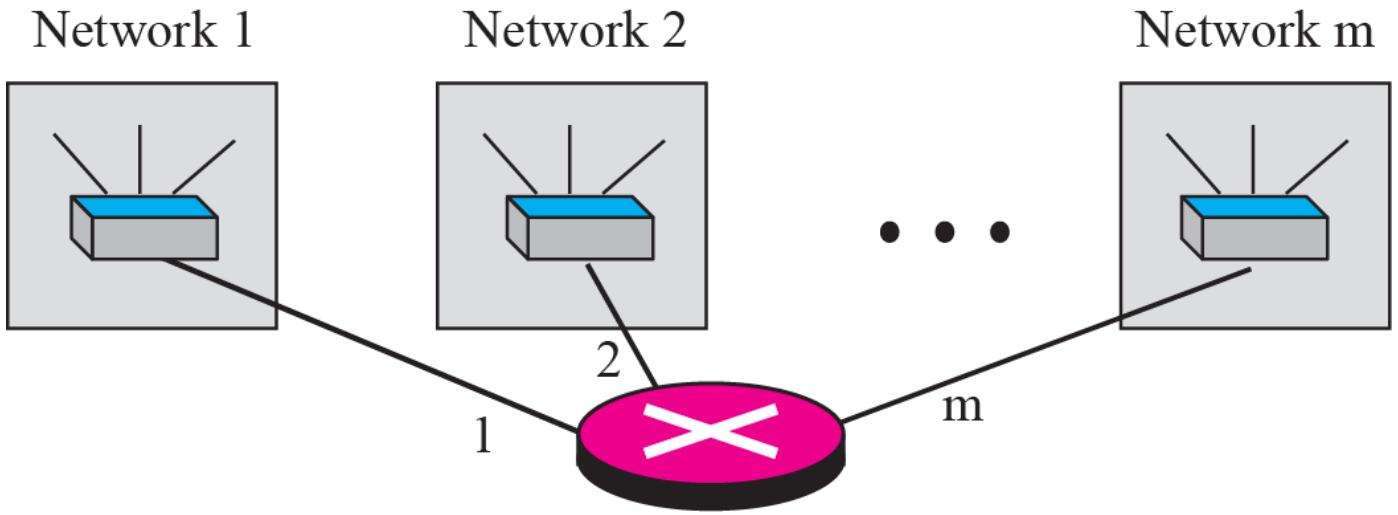


네트워크 주소는
네트워크의 식별자이다.

그림 5.19 샘플 인터넷 (Sample Internet)

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 2계층 주소지정



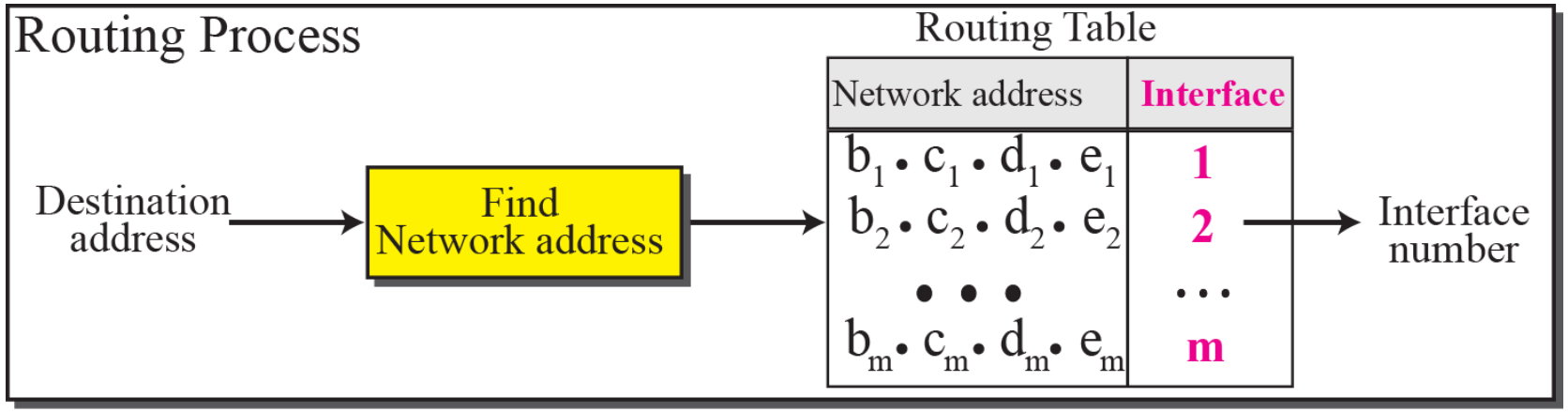


그림 5.20 네트워크 주소 (Network address)

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 2계층 주소지정

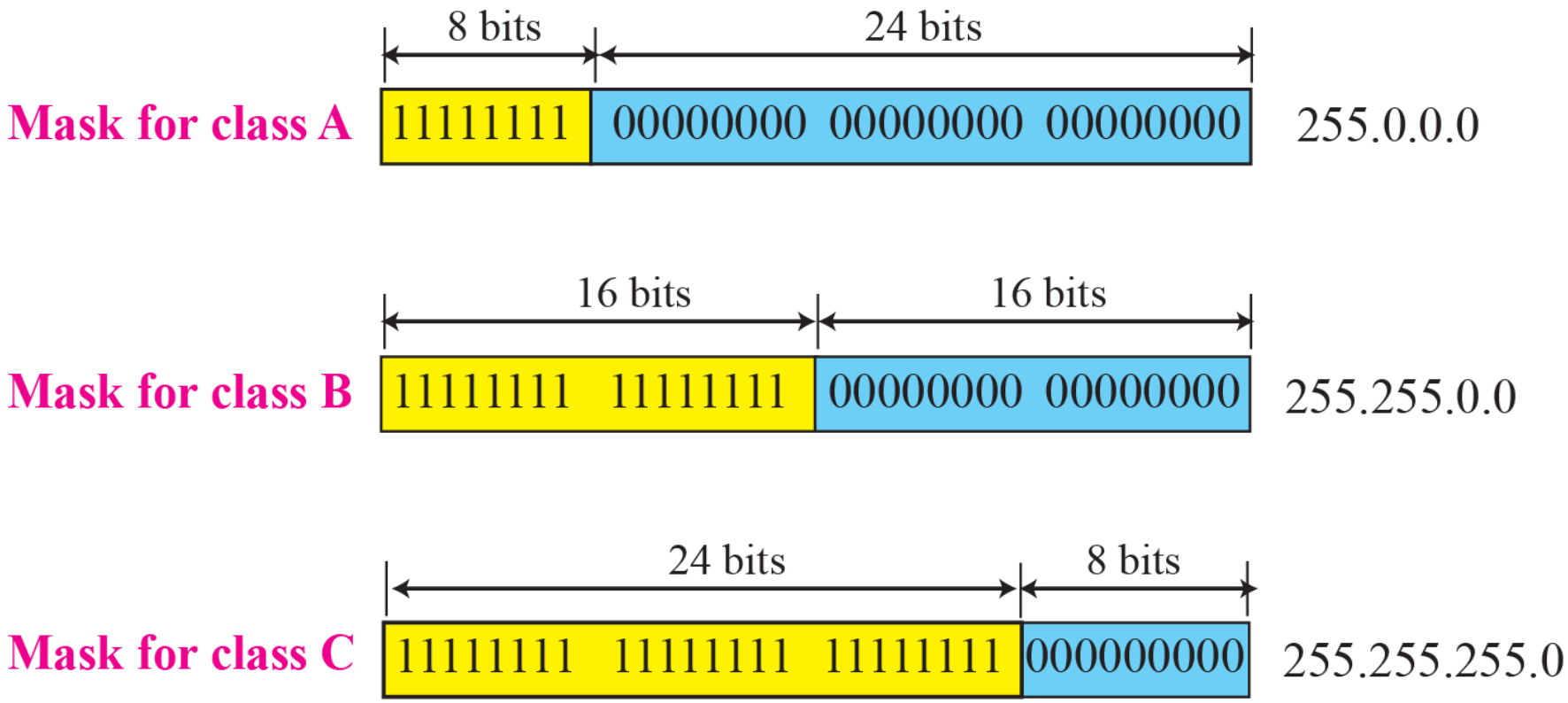


그림 5.21 네트워크 마스크 (Network mask)

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 2계층 주소지정

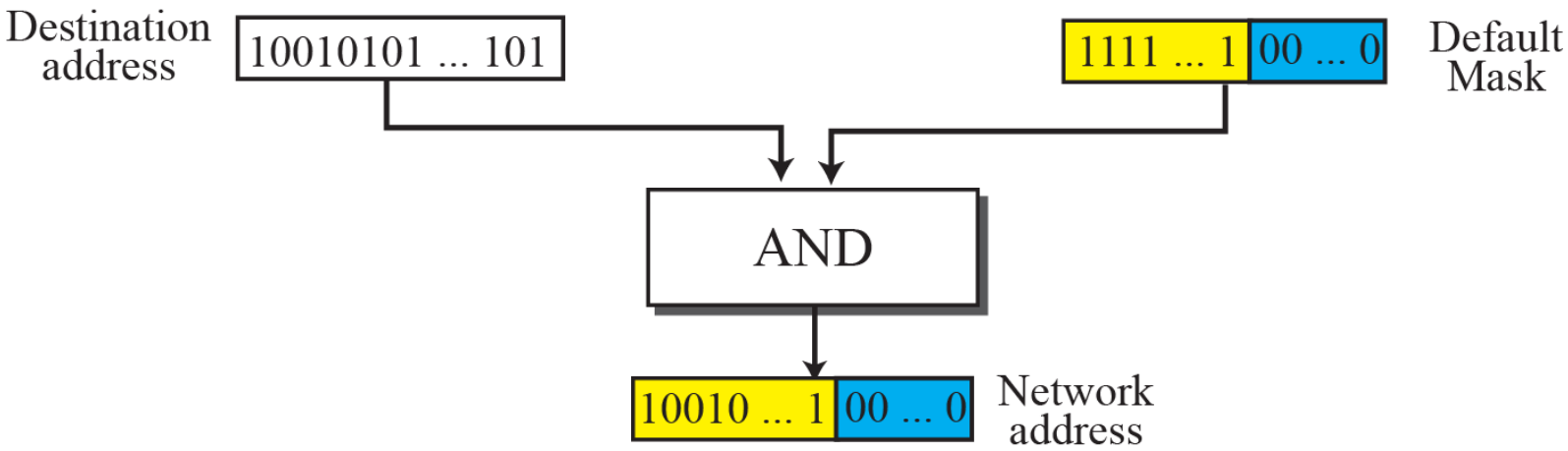


그림 5.22 디폴트 마스크를 사용한 네트워크 주소 찾기
(Finding a network address using the default mask)

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 2계층 주소지정 – Example 5.16

✓ 라우터가 **목적지 주소가 201.24.67.32**인 패킷을 받는다.
✓ 라우터가 패킷의 **네트워크 주소를 찾는 방법**을 보여라.

해답 (Solution)

✓ 주소의 클래스가 C이므로, 라우터는
- ➢ 네트워크 주소를 찾기 위해
- ➢ 클래스 C를 위한 기본 마스크인 255.255.255.0를 적용한다.

Destination address	→	201	.	24	.	67	.	32
Default mask	→	255	.	255	.	**255**	.	0
Network address	→	201	.	24	.	**67**	.	0

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 3계층 주소지정 – Example 5.17

✓ 전화번호의 로컬 파트를 교환기 (exchange)와 가입자 연결 (subscriber connection)의 두 부분으로 생각하면 전화 시스템을 **3계층 주소지정** (**three-level addressing**)으로 생각할 수 있다.

(626) 358 - 1301

여기서 **626**은 영역 코드 (area code)이고, **358**은 교환기 (exchange) 번호이며, **1301**은 가입자 연결 (subscriber connection)이다.

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 3계층 주소지정 – Example 5.18

- ✓ 그림 5.23은 클래스 B 주소를 사용하는 네트워크가 **서브네팅을 사용하기 전의 구성**을 보여준다.
- ✓ **거의 2^{16} 개의 호스트**를 가진 하나의 네트워크 만이 존재한다.
- ✓ 한 개의 연결을 통하여, 전체 네트워크가 **Internet 내의 한 개의 라우터에 연결**되어 있다.
- ✓ 클래스 B **netid**의 길이를 보이기 위하여 **/16**을 표기하였다.

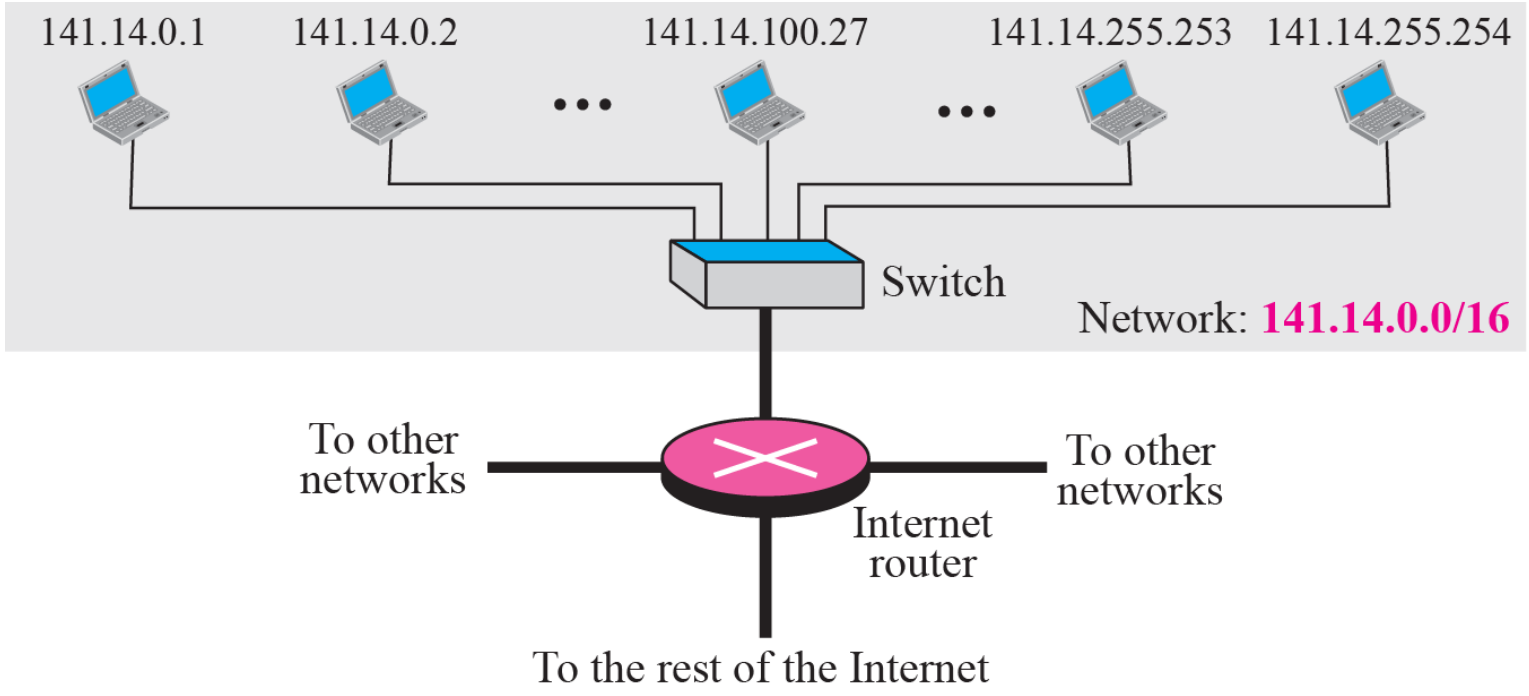


그림 5.23 예제 5.18

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 3계층 주소지정 – Example 5.19 (1)

- ✓ 그림 5.24는, 그림 5.23과 같은 네트워크에 대하여, **서브네팅을 사용한 이후의 구성**을 보여준다.
- ✓ 전체 네트워크는 여전히 동일한 라우터를 통하여 인터넷에 연결된다.
- ✓ 그러나, 네트워크는 **사설 라우터 (private router)**를 사용하여 네트워크를 **네 개의 서브네트워크 (subnetworks)로 나뉘어져** 있다.
- ✓ **Internet의 나머지 부분**은 이 네트워크를 **여전히 한 개의 네트워크로 보고 있지만,**
  - ➢ 내부적으로 네트워크는 네 개의 서브네트워크로 구성되어 있다.

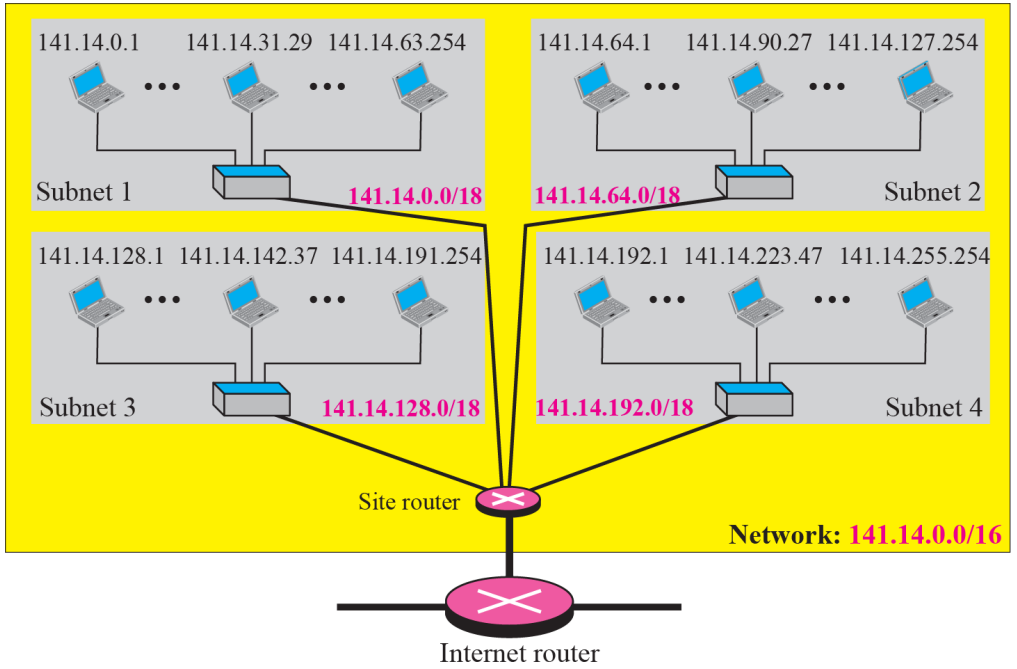


그림 5.24 예제 5.19

(다음 페이지에 설명 계속)

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 3계층 주소지정 – Example 5.19 (2)

- ✓ 각 서브네트워크는 **거의 2^{14}개의 호스트**를 가질 수 있다.
- ✓ 네트워크는 **네 개의 다른 학부 (건물)**를 가진 대학 캠퍼스에 설치되어 있을 수 있다.
- ✓ 서브네팅 후, 각 학부는 자신의 서브네트워크를 가지지만, 외부 네트워크는 전체 네트워크를 한 개의 네트워크로 인식한다.
- ✓ **/16**과 **/18**은 **netid**와 **subnetid**의 길이를 의미한다.

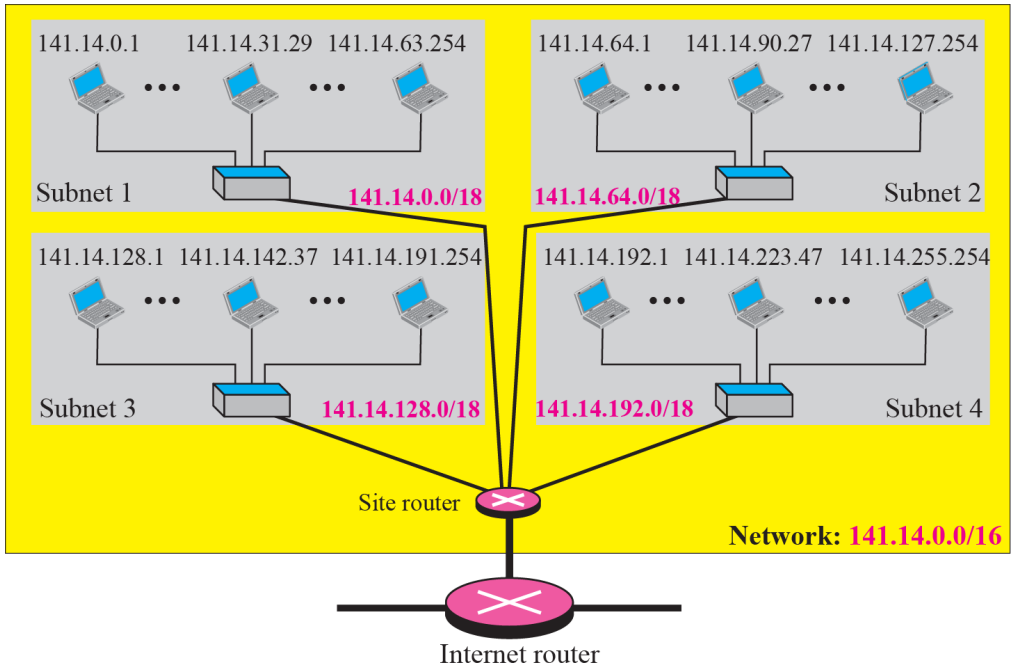


그림 5.24 예제 5.19

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 3계층 주소지정

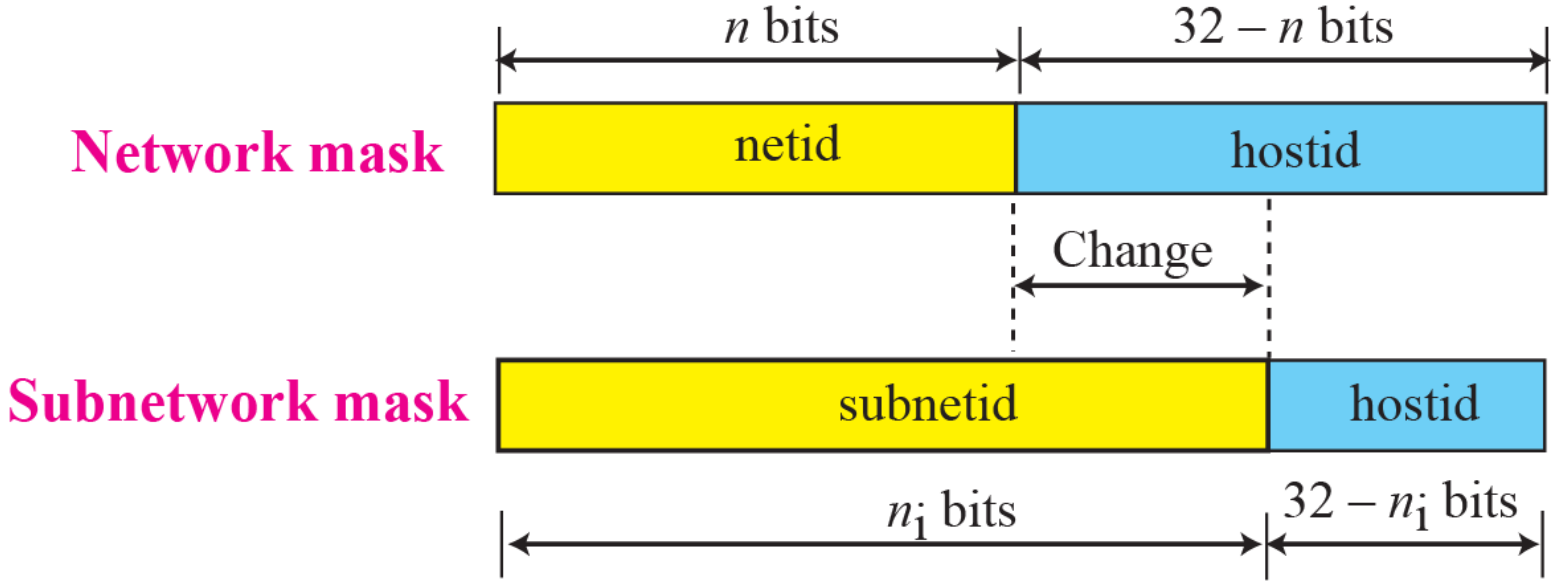


그림 5.25 네트워크 마스크와 서브네트워크 마스크
(Network mask and subnetwork mask)

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 3계층 주소지정 – Example 5.20

✓ 예제 5.19에서, 클래스 B 네트워크를 **네 개의 서브네트워크 (four subnetworks)**로 나누었다.

n = 16이고, $n_1 = n_2 = n_3 = n_4$ = 16 + **$\log_2 4$** = 18이다.

✓ 이것은 서브넷 마스크가 **18개의 1**을 가지고 있고, **14개의 0**을 가진다는 것을 의미한다.

✓ 다시 설명하면,

- 서브넷 마스크 (subnet mask)는 **255.255.192.0** 이고,
- **클래스 B의 네트워크 마스크 (network mask)**인 **255.255.0.0**과 다르다는 것을 알 수 있다.

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 3계층 주소지정 – Example 5.21

- ✓ 예제 5.19에서, 하나의 네트워크가 네 개의 서브넷 (subnet)로 나뉘었음을 확인하였다.
- ✓ **서브넷 2 (subnet 2)의 주소 중 하나**가 **141.14.120.77**이므로, 아래와 같이 **서브넷 주소 (subnet address)**를 찾을 수 있다.

Address	→	141	.	14	.	120	.	77
Mask	→	255	.	255	.	192	.	0
Subnet Address	→	141	.	14	.	64	.	0

- ✓ **첫 번째, 두 번째, 네 번째 바이트**의 값은 AND 연산의 **첫 번째 빠른 방법 (the first short cut)을 사용**하여 계산할 수 있다.
- ✓ **세 번째 바이트**의 값은 AND 연산의 **두 번째 빠른 방법 (the second short cut)**을 사용하여 계산할 수 있다.

Address (120)	0	+	64	+	32	+	16	+	8	+	0	+	0	+	0
Mask (192)	128	+	64	+	0	+	0	+	0	+	0	+	0	+	0
Result (64)	0	+	64	+	0	+	0	+	0	+	0	+	0	+	0

[from TCP/IP Protocol Suite Fourth Edition]

5.2 클래스기반 주소지정 – 슈퍼네팅

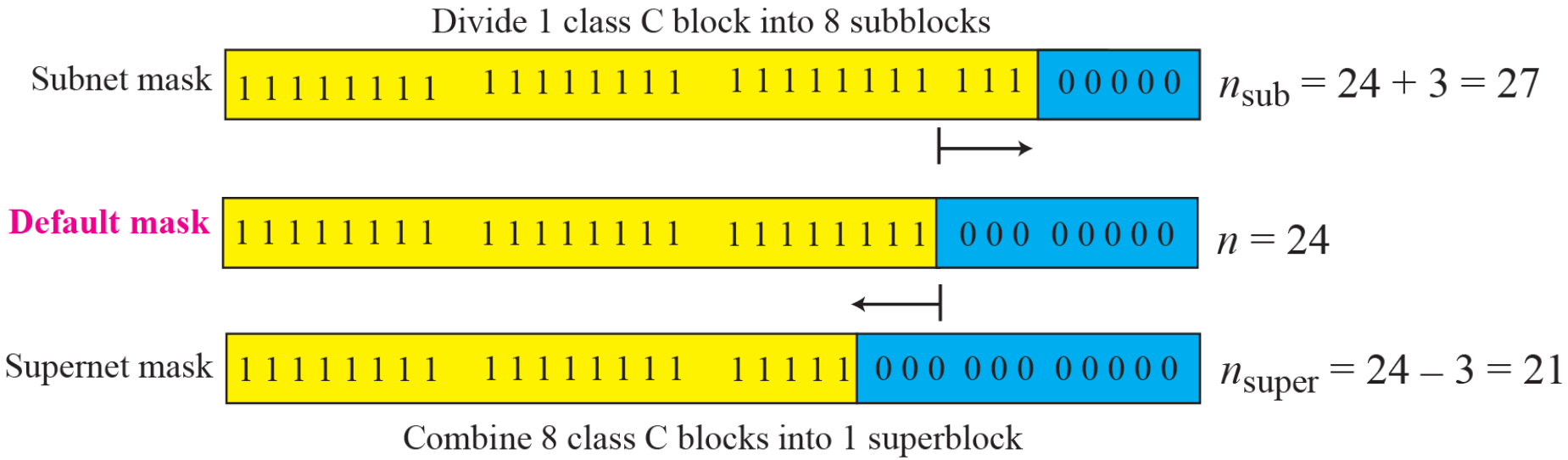


그림 5.26 서브넷, 디폴트, 슈퍼넷 마스크의 비교
(Comparison of subnet, default, and supernet mask)

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 (Classless Addressing)

- ✓ **클래스기반 주소지정 (classful addressing) 방식**에서 **서브네팅 (subnetting)**과 **수퍼네팅 (supernetting)**은 **주소 고갈 문제 (address depletion problem)**를 해결하지 **못하고** 있음

- ✓ 인터넷이 성장함에 따라, **좀 더 긴 주소 공간 (a larger address space)을 확보**하는 것이 **장기적인 해결책 (long-term solution)**임
- ✓ 비록 **IPv6**라고 하는 **장기적인 해결책 (long-range solution)**이 이미 고안되었지만, 동일한 주소 공간을 사용하면서도 기관들에게 균등하게 주소를 배분할 수 있도록 하기 위한, **단기적인 해결책 (short-term solution)**도 역시 **제시**되었음
- ✓ **단기적인 해결책**은 IPv4 주소를 그대로 사용하며, **클래스없는 주소지정 (classless addressing)**이라고 불림

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – Topics

1) 가변 길이 블록 (Variable-Length Blocks)
2) 2단계 주소 체계 (Two-Level Addressing)
3) 블록 할당 (Block Allocation)
4) 서브네팅 (Subnetting)

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 가변 길이 블록

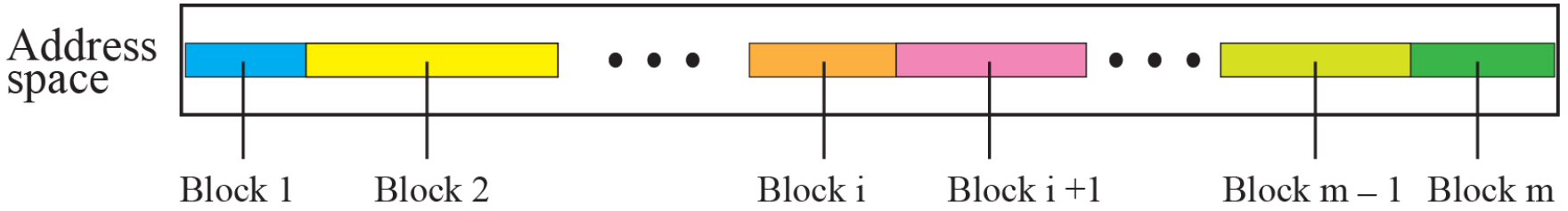


그림 5.27 클래스없는 주소지정 방식에서 가변 길이 블록
(Variable-length blocks in classless addressing)

클래스없는 주소지정 방식에서,
프리픽스 **(prefix)**는 네트워크를 정의하고
서픽스 **(suffix)**는 호스트를 정의한다.

클래스없는 주소지정 방식에서,
프리픽스의 길이는 **0**부터 **32**까지이다.

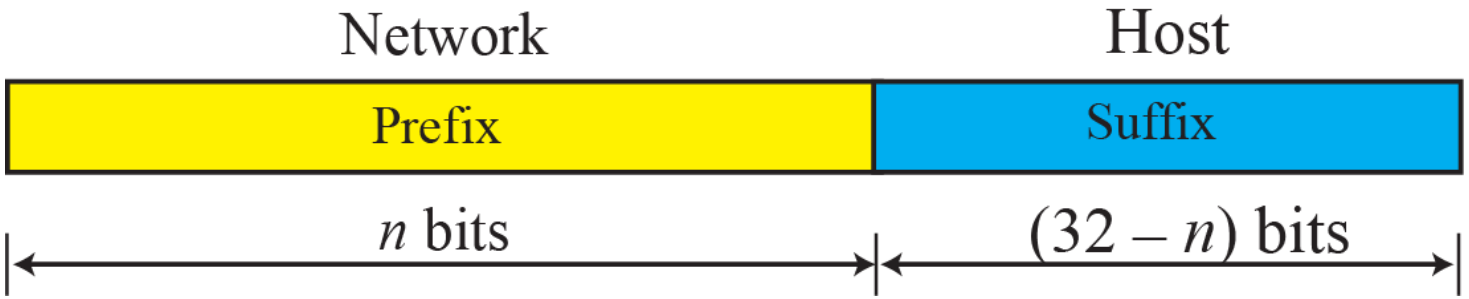


Figure 5.28 Prefix and suffix

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 가변 길이 블록 – Example 5.22

✓ 전체 Internet을 **4,294,967,296**개의 주소를 갖는 **하나의 단일 블록 (one single block)**이라고 하면, **프리픽스 길이 (prefix length)**와 **서픽스 길이 (suffix length)**는 얼마인가?

해답 (Solution)

✓ 이 경우, **프리픽스 길이는 0**이고, **서픽스 길이는 32**이다.

✓ **32개의 모든 비트**가 단일 블록에서, **2^{32} = 4,294,967,296개의 호스트를 구분**하기 위하여 사용된다.

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 가변 길이 블록 – Example 5.23

✓ Internet이 **4,294,967,296** 개의 블록으로 나누어지고, 각각의 블록은 **단지 하나의 주소만 (one single address)**을 포함하는 경우, **프리픽스 길이 (prefix length)**와 **서픽스 길이 (suffix length)**는 얼마인가?

해답 (Solution)

✓ 이 경우, 각 블록의 **프리픽스 길이는 32**이고, **서픽스 길이는 0**이다.

✓ **2^{32} = 4,294,967,296 개의 블록**을 나타내기 위하여, **32개의 비트**가 다 필요하다.

✓ 각 블록에 포함되는 단일 주소는 블록 자체를 정의한다.

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 가변 길이 블록 – Example 5.24

✓ 블록 내에 있는 **주소의 수**는 **프리픽스의 길이 (prefix length)**인 ***n***과 **역 (inverse)의 관계**를 가진다.

- ***n***이 **작다**는 것은 **블록의 크기가 크다**는 것을 의미하며,
- ***n***이 **크다**는 것은 **블록의 크기가 작다**는 것을 의미한다.

클래스없는 주소지정 방식에서, 블록 정보를 알기 위해서는 블록 내의 주소 하나 **(one of the addresses in the block)**와 프리픽스 길이 **(prefix length)**를 알아야 한다.

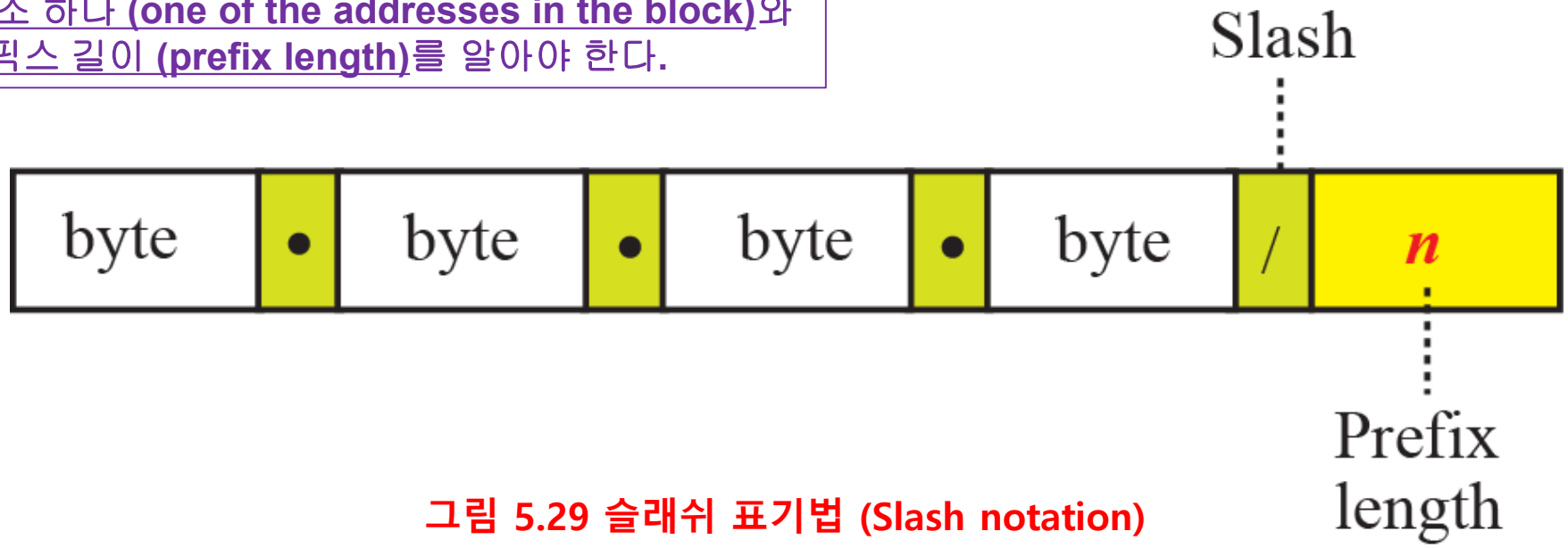


그림 5.29 슬래쉬 표기법 (Slash notation)

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 가변 길이 블록 – Example 5.25

✓ 클래스없는 주소지정 방식에서는 **주소 정보만을 이용하여** 주소가 속하는 **블록을 알 수 없다**.

- 예를 들어, 주소 **230.8.24.56**은 여러 개의 블록에 속할 수 있다.
- 다음은 **이 주소가 포함될 수 있는 여러 블록**과 이 **블록의 프리픽스 길이**를 보여준다.

Prefix length:16	→	Block:	230.8.0.0	to	230.8.255.255
Prefix length:20	→	Block:	230.8.16.0	to	230.8.31.255
Prefix length:26	→	Block:	230.8.24.0	to	230.8.24.63
Prefix length:27	→	Block:	230.8.24.32	to	230.8.24.63
Prefix length:29	→	Block:	230.8.24.56	to	230.8.24.63
Prefix length:31	→	Block:	230.8.24.56	to	230.8.24.57

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 2단계 주소 체계 – Example 5.26

✓ 다음은 **슬래쉬 표기법 (slash notations)**으로 표시된 **주소**들이다.

a. 주소 12.23.24.78**/8**에서 네트워크 마스크 (network mask)는 **255.0.0.0**이다.
네트워크 마스크는 8개의 1과 24개의 0으로 구성된다.
프리픽스 길이 (prefix length)는 8이고, 서픽스 길이 (suffix length)는 24이다.

b. 주소 130.11.232.156**/16**에서 네트워크 마스크 (network mask)는 **255.255.0.0**이다.
네트워크 마스크는 16개의 1과 16개의 0으로 구성된다.
프리픽스 길이 (prefix length)는 16이고, 서픽스 길이 (suffix length)는 16이다.

c. 주소 167.199.170.82**/27**에서 네트워크 마스크 (network mask)는 **255.255.255.224**이다.
네트워크 마스크는 27개의 1과 5개의 0으로 구성된다.
프리픽스 길이 (prefix length)는 27이고, 서픽스 길이 (suffix length)는 5이다.

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 2단계 주소 체계 – Example 5.27 (1)

✓ **167.199.170.82/27**은 블록에 속하는 하나의 주소이다.

✓ 이 네트워크에 속하는 **주소의 개수**와 **첫 번째 주소 (first address)**와 **마지막 주소 (last address)**를 구하라.

Solution

✓ n 의 값은 **27**이다.

✓ 네트워크 마스크는 27개의 1과 5개의 0으로 구성된다.

✓ 이것을 점-10진 표기법으로 표기하면 **255.255.255.224**이다.

a. 네트워크에 속하는 주소의 개수는 $2^{32-n} = 2^{32-27} = 2^5 = 32$ 이다.

b. 첫 번째 주소 (네트워크 주소)를 찾기 위하여 AND 계산을 수행한다.

첫 번째 주소는 **167.199.170.64/27**이다.

Address in binary:	10100111 11000111 10101010 01010010
Network mask:	11111111 11111111 11111111 11100000
First address:	10100111 11000111 10101010 01000000

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 2단계 주소 체계 – Example 5.27 (2)

c. **마지막 주소**를 찾기 위해서는 **네트워크 마스크에 보수 (complement of the network mask)**를 취한 후, 그 값과 주소를 OR 계산한다.

마지막 주소는 **167.199.170.95/27**이다.

Address in binary:	10100111	11000111	10101010	01010010
Complement of network mask:	00000000	00000000	00000000	00011111
Last address:	10100111	11000111	10101010	01011111

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 2단계 주소 체계 – Example 5.28

✓ **17.63.110.114/24**는 블록에 속하는 하나의 주소이다.

✓ 이 블록에 속하는 **주소의 개수**와 **첫 번째 주소**, **마지막 주소**를 구하라.

해답 (Solution)

✓ 네트워크 마스크는 255.255.255.0이다.

a. 네트워크에 속하는 주소의 개수는 $2^{32-24} = 256$이다.

b. **첫 번째 주소**를 찾기 위하여, 이 장의 앞에서 설명했던 **빠른 방법 (short cut methods)**을 사용한다.

첫 번째 주소는 **17.63.110.0/24** 이다.

Address:	17	.	63	.	110	.	114
Network mask:	255	.	255	.	255	.	0
First address (AND):	17	.	63	.	110	.	0

c. **마지막 주소**를 찾기 위하여, 네트워크 마스크에 보수 (complement of the network mask)와 앞에서 언급한 첫 번째 빠른 방법을 이용한다.

마지막 주소는 **17.63.110.255/24** 이다.

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 2단계 주소 체계 – Example 5.29 (1)

✓ **110.23.120.14/20**은 블록에 속하는 하나의 주소이다.

✓ 이 블록에 속하는 **주소의 개수**와 **첫 번째 주소**, **마지막 주소**를 구하라.

해답 (Solution)

✓ 네트워크 마스크는 **255.255.240.0**이다.

a. 네트워크에 속하는 주소의 개수는 $2^{32-20} = 4096$이다.

b. **첫 번째 주소**를 찾기 위하여, 첫 번째, 두 번째, 네 번째 바이트에 첫 번째 빠른 방법 (first short cut)을 사용하고, 세 번째 바이트에는 두 번째 빠른 방법 (second short cut)을 사용한다.

첫 번째 주소는 **110.23.112.0/20**이다.

Address:	110	.	23	.	120	.	14
Network mask:	255	.	255	.	240	.	0
First address (AND):	110	.	23	.	112	.	0

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 2단계 주소 체계 – Example 5.29 (2)

c. **마지막 주소**를 찾기 위하여, 첫 번째, 두 번째, 네 번째 바이트에 첫 번째 빠른 방법 (first short cut)을 사용하고, 세 번째 바이트에는 두 번째 빠른 방법 (second short cut)을 사용한다.

네트워크 마스크에 보수 (complement of the mask)를 취한 값과 OR 계산을 수행한다.

마지막 주소는 **110.23.127.255/20**이다.

Address:	110	.	23	.	120	.	14
Network mask:	0	.	0	.	15	.	255
Last address (OR):	110	.	23	.	127	.	255

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 블록 할당 – Example 5.30

✓ ISP는 **1,000개의 주소를 갖는 블록**을 요청하였다.

✓ 할당된 블록은 다음과 같다.

a. **1,000이 2의 거듭제곱이 아니**기 때문에, **1,024(1,024 = 2^{10})개의 주소가 할당**된다.

b. 블록의 프리픽스 길이는 $n = 32 - \log_2 1024 = 22$이다.

c. **18.14.12.0 (1,024로 나누어지는 값)**를 **시작 주소로 선택**하였다.

✓ **할당된 블록 (granted block)**은 **18.14.12.0/22**이다.

✓ **첫 번째 주소**는 **18.14.12.0/22**이며, **마지막 주소**는 **18.14.15.255/22**이다.

5.3 클래스없는 주소지정 – 블록 할당 – Example 5.31

✓ **73.0.0.0**의 **클래스 A 블록**을 할당받은 기관은 **클래스없는 주소지정에서 73.0.0.0/8 블록을 할당**받은 것으로 **간주**한다.

Table 5.1 *Prefix length for classful addressing*

Class	*Prefix length*	*Class*	*Prefix length*
A	/8	D	/4
B	/16	E	/4
C	/24		

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 서브네팅 – Example 5.32 (1)

- ✓ **130.34.12.64/26 블록**이 기관에게 할당되었다.
- ✓ 기관은 **각각이 동일한 개수**의 호스트를 갖는 **네 개의 서브네트워크를 구성**하고자 한다.
- ✓ 서브네트워크를 **설계**하고, 각각의 서브네트워크에 대한 **정보를 구하여라**.

해답 (Solution)

- ✓ 전체 네트워크에 **할당된 주소의 개수 N**은 $2^{32-26} = 64$이다.
- ✓ 네트워크의 **첫 번째 주소**는 **130.34.12.64/26**이고, **마지막 주소**는 **130.34.12.127/26**이다.
- ✓ 이제 서브네트워크를 설계해 보자.

1. **첫 번째 조건을 만족**하도록, 각각의 서브네트워크에게 **16 (16은 2의 거듭제곱)개의 주소를 할당**한다.
2. 각각의 서브네트워크를 위한 **서브네트워크 마스크**는 다음과 같다.

$$n_1 = n_2 = n_3 = n_4 = n + \log_2 (N/N_i) = 26 + \log_2 4 = 28$$

(다음 페이지 계속)

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 서브네팅 – Example 5.32 (2)

3. **각각의 서브네트워크에** 첫 번째 주소부터 시작하여 **16개의 주소를 할당**한다.
   그림 5.30은 각각의 서브넷을 위한 부-블록 (subblock)을 보여준다.
   각각의 서브네트워크에서 **첫 번째 주소는** 그 서브네트워크에 속하는 **주소의 개수로 나누어질 수 있어야 한다**.

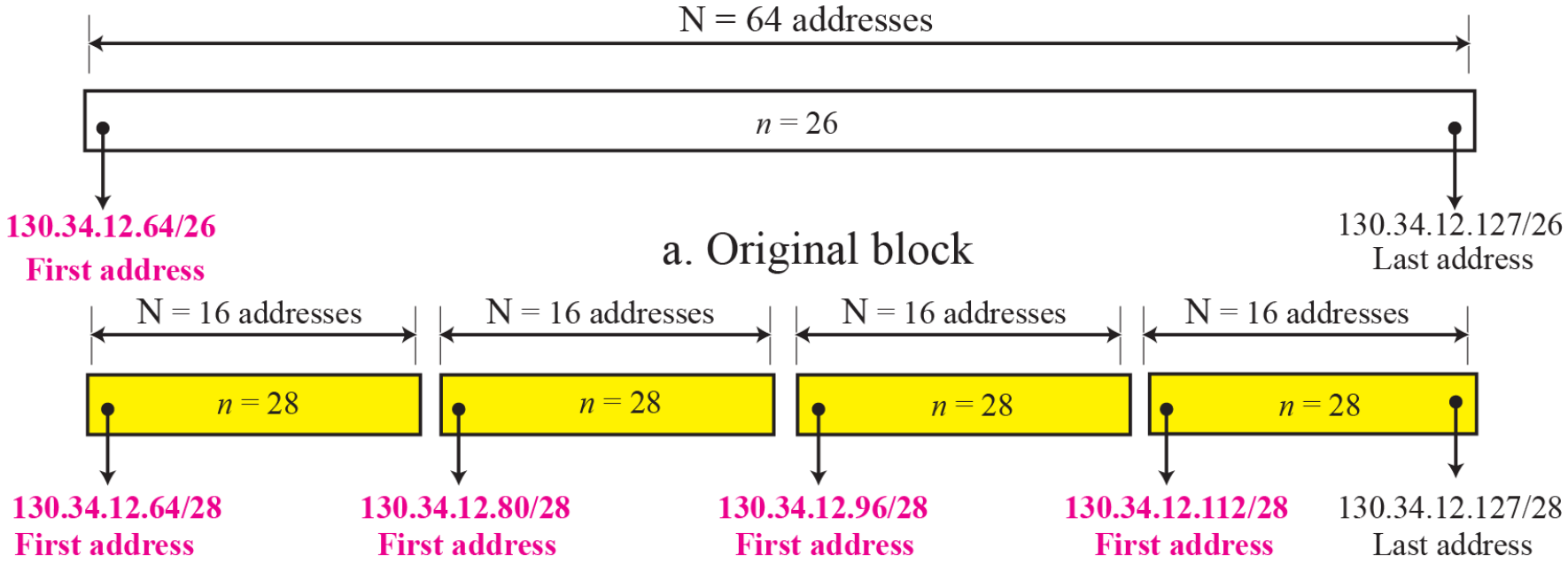


그림 5.30 예제 5.32의 답

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 서브네팅 – Example 5.33 (1)

- ✓ **14.24.74.0/24라는 시작 주소**를 갖는 하나의 주소 블록이 기관에게 할당되었다.
- ✓ 기관은 **다음과 같은 개수의 주소**를 갖는 **3개의 부-블록 (subblock)**을 구성하고자 한다.
  - ❑ **120** 개의 주소를 갖는 하나의 부-블록 (subblock)
  - ❑ **60** 개의 주소를 갖는 하나의 부-블록 (subblock)
  - ❑ **10** 개의 주소를 갖는 하나의 부-블록 (subblock)

해답 (Solution)

- ✓ 이 블록에는 $2^{32-24} = 256$**개의 주소**가 있다.
- ✓ **첫 번째 주소**는 **14.24.74.0/24**이고, **마지막 주소**는 **14.24.74.255/24**이다.

a. **첫 번째 부-블록**에 속하는 **주소의 개수**는 **2의 거듭제곱이 아니다**.
따라서 **128개의 주소**가 할당되고, **서브넷 마스크**는 **25**이다.
첫 번째 주소는 **14.24.74.0/25**이고, **마지막 주소**는 **14.24.74.127/25**이다.

(다음 페이지 계속)

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 서브네팅 – Example 5.33 (2)

b. **두 번째 부-블록**에 속하는 **주소의 개수**는 **2의 거듭제곱이 아니다**.
따라서 **64개의 주소**가 할당되고, **서브넷 마스크**는 **26**이다.
첫 번째 주소는 **14.24.74.128/26**이고, **마지막 주소**는 **14.24.74.191/26**이다.

c. **세 번째 부-블록**에 속하는 **주소의 개수**는 **2의 거듭제곱이 아니다**.
따라서 **16개의 주소**가 할당되고, **서브넷 마스크**는 **28**이다.
첫 번째 주소는 **14.24.74.192/28**이고, **마지막 주소**는 **14.24.74.207/28**이다.

d. 세 개의 부-블록에 할당된 모든 주소를 다 더하면 결과는 208이 되며, **48개의 주소가 남아있다**.
이 범위의 **첫 번째 주소**는 **14.24.74.208**이고, 마지막 주소는 **14.24.74.255**이다.

e. 그림 5.31은 블록이 어떻게 구성되었는지를 보여준다.
이 그림에서는 각각의 블록에 대한 첫 번째 주소를 볼 수 있다.

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 서브네팅 – Example 5.33 (3)

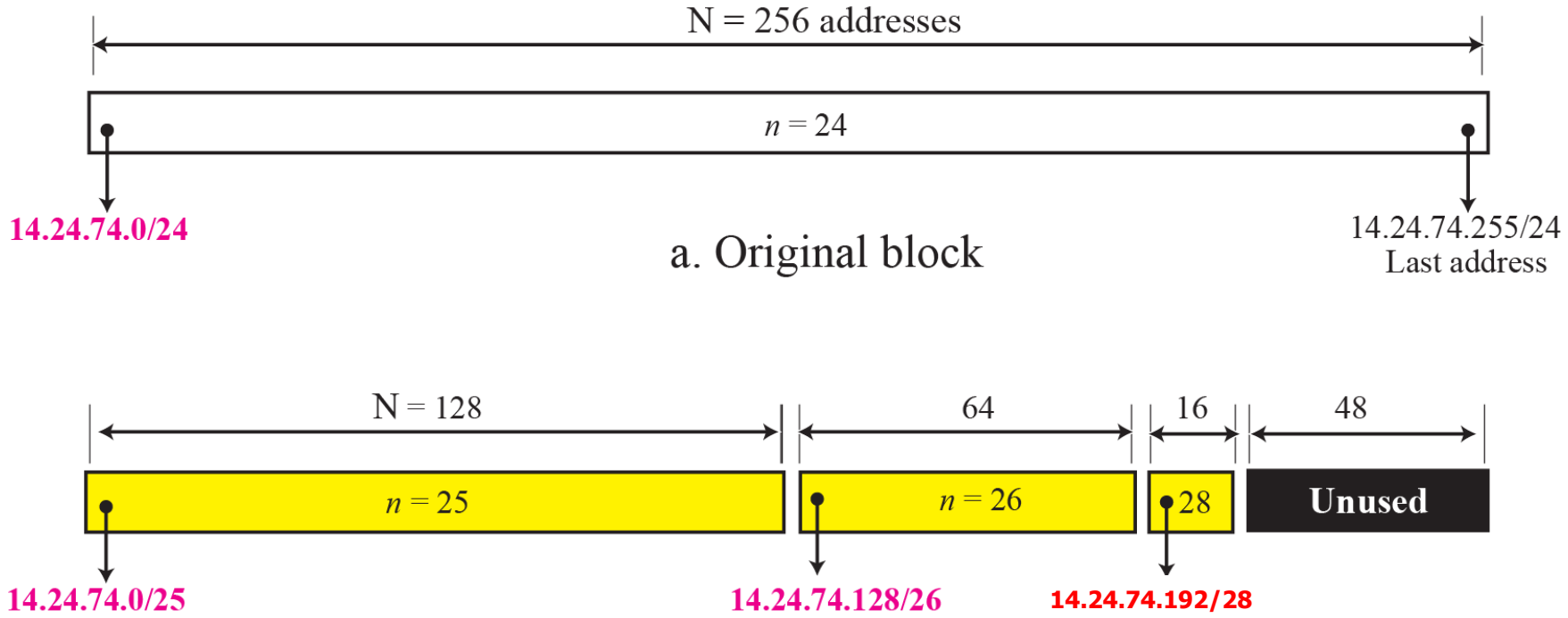


그림 5.31 예제 5.33의 답

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 서브네팅 – Example 5.34 (1)

- ✓ 어떤 회사가 중앙, 동쪽, 서쪽에 3개의 사무실을 가지고 있다고 가정해보자.
- ✓ 중앙 사무실 (**Central office**)은 사설 (private) WAN 선로를 이용하여 동쪽과 서쪽 사무실 (**East and West offices**)과 연결되어 있다.
- ✓ 이 회사가 **할당 받은 블록**은 **70.12.100.128/26**이라는 시작 주소를 갖는 **64개의 주소**로 이루어진다.
- ✓ 관리자는 **중앙 사무실에 32개의 주소를 할당**하고, **나머지 주소**를 **다른 두 개의 사무실에 할당**하기로 결정하였다.

1. 주소는 다음과 같이 할당되었다.

Central office $N_c = 32$	East office $N_e = 16$	West office $N_w = 16$

2. 각각의 서브네트워크를 위한 프리픽스 길이는 다음과 같다.

$n_c = n + \log_2(64/32) = 27$	$n_e = n + \log_2(64/16) = 28$	$n_w = n + \log_2(64/16) = 28$

(다음 페이지 계속)

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 서브네팅 – Example 5.34 (2)

3. 그림 5.32는 관리자에 의해서 설계된 구성도를 보여준다.

중앙 사무실이 사용한 주소는 **70.12.100.128/27**부터 **70.12.100.159/27**까지 이다.

회사는 이 주소 중에서 세 개의 주소를 라우터에 설정하기 위하여 사용하였고, 마지막 주소를 (다른 목적을 위하여) 예약해 놓았다.

동쪽 사무실은 **70.12.100.160/28**부터 **70.12.100.175/28**까지의 주소를 사용한다.

회사는 이 주소 중에서 한 개의 주소를 라우터에 설정하기 위하여 사용하였고, 마지막 주소를 (다른 목적을 위하여) 예약해 놓았다.

서쪽 사무실은 **70.12.100.176/28**부터 **70.12.100.191/28**까지의 주소를 사용한다.

회사는 이 주소 중에서 한 개의 주소를 라우터에 설정하기 위하여 사용하였고, 마지막 주소를 (다른 목적을 위하여) 예약해 놓았다.

WAN의 점-대-점 연결 (point-to-point connections)에는 주소가 필요하지 않다.

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 서브네팅 – Example 5.34 (3)

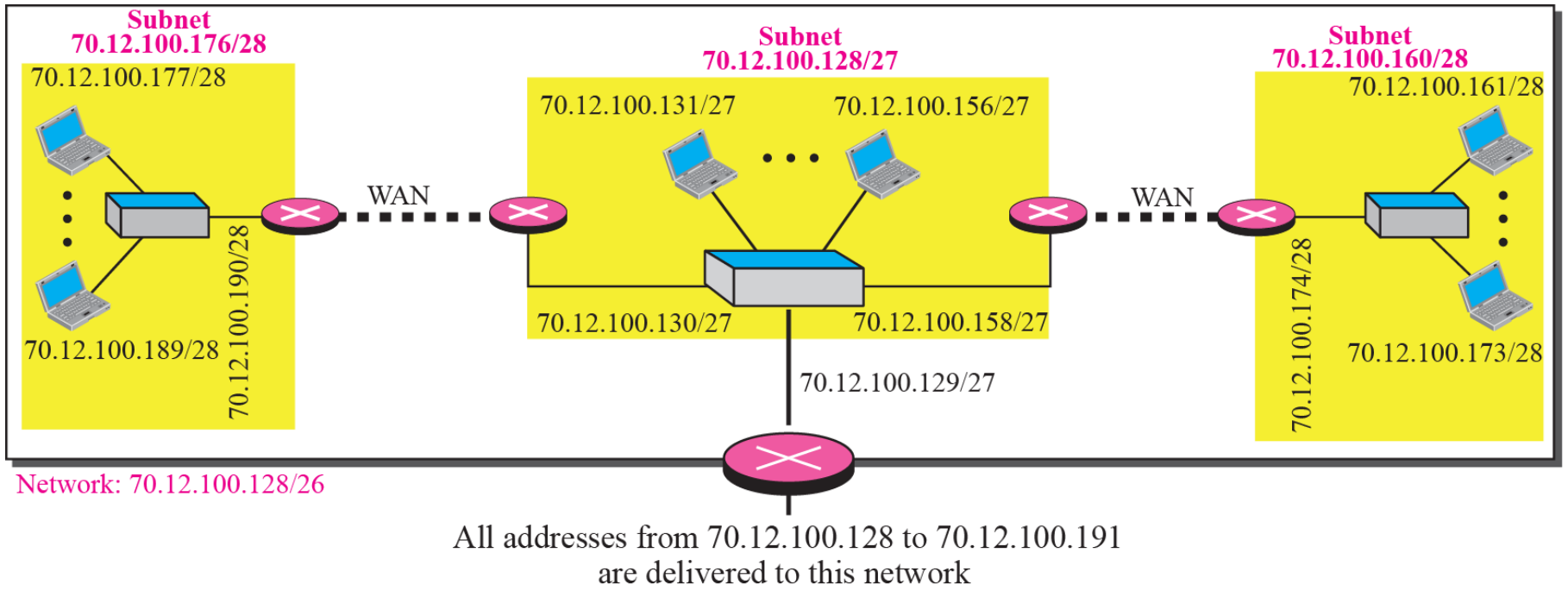


그림 5.32 예제 5.34의 답

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 서브네팅 – Example 5.35 (1)

- ✓ ISP가 **190.100.0.0/16 (65,536 주소)**부터 시작되는 주소 블록을 할당받았다.
- ✓ ISP는 이 주소 블록을 다음과 같은 **세 그룹**의 고객들에게 배분하고자 한다.
  - ❑ 첫 번째 그룹은 **64개의 고객**을 갖고 있으며, 각각은 약 **256개의 주소**를 필요로 한다.
  - ❑ 두 번째 그룹은 **128개의 고객**을 갖고 있으며, 각각은 약 **128개의 주소**를 필요로 한다.
  - ❑ 세 번째 그룹은 **128개의 고객**을 갖고 있으며, 각각은 약 **64개의 주소**를 필요로 한다.
- ✓ 부-블록을 설계하라. 각각의 부-블록에게 주소를 할당한 후에 얼마나 많은 수의 주소가 사용 가능한가?

해답 (Solution)

(다음 페이지 계속)

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 서브네팅 – Example 5.35 (2)

해답 (Solution)

- ✓ 다음의 **두 단계로 문제를 해결**하고자 한다.
- ✓ **첫 번째 단계**에서는 주소의 부-블록을 각각의 그룹에 할당한다.
- ✓ 각각의 그룹에 할당된 주소의 전체 개수와 각각의 부-블록을 위한 프리픽스 길이는 다음과 같다.

Group 1: $64 \times 256 = 16{,}384$	$n_1 = 16 + \log_2 (65536/16384) = 18$
Group 2: $128 \times 128 = 16{,}384$	$n_2 = 16 + \log_2 (65536/16384) = 18$
Group 3: $128 \times 64 = 8192$	$n_3 = 16 + \log_2 (65536/8192) = 19$

- ✓ 그림 5.33은 **첫 단계**를 위한 설계를 보여준다.
- ✓ 그림 5.34는 **두 번째 단계**를 보여준다.
- ✓ 여기에서, 각 고객을 위한 **첫 번째 주소**는 **서브넷 주소 (subnet address)**로 사용하였고, **마지막 주소**는 **특수 주소 (special address)** 목적으로 예약해 두었다.

(다음 페이지 계속)

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 서브네팅 – Example 5.35 (3)

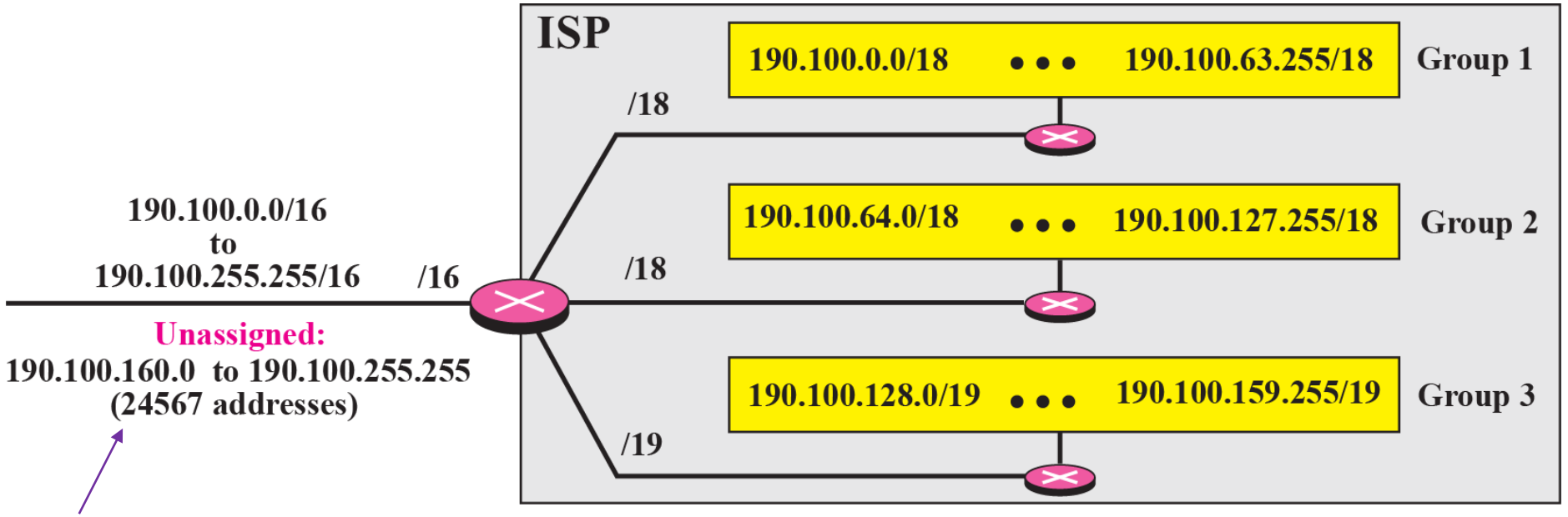


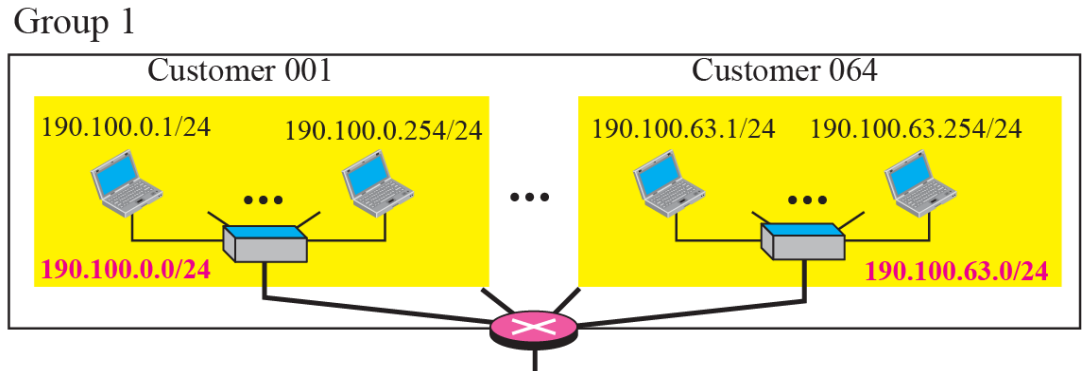
그림 5.33 예제 5.35의 답: 첫 번째 단계

[from TCP/IP Protocol Suite Fourth Edition]

5.3 클래스없는 주소지정 – 서브네팅 – Example 5.35 (4)

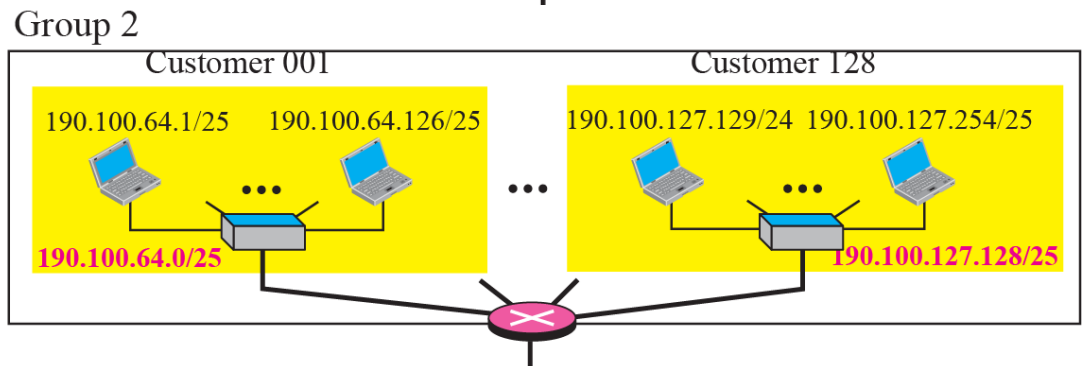
Group: $n = 18$

Subnet: $n = 18 + \log_2 (16385/256) = 24$



Group: $n = 18$

Subnet: $n = 18 + \log_2 (16385/128) = 25$



Group: $n = 19$

Subnet: $n = 19 + \log_2 (8192/64) = 26$

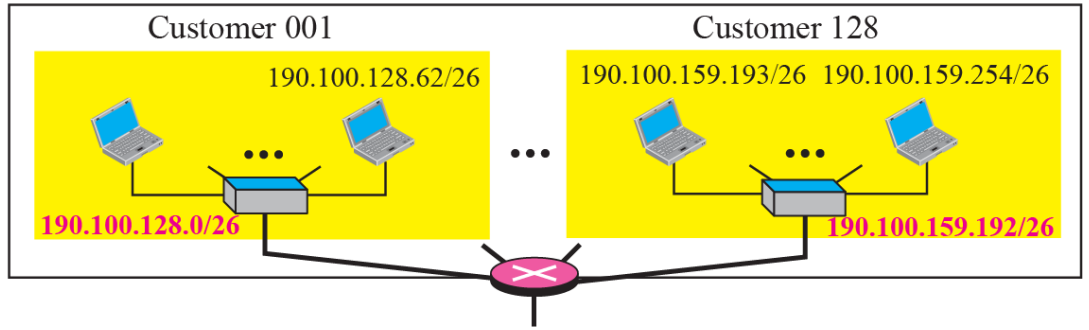


그림 5.34 예제 5.35의 답: 두 번째 단계

[from TCP/IP Protocol Suite Fourth Edition]

5.4 특수 주소 (Special Addresses)

- ✓ **클래스기반 주소지정 방식**에서 **일부의 주소 (some addresses)**는 **특수 목적 (special purposes)을 위하여 예약**되었음

- ✓ **클래스없는 주소지정 방식**에서도, 클래스기반 주소지정 방식과 **동일한 목적의 특수 주소들이 정의**됨

[from TCP/IP Protocol Suite Fourth Edition]

5.4 특수 주소 (Special Addresses) – Topics

1) 특수 블록 (Special Blocks)

2) 블록에 속하는 특수 주소 (Special Addresses in Each Block)

[from TCP/IP Protocol Suite Fourth Edition]

5.4 특수 주소 (Special Addresses) – 특수 블록

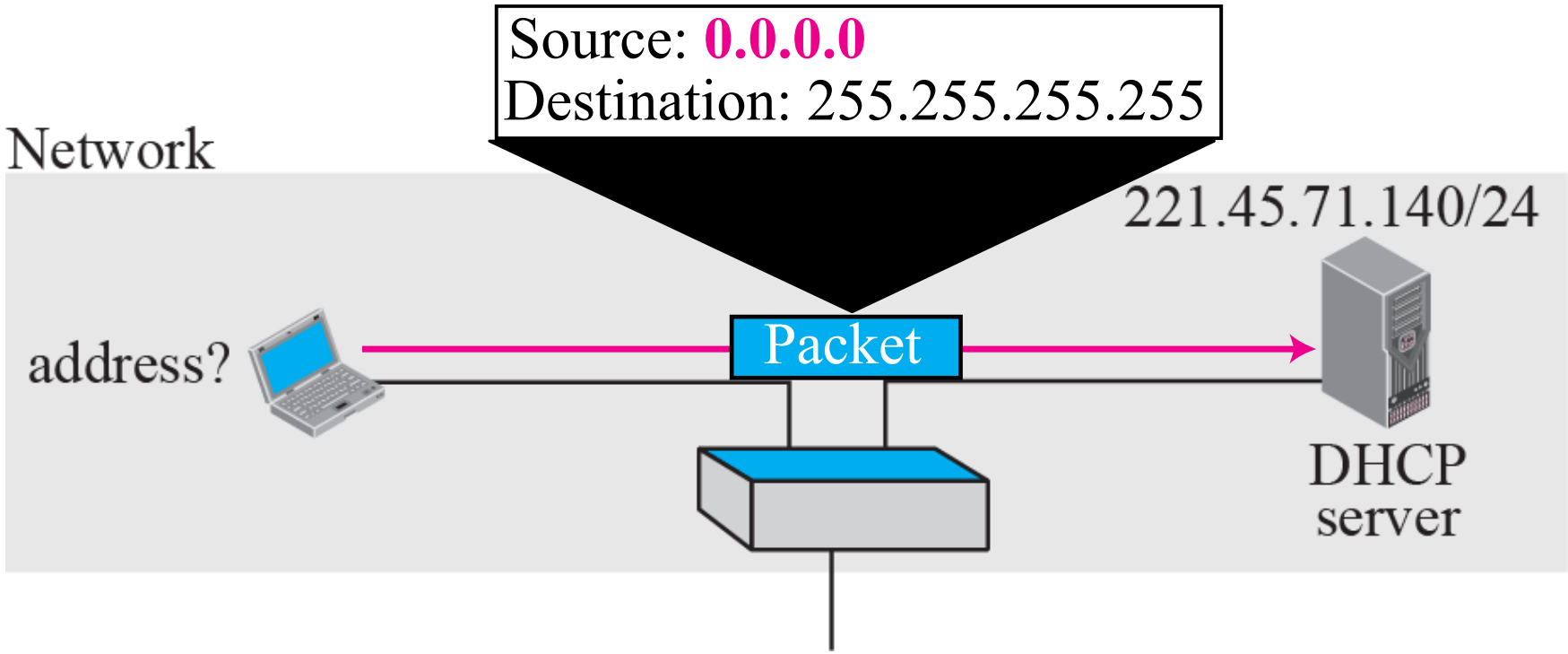


그림 5.35 all-zero 주소의 이용 예
(Example of using the all-zero address)

[from TCP/IP Protocol Suite Fourth Edition]

5.4 특수 주소 (Special Addresses) – 블록에 속하는 특수 주소

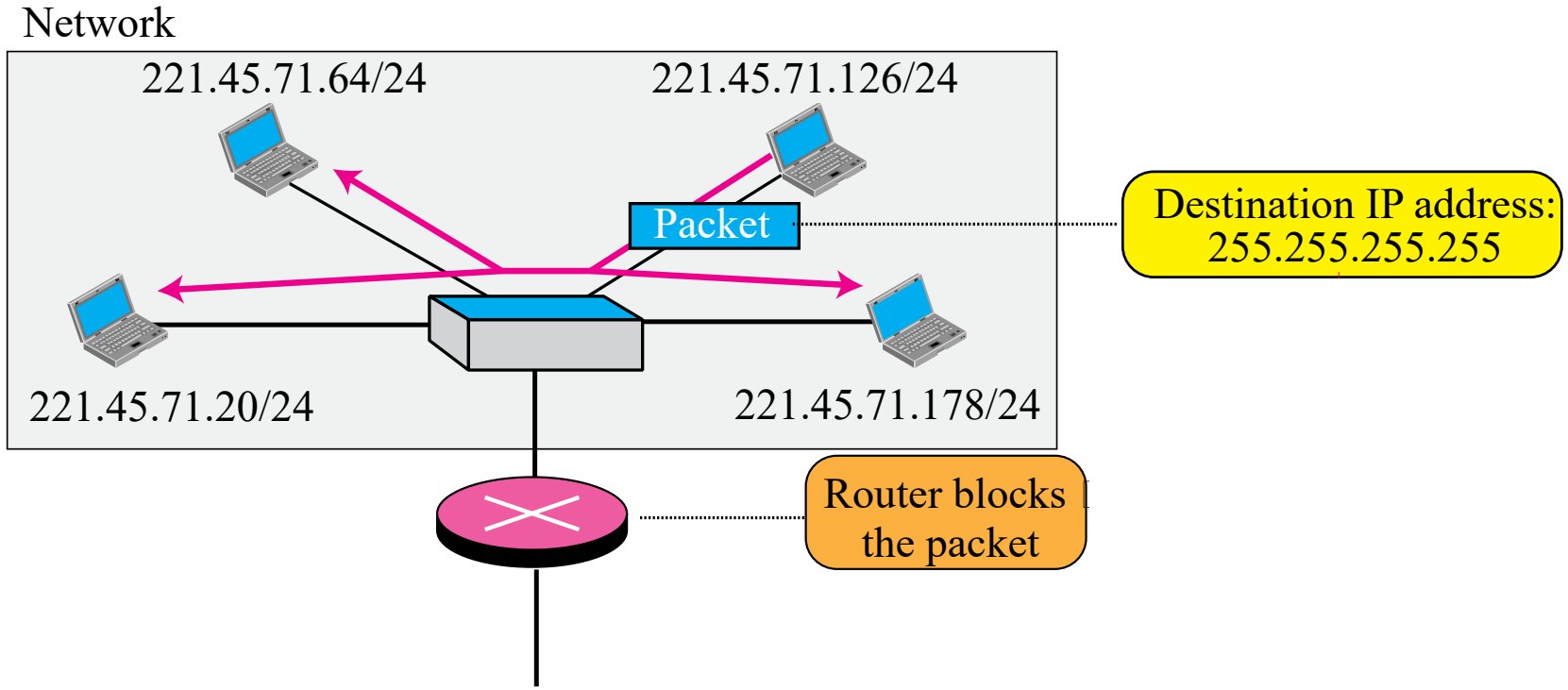


그림 5.36 제한된 브로드캐스트 주소의 예
(Example of limited broadcast address)

[from TCP/IP Protocol Suite Fourth Edition]

5.4 특수 주소 (Special Addresses) – 특수 블록

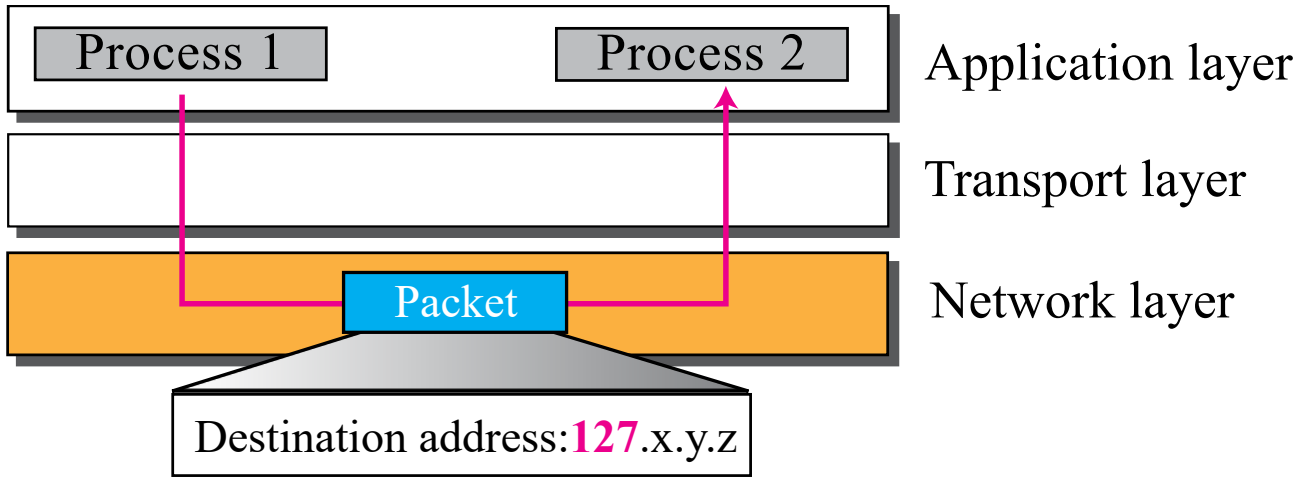


그림 5.37 루프백 주소의 예
(Example of loopback address)

[from TCP/IP Protocol Suite Fourth Edition]

5.4 특수 주소 (Special Addresses) – 블록에 속하는 특수 주소

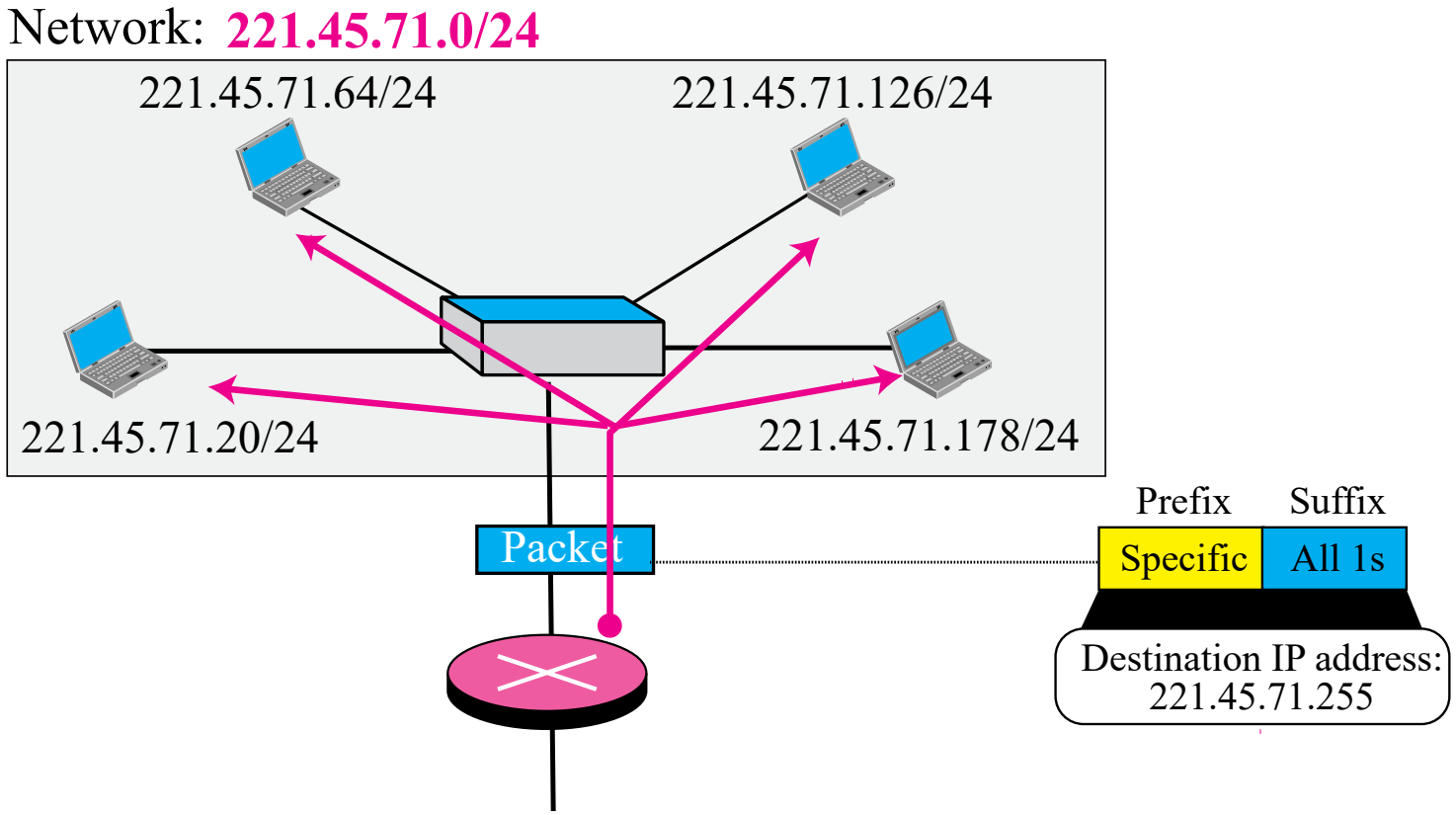


그림 5.38 직접 브로드캐스트 주소
(Example of a directed broadcast address)

[from TCP/IP Protocol Suite Fourth Edition]

5.4 특수 주소 (Special Addresses) – 특수 블록

Table 5.2 *Addresses for private networks*

Block	*Number of addresses*	*Block*	*Number of addresses*
10.0.0.0/8	16,777,216	192.168.0.0/16	65,536
172.16.0.0/12	1,047,584	169.254.0.0/16	65,536

[from TCP/IP Protocol Suite Fourth Edition]

5.5 NAT

- ✓ ISP를 통하여 **주소를 배분 (distribution of addresses)**하는 것은 **새로운 문제 (new problem)를 야기**함
- ✓ 사업이 번창하거나 또는 가정집에서 좀 더 큰 범위를 필요로 하는 경우, **ISP가 주소들을 다른 네트워크에 이미 할당하였다면** 이러한 요청을 **받아들일 수 없을 것**임
- ✓ 그러나, **대부분의 경우**, 하나의 네트워크에 속하는 **컴퓨터들의 일부분만**이 **동시에 (simultaneously) Internet과의 접속을 필요**로 함
- ✓ 이러한 경우, 적용 가능한 기술이 **네트워크 주소 변환 (NAT: Network Address Translation)**

[from TCP/IP Protocol Suite Fourth Edition]

5.5 NAT – Topics

1) 주소 변환 (Address Translation)

2) 변환 테이블 (Translation Table)

[from TCP/IP Protocol Suite Fourth Edition]

5.5 NAT – 주소 변환

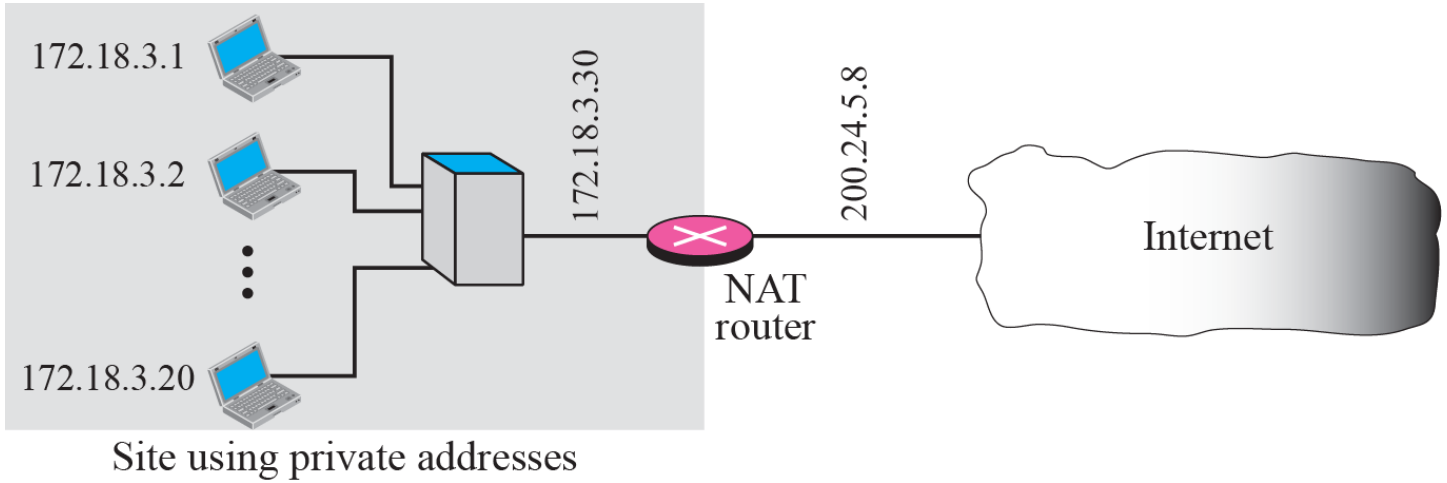


Figure 5.39 NAT

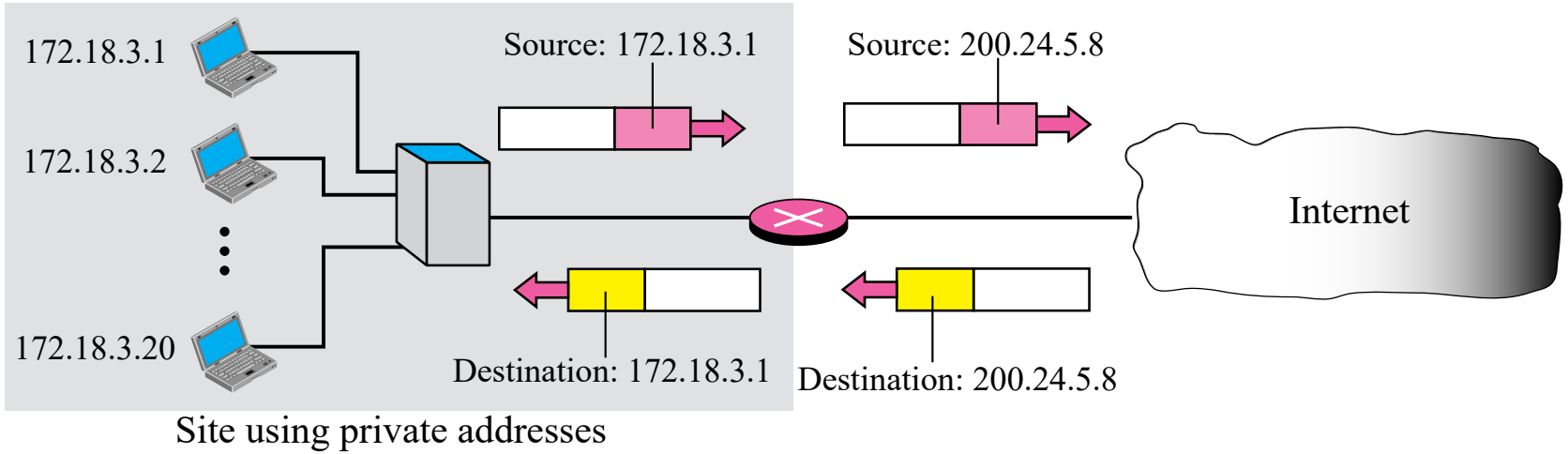


Figure 5.40 Address translation

[from TCP/IP Protocol Suite Fourth Edition]

5.5 NAT – 변환 테이블

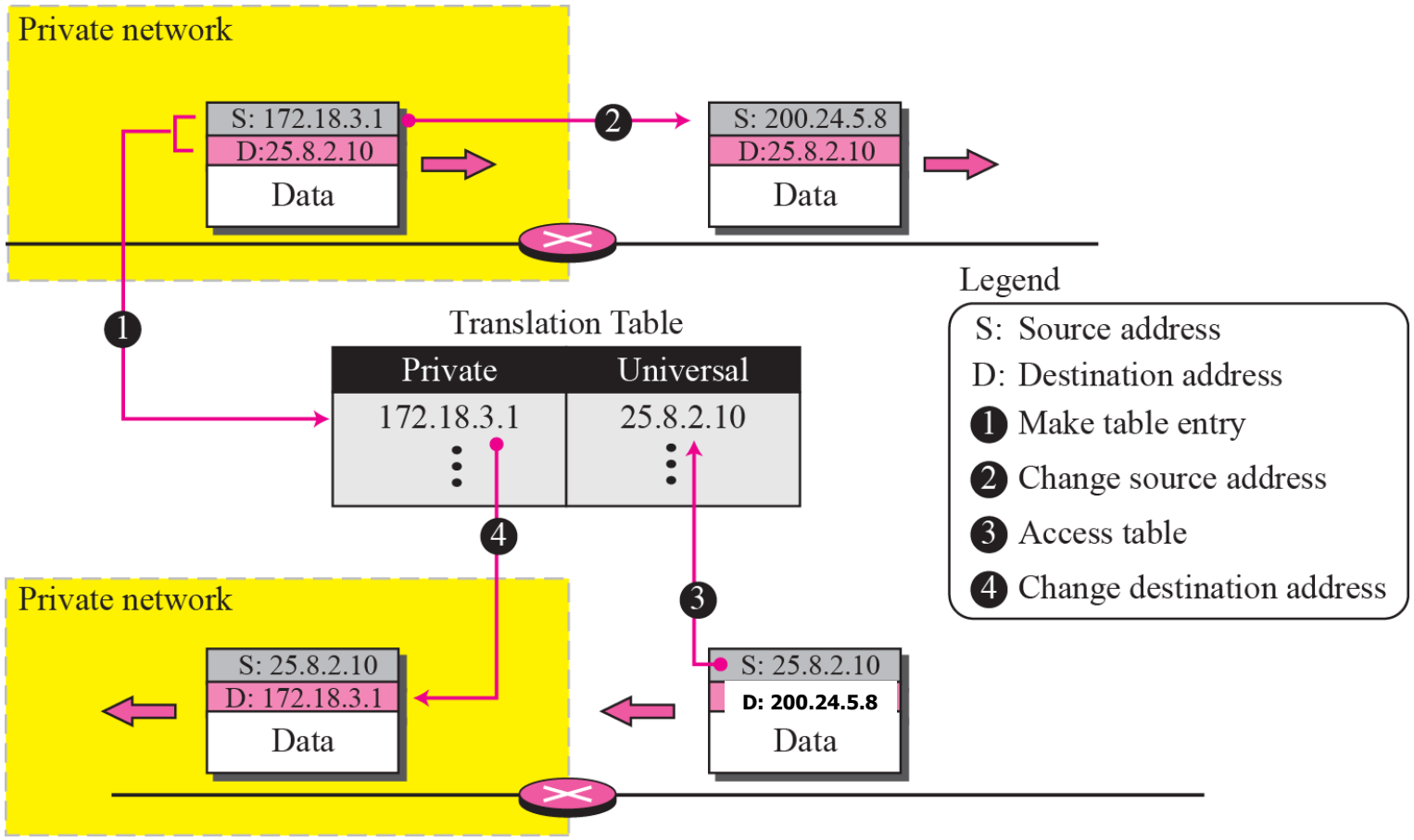


Figure 5.41 Translation

[from TCP/IP Protocol Suite Fourth Edition]

5.5 NAT – 변환 테이블

Table 5.3 *Five-column translation table*

Private Address	*Private Port*	*External Address*	*External Port*	*Transport Protocol*
172.18.3.1	1400	25.8.3.2	80	TCP
172.18.3.2	1401	25.8.3.2	80	TCP
…	…	…	…	…

[from TCP/IP Protocol Suite Fourth Edition]