



# CS 524 A

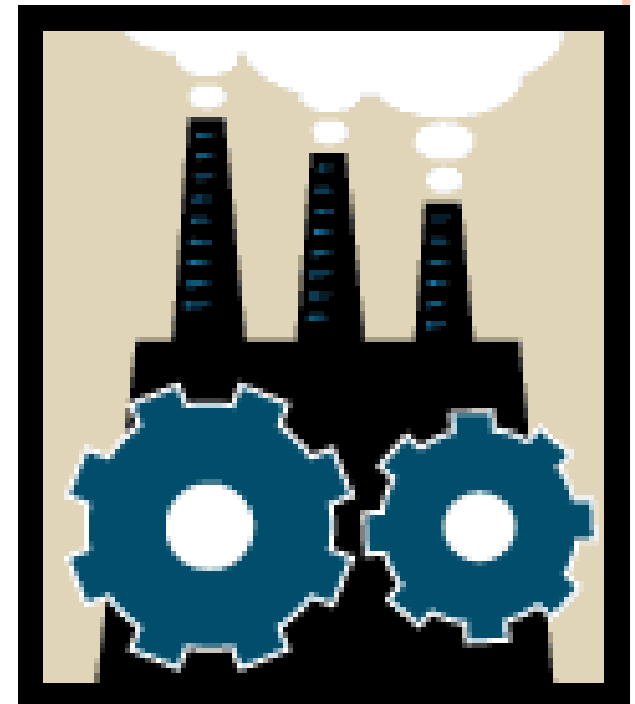
Introduction to Cloud Computing

**Module 6: Data Networking and Distributed Computation**  
(Part 2)

How the Cloud *pipes* are made

# OUTLINE

- The business side: New Cloud service Use Cases and examples
- A bit more detail on the Internet network layer
  - Subnets (a different meaning) and masks
  - *Autonomous Systems, Peering, and Border Gateway Protocol (BGP)*
  - *Private and Public* IP address spaces
  - *Network Address Translators (NATs)*
  - IPv6
- QoS
  - Packet Scheduling Disciplines
  - Integrated Services
  - Differentiated Services
  - Multi-Protocol Label Switching (MPLS)
  - Generalized MPLS
  - Virtual Private Networks



# CLOUD SERVICE EXAMPLES:

- Amazon Virtual Private Cloud (<http://aws.amazon.com/vpc/>)
- AWS Direct Connect (<http://aws.amazon.com/directconnect/>)
- Amazon S3 service transfer (<http://aws.amazon.com/s3/#importexport>)
- AT&T VPN Services (<http://www.business.att.com/enterprise/Family/network-services/ip-vpn/?GUID=BD979ABF-5055-4624-9529-9BA72B69AA5D&WT.srch=1>)
- You will find other examples as part of the homework

Question: What does *Netflix* use to stream movies?

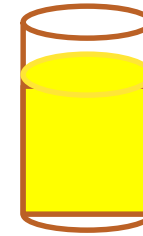
Answer: <https://aws.amazon.com/solutions/case-studies/netflix/>

# REMEMBER THE IP ADDRESSING SCHEME (IPv4)?

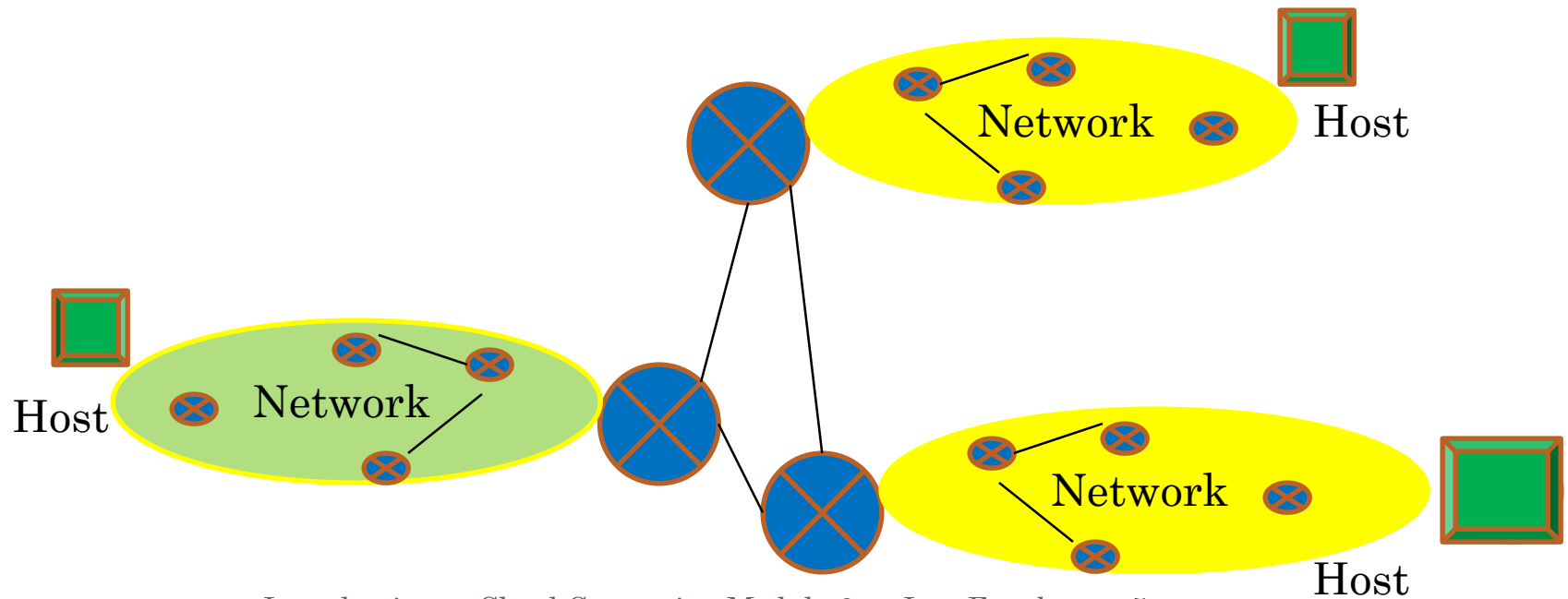
- IP addresses are assigned to all hosts and routers
- All IP addresses are 32-bit-long; they are normally written in decimal form byte-by-byte, separated by “.”s (e.g., 123.100.86.35)
- Each address has a form of **either**
  - <class> <network><host> (for classes A [0], B [10], and C [110]) **or**
  - 1110 <Multicast Address> (for class D [1110]) **or**
  - 11110 <Reserved for future use>

# THE (INITIAL) IP ADDRESS STRUCTURE AND ROUTING

1. Class (before CIDR)
2. Network address
3. Host address (within the network)



Pronounced  
*"cider"*



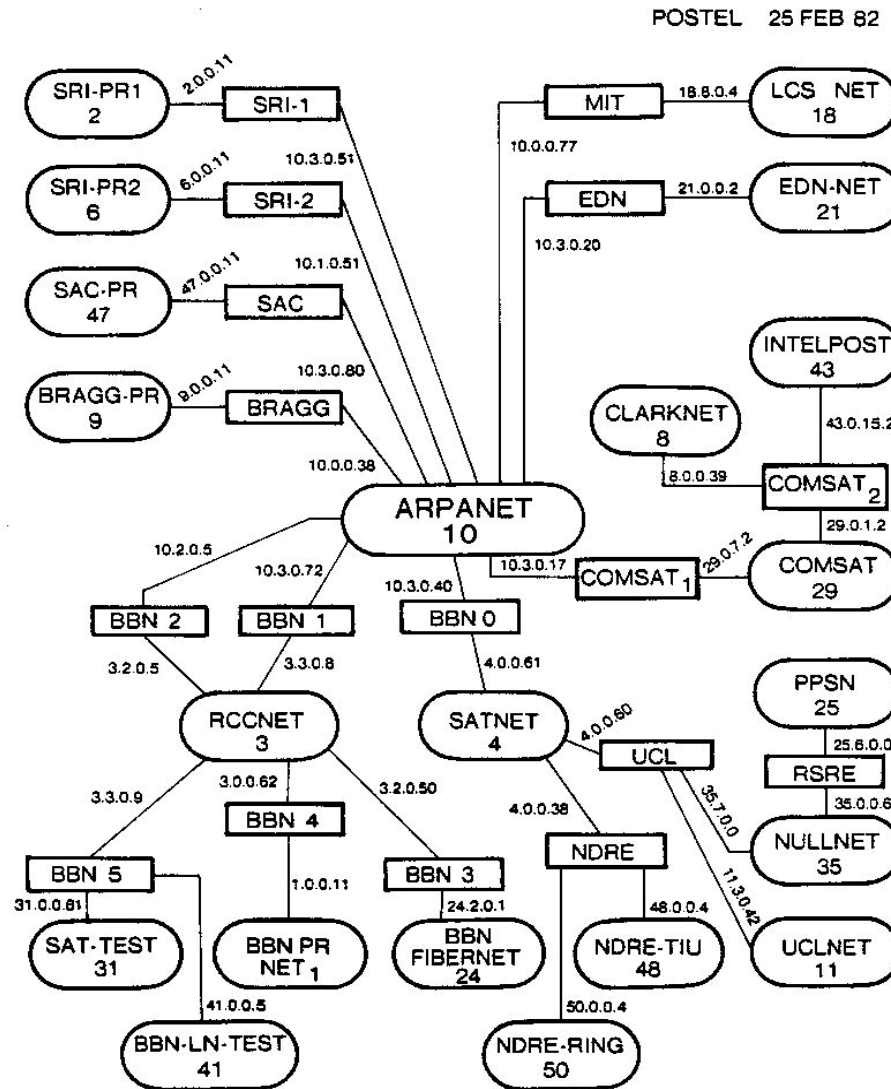
## AS DEFINED IN RFC 791

- With **Class A** [0] (intended for huge networks)
  - Network address is 7-bit long
  - Host address is 24-bit long
- With **Class B** [10] (intended for average-size networks)
  - Network address is 14-bit long
  - Host address is 16-bit long
- With **Class C** [110] (intended for small networks)
  - Network address is 21-bit long
  - Host address is 8-bit long

The idea: The smaller the network, the more of them will be out there!

# Jon Postel's map of the Internet in 1982

(Source: [http://commons.wikimedia.org/wiki/File%3AInternet\\_map\\_in\\_February\\_82.png](http://commons.wikimedia.org/wiki/File%3AInternet_map_in_February_82.png)  
By Jon Postel [Public domain], via Wikimedia Commons)

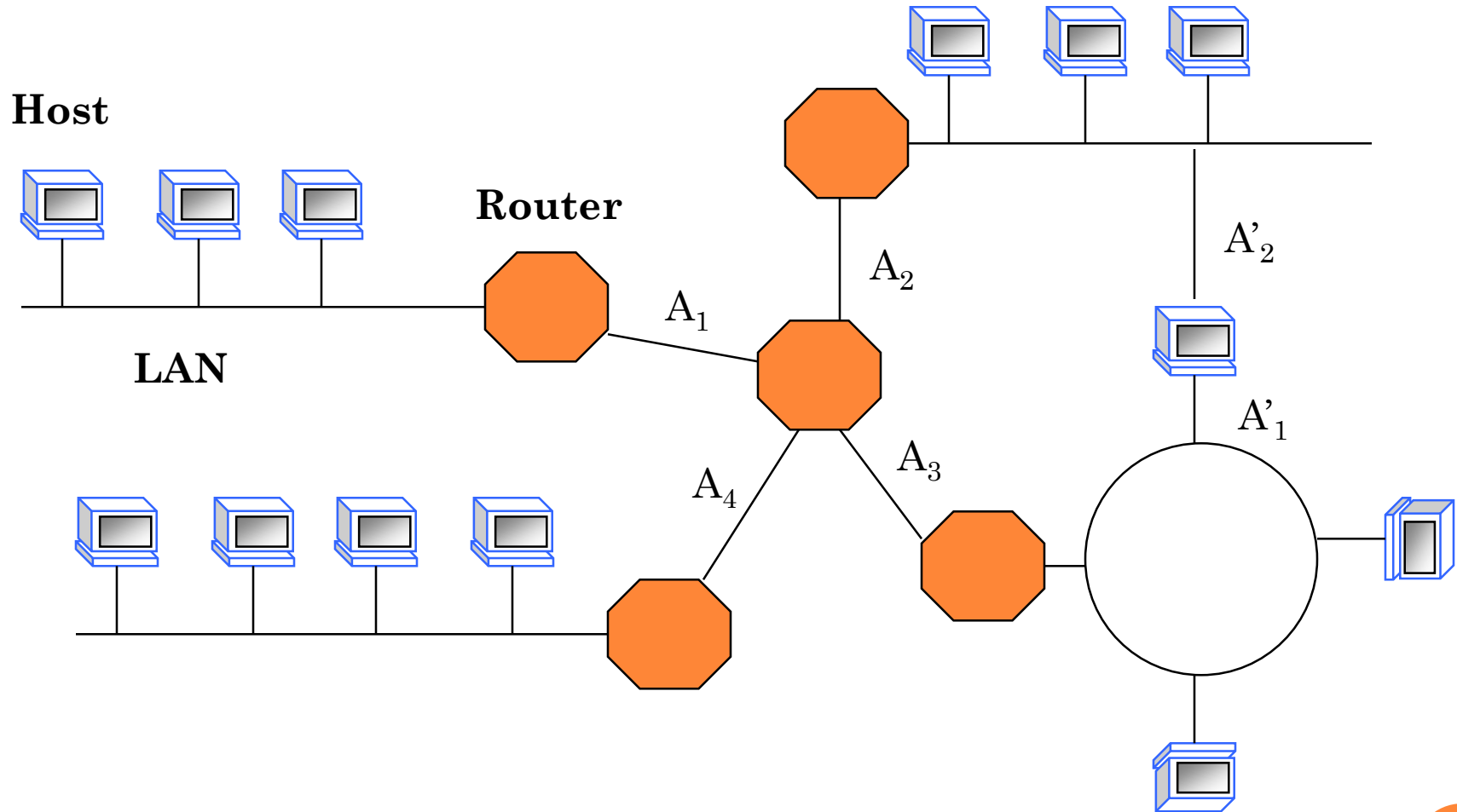


# CLASS-BASED IP ADDRESS ASSIGNMENT PROBLEM

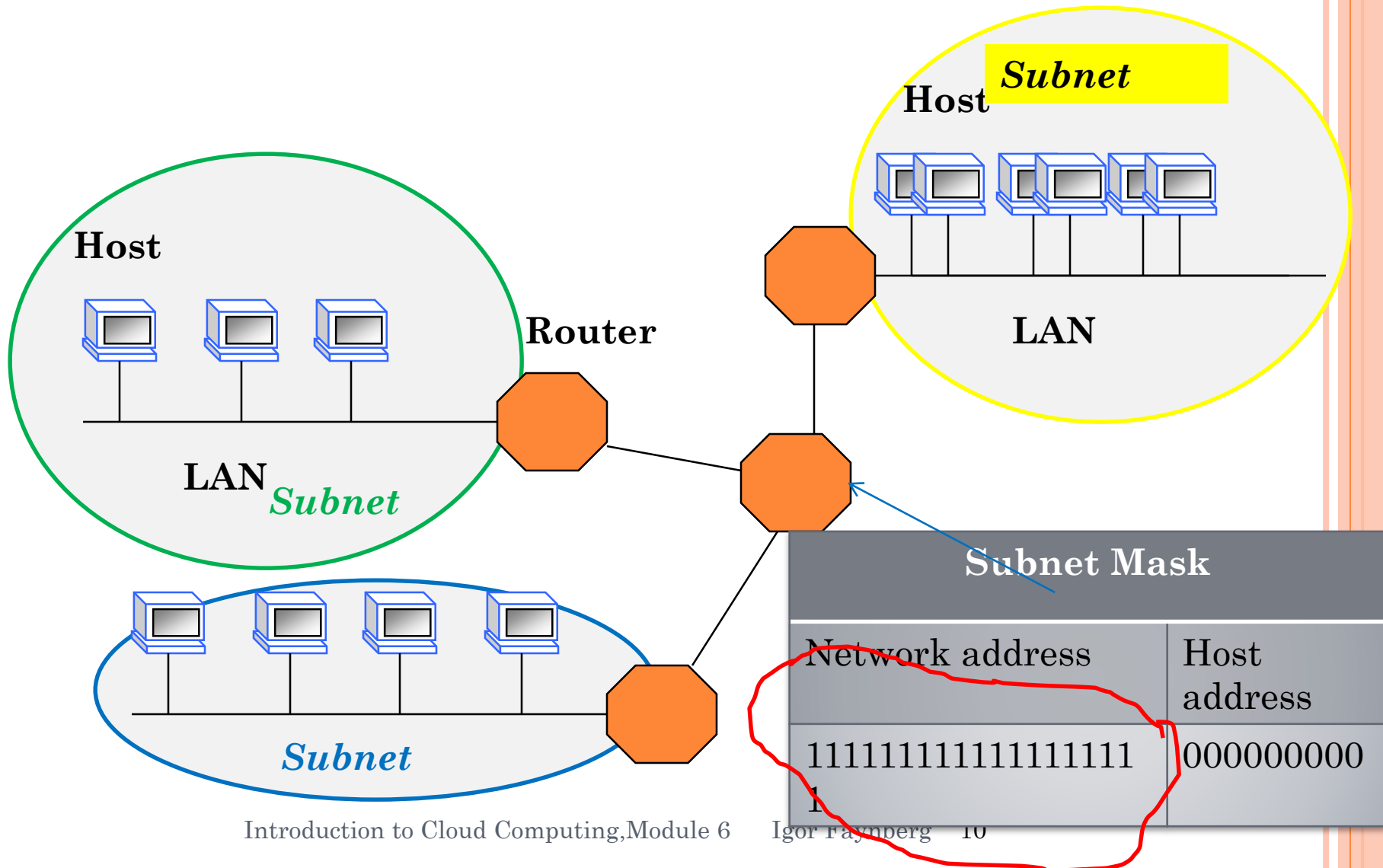
- Say, Company *X* has been allocated  $2^{16}$  (65 536) host addresses for its *Class B* network. The network started with **one** LAN and only 200 hosts)
- As more departments are added, each gets its own LAN and more hosts, but only so many L2 bridges can be put in...
- And so the new LANs need to attach to routers and become *separate* networks
- Now, the company has 10,000 hosts, but it cannot do much with the remaining addresses because they are on the *same* network
- Getting new addresses is problematic (and expensive), and the old ones have not been even used!



# THE RULE: A SEPARATE IP ADDRESS FOR EACH NETWORK INTERFACE



# NEW (*SUBNET* IS DEFINED BY THE *MASK*)

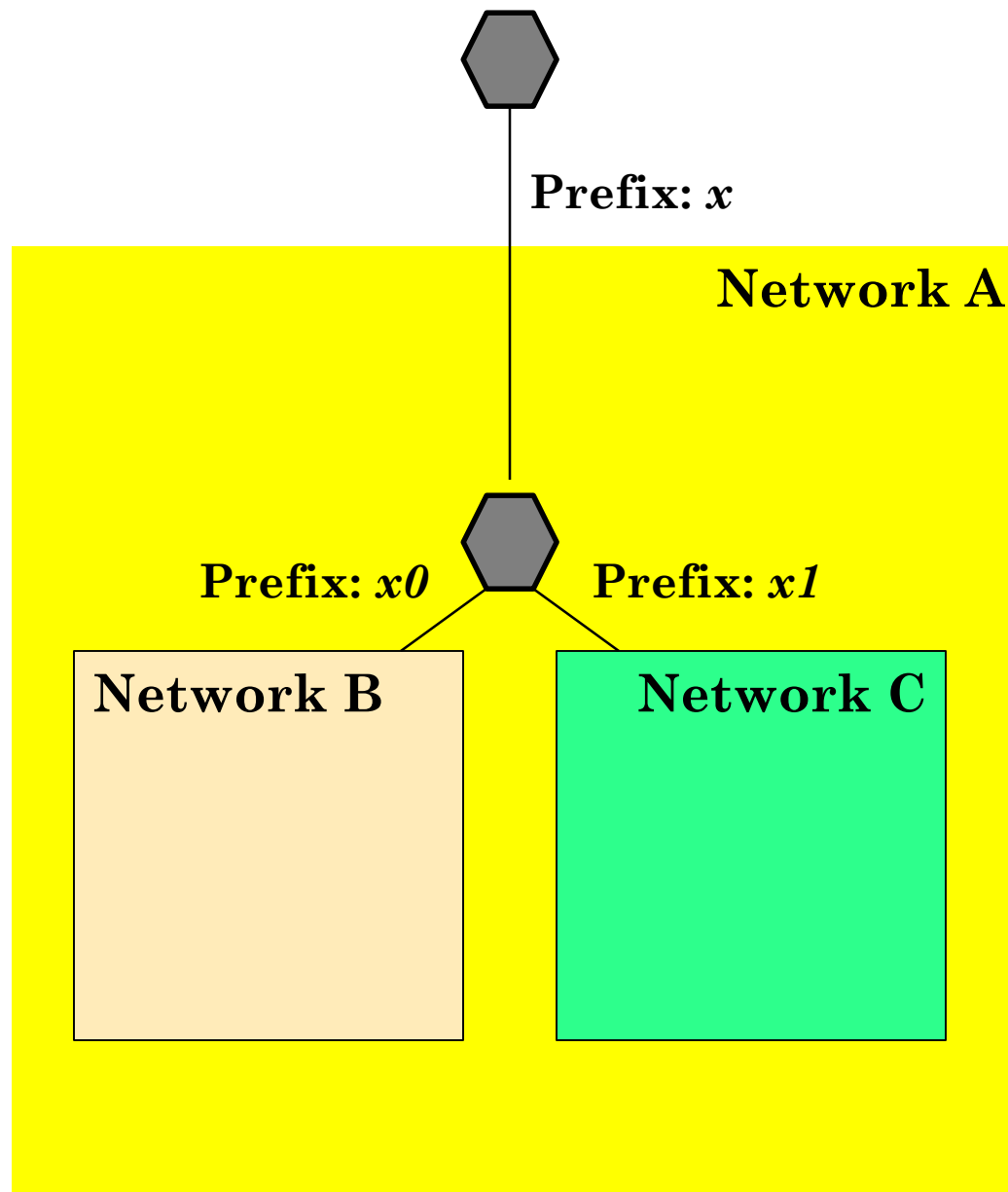


# MASKS AND ROUTING TABLES

- Masks are shown with a “dash notation” indicating the number of bits in the network address: **124.32.17.21/16**
- Now *Company X* can subdivide its address space into subnets
  - /17 (half of all addresses)
  - /18 (a quarter of all addresses)
  - /19 (an eighth part of addresses)
- With that, the edge router of Company X **does not need to** keep all hosts addresses; it keeps only *prefixes* for the subnet:

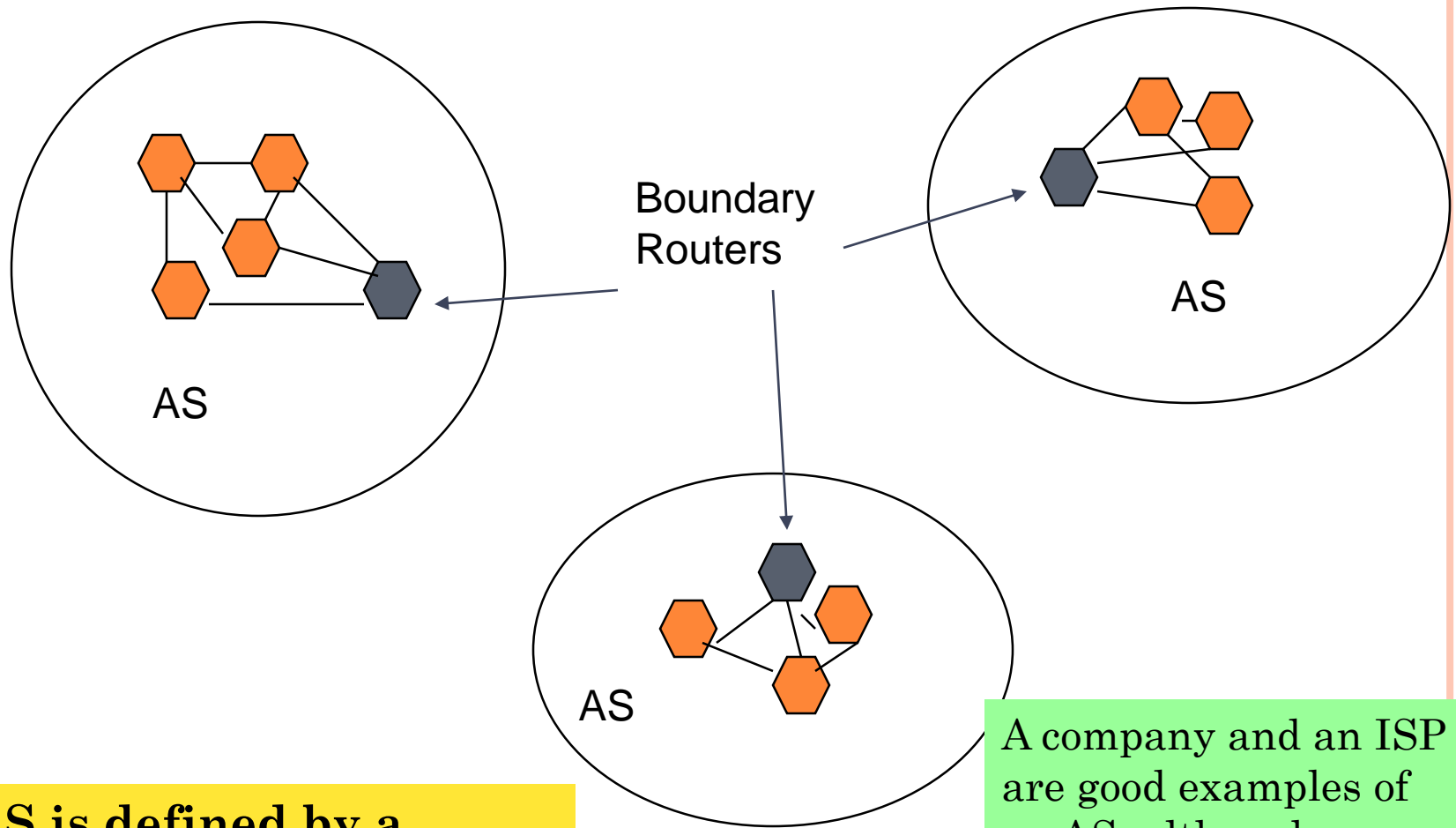
*Prefix = destination address AND mask*

- *Classless Inter-Domain Routing* (CIDR) system is built on the same prefixing idea: The routers *aggregate* and propagate an IP packet in the direction of the longest prefix (*supernet*)



**Figure 4.11: CIDR Aggregation**

# NEXT TOPIC: AUTONOMOUS SYSTEMS



**An AS is defined by a policy.**

A company and an ISP are good examples of an AS, although an ISP may have more than one AS

# THE BORDER GATEWAY ROUTING PROTOCOL (BGP)

(Some aspects of this you will encounter in the homework, as you will study the *Amazon VPC* service)

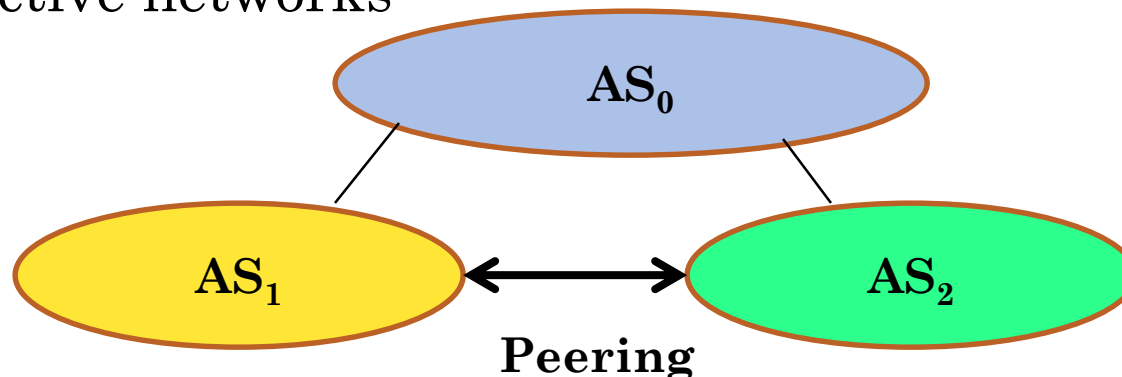
- The current version (4) of BGP is defined in RFC 4271 (<http://tools.ietf.org/html/rfc4271>) and other BGP RFCs that you may need to take a look at
- BGP runs on top of TCP (why?)
- **Routing constraints** are expressed by *policies*.

Examples:

- Do not traverse network *A* when traffic originates and terminates in network *B*
- When it is possible to reach a destination through either networks *C* or network *D*, always choose *D*
- Don't use network *X* in country *Y*

# ROUTING POLICIES ARE DEFINED ACCORDING TO BUSINESS NEEDS

- **Business:** who pays *whom* and *what* for *transit service*
- $AS_1$  and  $AS_2$  may start by paying  $AS_0$  for transit
- When  $AS_1$  and  $AS_2$  find that they need to exchange a lot of data, they may avoid services of  $AS_0$ , connect their networks, and use the *peering policy* to reduce their bills
- To implement *peering*,  $AS_1$  and  $AS_2$  advertise to each other (via BGP) the addresses that reside in their respective networks



# MULTIHOMING AND AUTONOMOUS SYSTEM NUMBERS (ASNs)

- When a network's router is connected to more than one *network provider*, this is called *multihoming*
- *RFC 1930* (<http://tools.ietf.org/html/rfc1930>) defines *multihoming* as a major case for having a private autonomous system number (ASN)
- The ASN uniquely identifies each network on the Internet.
- These numbers (started with 16-bit sizes, which grew to 32 bits) are assigned in blocks by the *Internet Assigned Numbers Authority* (IANA) to *Regional Internet Registries* (RIRs)
- A unique ASN is assigned to each AS by an RIR for use in BGP routing
- An ASN fully identifies each AS on the Internet.

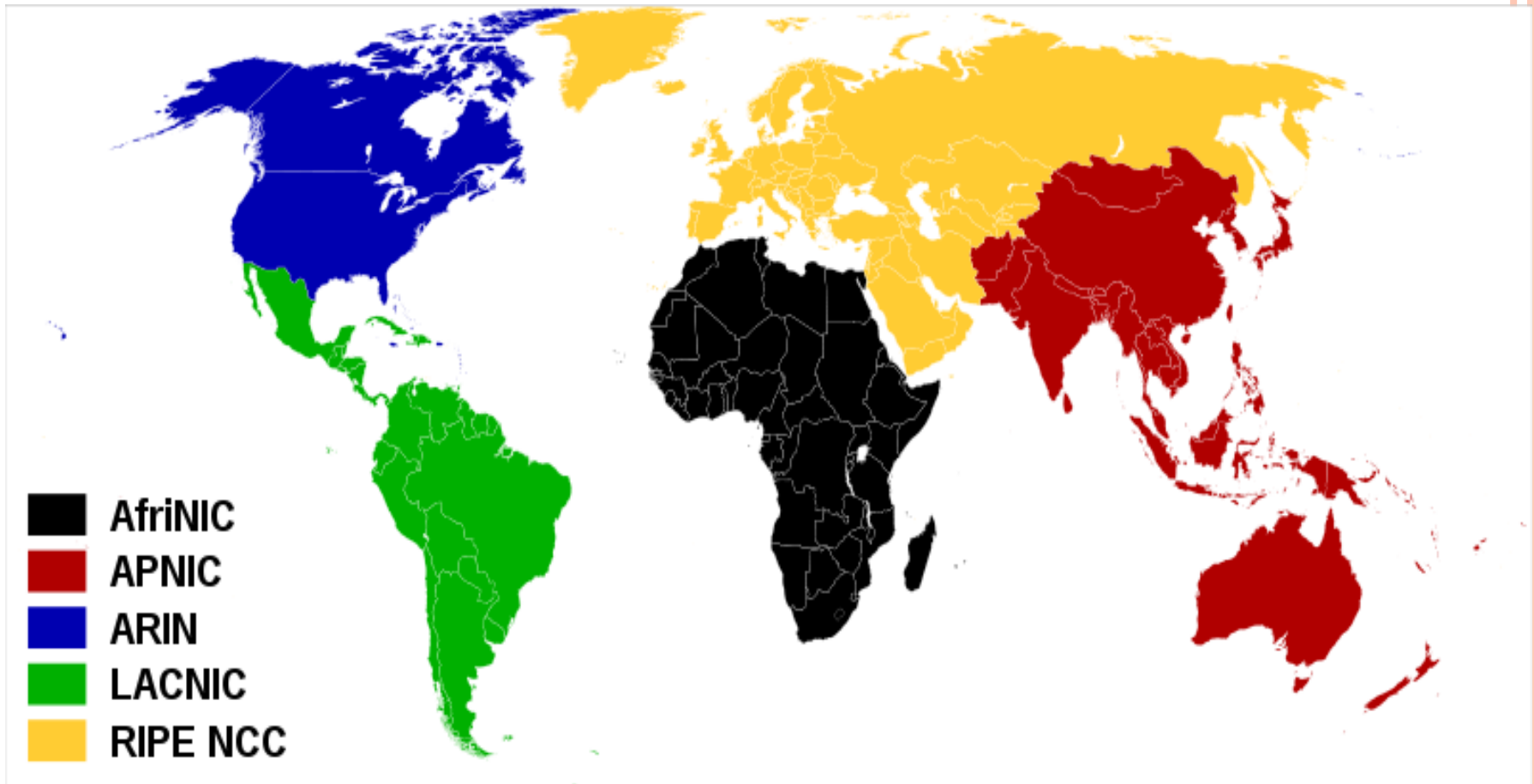


# PRESENT *REGIONAL INTERNET REGISTRIES (RIRs)*

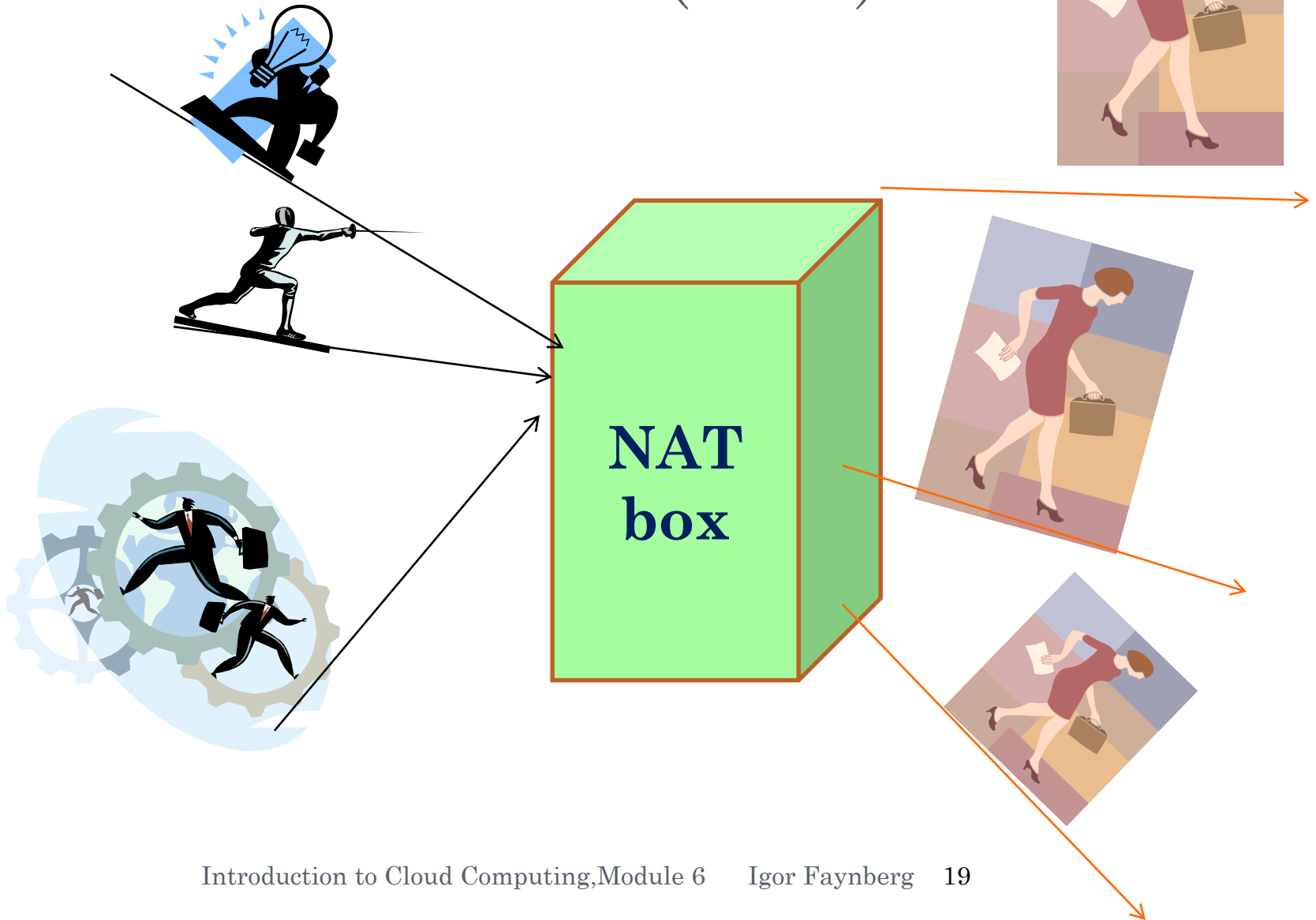
- African Network Information Centre (AfriNIC) for Africa
- American Registry for Internet Numbers (ARIN) for the United States, Canada, several parts of the Caribbean region, and Antarctica.
- Asia-Pacific Network Information Centre (APNIC) for Asia, Australia, New Zealand, and neighboring countries
- Latin America and Caribbean Network Information Centre (LACNIC) for Latin America and parts of the Caribbean region
- Réseaux IP Européens Network Coordination Centre (RIPE NCC) [for Europe, the Middle East, and Central Asia

# GEOGRAPHIC DISTRIBUTION OF RIRs

(SOURCE: WIKIMEDIA)



# NEXT TOPIC: NETWORK ADDRESS TRANSLATORS (NATs)



# IPv4 HEADER

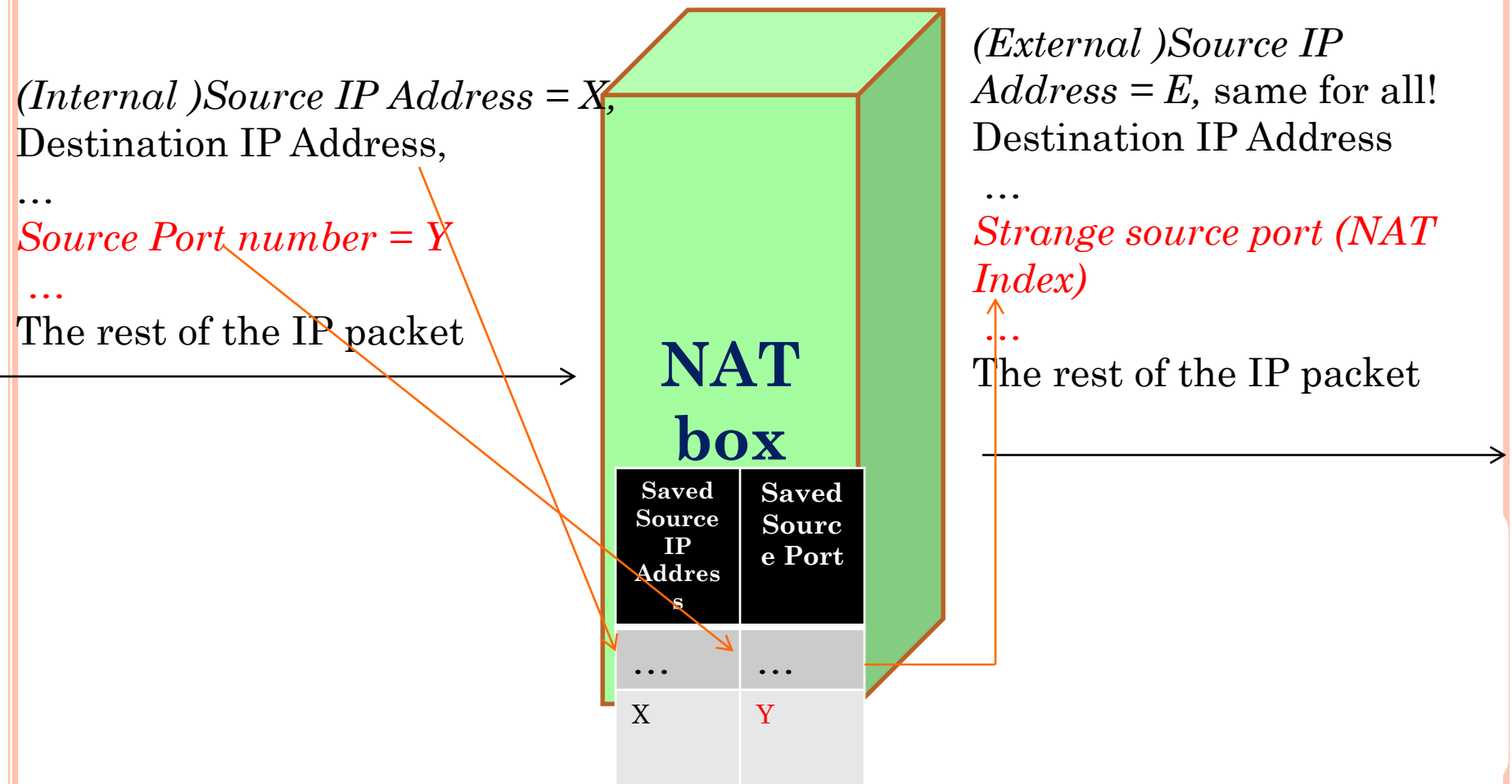
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31			
Version				IHL (Header Length)				TOS (Differentiated services)								Total length (in bytes)																		
Identification (common to all fragments)																Used	DF	MF	Fragment offset															
Time to live								Protocol								Header Checksum																		
Source Address																																		
Destination Address																																		
Options (0 to maximum length)																																		
...																																		

## WITH A 32-BIT IP ADDRESS FIELD...

There are only 4 294 967 296 IP addresses

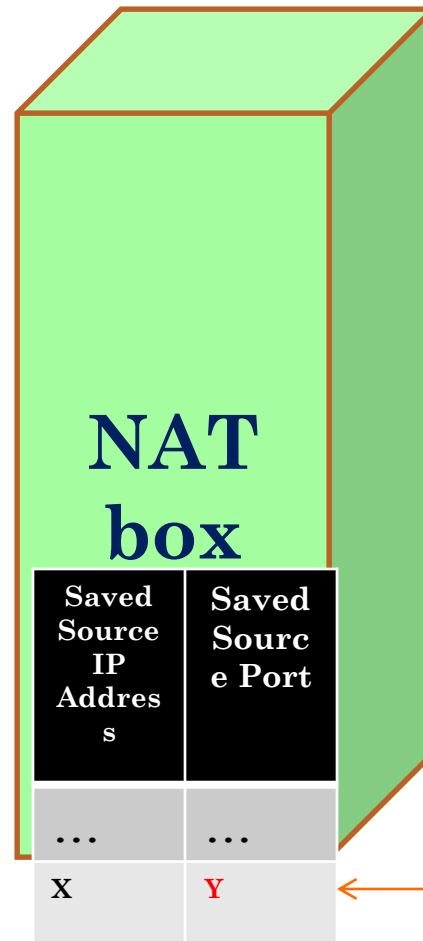
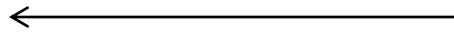
- Not enough for every person's computer, phone, washing machine, tooth pick, etc...
- Hence *Network Address Translation (NAT)* boxes which
  - are (typically) combined with firewalls
  - serve two *purposes*: 1) effective management of scarce IP addresses and 2) **obfuscation of the internal network architecture**

# HOW NAT WORKS (OUTGOING TRAFFIC)



# HOW NAT WORKS (INCOMING TRAFFIC)

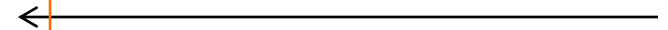
(Internal )Source IP Address,  
Destination IP Address =X,  
...  
*Source Port number = Y*  
...  
The rest of the IP packet



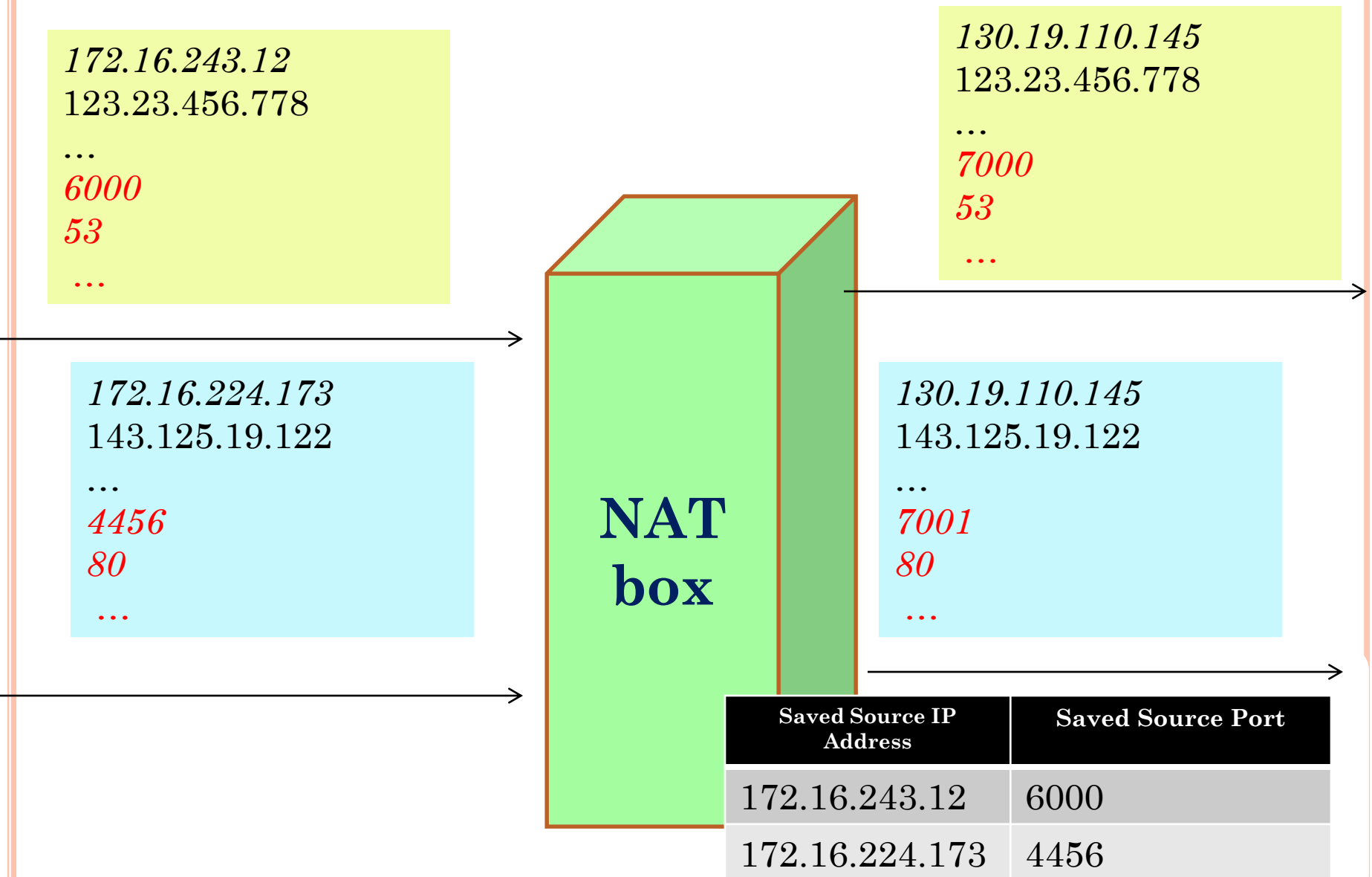
Source IP Address,  
*E*

...  
*NAT index*

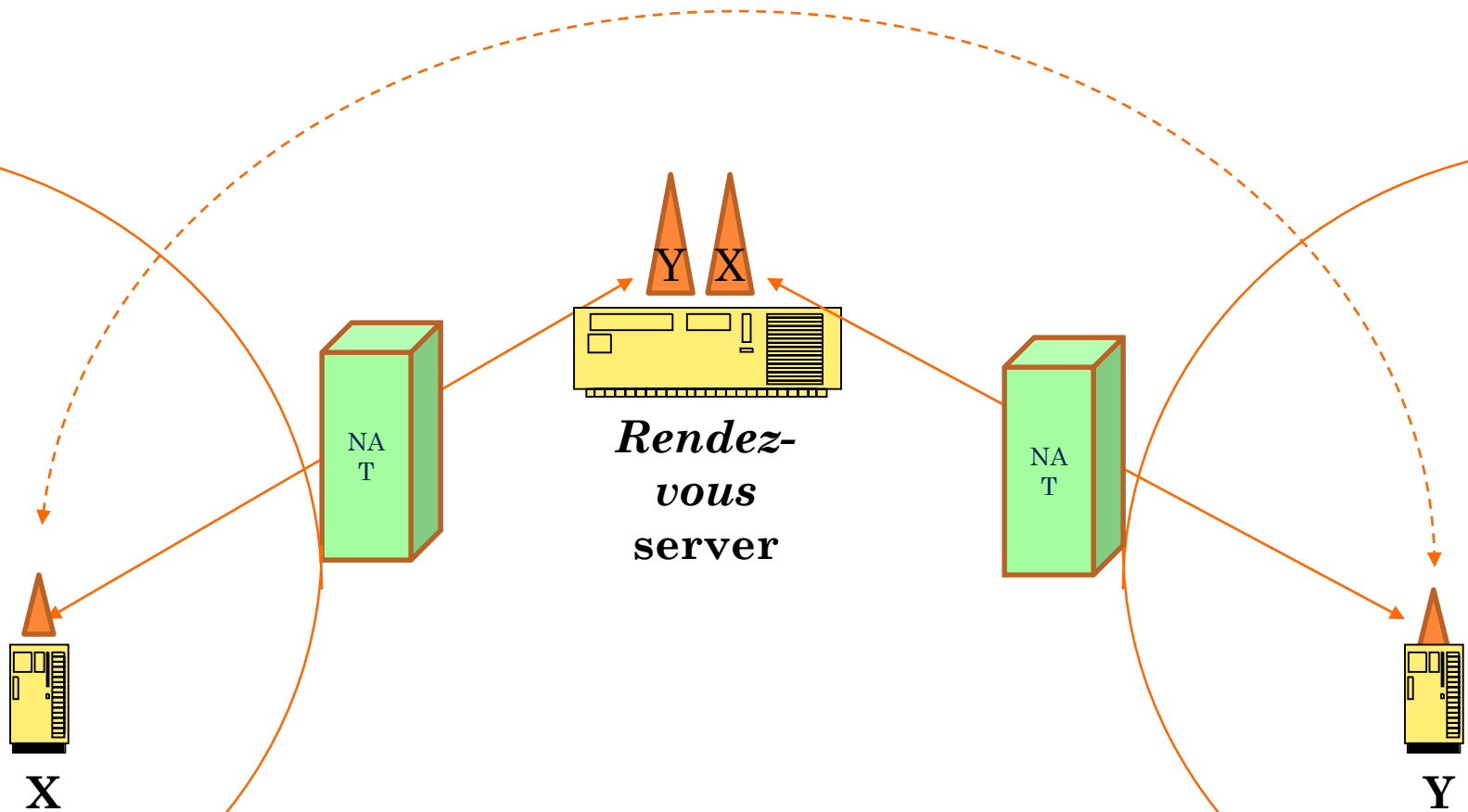
...  
The rest of the IP packet



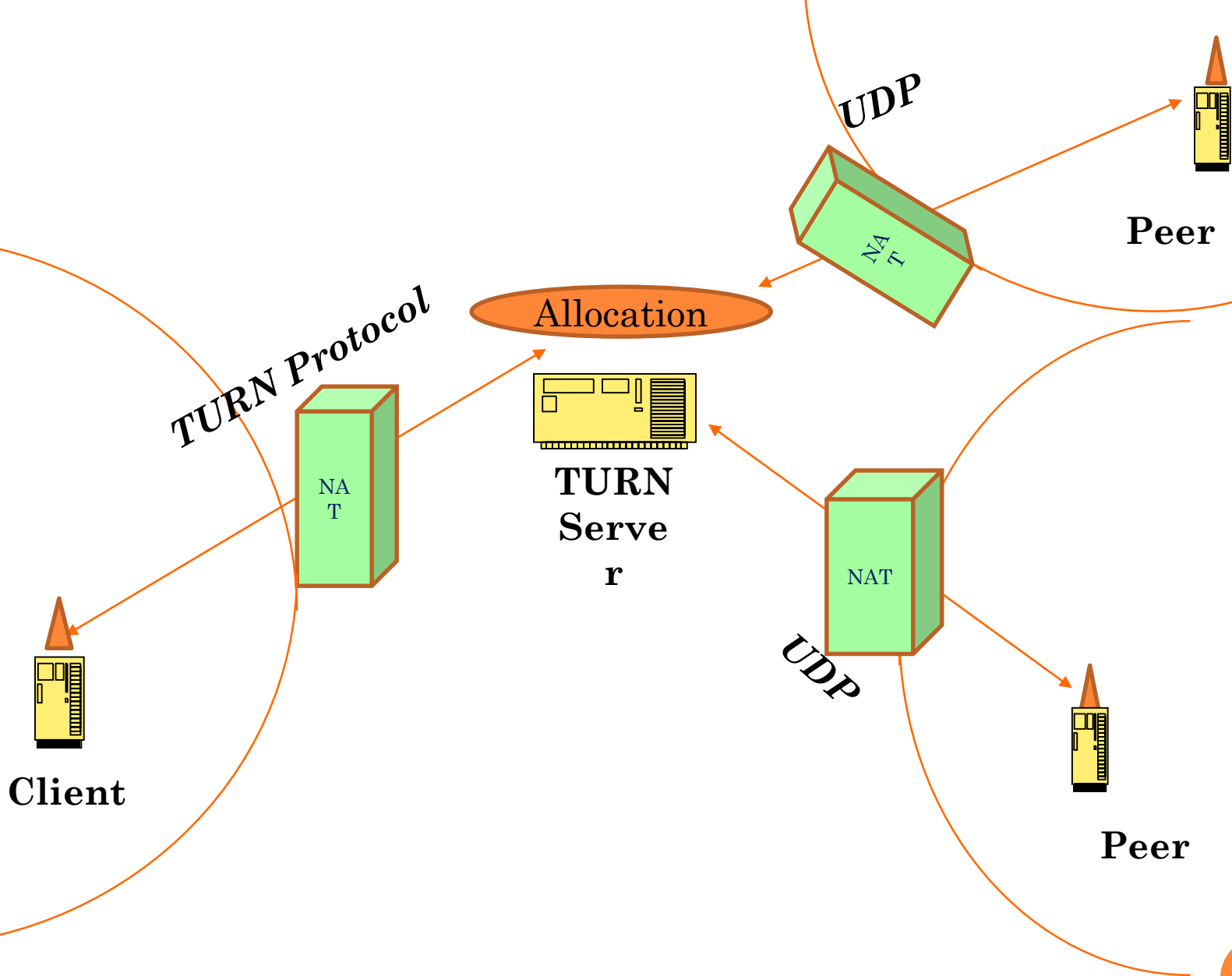
# AN EXAMPLE (OUTGOING TRAFFIC)







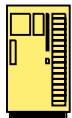
## Working around NATs: A *rendez-vous* relay



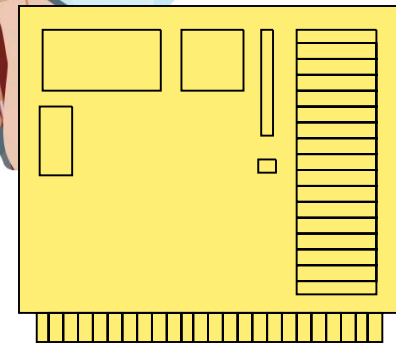
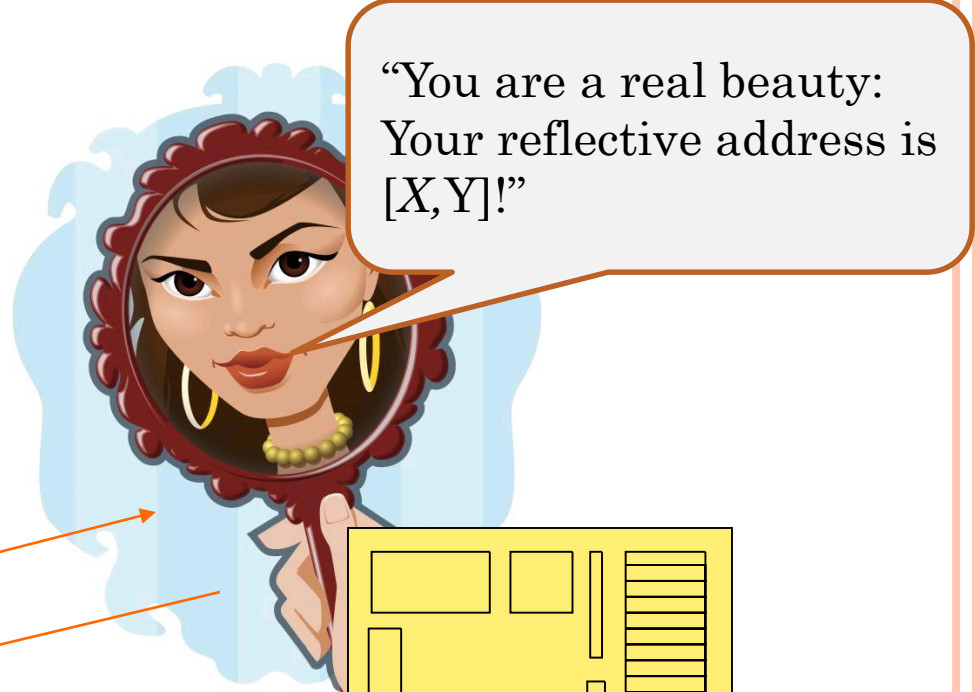
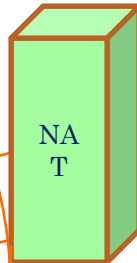
# Working around NATS: Traversal Using Relays around NAT (TURN)

“STUN, STUN on the wall!..”

“You are a real beauty:  
Your reflective address is [X,Y]!”

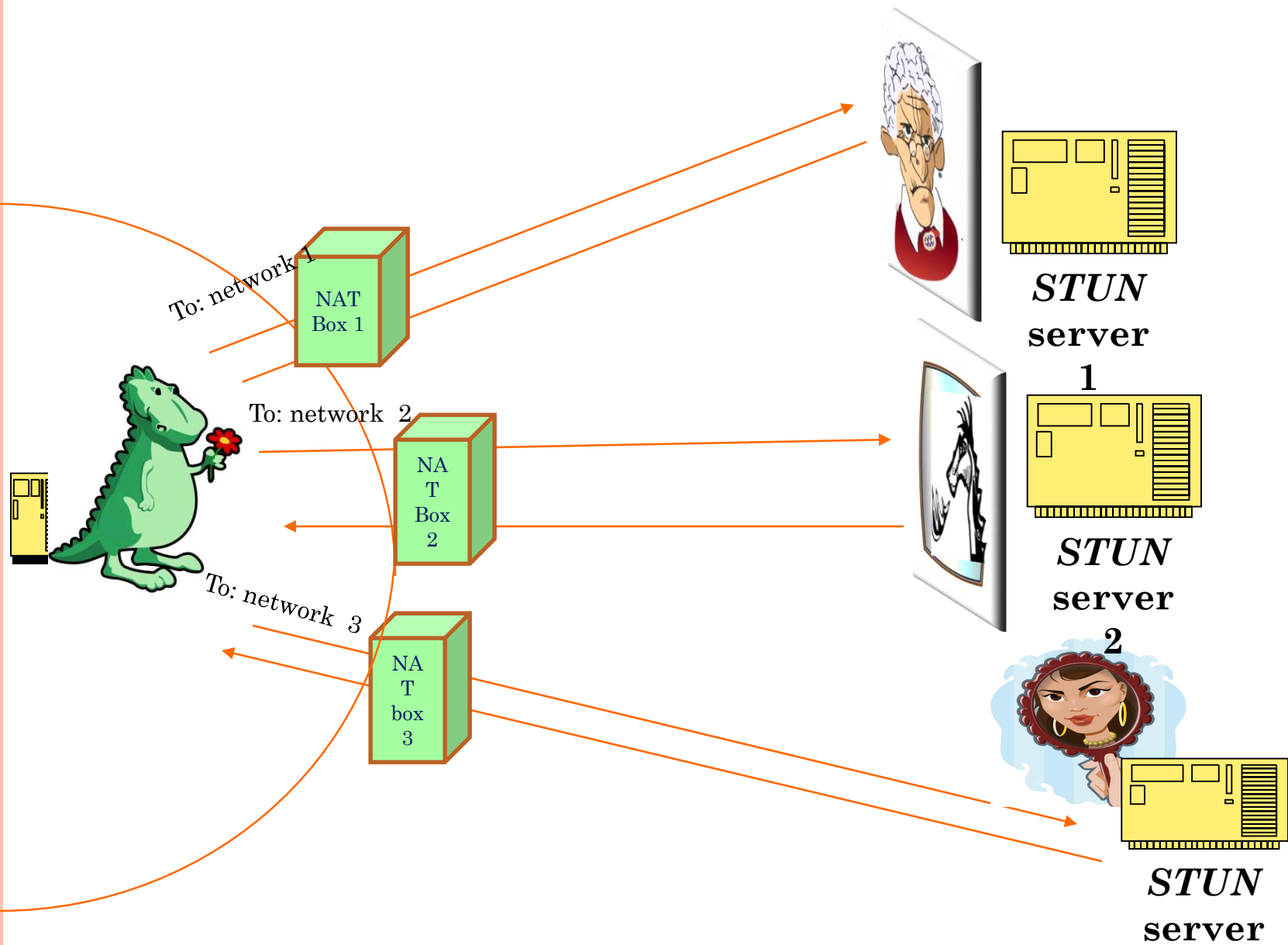


X



*STUN* server

Working Around NATs: Learning the reflective address from a STUN server



**Working around NATs: Different NATs for different paths<sup>3</sup>...**

# NATs

- **Solve** the IPv4 address shortage problem
- **Help** re-using addresses (<http://tools.ietf.org/html/rfc1918>)
- **Hide** internal network structure (an essential security measure)
- **Violate** modularity (by assuming the transport layer addressing structure and interfering with the transport header)
- **Violate** an Internet principle that requires a unique IP address for *every* host on the Internet
- **Violate** an Internet principle that no *connection state* be kept in the network
- **Violate** an Internet end-to-end connectivity principle: Host *A* behind NAT can *not* receive packets from Host *B* until after it had send a packet to node *B*
- **Interfere** with those applications protocols (such as SIP, SDP, FTP...) that carry IP addresses in payload—this gave birth to NAT traversal technology (ICE/STUN)
- **Delay** the IPv6 deployment

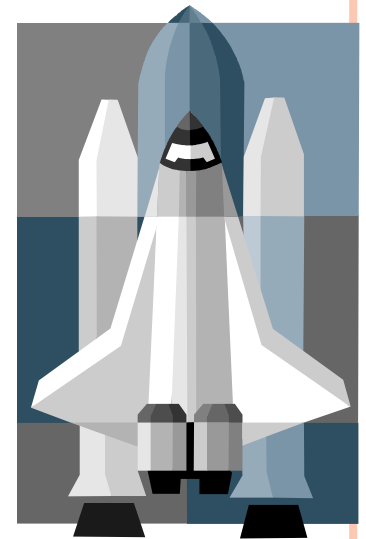
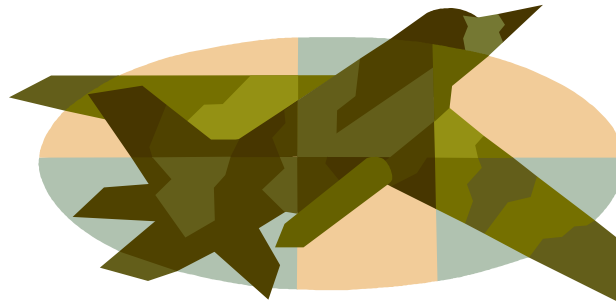
# IPv6 (MAIN) HEADER

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31				
<i>Version</i>				Differentiated services								<i>Flow Label</i> (to prevent layering violation, which we will study later)																							
<i>Payload length</i>																<i>Next Header</i>										<i>Hop limit</i>									
<i>Source Address</i>																																			
(128 bits)																																			
<i>Destination Address</i>																																			
(128 bits)																																			

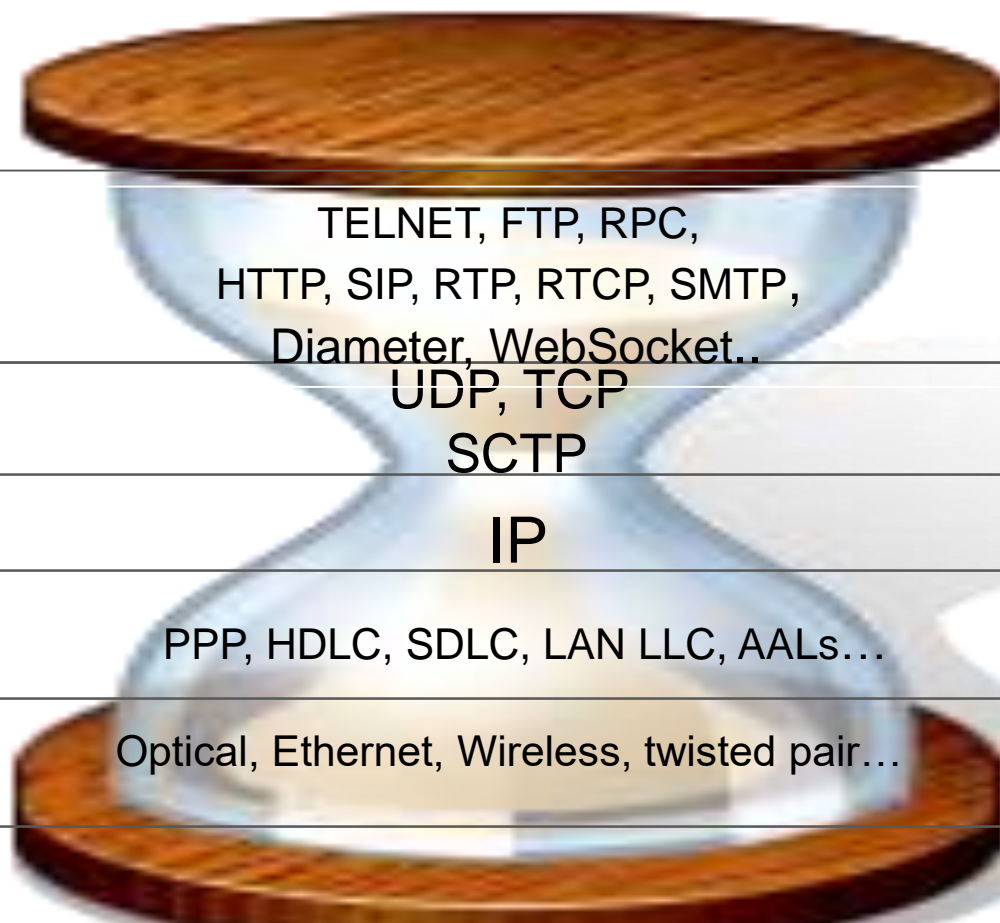
With 128-bits, we have  $3.4028236789088481 \times 10^{38}$  IP addresses

A question: Will we still need NATs?

# NEXT TOPIC: QUALITY OF SERVICE (QoS) IN THE NETWORK



# The Internet Protocol Hourglass (after Steve Deering)



Application	TELNET, FTP, RPC, HTTP, SIP, RTP, RTCP, SMTP, Diameter, WebSocket...
Transport	UDP, TCP SCTP
Network	IP
Link	PPP, HDLC, SDLC, LAN LLC, AALs...
Physical	Optical, Ethernet, Wireless, twisted pair...

AAL: ATM Adaptation Layer  
 HTTP: Hyper Text Transfer Protocol  
 HDLC: High? Data Link Control  
 LLC: Logical Link Control  
 PPP: Point-to-Point Protocol  
 RPC: Remote Procedure Call  
 RTP: Real Time Protocol

RTCP: Real Time Control Protocol  
 SDLC: High? Data Link Control  
 SMTP: Simple Mail Transfer Protocol  
 SCTP: Stream Control Transmission Protocol  
 SIP: Session Initiation Protocol  
 TCP: Transmission Control Protocol  
 UDP: User Datagram Protocol



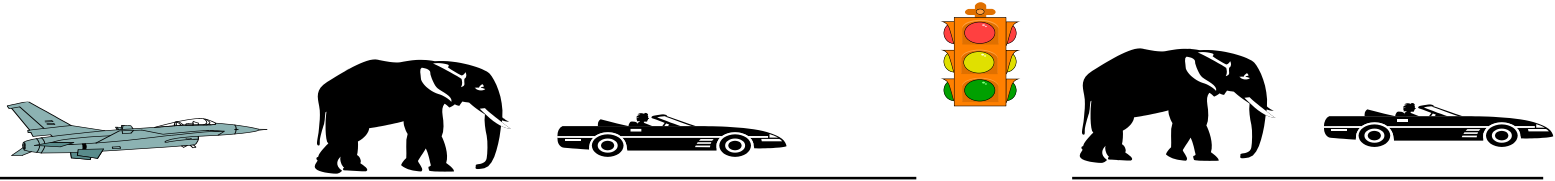
# QUALITY OF SERVICE (QOS) IN IP NETWORKS

- Different applications *define* it differently
  - for *telemedicine* the accuracy of the delivery—no packets lost—is more important than *delay* or packet delay variation (*jitter*)
  - for IP telephony, *jitter* and *delay* must be minimized, but packet loss is tolerable
  - for *streaming broadcast* , *initial delay* is not a problem; *jitter* is controllable (how), but subsequent long delay is unacceptable
  - for e-mail, neither *jitter*, nor *delay* is problematic
  - For instant messaging, *delay* is a bit problematic; *jitter* is not
- Solutions provided by the IP routers deal with
  - recognizing the type of treatment a particular packet deserves
  - scheduling and forwarding the packet accordingly

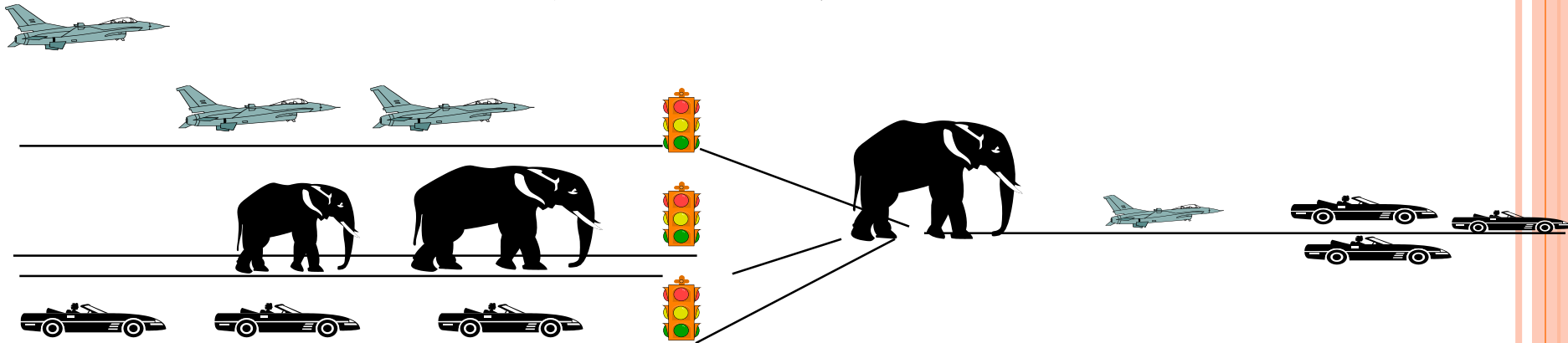
# SCHEDULING DISCIPLINES

- *Best Effort*: packets are sent on the first-come, first-served basis; they are dropped when queues become too large
- *Fair Queuing*: packets are queued for each flow and transmitted round-robin to guarantee each flow an equal share of bandwidth; bigger things wait for smaller, faster things
- *Weighted Fair Queuing*: same as fair queuing, but bandwidth is allocated according to priority

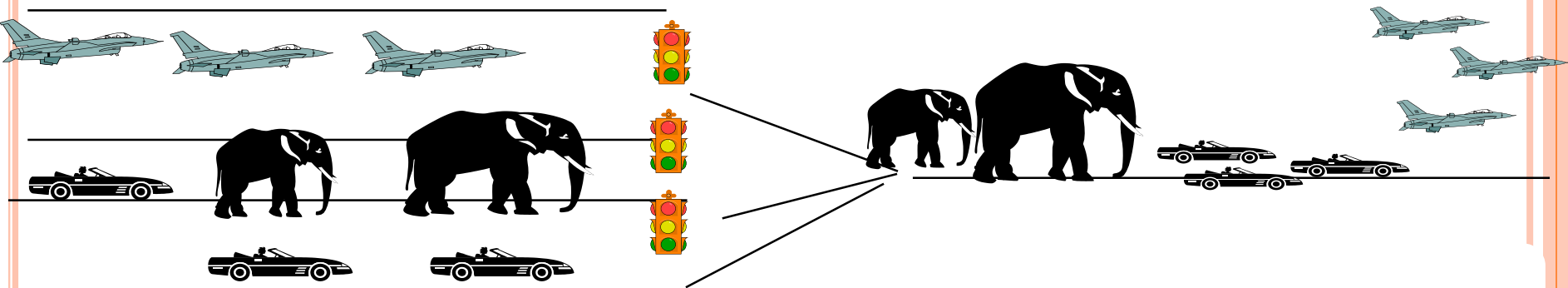
# SCHEDULING DISCIPLINES: AN ILLUSTRATION



a) First-come, first-served



b) Fair queuing

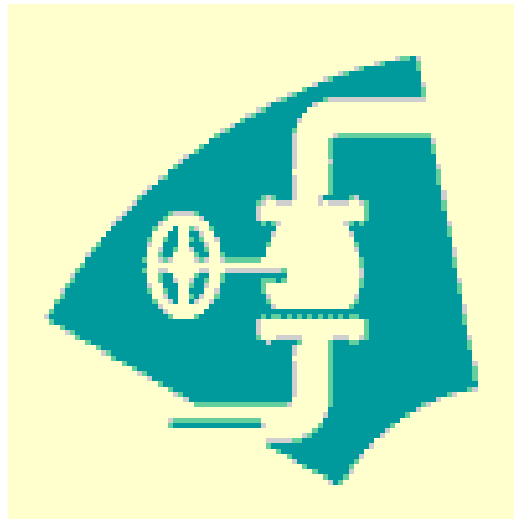


c) Weighted fair queuing

# THE INTEGRATED SERVICES MODEL

The Integrated Services model deals with two end-to-end services

1. *Guaranteed* service and
2. *Controlled-load* service on a *per-flow* basis.

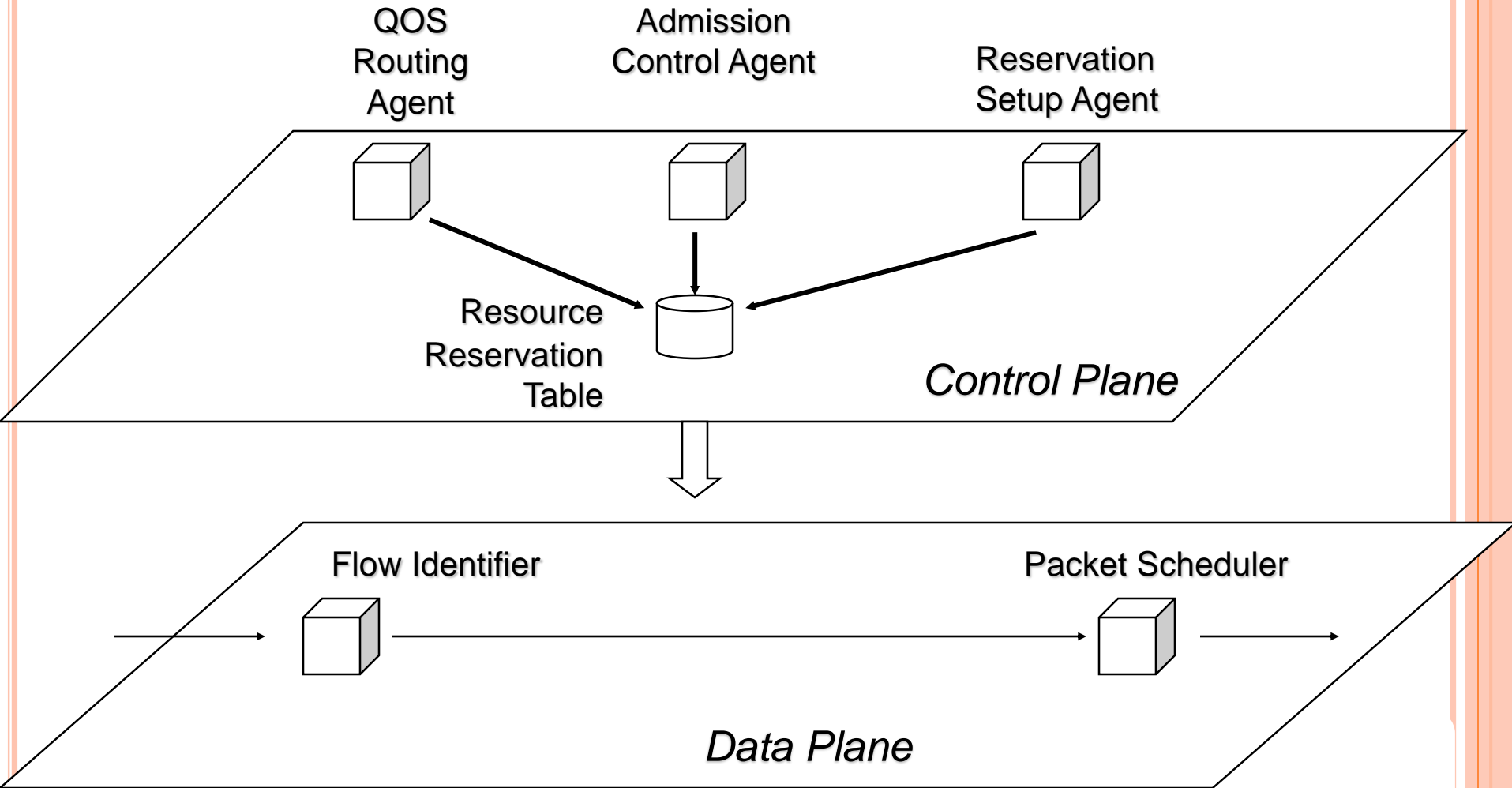


# WHAT IS A FLOW?

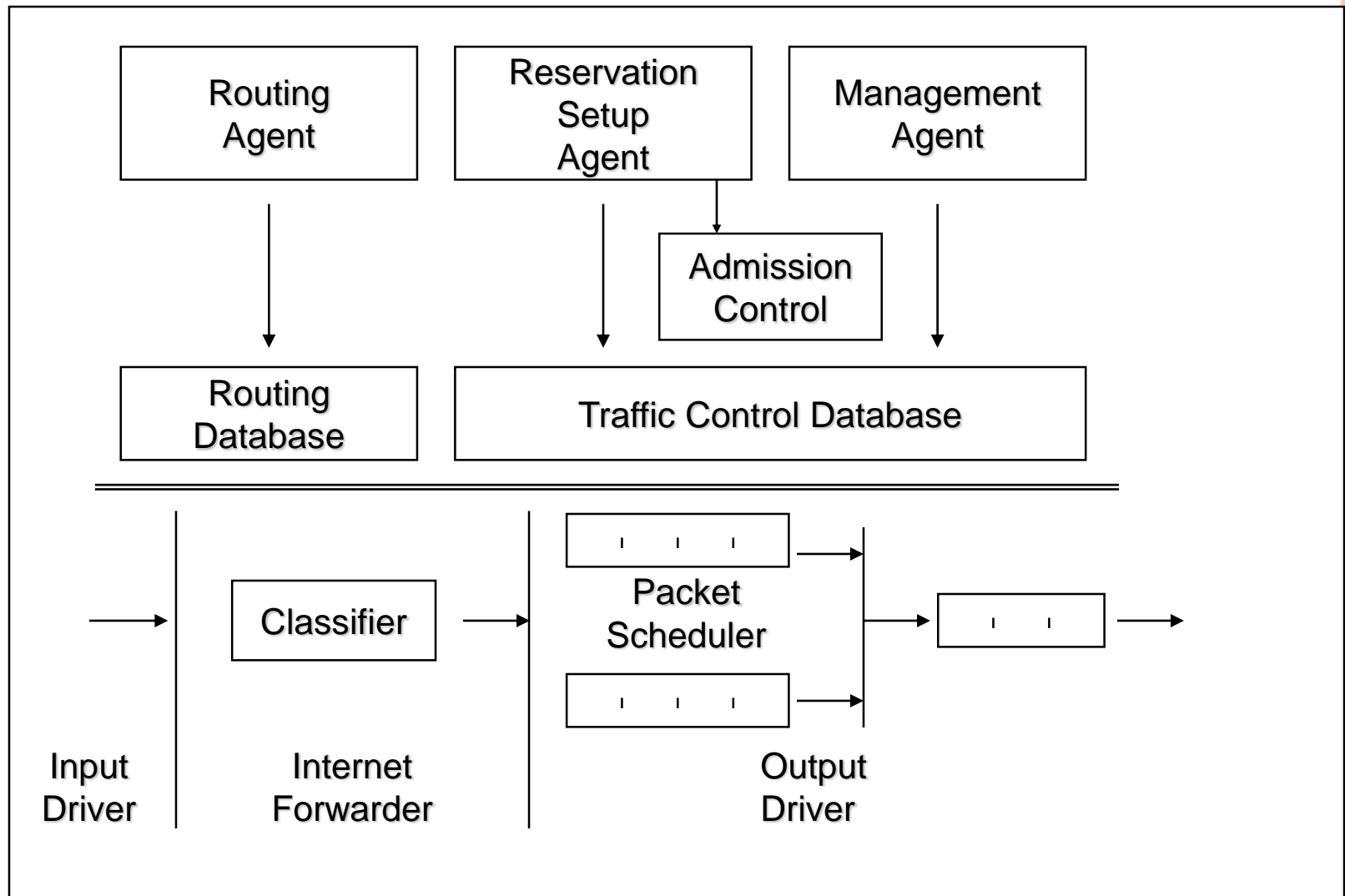
A flow is identified by a *quintuple* carried in both the IP- and transport-layer (TCP, UDP, SCTP) packet headers:

1. Source IP Address
2. Destination IP Address
3. Protocol ID
4. Source Port
5. Destination Port

# INTEGRATED SERVICES: *CONTROL* AND *DATA* PLANES



# IMPLEMENTATION INTSERV MODEL FOR ROUTERS



(after Figure 1 of RFC 1633)

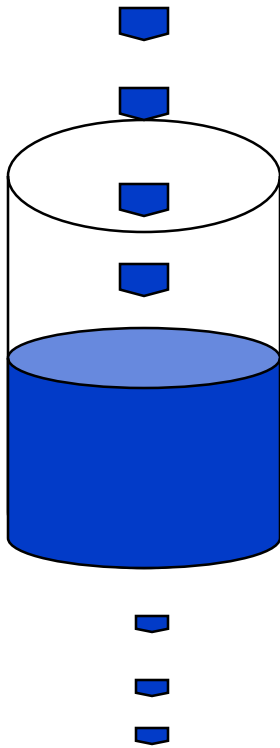
# FLOW SPECIFICATION (COMMON PARAMETERS)

- *Peak rate*: the highest rate at which a source may generate traffic (measured in bytes/sec)
- *Average rate*: the average transmission rate over a time interval
- *Burst size*: the maximum amount of data that can be injected into the network at peak rate
- *Minimum bandwidth*: the minimum amount of bandwidth required by an application flow over a specified period of time (measured in bytes/sec)
- *Delay*: either the average or worst-case end-to-end delay
- *Jitter*: the difference between the maximum delay and minimum delay
- *Loss rate*: the ratio of the number of lost packets to the total number of transmitted packets

Note: The full set of the parameters is defined in RFC 2215.



# TRAFFIC SPECIFICATION MODEL (LEAKY BUCKET)

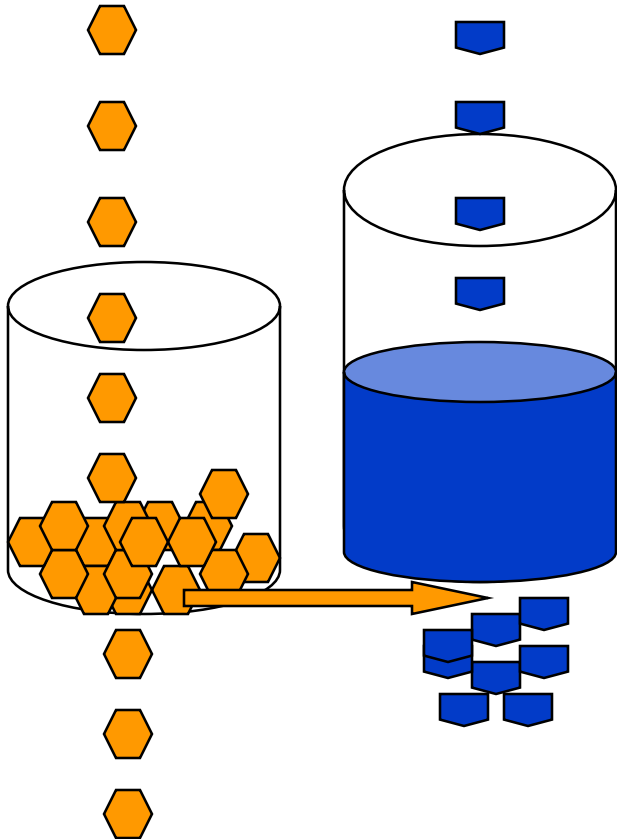


In this model, a host outputs packets into a network-controlled queue of size  $b$  bytes (bucket depth), which is processed at a rate  $r$  bytes/sec.

If the bucket is full, and packets arrive at a higher rate than  $r$ , the overflow packets get dropped.

**Result:** Bursts are eliminated

# TRAFFIC SPECIFICATION MODEL (TOKEN BUCKET)



In this model, tokens are output into a token bucket of size  $b$  (token bucket depth), at a rate of  $r$ . Tokens stop arriving when the bucket is full.

A token allows output of a fixed number of bytes from the queue, and is then destroyed. No packet can be transmitted if there are no tokens in the token bucket, but the tokens can be saved up.

Result: **Controlled bursts are allowed**

# TOKEN BUCKET PROPERTIES

- The total traffic volume  $V(t)$  the host can send over period  $t$  is bounded by a linear function:

$$V(t) \leq rt + b$$

- Given the maximum output rate of  $M$  (bytes/sec), the maximum burst time  $T$  can be found (this is another homework assignment)
- The token arrival rate defines the long term *average rate* of the traffic

**Note:** The tokens can be saved, up to the full size  $b$  of the token bucket

# MIX AND MATCH!

- We can shape the traffic never to exceed a given rate  $R$  *while allowing controlled bursts* by placing a token bucket after a leaky bucket that enforces rate  $R$ :

