Chapter 5: IPv4 Addresses

Exercises

1. What is the address space in each of the following systems?
a. a system with 8-bit addresses
Ans: 2^8 = **256 addresses**
b. a system with 16-bit addresses
Ans: 2^{16} = **65,536 addresses**
c. a system with 64-bit addresses
Ans: 2^{64} = **$1.844674407 \times 10^{19}$ addresses**
2. An address space has a total of 1,024 addresses. How many bits are needed to represent an address?
Ans: $\log_2(1024)$ = **10 bits**
3. An address space uses three symbols: 0, 1, and 2 to represent addresses. If each address is made of
10 symbols, how many addresses are available in this system?
Ans: 3^{10} = **59,049 addresses**
4. Change the following IP addresses from dotted-decimal notation to binary notation:
a. 114.34.2.8
Ans: 01110010 00100010 00000010 00001000
b. 129.14.6.8
c. 208.34.54.12
d. 238.34.2.1
In python, write `bin(129)` for conversion!
5. Change the following IP addresses from dotted-decimal notation to hexadecimal notation:
a. 114.34.2.8
b. 129.14.6.8
c. 208.34.54.12
d. 238.34.2.1
In python, write `hex(129)` for conversion!
6. Change the following IP addresses from hexadecimal notation to binary notation:
a. 0x1347FEAB
b. 0xAB234102
c. 0x0123A2BE
d. 0x00001111
In python, write `bin(0x1347FEAB)` for conversion!
7. How many hexadecimal digits are needed to define the netid in each of the following classes?
a. Class A
Ans: 2 hex digits (8 bits)
b. Class B
Ans: 4 hex digits (16 bits)
12c. Class C
Ans: 6 hex digits (24 bits)
8. Change the following IP addresses from binary notation to dotted-decimal notation:
a. 01111111 11110000 01100111 01111101
Ans: 127.240.103.125
b. 10101111 11000000 11111000 00011101
c. 11011111 10110000 00011111 01011101
d. 11101111 11110111 11000111 00011101
In python, write `int(0b10101111)` for conversion!

9. Find the class of the following IP addresses:
a. 208.34.54.12
Ans: Class C
b. 238.34.2.1
Ans: Class D
c. 242.34.2.8
Ans: Class E
d. 129.14.6.8
Ans: Class B
10. Find the class of the following IP addresses:
a. 11110111 11110011 10000111 11011101
Ans: Class E
b. 10101111 11000000 11110000 00011101
Ans: Class B
c. 11011111 10110000 00011111 01011101
Ans: Class C
d. 11101111 11110111 11000111 00011101
Ans: Class D
11. Find the netid and the hostid of the following IP addresses:
a. 114.34.2.8
Ans:
Class Netid Hostid
A
114
34.2.8
b. 132.56.8.6
Ans:
Class Netid
Hostid
B
132.56 8.6
c. 208.34.54.12
Ans:
13Class Netid
Hostid
C
208.34.54 12
d. 251.34.98.5
Ans: It's class E, which doesn't have a defined netid and hostid. Therefore, all bits in a Class E address are used for the network ID, and it is not typically used for standard networking. (read more here…)
12. Find the number of addresses in the range if the first address is 14.7.24.0 and the last address is 14.14.34.255.
Ans: We can subtract the first address from the last address to find the number of addresses in the range.
The result is 0.7.10.255. We convert this number to base 10 and add 1 to it,
Number of addresses = $(0 * 256^3) + (7 * 256^2) + (10 * 256^1) + (255 * 256^0) + 1 = 461568$
13. If the first address in a range is 122.12.7.0 and there are 2048 addresses in the range, what is the
last address?

Ans: We convert the number of addresses minus 1 to base 256, which is 0.0.7.255. Then we add this number to the first address,
The last address is (122.12.7.0 + 0.0.7.255) = 122.12.14.255
This one is a bit critical, think like money divide algorithm!
Courtesy **Imamul Anan**
14. Find the result of each operation:
a. NOT (22.14.70.34)
Ans: 233.241.185.221
b. NOT (145.36.12.20)
Ans: 110.219.243.235
c. NOT (200.7.2.0)
Ans: 55.248.253.255
d. NOT (11.20.255.255)
Ans: 244.235.0.0
(255 - decimal) for not conversion!
15. Find the result of each operation:
a. (22.14.70.34) AND (255.255.0.0)
Ans: 22.14.0.0
b. (12.11.60.12) AND (255.0.0.0)
Ans: 12.0.0.0
c. (14.110.160.12) AND (255.200.140.0)
Ans: 14.70.128.0
d. (28.14.40.100) AND (255.128.100.0)
Ans: 28.0.32.0
AND, OR calculator এই করসা সা ব :)
1416. Find the result of each operation:
a. (22.14.70.34) OR (255.255.0.0)
Ans: 255.255.70.34
b. (12.11.60.12) OR (255.0.0.0)
Ans: 255.11.60.12
c. (14.110.160.12) OR (255.200.140.0)
Ans: 255.238.172.12
d. (28.14.40.100) OR (255.128.100.0)
Ans: 255.142.108.100
17. In a class A subnet, we know the IP address of one of the hosts and the subnet mask as given below:
IP Address: 25.34.12.56
Subnet mask: 255.255.0.0
What is the first address (subnet address)? **25.34.0.0**
What is the last address? **25.34.255.255**
18. In a class B subnet, we know the IP address of one of the hosts and the subnet mask as given below:
Address: 131.134.112.66
Subnet mask: 255.255.224.0
What is the first address (subnet address)? **131.134.96.0**
What is the last address? **131.134.127.255**
19. In a class C subnet, we know the IP address of one of the hosts and the subnet mask as given below:
Address: 202.44.82.16
Subnet mask: 255.255.255.192

What is the first address (subnet address)? 202.44.82.0
What is the last address? 202.44.82.63
20. Find the subnet mask in each case:
a. 1024 subnets in class A **255.255.192.0**
b. 256 subnets in class B **255.255.255.0**
c. 32 subnets in class C **255.255.255.224**
d. 4 subnets in class C **255.255.255.252**
21. In a block of addresses, we know the IP address of one host is 25.34.12.56/16. What is the first address (network address) and the last address (limited broadcast address) in this block?
Ans: The first address is 25.34.0.0 and the last address is 25.34.255.255.
22. In a block of addresses, we know the IP address of one host is 182.44.82.16/26. What is the first address (network address) and the last address (limited broadcast address) in this block?
Ans: The first address is 182.44.82.0 and the last address is 182.44.82.63.

23) Fixed-length subnetting — bits to add (`2^s ≥ subnets`)

a) 2 → **s = 1**
b) 62 → **s = 6**
c) 122 → **s = 7**
d) 250 → **s = 8**

24) Block 16.0.0.0/8, need 500 subnets

- a) New mask: **/17** → **255.255.128.0** (since 2^9=512 ≥ 500)
- b) Addresses/subnet: **2^(32−17) = 32,768**
- c) First subnet range: **16.0.0.0 – 16.127.255.255**
- d) Subnet 500 (1-based) range: **16.249.128.0 – 16.249.255.255**

---

25) Block 130.56.0.0/16, need 1024 subnets

- a) New mask: **/26** → **255.255.255.192** (2^10 = 1024)
- b) Addresses/subnet: **2^(32−26) = 64**
- c) First subnet range: **130.56.0.0 – 130.56.0.63**
- d) Subnet 1024 (1-based) range: **130.56.255.192 – 130.56.255.255**

---

26) Block 211.17.180.0/24, need 32 subnets

- a) New mask: **/29** → **255.255.255.248** (2^5 = 32)
- b) Addresses/subnet: **8**
- c) First subnet range: **211.17.180.0 – 211.17.180.7**
- d) Subnet 32 (1-based) range: **211.17.180.248 – 211.17.180.255**

---

27) Slash notation

a) 255.255.255.0 → **/24**
b) 255.0.0.0 → **/8**
c) 255.255.224.0 → **/19**
d) 255.255.240.0 → **/20**

---

28) Ranges of CIDR blocks

a) **123.56.77.32/29** → size 8 → **123.56.77.32 – 123.56.77.39**
b) **200.17.21.128/27** → size 32 → **200.17.21.128 – 200.17.21.159**
c) **17.34.16.0/23** → size 512 → **17.34.16.0 – 17.34.17.255**
d) **180.34.64.64/30** → size 4 → **180.34.64.64 – 180.34.64.67**

---

29) First & last known → find prefix?

Yes. Block size = `last - first + 1`; `n = 32 - log2(size)`.
Example: first **10.0.0.0**, last **10.0.3.255** → size **1024** → **/22**.

---

30) First & count known → find prefix?

Yes. `n = 32 - log2(N)` (N must be a power of 2, and first must be N-aligned).
Example: first **192.168.4.0**, N **512** → **/23**; range **192.168.4.0 – 192.168.5.255**.

---

31) Can two blocks share the same prefix length?

Yes. Many distinct blocks can have the same `/n` (same size) but different starting addresses.

---

32) First address & an interior address known → can we find prefix?

In general, no (not uniquely). Many different block sizes/prefixes could contain that interior address while starting at different valid aligned boundaries. You need either the last address or the count to determine a unique prefix.

---

33) ISP 150.80.0.0/16 for 2600 customers

- Group A: 200 × ~128 → **/25** each (128 addresses)
  - Use **200 /25s** from **150.80.0.0/25** up to **150.80.99.128/25** (covers 100 /24s).
- Group B: 400 × ~16 → **/28** each (16 addresses)
  - Use **400 /28s** from **150.80.100.0/28** up to **150.80.124.240/28**.
- Group C: 2000 × 4 → **/30** each (4 addresses)
  - Use **2000 /30s** from **150.80.125.0/30** up to **150.80.156.60/30** (last block spans .60–.63).

Addresses remaining:
Total used = 25,600 + 6,400 + 8,000 = **40,000** → remaining **65,536 − 40,000 = 25,536**.
Remaining range starts at **150.80.156.64** through **150.80.255.255**.

---

34) ISP 120.60.4.0/20 for 100 orgs, 8 each

- Each org gets **/29** (8 addresses).
- Assign **100 /29s** from **120.60.4.0/29** upward to **120.60.7.24/29** (100th; covers .24–.31).
- **Addresses remaining:** 4096 − 800 = **3296**, starting from **120.60.7.32** to **120.60.19.255**.

---

35) ISP has a block of 1024 addresses; 1024 customers

Yes—**subdivide into 1024 × /32 blocks** (one address per customer). That's subnetting at host-route granularity; it wastes no addresses but yields many routes (1024 /32s).