COMP 3721
Introduction to Data Communications

10b - Week 10 - Part 2

Learning Outcomes

- By the end of this lecture, you will be able to
  - Explain what is IPv4 addressing.
  - Explain classful and classless addressing schemes.
  - Explain how the destination-based forwarding works.
  - Explain the role of DHCP.
  - Explain what is NAT.

IPv4 Address

- An **IPv4** (Internet Protocol version 4) address is a **32-bit address** that **uniquely** and **universally** defines the connection of a **host** or a **router** to the Internet.
  - **Uniqueness**: each address defines one, and only one, connection to the Internet.
  - **Universality**: the addressing system must be accepted by any host that wants to be connected to the Internet.
- The **IP address** is the address of the **connection**, not the host or the router.
  - If the device is moved to another network, the IP address may be changed
  - Example: if a device has two connections to the Internet, via two networks, it has two IPv4 addresses

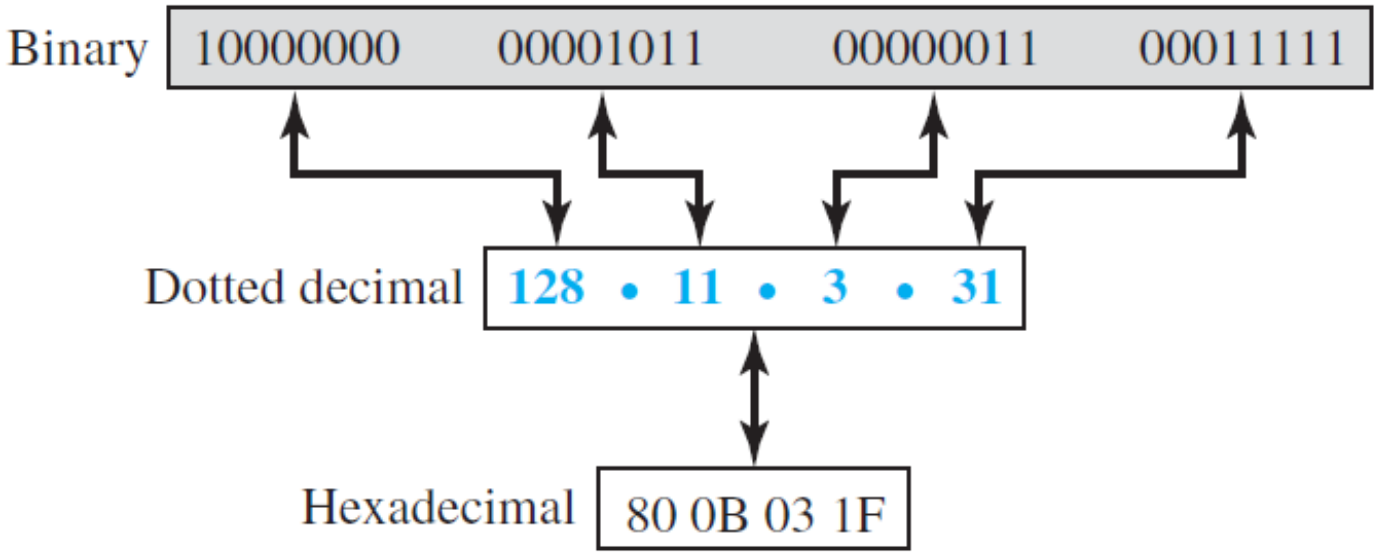
IPv4 Address Space

- **Address space**
  - The total number of addresses used by the protocol.
  - If a protocol uses *b* bits to define an address, the **address space is 2^b**.
- IPv4 uses 32-bit addresses, which means that the address space is:

$$2^{32} = 4{,}294{,}967{,}296$$

Three Common Notations to Show an IPv4 Address

1. **Binary Notation**
2. **Dotted-decimal Notation** → more compact and easier to read
3. **Hexadecimal Notation** → used in network programming

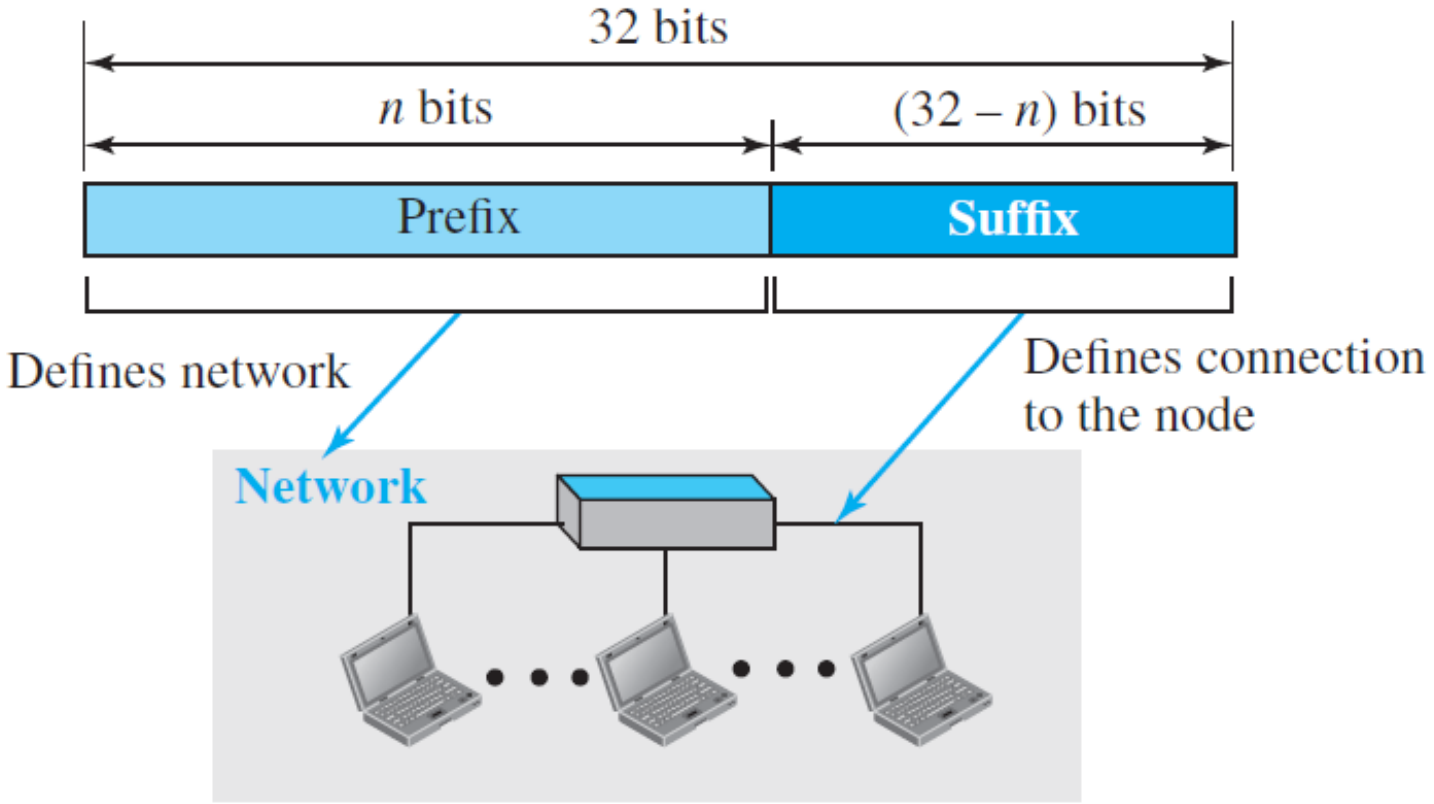


Hierarchy in Addressing – Two Analogies

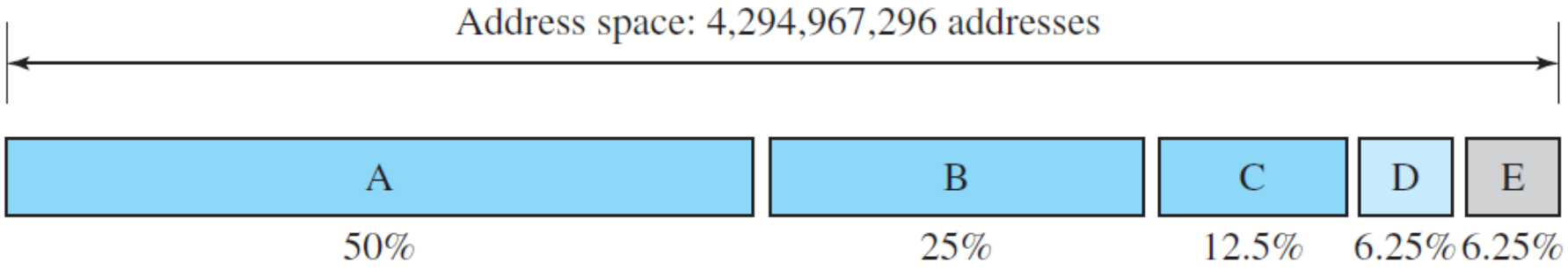
- Postal Network
  - Postal address (mailing address) includes the country, state, city, street, house number, and the name of the mail recipient
- Telephone Network
  - A telephone number is divided into the country code, area code, local exchange, and the connection

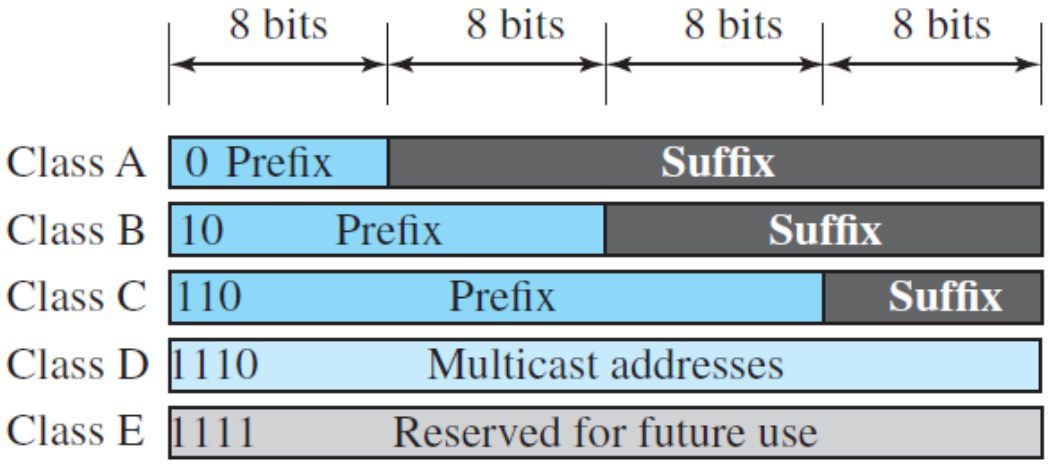
Hierarchy in Addressing

- A 32-bit IPv4 address is hierarchical and divided into two parts:



Classful Addressing





Class	Prefixes	First byte
A	$n = 8$ bits	0 to 127
B	$n = 16$ bits	128 to 191
C	$n = 24$ bits	192 to 223
D	Not applicable	224 to 239
E	Not applicable	240 to 255

- This scheme is now **obsolete**.
  - Due to **address depletion** (resulting from improper distribution of addresses in classes).

Classless Addressing

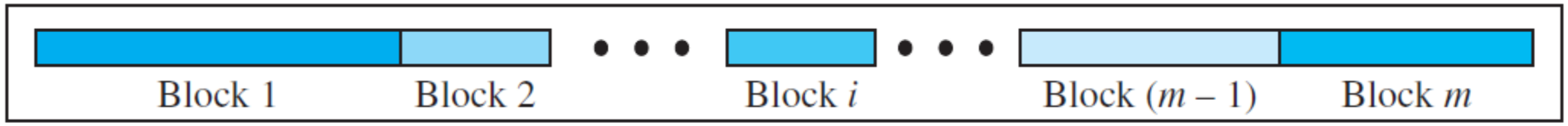
- With the growth of the Internet, it was clear that a larger address space was needed (**address depletion**/**shortage problem**).
- Short-term solution:
  - **Classless addressing** (introduced in 1996)
    - Use the same IPv4 address space
    - Changes the distribution of addresses to provide a fair share to each organization.
  - **NAT** (Network Address Translation)
- Long-term solution: **IPv6**
  - Uses a larger address space.
  - The length of IP addresses is required to be increased, which means the format of the IP packets needs to be changed.
  - 2^{128} = 340,282,366,920,938,463,463,374,607,431,768,211,456

Classless Addressing – Motivation

- IPv4 address depletion.
- The emergence of ISPs (Internet Service Providers).
  - An ISP is granted a large range of addresses and then subdivides the addresses and assigns them to customers.

Classless Addressing

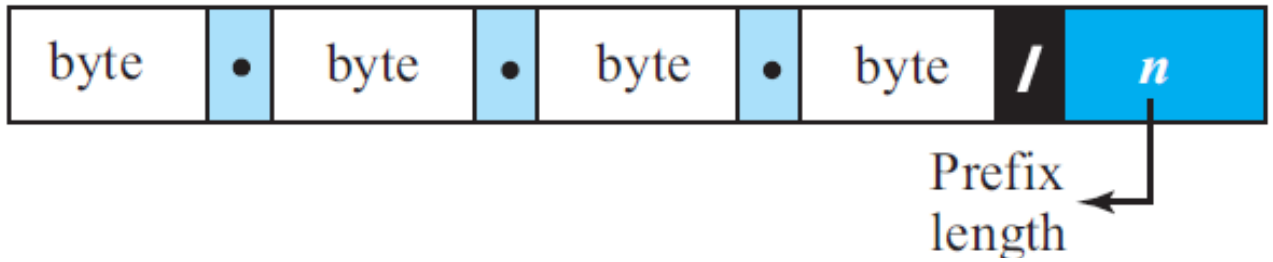
- The whole address space is divided into **variable length blocks**.
- The **prefix** in an address defines the **block** (**network**); the **suffix** defines the **node** (**device**).
- Unlike classful addressing, the length of prefix is **variable** (0 – 32).
- **Restriction**: the number of addresses in a block needs to be a **power of 2** (we can have a block of 2^0, 2^1, 2^2, …, 2^{32} addresses).
- The size of the network is inversely proportional to the length of the prefix.
  - A small prefix means a larger network; a large prefix means a smaller network.
- An organization can be granted one block of addresses.



Address space

Classless Interdomain Routing (CIDR)

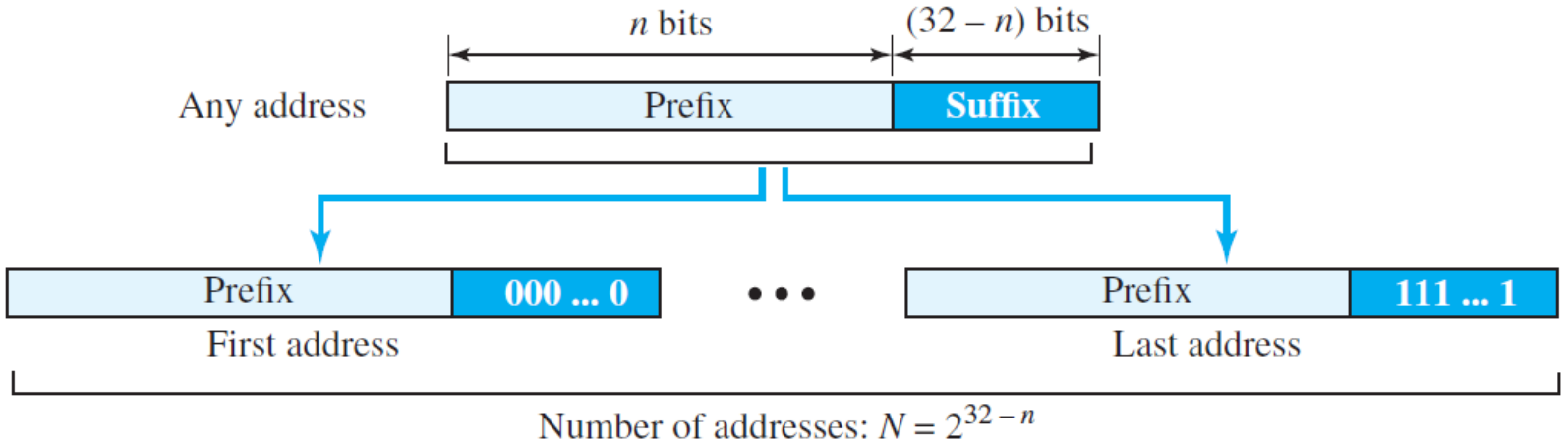
- The **prefix length** (also called **mask**), *n*, is added to the address, separated by a **slash**.
- This notation is referred to as **CIDR** (pronounced cider) or **slash notation**.



Examples:
12.24.76.8/8
23.14.67.92/12
220.8.24.255/25

Extracting Information from an Address

- Given any address in a block, three pieces of information should be extracted.
  1. The number of addresses (**$N = 2^{32-n}$**).
  2. The first address in the block (**keep the *n* leftmost bits** and **set the (32–*n*) rightmost bits all to 0s**).
  3. The last address in the block (**keep the n leftmost bits** and **set the (32–*n*) rightmost bits all to 1s**).



Network Address

- The **first address** in the block is called the **network address**.
- **Network address** is important since it is **used in routing the packet to its destination**.

Classless Addressing – Example 1

- A classless address is given as 167.199.170.82/27. Find the number of addresses as well as the first and last addresses.

Classless Addressing – Example 1

- A classless address is given as 167.199.170.82/27. Find the number of addresses as well as the first and last addresses.

1. The **number of addresses** in the network is $2^{32-n} = 2^{32-27} = 2^5$ = **32 addresses**.
2. The **first address** can be found by **keeping the first 27 bits** (27 leftmost bits) and changing the **rest of the bits to 0s**.

Address: 167.199.170.82/27	10100111	11000111	10101010	01010010
First address: 167.199.170.64/27	10100111	11000111	10101010	01000000

3. The last address can be found by **keeping the first 27 bits** and changing the **rest of the bits to 1s**.

Last address: 167.199.170.95/27	10100111	11000111	10101010	01011111

Classless Addressing – Example 2

- In classless addressing, an address cannot per se define the block the address belongs to.
- For example, the address 230.8.24.56 can belong to many blocks. Some of them are shown below with the value of the prefix associated with that block.

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

Classless Addressing – Example 2

- 230.8.24.56/20

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

- Each decimal represent 8 bits. To keep the first 20 bits, fix the first 2 decimals (16 bits), then fix 4 more bits.
- 24 = 00011000 → 230.8.00010000.0 → 230.8.16.0
- 00011111 = 31 → 230.8.00011111.255 → 230.8.31.255

Address Mask (Subnet Mask)

- The address mask is a 32-bit number in which the n leftmost bits are set to 1s and the rest of the bits (32−*n*) are set to 0s.
- E.g., 167.199.170.82/**27** subnet mask is
  - **/27** → **11111111**.**11111111**.**11111111**.**111**00000 = 255.255.255.224

Block Allocation

- Block allocation is the responsibility of a **global authority** called the **Internet Corporation for Assigned Names and Numbers** (**ICANN**).
- ICANN **assigns a large block of addresses to an ISP** (or a larger organization that is considered an ISP in this case).

Block Allocation

- For proper operation of CIDR, **two restrictions** are applied to the allocated block:
  1. **The number of requested addresses, *N*, needs to be a power of 2** (if not a power of 2, then a number that is a power of 2 and is greater than the requested number will be considered), e.g., 1000 addresses requested → 1024 granted
     - **Reason**: $N = 2^{32-n}$ or $n = 32 - \log_2 N$ (if *N* is not a power of 2, we cannot have an integer value for n)
  2. **The first address needs to be divisible by the number of addresses in the block**.
     - **Reason**: the first address needs to be the prefix followed by (32−*n*) number of 0s. The decimal value of the first address is then

$$\text{first address} = (\text{prefix in decimal}) \times 2^{32-n} = (\text{prefix in decimal}) \times N.$$

Block Allocation Example

- An ISP has requested a block of 1000 addresses. Show how many addresses will be granted by the ISP and verify the allocation.

Block Allocation Example

- An ISP has requested a block of 1000 addresses. Show how many addresses will be granted by the ISP and verify the allocation.
- **Answer**:
  - Since 1000 is not a power of 2, **1024** addresses are granted. The prefix length is calculated as $n = 32 - \log_2 1024 =$ **22**.
  - An available block, **18.14.12.0/22**, is granted to the ISP.
  - It can be seen that the first address in decimal is 302,910,464, which is divisible by 1024.
  - 00010010 00001110 00001100 00000000 = 302,910,464
  - 302,910,464 mod 1024 = 0

Subnetting

- Subnetting is the **process** of **creating a subnetwork**.
- Creates more levels of hierarchy
- An organization (or an ISP) that is granted a range of addresses may divide the range into several subranges and assign each subrange to a subnetwork (or subnet).
  - A subnetwork can be divided into several sub-subnetworks.
  - A sub-subnetwork can be divided into several sub-sub-subnetworks, and so on.

Designing Subnets

- Our assumptions
  - ***N***: the total number of addresses granted to the organization
  - ***n***: the prefix length
  - N_{sub}: the assigned number of addresses to each subnetwork
  - n_{sub}: the prefix length for each subnetwork
- To guarantee the proper operation of the subnetworks:
  1. The number of addresses in each subnetwork should be a power of 2.
  2. The prefix length for each subnetwork should be found using the following formula: $n_{\text{sub}} = 32 - \log_2 N_{\text{sub}}$
  3. The starting address in each subnetwork should be divisible by the number of addresses in that subnetwork. This can be achieved if we **first assign** addresses to **larger subnetworks**.

Designing Subnets

- The subnetworks in a network should be carefully designed to enable the routing of packets.
- After designing the subnetworks, the information about each subnetwork, such as first and last address, can be found using the process we described to find the information about each network in the Internet.

Designing Subnets Example

- An organization is granted a block of addresses with the beginning address **14.24.74.0/24**. The organization needs to have **3 subblocks** of addresses to use in its three subnets: one subblock of **10 addresses**, one subblock of **60 addresses**, and one subblock of **120 addresses**. Design the subblocks.

Designing Subnets Example

- There are 2^{32-24} = **256 addresses** in this block.
  - The first address is **14.24.74.0/24**;
  - The last address is **14.24.74.255/24**.

- To satisfy the subnetting requirements, we assign addresses to subblocks, starting with the largest and ending with the smallest one:

Designing Subnets Example

a) The number of addresses in the largest subblock, which requires **120 addresses**, is not a power of 2. We allocate **128 addresses**. The subnet mask for this subnet can be found as $n_1 = 32 - \log_2 128 = 25$.

- The first address in this block is **14.24.74.0/25**.
- The last address is **14.24.74.127/25**.

Designing Subnets Example

b) The number of addresses in the second largest subblock, which requires **60 addresses**, is not a power of 2 either. We allocate **64 addresses**. The subnet mask for this subnet can be found as $n_2 = 32 - \log_2 64 = 26$.

- The first address in this block is **14.24.74.128/26**.
- The last address is **14.24.74.191/26**.

Designing Subnets Example

c) The number of addresses in the second largest subblock, which requires **10 addresses**, is not a power of 2 either. We allocate **16 addresses**. The subnet mask for this subnet can be found as $n_2 = 32 - \log_2 16 = 28$.
   - The first address in this block is **14.24.74.192/28**.
   - The last address is **14.24.74.207/28**.

- If we add all addresses in the previous subblocks, the result is 208 addresses, which means 48 addresses are left in reserve. The first address in this range is 14.24.74.208. The last address is 14.24.74.255. We don't know about the prefix length yet.

Special Addresses

- Five special addresses that are used for special purpose:

1. **This-host address**: the only address in the block **0.0.0.0/32**
   - Used whenever a host needs to send an IP datagram, but it does not know its own address to use as the source address.
2. **Broadcast address**: the only address in the block is **255.255.255.255/32**
   - Used whenever a router or a host needs to send a datagram to all devices in a network.
   - The routers in the network, however, block the packet having this address as the destination; the packet cannot travel outside the network.
3. **Loopback address**: the block **127.0.0.0/8**
   - A packet with one of the addresses in this block as the destination address never leaves the host; it will remain in the host.
   - Testing a piece of software in the machine. Running some services, etc.

Special Addresses

4. **Private addresses**
   - Four blocks are assigned as private addresses: **10.0.0.0/8**, **172.16.0.0/12**, **192.168.0.0/16**, and **169.254.0.0/16**.
5. **Multicast addresses**
   - The block **224.0.0.0/4** is reserved for multicast addresses

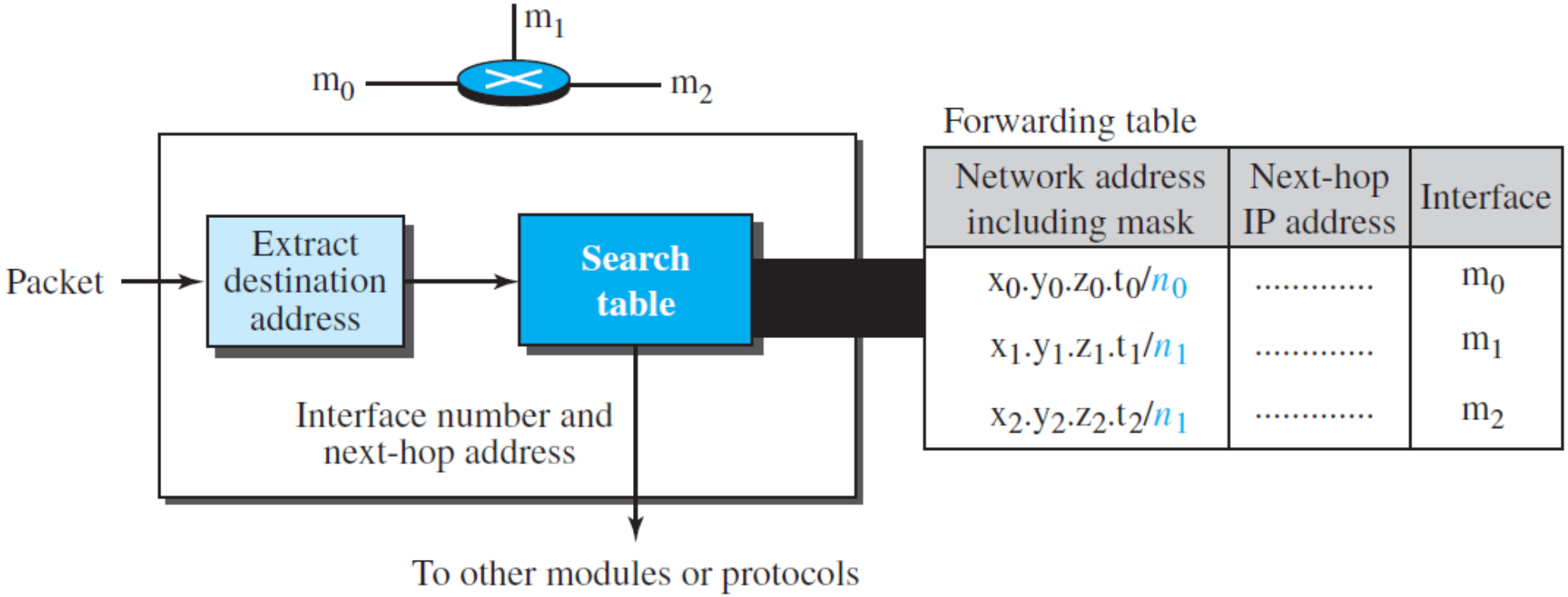
Forwarding of IP Packets

- The Internet today is made of a combination of links (networks).
- **Forwarding**: to deliver the packet to the next hop (which can be the final destination or the intermediate connecting device).
- IP protocol was originally designed as a **connectionless** protocol.
- **Forwarding** is based on the **destination address** of the IP datagram.
  - Traditional approach and prevalent today

Forwarding Based on Destination Address

- Destination-based forwarding requires a host or a router to have a **forwarding** (**routing**) **table**.
- In classless addressing, one row of information is added to the forwarding table for each block involved.
  - Four pieces of information: **mask**, **network address**, **interface number**, and **IP address** of the next router.
  - The first two are usually combined.
- The forwarding table needs to be **searched** based on the **network address** (i.e., the first address in the block).

Forwarding Based on Destination Address

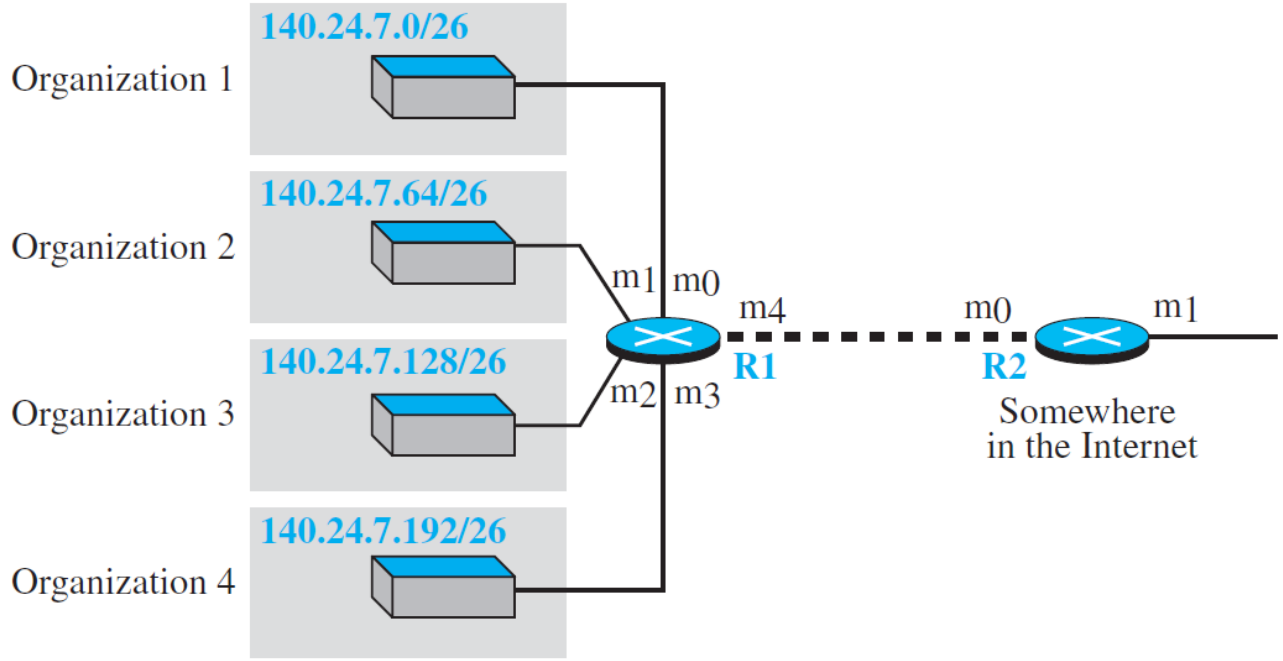


- Search: The router looks at each row. The *n* leftmost bits of the destination address (prefix) are kept, and the rest of the bits (suffix) are set to 0s. If the resulting address (network address), matches with the address in the first column, the information in the next two columns is extracted; otherwise, the search continues.

Address Aggregation (Address Summarization/Route Summarization)

- When the number of rows in the forwarding table increases → the time to search the table increases
- **Address Aggregation** → to alleviate the above problem
  - When blocks of addresses are **combined** to create a larger block, **routing** can be done based on the **prefix** of the **larger block**.
  - One of the **advantages** of the CIDR strategy.
- ICANN assigns a large block of addresses to an ISP
  - Each ISP in turn divides its assigned block into smaller subblocks and grants the subblocks to its customers.

Address Aggregation Example



Forwarding table for R1

Network address/mask	Next-hop address	Interface
140.24.7.0/26	----------	m0
140.24.7.64/26	----------	m1
140.24.7.128/26	----------	m2
140.24.7.192/26	----------	m3
0.0.0.0/0	address of R2	m4

Forwarding table for R2

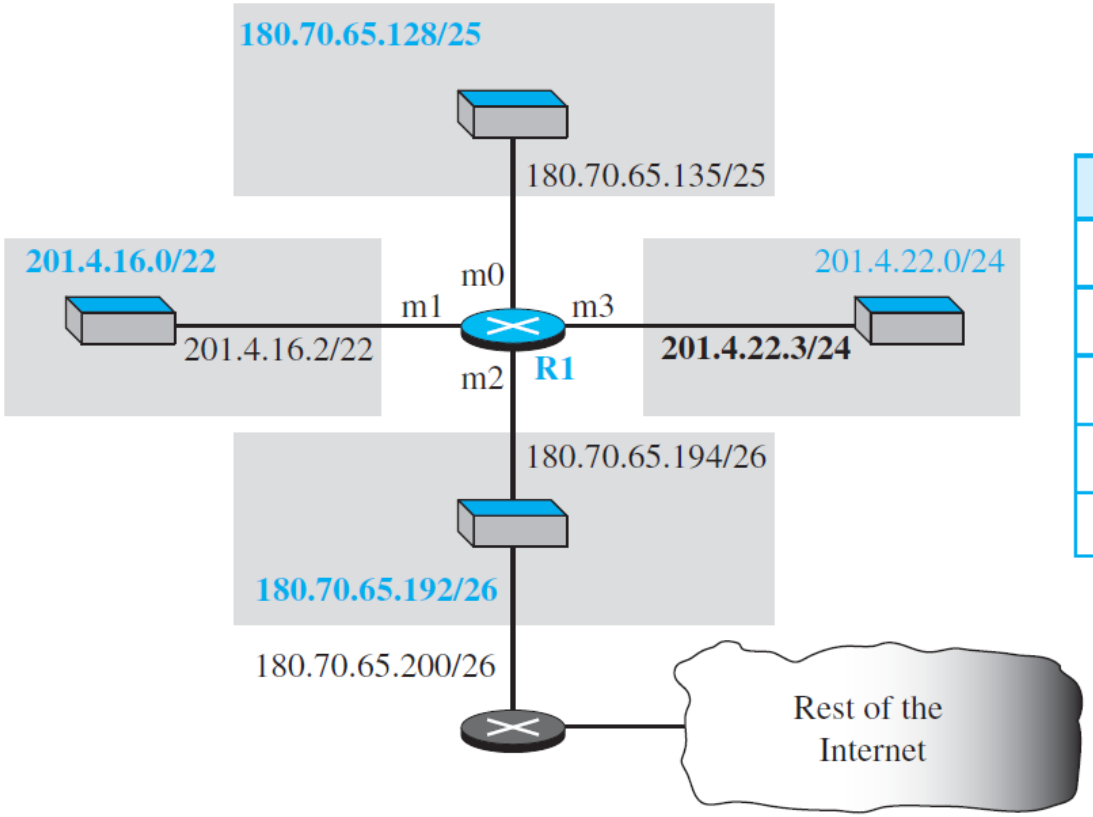
Network address/mask	Next-hop address	Interface
140.24.7.0/24	----------	m0
0.0.0.0/0	default router	m1

Longest Mask Matching

- Longest mask matching
  - A **routing principle** in classless addressing.
  - States that the forwarding table is **sorted** from the **longest mask** to the shortest mask.
  - When a router receives the IP packet, it compares the **destination IP** address **bit-by-bit** with **prefixes** in the routing table. The prefix with the **most matching bits** is the prefix that the router will use.

Longest Mask Matching – Example

- Make a forwarding table for router R1 using the configuration in the following figure. Only fill in the columns for network address/mask and interfaces.

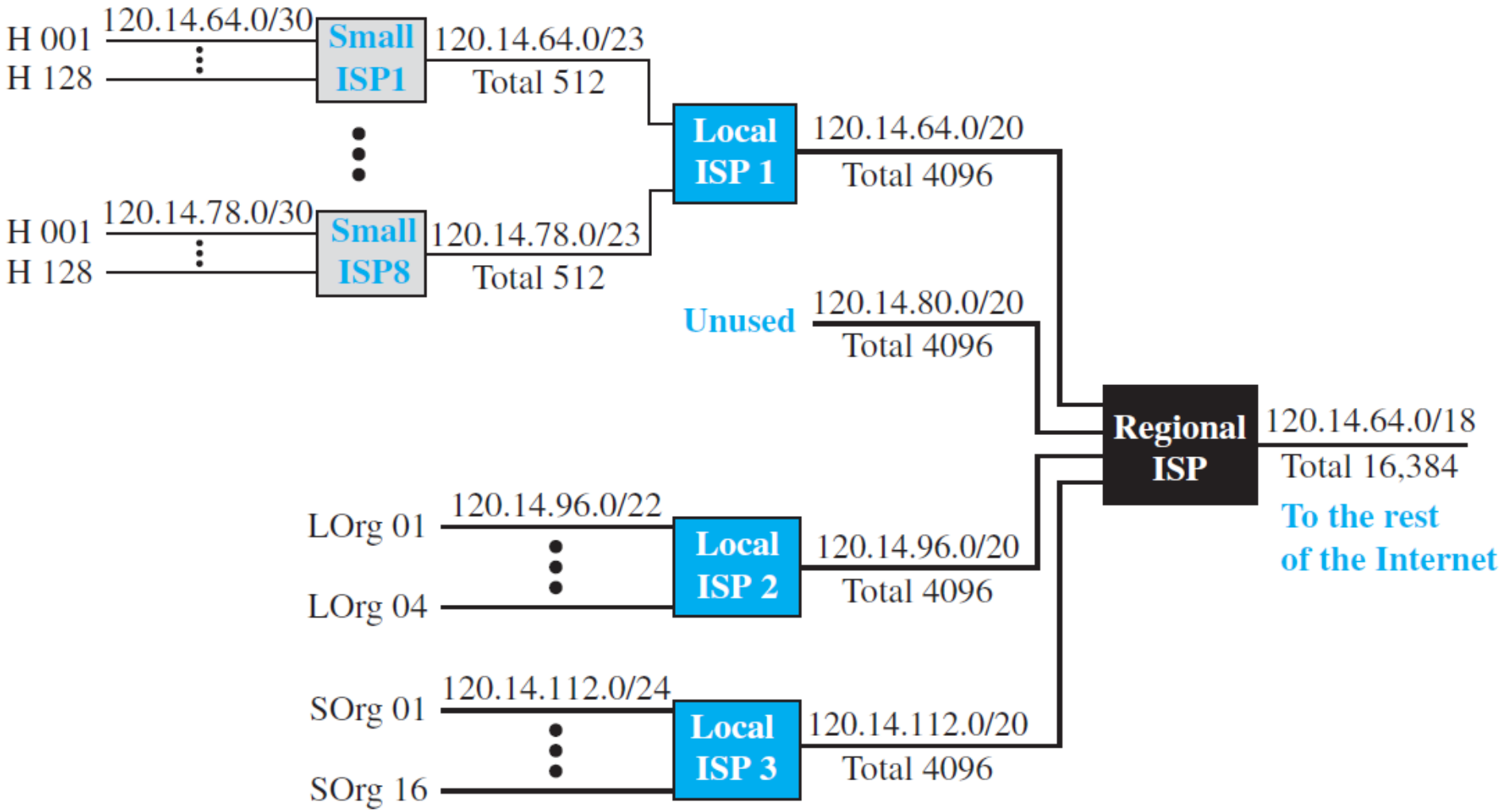


Network address/mask	*Next hop*	*Interface*
180.70.65.192/**26**	—	m2
180.70.65.128/**25**	—	m0
201.4.22.0/**24**	—	m3
201.4.16.0/**22**	—	m1
Default	180.70.65.200	m2

Hierarchical Routing

- To solve the problem of gigantic forwarding tables, we can create a sense of hierarchy in the forwarding tables:
- The Internet
  - Backbone ISPs
  - National ISPs
    - Regional ISPs
      - Local ISPs
- We have **unlimited levels** of hierarchy in **classless addressing** as long as the rules of addressing are followed.

Hierarchical Routing – Example



Dynamic Host Configuration Protocol (DHCP)

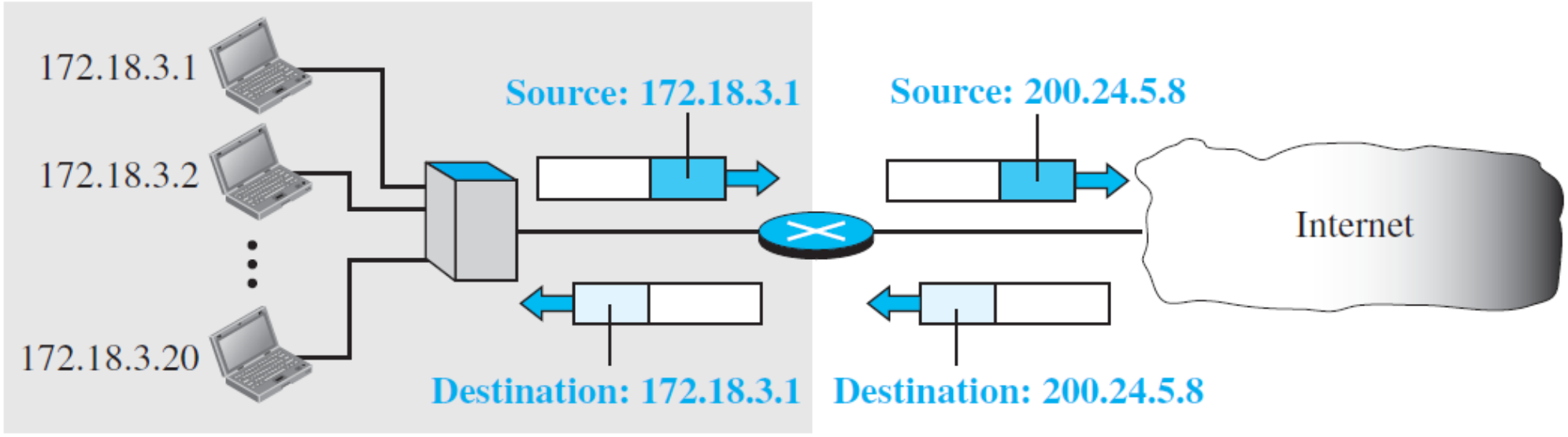
- An **application-layer program** (**protocol**), using the client-server paradigm, that actually helps TCP/IP at the network layer.
- After a **block of addresses** are assigned to an organization, IP addresses must be assigned to the **individual hosts** or **routers**.
  - **Manual assignment**: done by the network administration.
  - **Automatic**/**dynamic assignment**: using DHCP (network admin configures it).
- DHCP can be configured to assign
  - **permanent** IP addresses to the host and routers.
  - **temporary**, on demand, IP addresses to hosts (e.g., traveller staying at a hotel to connect his/her laptop to the Internet).

Network Address Translation (NAT)

- In most situations, only **a portion of computers** in a small network need access to the Internet **simultaneously**.
  - The number of **allocated addresses** does not have to match the **number of computers** in the network.
  - A company usually uses **private IP addresses** for **internal communication**.
  - A set of **universal** (public/global) **IP addresses** assigned by an ISP are used for **universal communication**.
- **NAT** is a technology that provides mapping between the **private** and **universal IP addresses**.

NAT Example

- All of the **outgoing packets** go through the NAT router, which replaces the **source address** in the packet with the **global NAT address**. All **incoming packets** also pass through the NAT router, which replaces the **destination address** in the packet (the NAT router global address) with the appropriate **private address**.

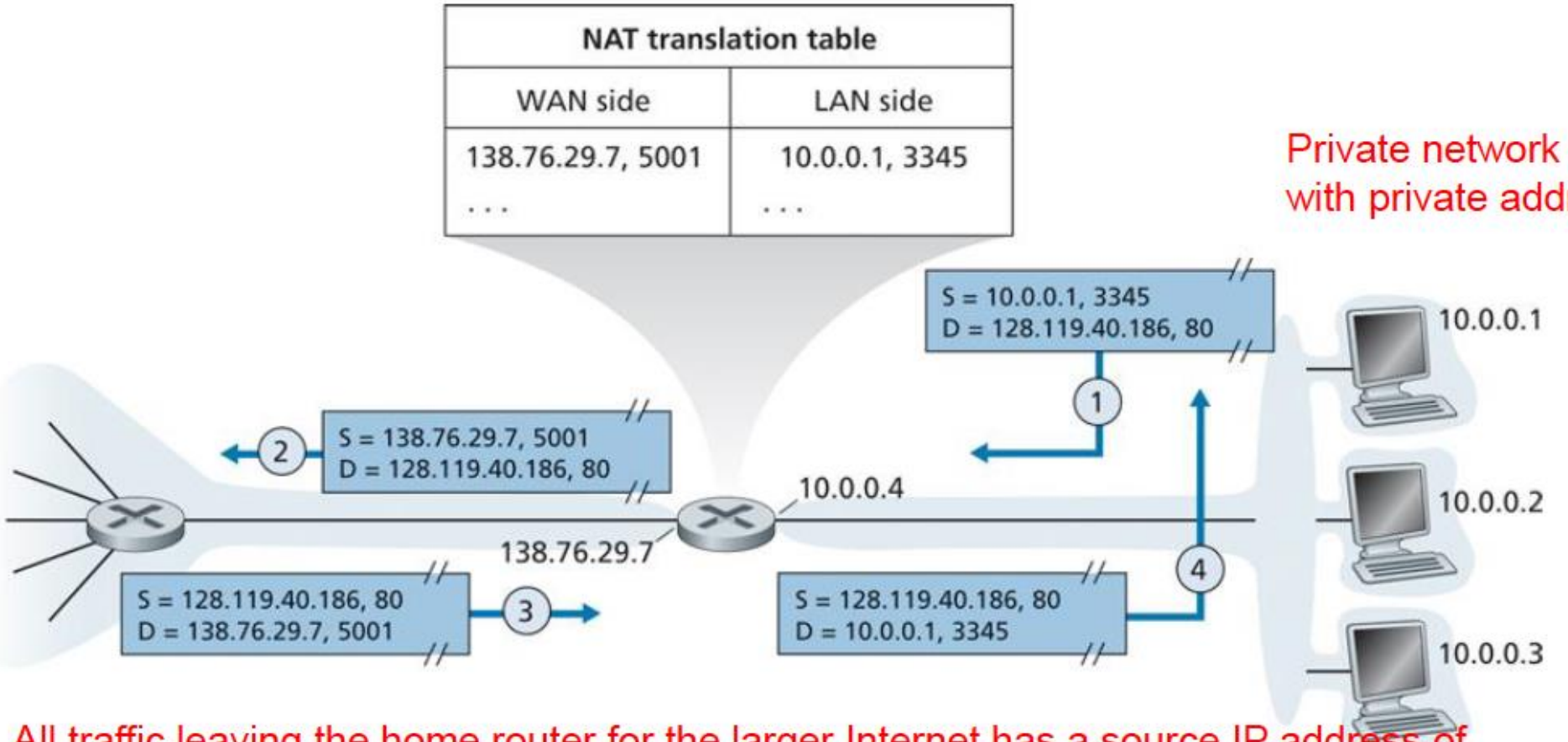


Site using private addresses

NAT Translation Table

- Translating the **source addresses** for an outgoing packet is straightforward.
- How does the NAT router know the **destination address** for a packet coming from the Internet?
  - NAT router has a **translation table**.

NAT Translation Table



All traffic leaving the home router for the larger Internet has a source IP address of 138.76.29.7, and all traffic entering the home router must have a destination address of 138.76.29.7. In essence, the **NAT-enabled router** is hiding the details of the home network from the outside world.

51

NAT Advantages

- Just **one** IP address can be used from the ISP for all devices.
- The **addresses** of hosts in a **local network** can be **changed** without notifying outside world.
- A **different ISP** can be chosen (to receive service from) without changing addresses of devices in local network.
- **Security**: devices inside local net not directly addressable, visible by outside world.
- NAT has been extensively used in home and institutional nets, 4G/5G cellular nets.

Summary

- The **main services** provided by the network layer are **packetizing** and **routing** the packet from the source to the destination.
- Addressing in IPv4
  - Two address distribution mechanisms: classful and classless addressing.
- **Destination-based routing** by the routers.
- **DHCP** to dynamically assign IP addresses to hosts and routers.
- Temporary alleviating address shortage in IPv4 using **NAT**.

References

[1] Behrouz A.Forouzan, Data Communications & Networking with TCP/IP Protocol Suite, 6th Ed, 2022, McGraw-Hill companies.

Reading

- Chapter 7 of the textbook, section 7.4.1.
- Chapter 7 of the textbook, section 7.8 (Practice Test)