Lecture 2

Logical Addressing- IPv4 Addresses

Dr. Vandana Kushwaha

Department of Computer Science

Institute of Science, BHU, Varanasi

Introduction

- Communication at the **network layer** is **end-to-end(source to destination)**;
- A computer somewhere in the world needs to communicate with another computer somewhere else in the world.
- For this **level of communication**, we need a **global addressing scheme**; we called this **logical addressing** or **IP address**.

IPv4 ADDRESSES

- An **IPv4** address is a **32-bit** address.
- They are **unique** in the sense that **each address** defines one, and only one, connection to the **Internet.**
- **Two devices** on the **Internet** can **never have** the **same address** at the **same time.**

IPv4 ADDRESSES

- But by using **some strategies**, an **address** may be assigned to a device for a time period and then taken away and assigned to another device.
- On the other hand, if a device operating at the **network layer** has ***m*** **connections** to the **Internet,** it needs to have ***m*** **addresses.**
- A **router** is such a **device** which **needs** as many **IP addresses** as the **number of ports** are there in it.

IPv4 Address Space

- *An **address space** is the **total number of addresses** used by the **protocol**.*
- If a protocol uses ***N* bits** to define **an address**, the **address space is 2^N** because each bit can have two different values (0 or 1) and ***N* bits** can have **2^N** values.
- **IPv4** uses **32-bit addresses**, which means that the **address space is 2^{32}** or **4,294,967,296 (more than 4 billion).**
- This means that, **theoretically**, if there were no restrictions, **more than 4 billion devices** could be connected to the **Internet.**
- But the **actual number** is **much less** because of the **restrictions** imposed on the **addresses.**

IPv4 Address Notations

There are **two prevalent notations** to show an **IPv4 address**:

i. ***Binary notation***

ii. ***Dotted decimal notation.***

i. Binary Notation

- In **binary notation**, the **IPv4 address** is displayed as **32 bits.**
- Each **octet** is often referred to as a **byte.**
- So an **IPv4 address** referred to as a **32-bit address** or a **4-byte address**.
- The following is an **example** of an **IPv4 address** in **binary notation:**

01110101 10010101 00011101 00000010

IPv4 Address Notations

ii. Dotted-Decimal Notation

- To make the **IPv4 address** more **compact and easier to read**, Internet addresses are usually written in **decimal form** with a **decimal point (dot)** separating the **bytes.**
- The following is the **dotted decimal notation** of the previous address:

  117.149.29.2

- Note that because each **byte (octet)** is **8 bits**, each number in **dotted-decimal notation** is a value ranging from **0 to 255**.

Types of IPv4 Addressing Schemes

There are **two types of IPv4 addressing** schemes:

i. ***Classful Addressing***

ii. ***Classless Addressing***

i. Classful Addressing

- **IPv4 addressing**, at its **inception**, used the concept of **classes.** Although this scheme is becoming **obsolete**.
- In **classful addressing**, the **address space** is divided into **five classes**: **A, B, C, D**, and **E**.
- Each **class** occupies some **part** of the **address space.**
- If the address is given in **binary notation**, the **first few bits** can immediately tell us the **class of the address.**
- If the **address** is given in **decimal-dotted notation**, the **first byte defines** the **class.**

Finding the Classes in Binary and Dotted-decimal notation

	First byte	Second byte	Third byte	Fourth byte
Class A	0			
Class B	10			
Class C	110			
Class D	1110			
Class E	1111			

a. Binary notation

	First byte	Second byte	Third byte	Fourth byte
Class A	0–127			
Class B	128–191			
Class C	192–223			
Class D	224–239			
Class E	240–255			

b. Dotted-decimal notation

Finding the classes in binary and dotted-decimal notation

Example

Find the class of each address.

a. 00000001 00001011 00001011 11101111

b. 11000001 10000011 00011011 11111111

c. 14.23.120.8

d. 252.5.15.111

Solution

a. The first bit is 0. This is a **class A** address.

b. The first 2 bits are 1; the third bit is 0. This is a **class C** address.

c. The first byte is 14 (between 0 and 127); the **class is A**.

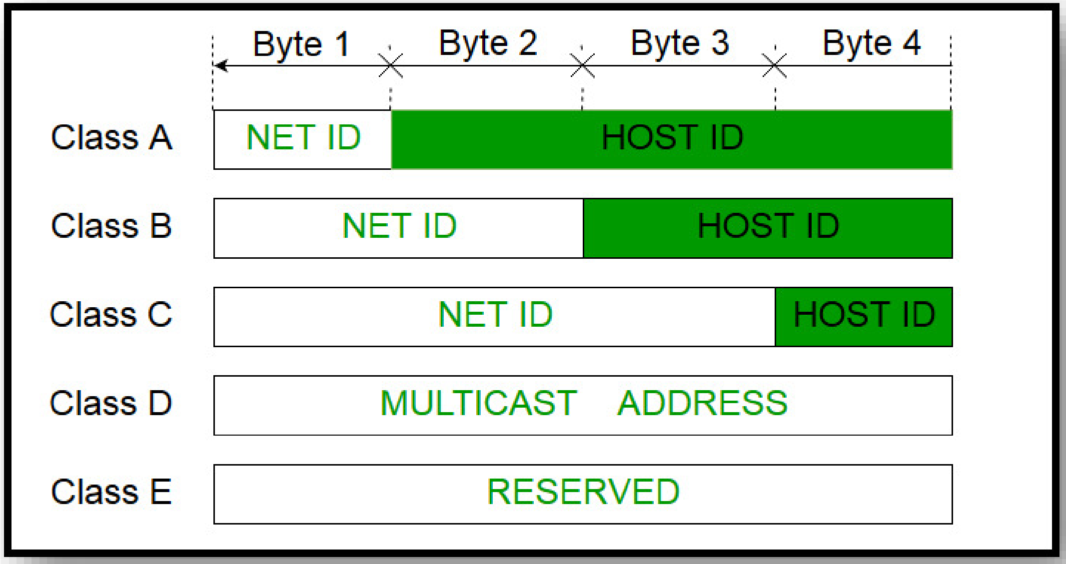
d. The first byte is 252 (between 240 and 255); the **class is E**.

IPv4 address

Each **IPv4 address** is divided into **two parts:**

- **Network ID**
- **Host ID**

- The **class** of **IP address** is used to **determine** the **bits** used for **network ID** and **host ID** and the **number of total networks** **and** **hosts possible** in that particular **class.**



Classes and Blocks

- In **Classful addressing** each **class** is **divided** into a **fixed number of blocks** with each **block** having a **fixed size** as shown in following **Table**:

Class	Number of Blocks	Block Size	Application
A	$2^7=128$	$2^{24}=16,777,216$	**Unicast** (large organizations)
B	$2^{14}=16,384$	$2^{16}=65,536$	**Unicast** (midsize organizations)
C	$2^{21}=2,097,152$	$2^8=256$	**Unicast** (small organizations)
D	1	$2^{28}=268,435,456$	**Multicast**
E	1	$2^{28}=268,435,456$	**Reserved**

Limitations of Classful Addressing

- A **block** in **class A** address is **too large(**16,777,216) for almost any organization.
- This means **most** of the **addresses in class A were wasted** and were **not used.**
- A block in **class B** is also **very large(65,536)**, probably **too large** for many of the organizations that received a **class B** block.
- A block in **class C** is probably **too small(256)** for many organizations.
- **Class D** addresses were designed for **multicasting**. Each address in this class is used to define **one group** of hosts on the Internet.
- The **Internet authorities** wrongly predicted a need for **268,435,456 groups**.
- This never happened and many **addresses were wasted** here too.
- And lastly, the **class E** addresses were **reserved** for **future use**; only a few were used, resulting in **another waste** of **addresses.**

Address Depletion Problem

- The **fast growth** of the **Internet** led to the near **depletion(shortage)** of the **available addresses** in **classful addressing scheme.**
- Yet the **number of devices** on the Internet is **much less than** the **2^{32}** address space.
- We have run **out of class A** and **B addresses**, and a **class C block** is **too small** for **most midsize organizations**.
- One **solution** that has alleviated the problem is the idea of **Classless addressing.**
- **Classful addressing**, which is almost **obsolete**, is **replaced with Classless addressing**.
- To overcome **address depletion** and give more organizations access to the Internet, **Classless addressing** was designed and implemented.

Classless Addressing

In **Classless Addressing** scheme, there are **no classes**, but the addresses are still granted in **blocks.**

Address Blocks

- In **classless addressing**, when an entity, small or large, needs to be connected to the Internet, it is **granted a block** (range) of **addresses.**
- The **size of the block** (the number of addresses) varies based on the nature and size of the entity. For **example**:
  - A **household** may be given only **two addresses**.
  - A **large organization** may be given **thousands of addresses**.
  - An **ISP**(Internet service provider) may be given **thousands** or **hundreds** of addresses based on the **number of customers** it may **serve.**

Classless Addressing

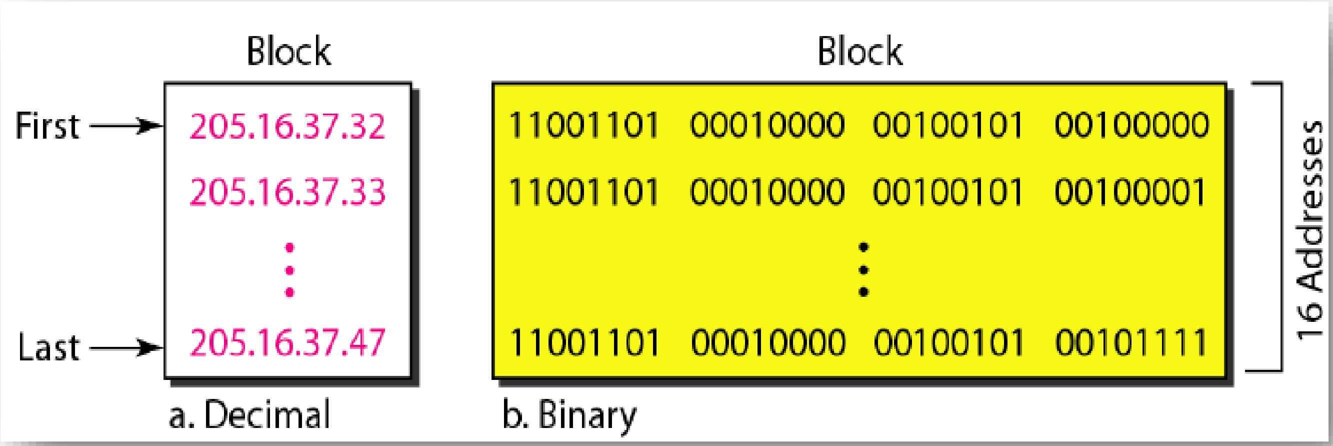
The **Internet authorities** impose **three restrictions** on **classless address blocks**:

1. The **addresses in a** ***block must be contiguous***, one after another.

2. The ***number of addresses*** in **a block** must be a ***power of 2*** **(1, 2, 4, 8, ...).**

3. The ***first address*** must be ***evenly divisible*** ***by*** *the* ***number of addresses***.

Example: Classless Addressing



- The addresses are **contiguous**.
- The **number of addresses** is a **power of 2** (**16 = 2^4**), and the **first address** is **divisible by 16**.
- **The first address**, when converted to a **decimal number, is 3,440,387,360**, which when **divided by 16** results in **215,024,210**.

Mask

- A **better** way to **define** a **block** of **addresses** is to select **any address** in the **block** and the **mask.**
- A **mask** is a **32-bit number** in which the **n leftmost bits are 1s and the 32 - n rightmost bits are 0s.**
- **Mask** is **represented** by the value of ***n* preceded by a slash(/)** which is known as **CIDR notation**.
- In **IPv4 addressing**, a **block of addresses** can be defined as **x.y.z.t/n** in which **x.y.z.t defines one of the addresses** and the **/n defines the mask**.
- The **address** and the **/n notation** completely **define the whole block** (the *first address, the last address, and the number of addresses*).

First Address of a block

- The **first address** in the block can be found by ***setting the 32 - n rightmost bits*** *in the* ***binary notation*** *of the address to* ***0s.***

Example

A **block** of addresses is granted to a small organization. We know that one of the addresses is **205.16.37.39/28.** What is the **first address** in the block?

Solution

The binary representation of the given address is

11001101 00010000 00100101 00100111

If we set **32 - 28** rightmost bits **to 0**, we get

11001101 0001000 00100101 00100000 or 205.16.37.32

This is actually the block shown in previous Figure.

Last Address

The **last address** in the block can be found by ***setting the 32 - n rightmost bits*** *in the binary notation of the address* ***to 1s*****.**

Example

Find the **last address** for the **block** in previous Example.

Solution

The binary representation of the given address is

11001101 00010000 00100101 00100111

If we set **32 - 28 rightmost bits to 1**, we get

11001101 00010000 00100101 0010 1111 or **205.16.37.47**

This is actually the **block** shown in previous Figure.

Number of Addresses:

The **number of addresses** in the block = $\boldsymbol{2^{32-n}}$

Example

Find the number of addresses in previous Example.

Solution

The value of *n* is 28, which means that number of addresses is $\mathbf{2^{32-28}}$ or **16**.

Method 2 for finding block information

- Another way to find the *first address*, the *last address*, and *the number of addresses* is to represent the **mask** as a **32-bit binary**.
- This is particularly useful when we are writing a **program** to find these pieces of information.
- In previous **Example** the **/28** can be represented as :

  11111111 11111111 11111111 11110000 (twenty-eight 1s and four 0s).

  First address of the block= (Given address) bit wise AND (Mask)

- **Address:** 11001101 00010000 00100101 00100111
- **Mask:** 11111111 11111111 11111111 11110000
- **First address:** 11001101 00010000 00100101 00100000

Last address & the No. of addresses

Last address of the block= (***Given address) bitwise OR (Complement of the mask***)

- **Address:** 11001101 00010000 00100101 00100111
- **Mask complement**: 00000000 00000000 00000000 00001111
- **Last address:** 11001101 00010000 00100101 00101111

The number of addresses = (mask complement)$_{10}$ + 1

- **Mask complement**: 000000000 00000000 00000000 00001111
- **Number of addresses**: 15 + 1 =16

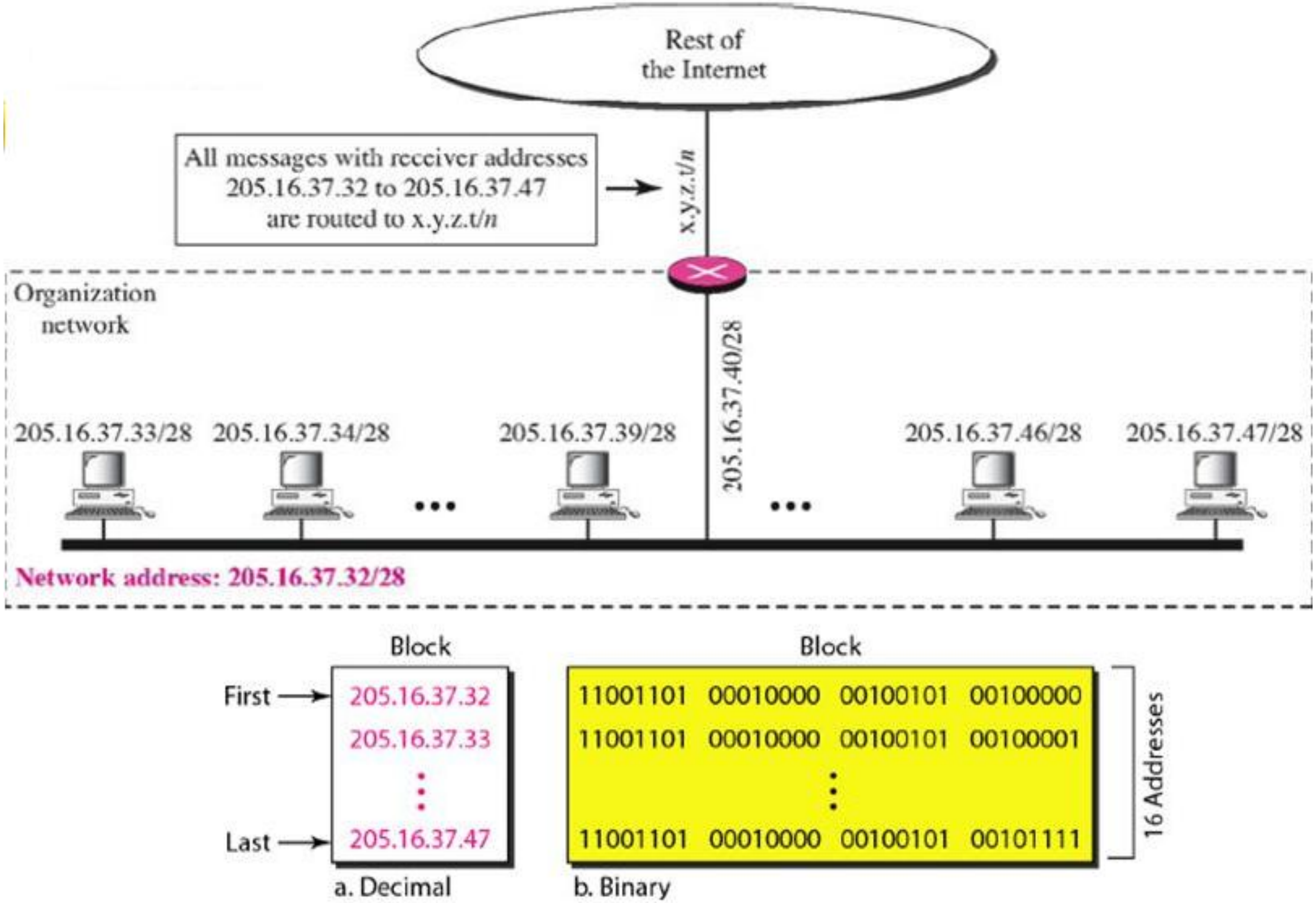
Network Addresses

- A very important concept in **IP addressing** is the **network address**.
- When an organization is given a **block** of **addresses,** the organization is free to allocate the addresses to the devices that need to be connected to the Internet.
- The **first address in the class**, however, is normally (not always**)** treated as a **special address.**
- *The **first address** is called the **network address** and defines the organization network*.
- It **defines** the **organization itself** to the rest of the world.
- Usually the **first address** is the one that is **used by routers** to direct the message sent to the organization from the **outside**.

Example

- Figure on next page shows an **organization** that is granted a **16-address block**.
- The organization network is connected to the **Internet** via a **router.**
- The **router** has **two** addresses. One belongs to the **granted block**; the other belongs to the **network** that is at the **other side** of the **router.**
- We call the **second address x.y.z.t/n** because we do not know anything about the network it is connected to at the other side.
- All messages **destined** for **addresses** in the **organization block** (205.16.37.32 to 205.16.37.47) are sent, to **x.y.z.t/n.**

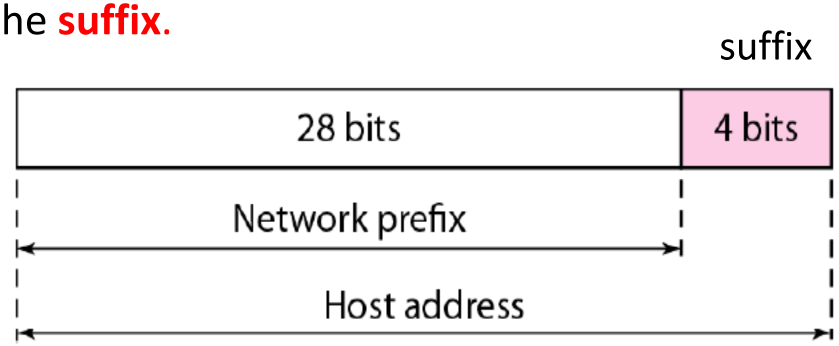
The first address in a block is normally not assigned to any device; it is used as the network address that represents the organization to the rest of the world.



Hierarchy of IP Addresses

Two-Level Hierarchy: No Subnetting

- An **IP address** can define only **two levels** of **hierarchy** when **not subnetted**.
- The ***n* leftmost bits** of the address ***x.y.z.t/n*** define the **network** (organization network).
- The **32 – *n* rightmost bits** define the **particular host** (computer or router) to the network.
- The two common terms are **prefix** and **suffix**.
- The part of the **address** that defines the **network** is called the **prefix**; the part that defines the **host** is called the **suffix**.



- The **prefix** is **common to all addresses** in the network; the **suffix changes** from one device to another.

Three-Levels of Hierarchy: Subnetting

- Suppose an **organization** is given the **block 17.12.14.0/26**, which contains **64 addresses**.
- The organization has **three offices** and needs to divide the **addresses** into **three sub blocks of 32, 16, and 16 addresses**.
- We can find the **new masks(subnet mask)** by using the following arguments:

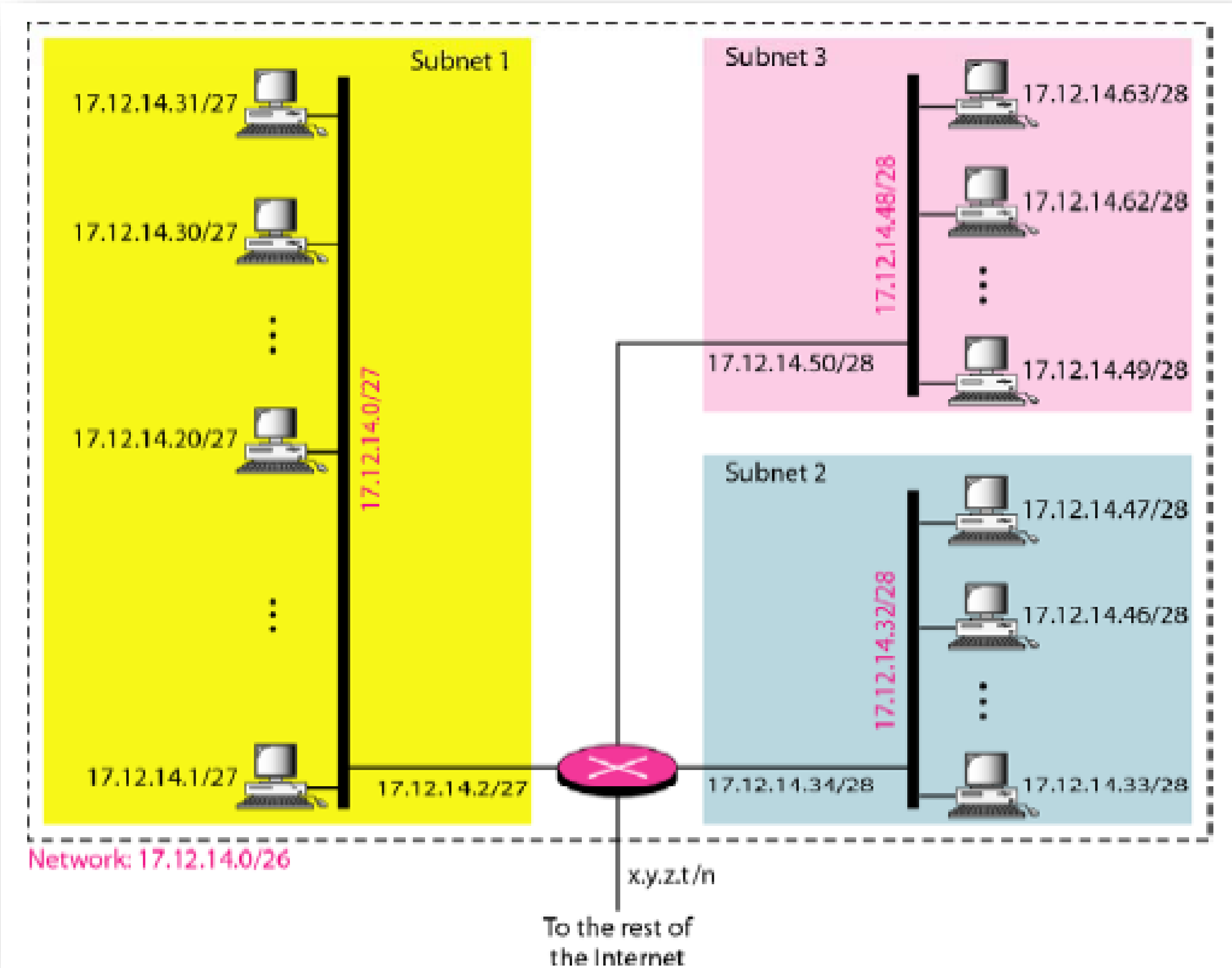
1. Suppose the mask for the **first subnet** is **n1**, then 2^{32-n1} must be **32,** which means that **n1 =27.**

2. Suppose the mask for the **second subnet** is **n2**, then 2^{32-n2} must be **16**, which means that **n2 = 28.**

3. Suppose the mask for the **third subnet** is **n3**, then 2^{32-n3} must be **16,** which means that **n3 =28**.

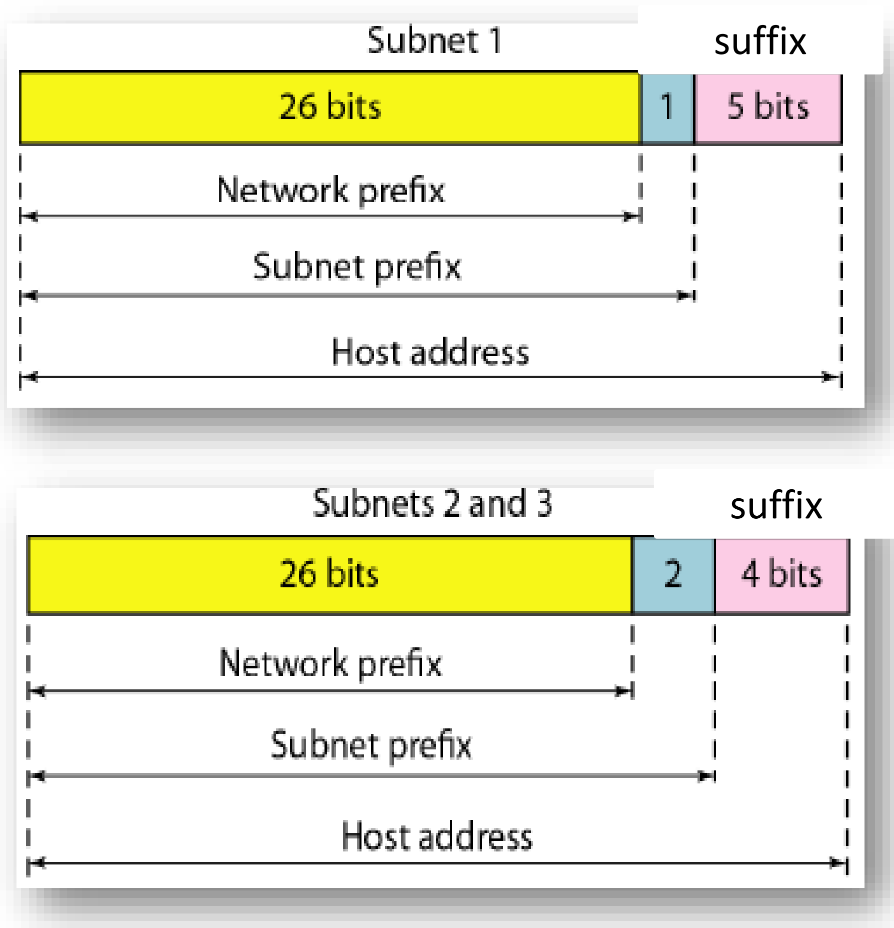
This means that we have the **subnet masks 27, 28, 28** with the **organization mask** being **26**.

Subnetting



Subnet 1
17.12.14.31/27
17.12.14.30/27
17.12.14.20/27
17.12.14.1/27
17.12.14.0/27
17.12.14.2/27
Subnet 3
17.12.14.63/28
17.12.14.62/28
17.12.14.48/28
17.12.14.50/28
17.12.14.49/28
Subnet 2
17.12.14.47/28
17.12.14.46/28
17.12.14.32/28
17.12.14.34/28
17.12.14.33/28
Network: 17.12.14.0/26
x.y.z.t/n
To the rest of
the Internet

Three-level hierarchy in an IPv4 address



Address Allocation

- The ultimate responsibility of **address allocation** is given to a **global authority** called the ***Internet Corporation for Assigned Names and Addresses*** (ICANN).
- However, **ICANN** does not normally allocate addresses to individual organizations.
- It assigns a **large block of addresses** to an **ISP.**
- Each **ISP**, in turn, **divides** its **assigned** **block** into **smaller** **sub blocks** and grants the **sub blocks** to its **customers.**
- In other words, an **ISP** **receives one large block** to be **distributed** to its **Internet users.**
- This is called **address aggregation**: many blocks of addresses are aggregated in one block and granted to one **ISP.**

Network Address Translation (NAT)

- The **number** of *home users* and *small businesses* that want to use the **Internet** is ever **increasing**.
- In the **beginning**, a user was connected to the **Internet** with a **dial-up line**, which means that he was connected for a **specific period of time**.
- An **ISP** with a **block of addresses** could **dynamically assign** an **address** to this **user.**
- An **address** was given to a **user** when it was needed, but the situation is different today.
- Home users and small businesses may **needs internet connectivity 24X7.**
- In addition, many are not happy with **one address**; many have created small networks with **several hosts** and need an **IP address** for **each host.**
- With the **shortage of addresses**, this is a **serious problem**.
- A **quick solution** to this problem is called **Network Address Translation (NAT)**.

Network Address Translation (NAT)

- **NAT** enables a **user** to have a **large set of addresses internally** and **one address, or a small set of addresses, externally**.
- The **traffic inside** can use the **large set;** the **traffic outside,** the **small set.**
- The **Internet authorities** have reserved **three sets of addresses** as **private addresses**, shown in **Table** below:

Table. Addresses for private networks

Range			*Total*
10.0.0.0	to	10.255.255.255	2^{24}
172.16.0.0	to	172.31.255.255	2^{20}
192.168.0.0	to	192.168.255.255	2^{16}

Network Address Translation (NAT)

- Any **organization** can **use** an **address** out of this set of **private addresses** **without permission** from the **Internet authorities**.
- As everyone knows that these **reserved addresses** are for **private networks**.
- They are **unique** **inside the organization**, but they are **not unique** **globally**.
- No **router** will **forward a packet** that has one of these **addresses** as the **destination address**.
- The **Organization site** must have a **connection** to the **global** **Internet** through a **router** that runs the **NAT software**.

Example

- Following **Figure** shows a simple **implementation of NAT**. As Figure shows, the **private network** uses **private addresses**.
- The **NAT router** that connects the **private network** to the Internet uses **one private address**(172.18.3.30) and **one global address**(200.24.5.8).
- The rest of the **Internet** sees only the **NAT router** with the address **200.24.5.8.**

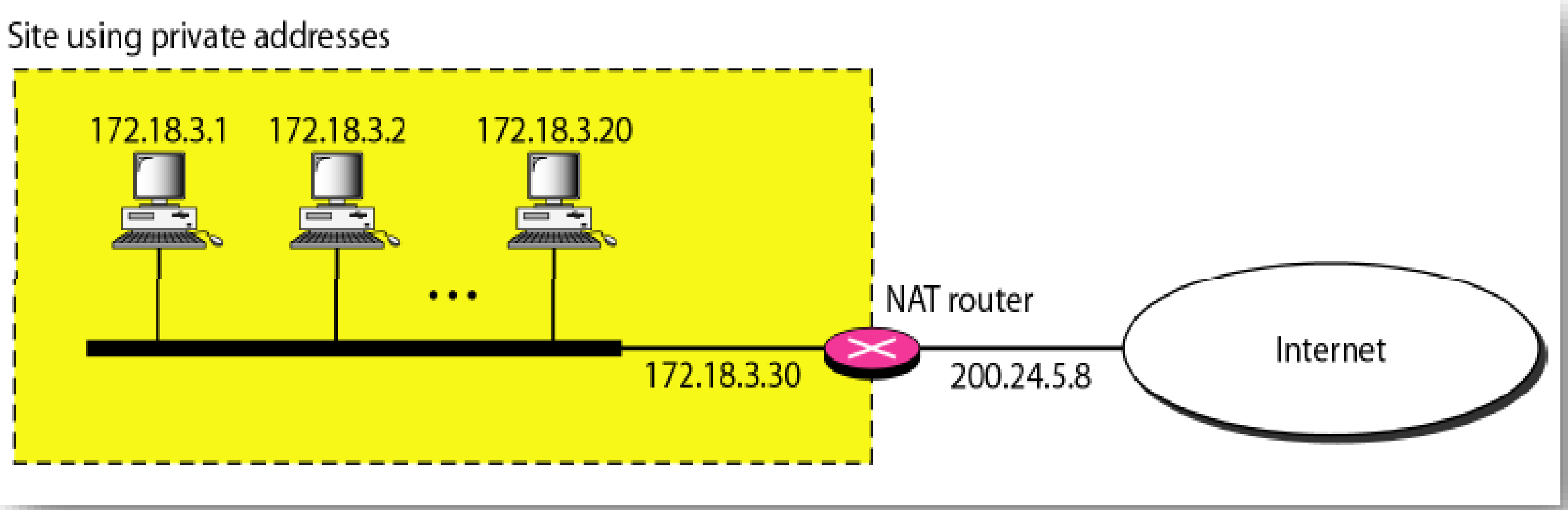
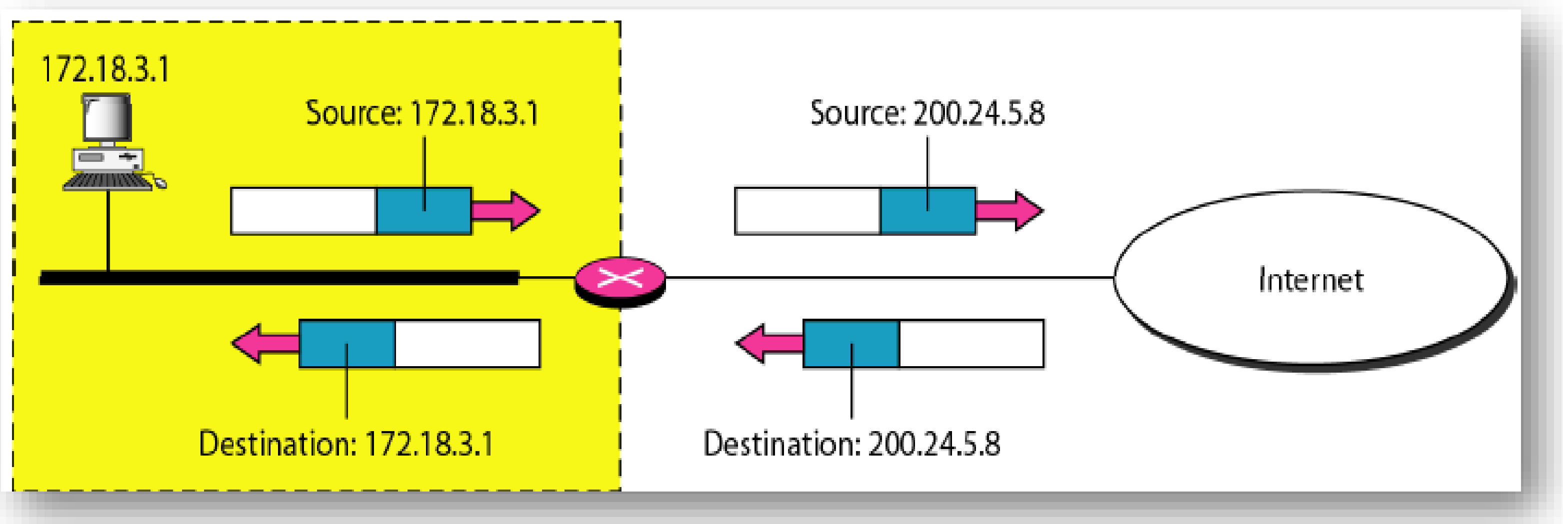


Figure. A NAT implementation

Address Translation

- All the **outgoing packets** go through the **NAT router**, which **replaces** the ***source address*** in the packet **with** the **global NAT address**(200.24.5.8).
- 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.**



Address Translation

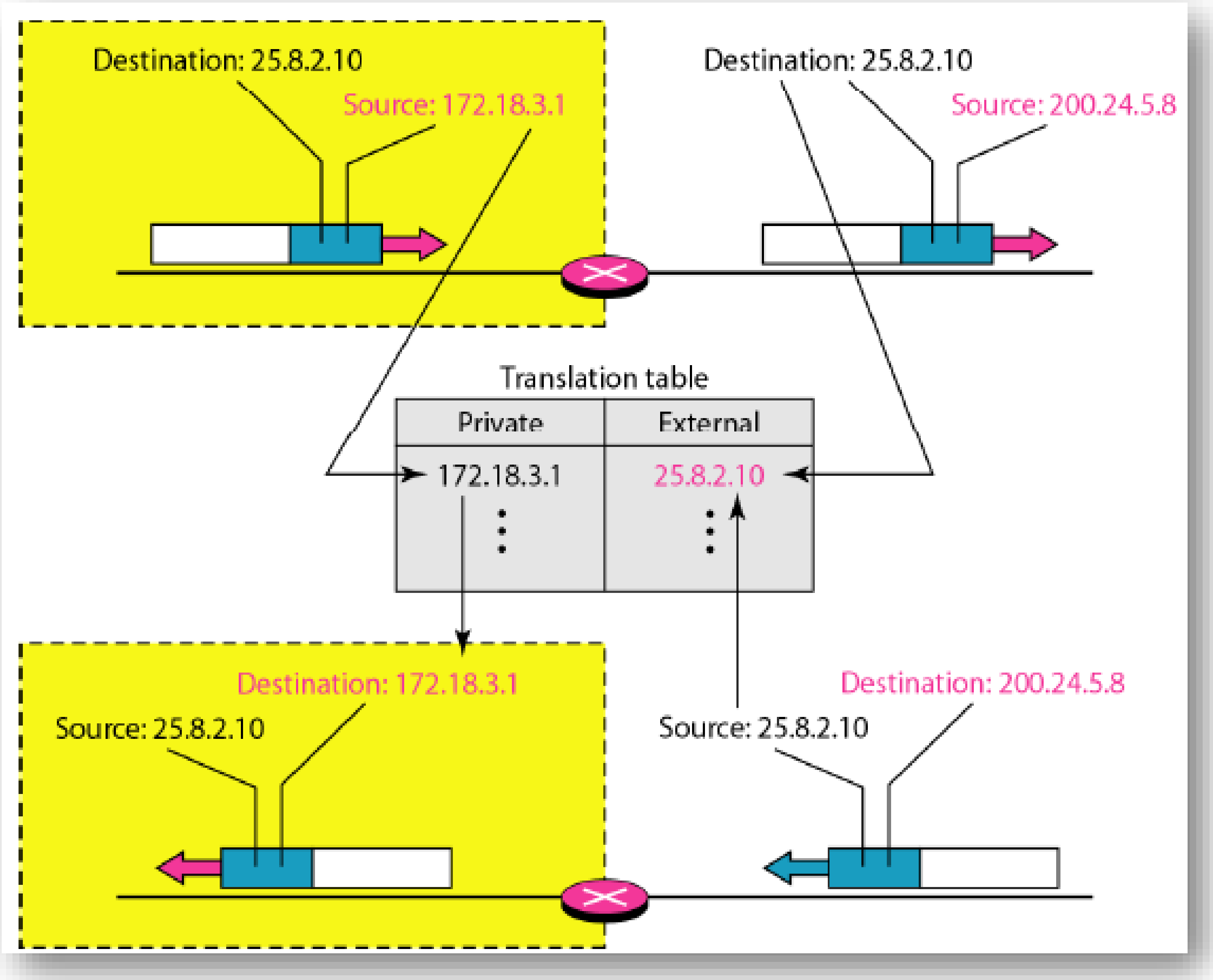
Translation Table

- **Translating** the **source addresses** for **outgoing packets** is **straightforward**.
- But **how does** the **NAT router know the destination address** for a **packet coming** from the **Internet?**
- There may be tens or hundreds of **private IP addresses**, each belonging to one specific host.
- The problem is solved if the **NAT router** has a **translation table**.

Case 1: Using One IP Address

- In its **simplest form**, a **translation table** has **only two columns**: the **private (source)address** and the **external address (destination address** of the packet).
- When the **router** translates the **source address** of the **outgoing packet**, it also makes **note** of the **destination address**-where the **packet** is **going.**
- When the **response comes back** from the **destination**, the router uses the **source address** of the **packet** (as the **external address**) to find the **private address** of the packet.

Case 1: Using One IP Address



Case 1:Using One IP Address

- In this strategy, **communication** must always be **initiated** by the **private network**.
- The **NAT mechanism** described **requires** that the **private network** **start the communication**.
- **Example:**
- **Communication** with the **Internet** is **always initiated** from the **customer site**, using a **client program** such as **HTTP, TELNET,** or **FTP** to access the corresponding **server program.**

Case 2: Using a Pool of IP Addresses

- Since the **NAT router** has only **one global address**, only **one private** network **host can access** the **same external host**.
- To **remove** this **restriction,** the **NAT router uses a pool** of **global addresses**.
- **For example,** instead of using only **one global address** (200.24.5.8), the **NAT router can use four addresses** (200.24.5.8, 200.24.5.9, 200.24.5.10, and 200.24.5.11).
- In this case, **four private network hosts** can communicate with the **same external host** at the **same time** because each pair of addresses defines a **connection.**
- However, there are **still some drawbacks**.
- In this **example, no more than four connections can be made** to the same **destination.**

Case 3: Using Both IP Addresses and Port Numbers

- To **allow a many-to-many relationship** between **private-network hosts** and **external server programs**, we need **more information** in the **translation table**.
- **Example**:
- Suppose **two hosts** with addresses **172.18.3.1** and **172.18.3.2** inside a **private network** need to access the same **HTTP server(port 80)** on **external host 25.8.3.2**.
- If the translation table has **five columns**, instead of **two**, that include the **source** and **destination port numbers** of the **transport layer protocol**, the ambiguity is eliminated.

Case 3: Using Both IP Addresses and Port Numbers

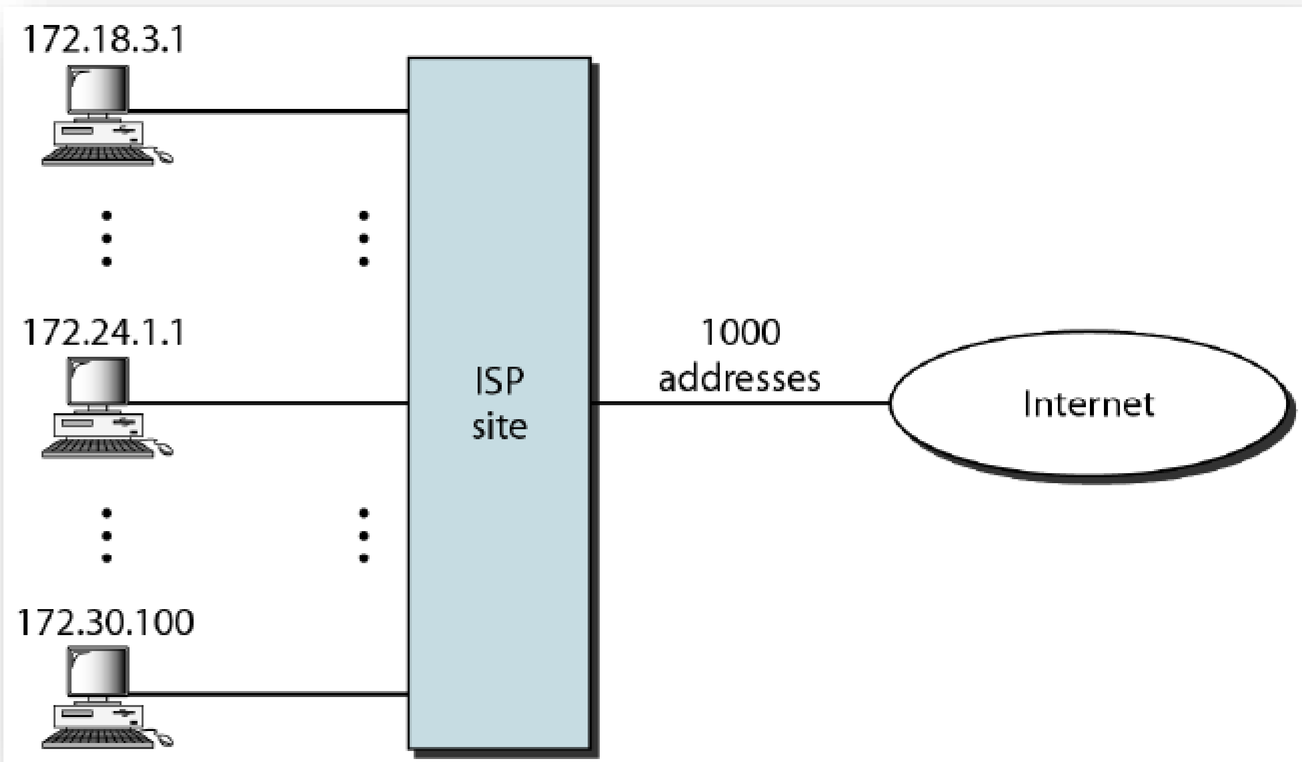
Table . Five-column translation table

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
...	...	...	...	...

NAT and ISP

- An **ISP** that serves **dial-up customers** can use **NAT technology** to **conserve addresses**.
- For **example**, suppose an **ISP** is granted **1000 addresses**, but has **100,000 customers**.
- Each of the customers is assigned a **private IP address**.
- The **ISP translates** each of the **100,000 source addresses** in **outgoing packets** to **one** of the **1000 global addresses**.
- It translates the **global destination address** in incoming packets to the corresponding **private address.**

NAT and ISP



172.18.3.1
172.24.1.1
172.30.100
ISP
site
1000
addresses
Internet

IPv6 ADDRESSES

- Despite all **short-term solutions**, such as **classless addressing** and **NAT, address depletion** is still a **long-term problem** for the **Internet.**
- Other **problems** in the **IP protocol** itself, such as :
  - ***lack of accommodation*** *for* ***real-time audio*** *and* ***video*** *transmission,*
  - **lack of encryption** and **authentication** of **data** for some applications,

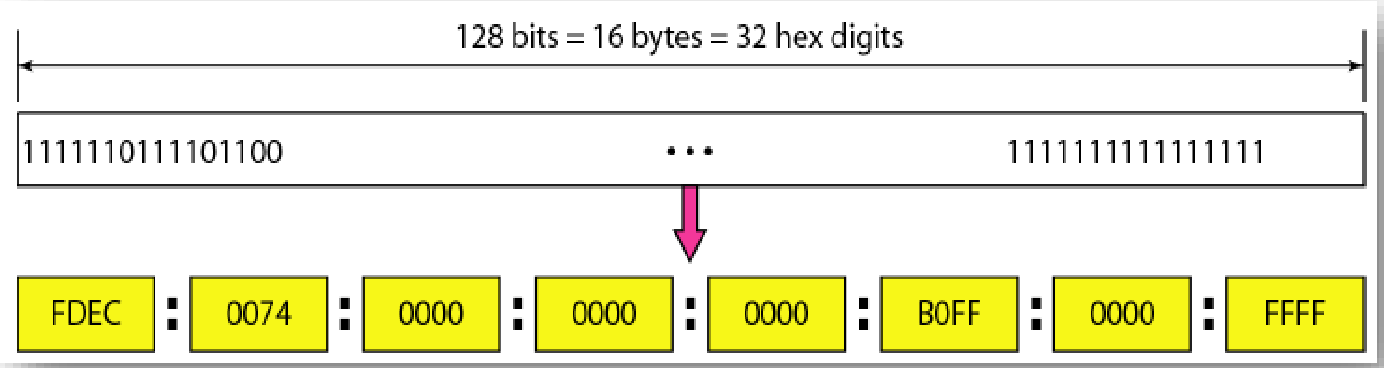
IPv6 Address Structure

- An **IPv6 address** consists of **16 bytes** (octets);
- It is **128 bits** long.

IPv6 ADDRESSES

Hexadecimal Colon Notation

- To make addresses more readable, **IPv6** specifies **hexadecimal colon notation**.
- In this notation, **128 bits** are divided into **eight sections**, each **2 bytes** in length.
- **Two bytes** in **hexadecimal notation** requires **four hexadecimal digits**.
- Therefore, the **address** consists of **32 hexadecimal digits**, with every **four digits separated** by a **colon**, as shown in Figure .



IPv6 ADDRESS SPACE

- **IPv6 addresses** are **128-bit** long, which means that there are **2^{128} =340 undecillion (1 followed by 36 zeros!)** possible addresses (the exact number is shown below).

  340,282,366,920,938,463,463,374,607,431,768,211,456

- The **Internet Assigned Numbers Authority (IANA)** allocates only a **small portion** of the whole **IPv6 space.**
- **IANA** provides **global unicast addresses** that start with leading leftmost bits **001.**
- A **small portion** of the **addresses** starting with **000** and **111** are allocated for **special types.**
- All **other possible addresses** are **reserved for future** use and are currently not being allocated.

IANA's Allocation of IPv6 Address Space