Web & Internet Names

- Most of the Internet sites have easy names like mail.google.com, irishtimes.com or ucd.ie
- Sometimes you have to put in the www. in front of it to make it work, sometimes the computer does it
- Once you have a name like this you can then make up others as long as they end in the name e.g.:

  www.ucd.ie, docs.ucd.ie, mail.ucd.ie or even www.mail.ucd.ie

More than .com

.com	mtv.com
.org	un.org, nationalacademies.org
.net	hea.net, eircom.net, esat.net
.edu	berkeley.edu, upenn.edu
.gov	fbi.gov, cia.gov, nasa.gov
.mil	navy.mil, army.mil, af.mil
.tv	Tuvalu Pacific Island
.cc	Cocos Indian Ocean Island
.ws	Samoa now Web Site
.ie	Ireland

Web & Internet Names

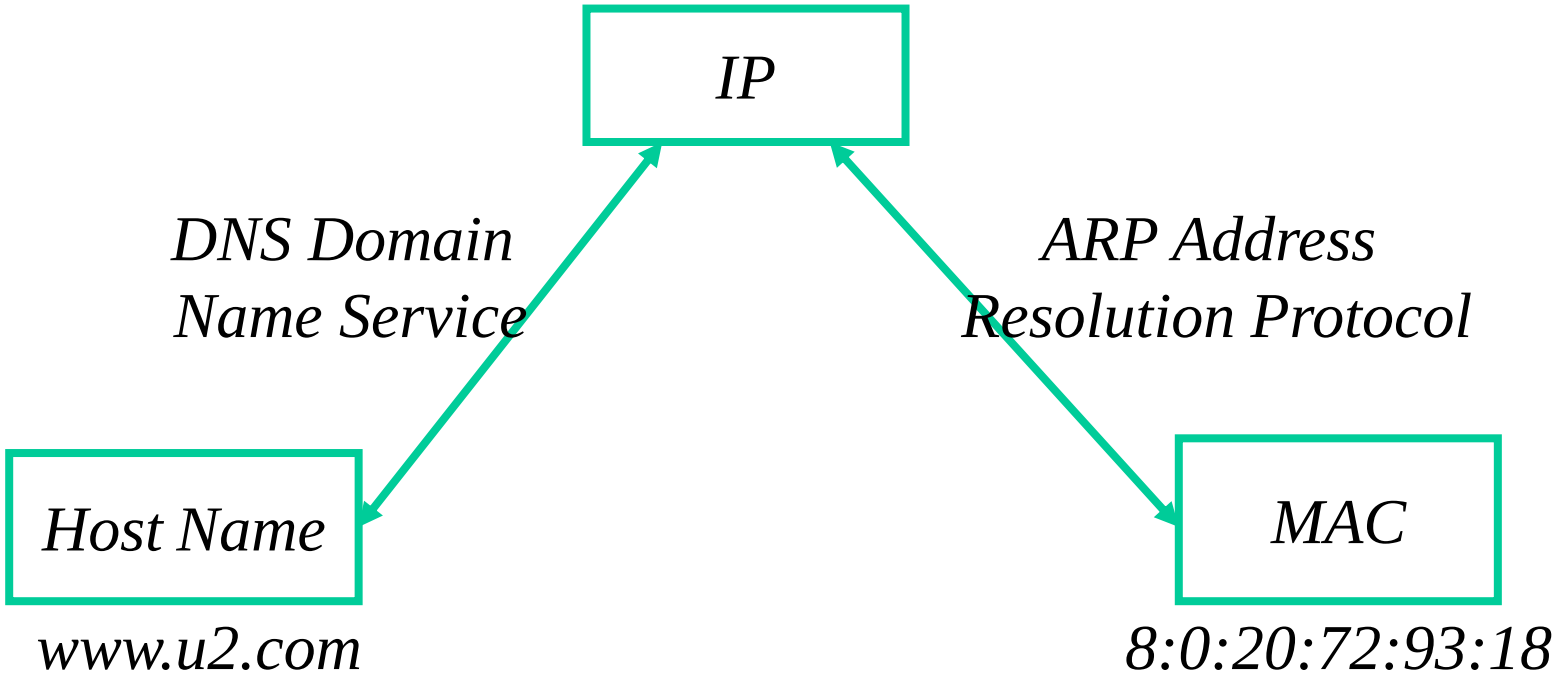
- This name that we have been talking about is the domain name of the site, or the host name
- Some big sites want to have many different computers serving that site
- Some smaller sites might want to have many different domain names on the one computer
- How does your computer find web addresses?
- All Internet devices (computers, internet enabled mobile phones, laptops, computer servers and network equipment) have a unique address

The Internet & Internet Protocols

- This is called the IP address of the device and is usually given in a dotted decimal format:

64 . 58 . 76 . 225

each number is between 0 and 255 (2^8)



IP Addresses

- The Internet address does not have to remain the same all the time, for example when you dial in you get assigned an address
- You can find the IP address on your computer by typing

ipconfig

- One way to find out which IP address is being used by a site would be to try the following command

nslookup www.ucd.ie

- This will give you information on which IP address might be for this site, or sometimes which set of IP addresses

How to get there!

- Does the computer that you are using have a direct link to the site that you want to look at?
- Well to find out you can look at the route that that is usually taken by using the command

tracert www.irishtimes.com

- This will show you all the pieces of network equipment that are used, the time taken between them and the IP address of all of them
- So what can we see about the network from UCD to the outside world
- Well to find out you can look at all the topology of the network by the following site

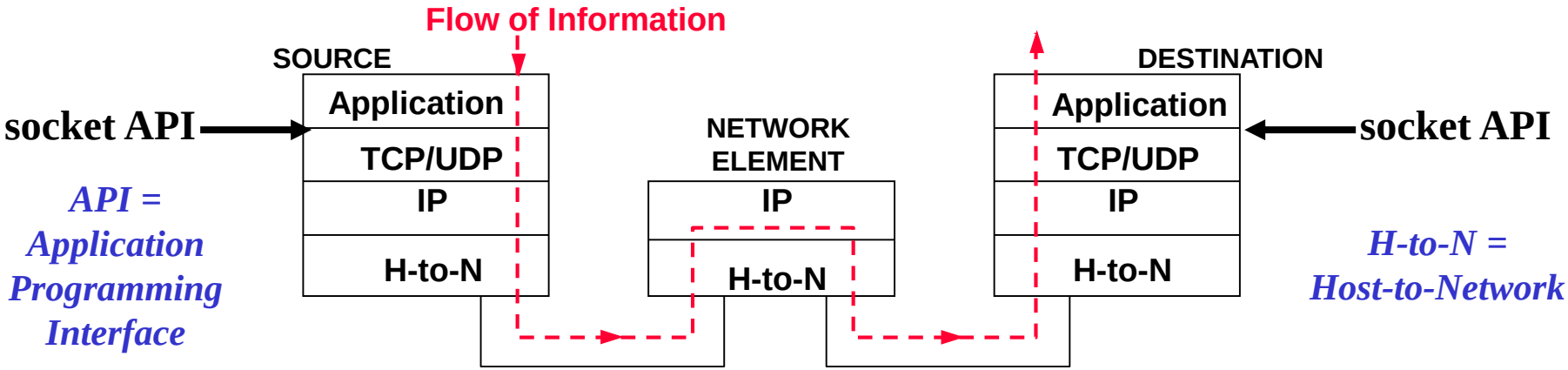
www.hea.net

Evolution of the Internet

- started in 1970 with 4 hosts, funded by US Defence Department
- ARPANet → NSFNet → today's Internet
  - US/non-US hosts split approximately 50/50
- remarkable sustained growth:
  - annual growth rate for Internet hosts >50% per year, 1993–present
  - number of Internet hosts: ~1,000,000,000 [1,387,000 in Ireland]
    - impossible to say how many Internet users there are !
- growth not just in size:
  - everyone has **`http://www`**... on their products, ads, business cards
  - **`.net`** and **`.com`** are the largest domains (together, 60% of all hosts)
  - ***E-Commerce:*** annual growth rate approx. >100%
  - Voice over IP (VoIP), audio/video streaming, specifications for ***multimedia communications*** [Real-time Transport Protocol (RTP), Real Time Streaming Protocol (RTSP), Session Initiation Protocol (SIP), …]
  - methods for ***supporting Quality of Service (QoS)*** [Integrated Services (intserv), Differentiated Services (diffserv), Resource reSerVation Protocol (RSVP), MultiProtocol Label Switching (MPLS), …]

Major features of the Internet

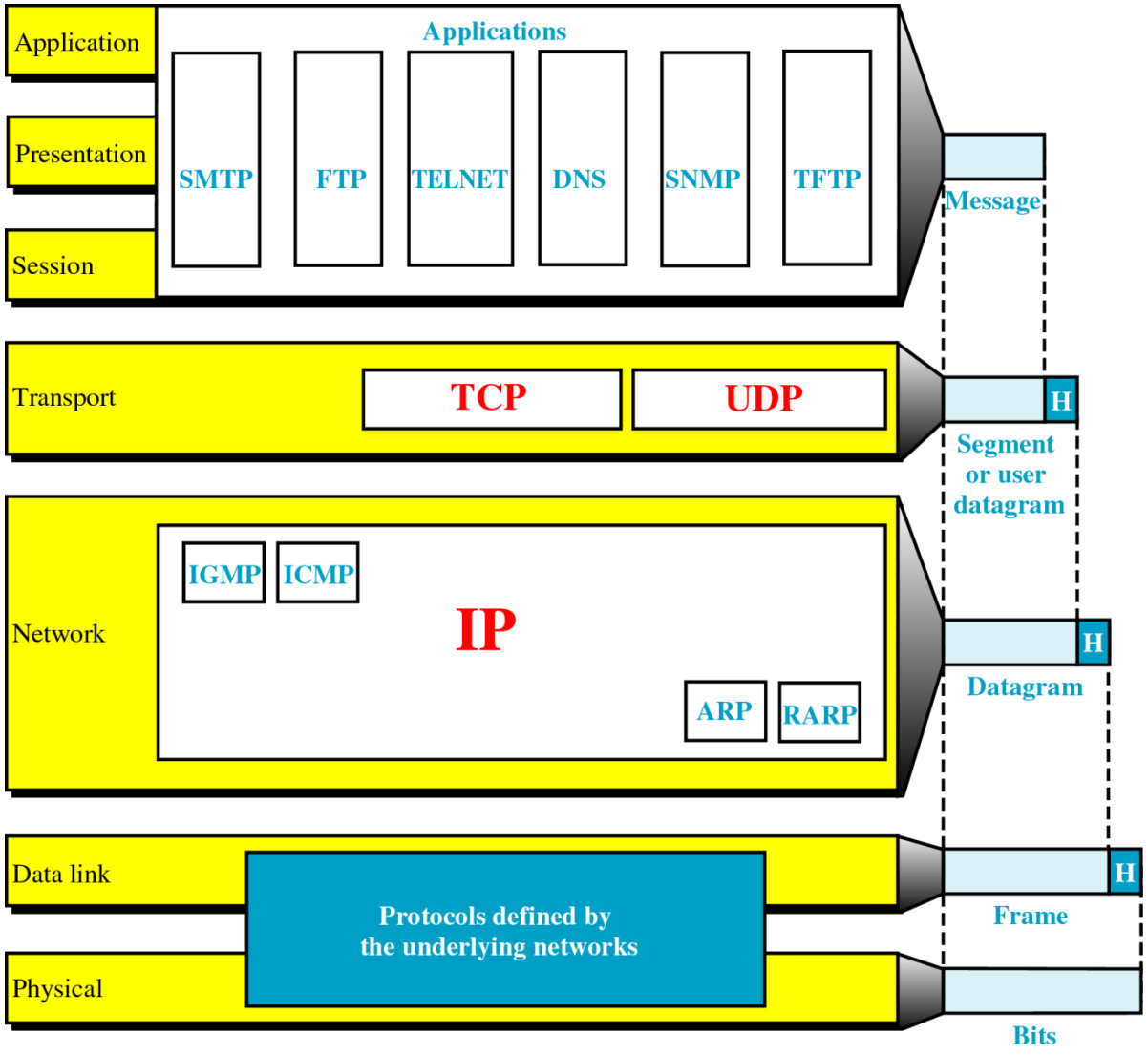
- IP uses datagram packet switching ⇒ ***connectionless***
- ***open*** design, ***open*** implementations, ***open*** standardisation process
  - *"We reject kings, presidents, and voting. We believe in rough consensus and running code"* (Dave Clark, MIT)
- ***independent*** of the physical medium
- ***scaleable:*** as evidenced by its growth
  - protocols have evolved over time, as problems arose and/or as application requirements changed
  - although "connections" are not welcomed in Internet circles, the desire to support QoS guarantees is adding connection-oriented concepts (such as *admission control* and *resource reservation*) to the Internet service model



Why does IP offer “best-effort datagram” service ?

- the Internet was originally intended for robust transfer of computer-to-computer data over long distances
  - best-effort service appropriate for data transfers
    - ➢ no real-time requirements
    - ➢ end-points can adapt to network conditions, if they want/need to
  - why was connectionless packet-switching preferred to circuit-switching ?
    - ➢ no set-up delay
    - ➢ no blocking *(fire-and-forget, forward-if-possible, deliver-as-received)*
      - ➢ however, there is no guarantee that any data reaches the destination
    - ➢ flexibility in transmission bit-rates
      - ➢ in contrast with circuit-switching, which is usually tied to a few pre-determined bit-rates
    - ➢ no “path” ⇒ more reliable *(route around problems)*
    - ➢ more efficient use of network resources when traffic is bursty
      - ➢ economics: transmission became more expensive than switching in late 1960s

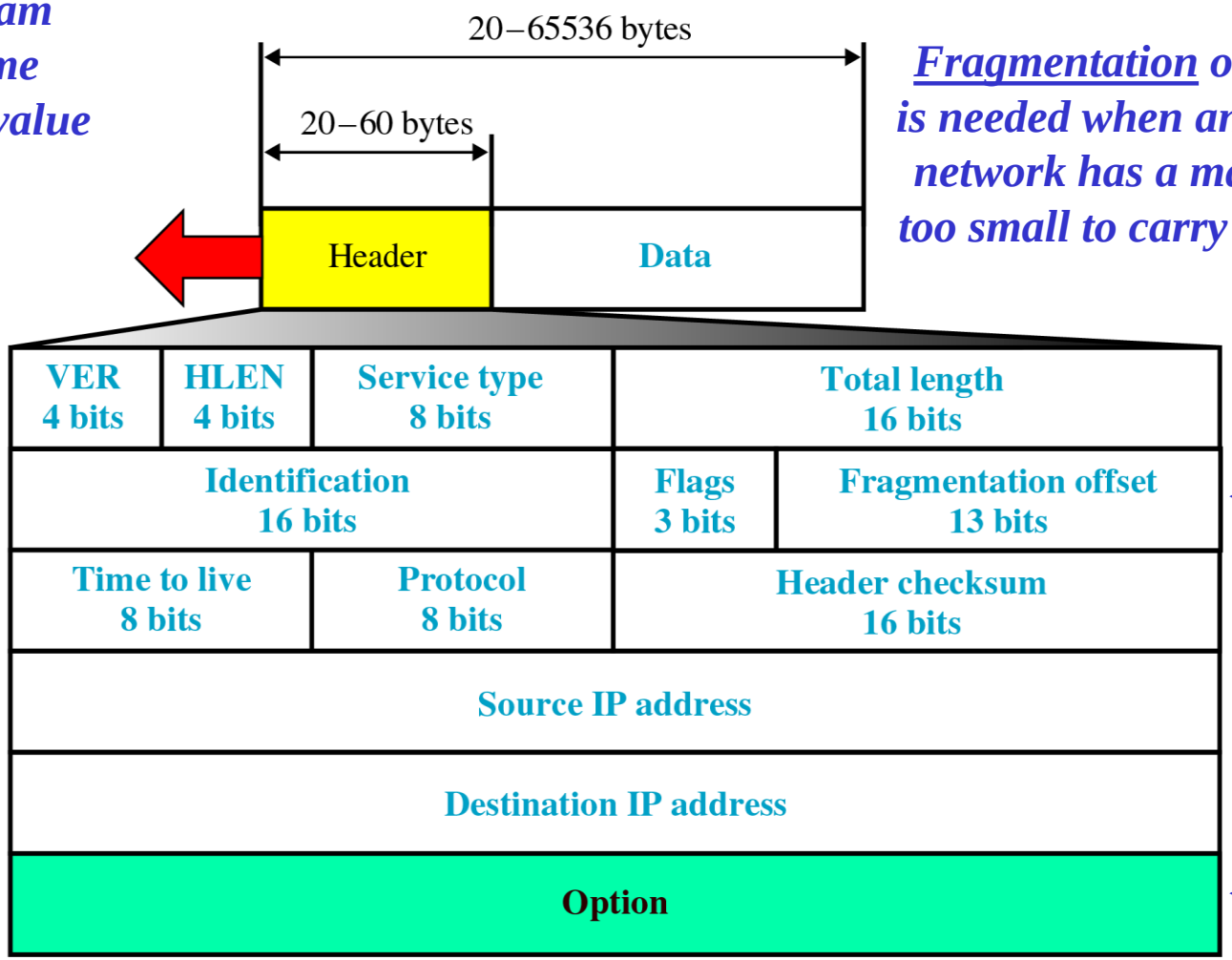
Internet Protocols



Datalink Frames have a Trailer also (not shown here)

The IPv4 Header

all fragments of a given datagram have the same Identification value



Fragmentation of a datagram is needed when an intermediate network has a max frame size too small to carry the datagram

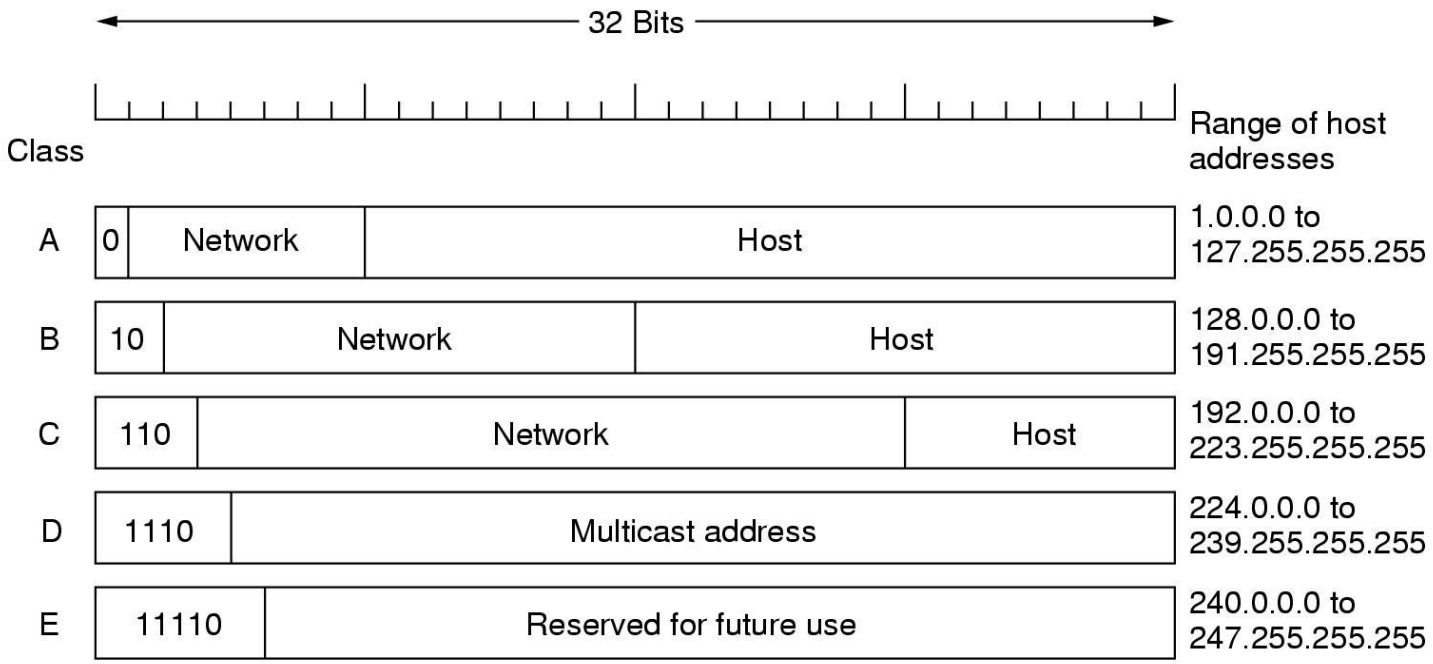
decreased by 1 at each "hop"

padded to a multiple of 4 bytes (32 bits); can carry fields that control routing, timing, management, security, ...

IPv4 Addresses

```
00000000 00000000 00000000 00000000     0.0.0.0
00000000 00000000 00000000 00000001     0.0.0.1
00000000 00000000 00000000 00000010     0.0.0.2
00000000 00000000 00000000 00000011     0.0.0.3
    ⋮        ⋮        ⋮        ⋮
10001001 00101011 00000001 00110001     137.43.1.49
10001001 00101011 00000001 00110010     137.43.1.50
10001001 00101011 00000001 00110011     137.43.1.51
    ⋮        ⋮        ⋮        ⋮
11111111 11111111 11111111 11111100     255.255.255.252
11111111 11111111 11111111 11111101     255.255.255.253
11111111 11111111 11111111 11111110     255.255.255.254
11111111 11111111 11111111 11111111     255.255.255.255
```

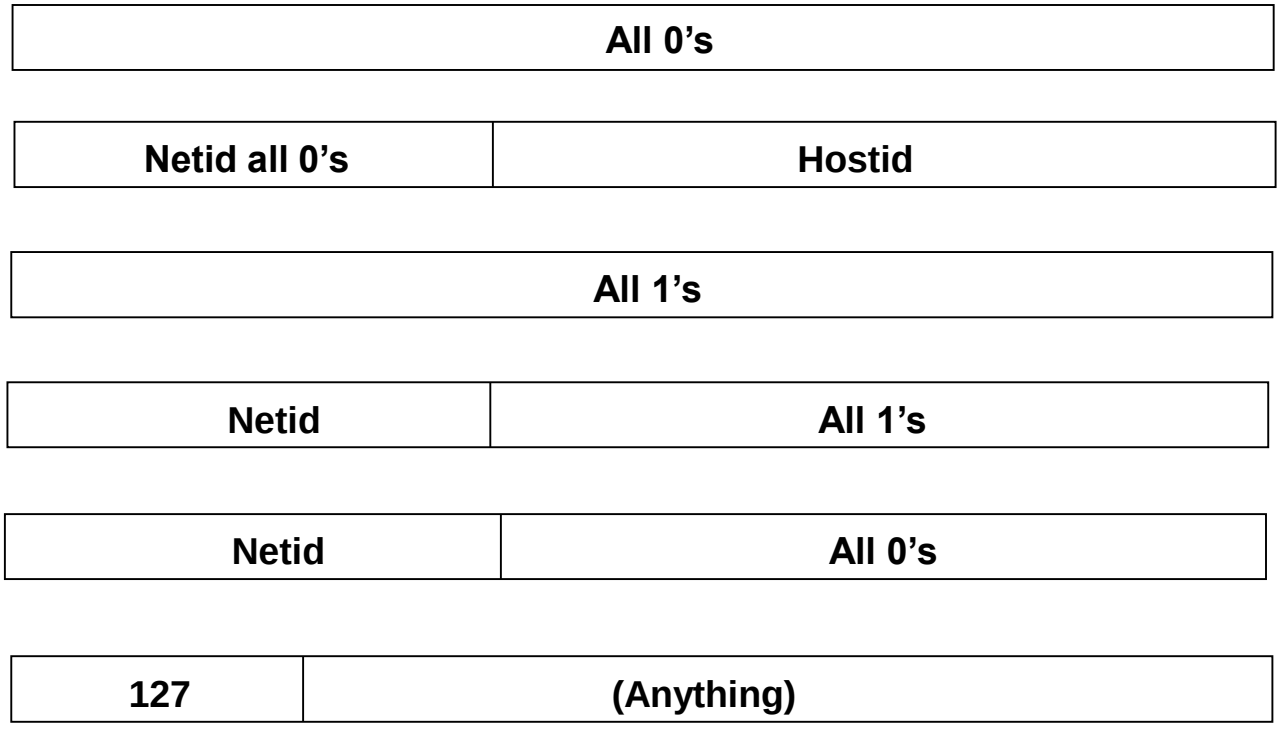
IPv4 Address Format



Class	Number of networks of this Class	Number of hosts on each network of this Class
A	126	$2^{24} - 2$ (≈ 16 million)
B	$2^{14} - 2 = 16{,}382$	$2^{16} - 2$ (≈ 65,000)
C	$2^{21} - 2$ (≈ 2 million)	254

all-0's and all-1's (in the Netid &/or Hostid parts) have special meanings

Special IPv4 Addresses



IP packets with loopback addresses are processed by the machine which generated them as if they were incoming packets (useful for debugging network software)

Who assigns IPv4 Addresses?

IP address assignment managed by ICANN (Internet Corporation for Assigned Names and Numbers) https://www.icann.org/

> ***ICANN is the non-profit corporation that was formed to assume responsibility for the IP address space allocation, protocol parameter assignment, domain name system management, and root server system management functions previously performed under U.S. Government contract by IANA and other entities***

ICANN assigns portions of the IP address space to other authorities.

Some of the authorities which assign addresses to ISPs:

APNIC: 61, 202, 203, 210, 211
Asia Pacific Network Information Centre (www.apnic.net)

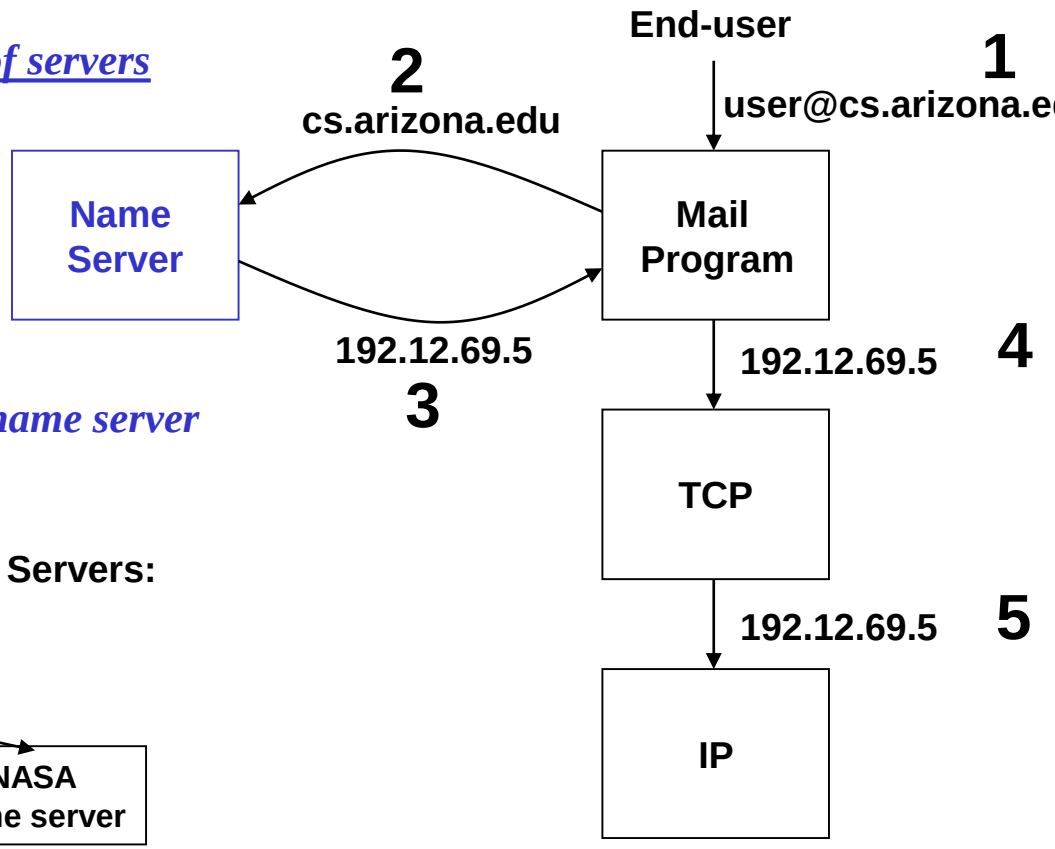
ARIN: 208, 209, 216
American Registry for Internet Numbers (www.arin.net)

RIPE NCC: 193, 194, 195, 212, 62
Réseaux IP Européens Network Coordination Centre (www.ripe.net)

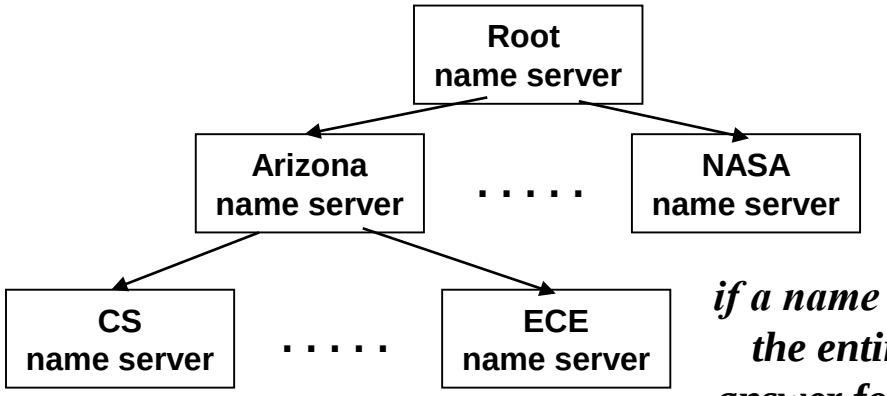
Names and IP addresses: the Domain Name System

There has to be a way to translate between "user-friendly" names and IP addresses:

- *"Name Server" is actually a hierarchy of servers called the Domain Name System (DNS)*
- *name-to-address mapping for one or more root servers is "well-known" and stored in the local name server's system configuration file*
- *hosts initialised with address of a local name server*

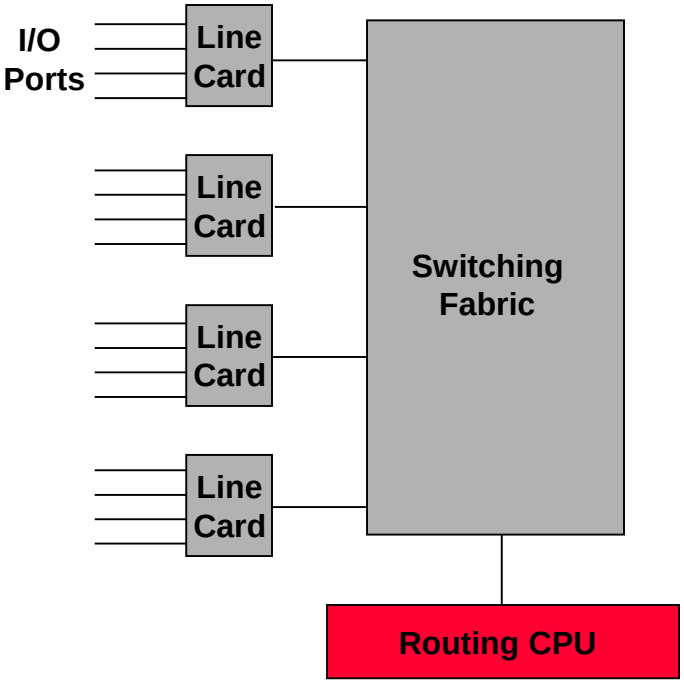


Example Hierarchy of Name Servers:



if a name server cannot "resolve" the entire name, it returns an answer for as much of it as it can

IP Router Architecture



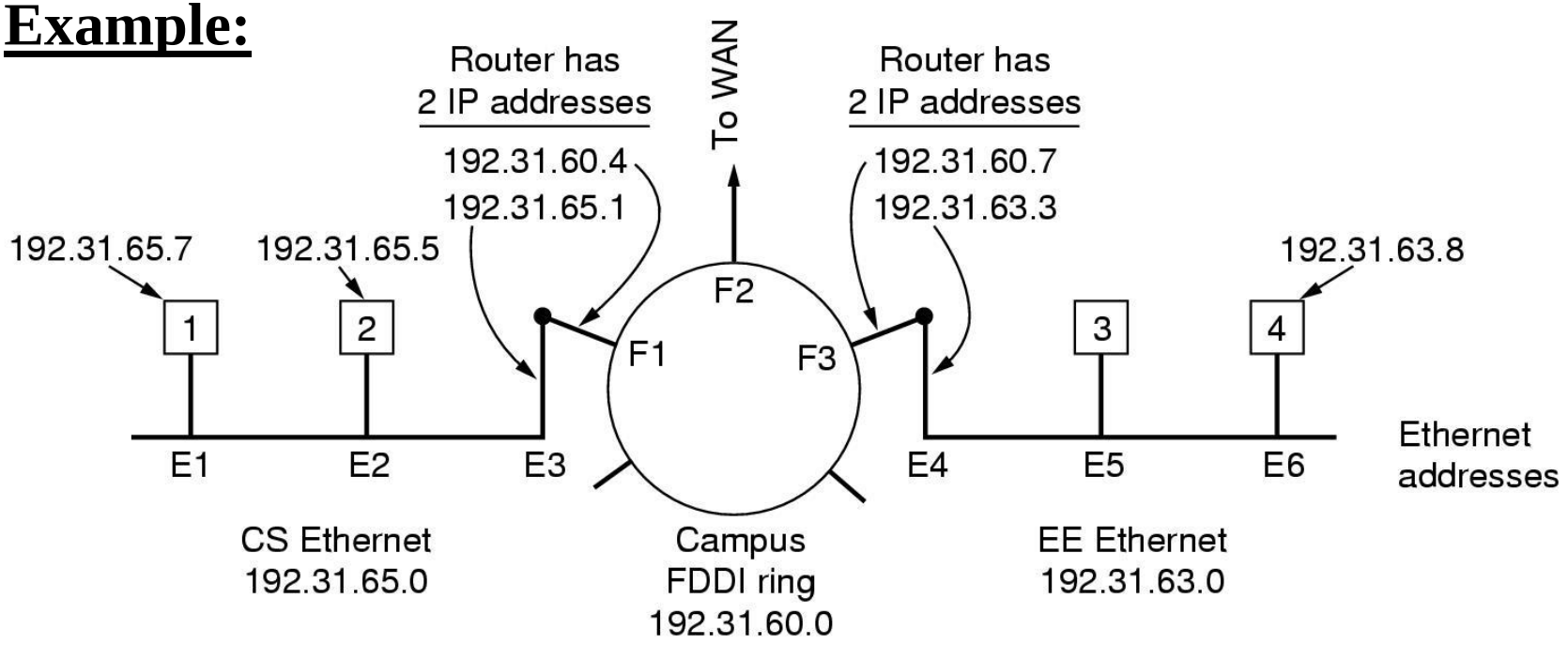
***IP packet arrives* (in a datalink frame):**
1. line card applies datalink layer logic to ensure frame is valid and packet successfully received;
2. validity check performed on IP header;
3. If destination address is a non-local host, routing table lookup performed to determine how to forward the packet;
4. packets may have to be classified into predefined service classes;
5. TTL field decremented, new header checksum computed, then packet sent to appropriate output port;
6. datalink layer logic on output port's line card inserts datalink layer header and transmits the packet inside a frame

[NOTE: some/all of steps 2-5 may be done in the input port, not the routing CPU]

- ***if this process fails, ICMP error message sent to packet's sender***
- ***if packet arrival rate exceeds router's forwarding capacity, packets must be buffered***
- ***if buffers are full, packets must be discarded***

Forwarding IP packets

Example:



Problems forwarding IP packets ⇒ Internet Control Message Protocol (ICMP) sends error message back to sending host. ICMP can also be used to tell hosts about better routes, using ICMP–Redirect messages.

How does a sending host know the physical address corresponding to the IP address of its intended destination ? Answer: it uses the Address Resolution Protocol (ARP)

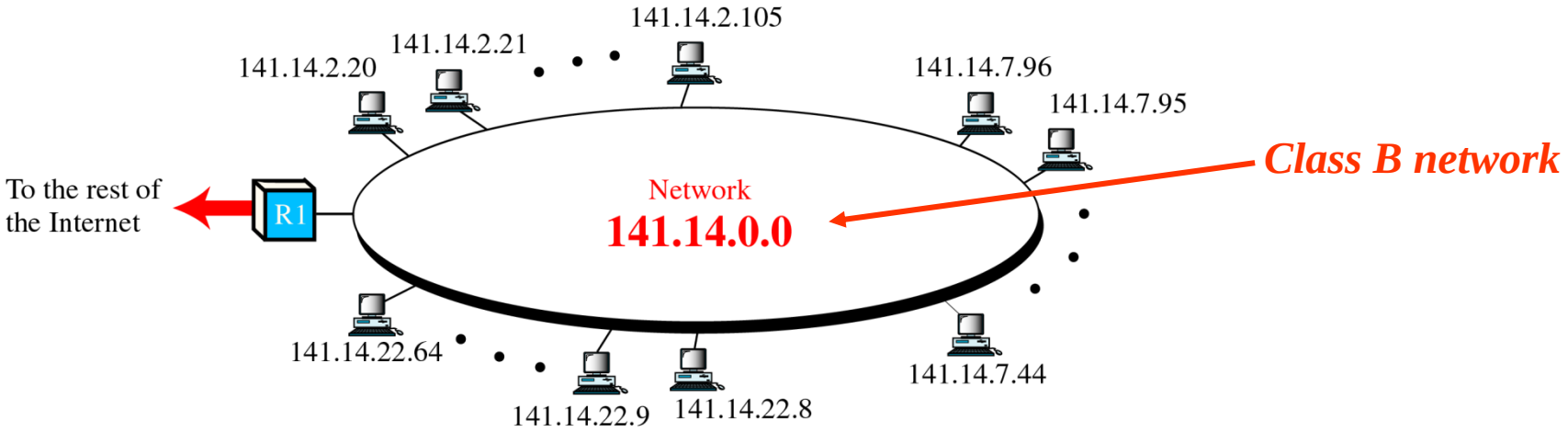
ARP and Reverse ARP

- each host maintains an *ARP cache* of mappings between IP addresses and physical addresses
  - ARP cache entries ***time out*** after (typically) 15 minutes
- if destination's IP address not in sending host's ARP cache, sending host broadcasts *ARP query* on local network
  - ARP query asks for physical address for destination's IP address
  - ARP query includes sending host's IP address and physical address (so other hosts can enter this in their ARP caches)
- if destination is a host on local network, it sends its physical address to sending host in an *ARP reply*
- if destination not on the local network, ARP server replies with its own physical address: *proxy ARP*

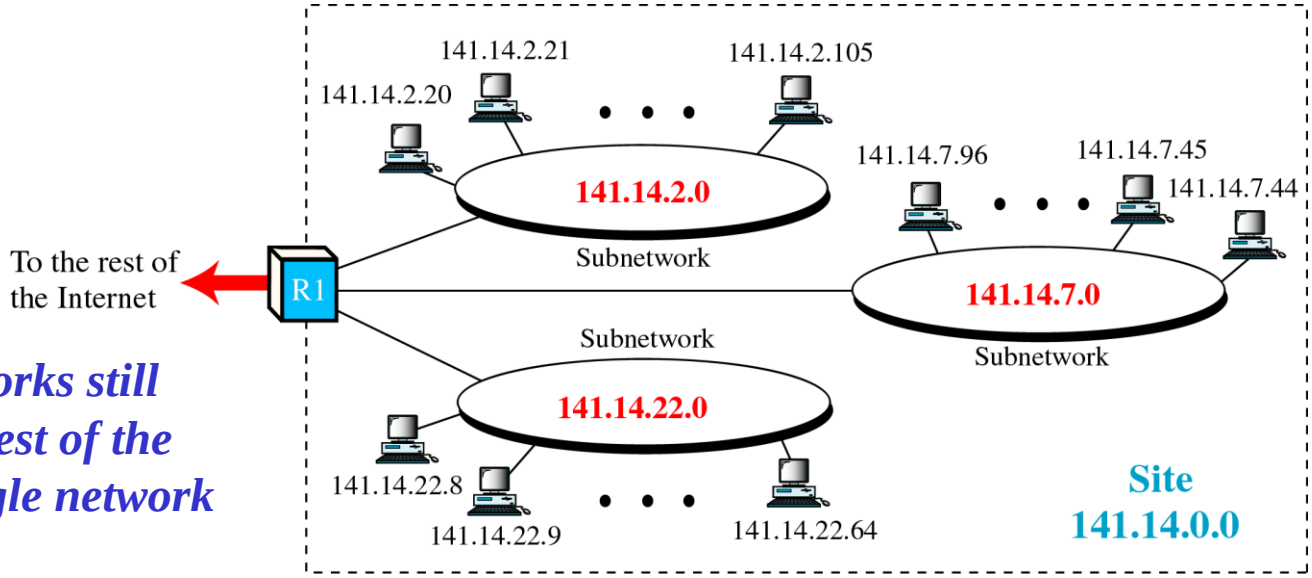
- if a host doesn't know its own IP address (e.g. diskless workstation being booted), it uses *Reverse ARP*
  - host broadcasts its physical address and asks for corresponding IP address; RARP server replies

IP Subnetting

- IP address is composed of a Netid part and a Hostid part ⇒ ***2-level hierarchy***
- sometimes a 2-level hierarchy is insufficient for an organisation's needs:



- a solution:
  subnetting

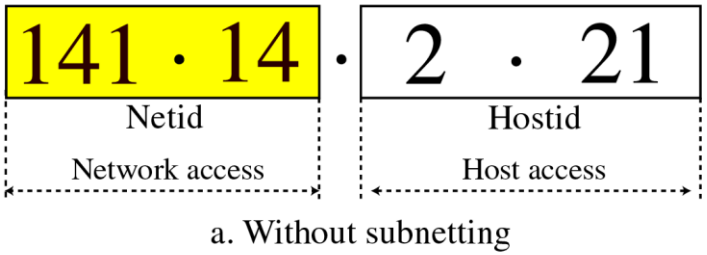


the 3 subnetworks still appear to the rest of the Internet as a single network

IP Subnetting (cont.)

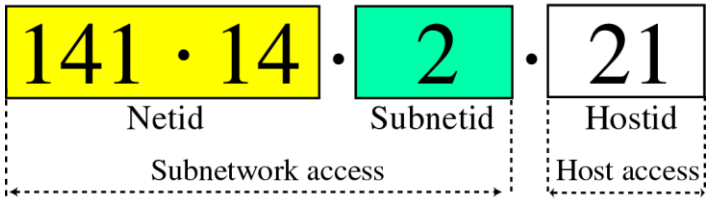
- an IP packet from some other network destined for host 141.14.2.21 still reaches router R1, since the destination address is still a Class B address with Netid 141.14 and Hostid 2.21 as far as the rest of the Internet is concerned

- when the packet reaches router R1, *the interpretation of the IP address changes*
  - R1 knows that there are 3 levels of hierarchy within the organisation, and that in this case, the Netid is 141.14, the Subnetid is 2, and the Hostid is 21



a. Without subnetting

with subnetting, the Netid defines the site, the Subnetid defines the physical network, and the Hostid defines the actual machine

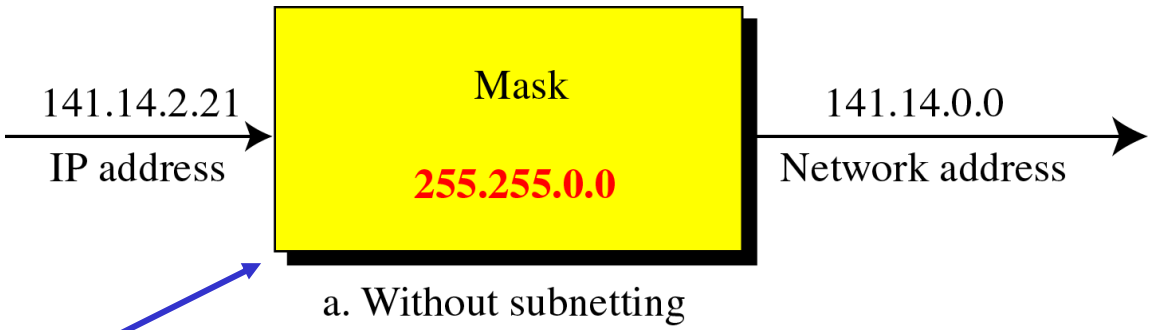


b. With subnetting

- how is this knowledge of the internal network hierarchy implemented in the organisation's routers ?
  - A: ***masking*** of IP addresses during the packet-forwarding process
  - masking is done whether or not subnetting is being used

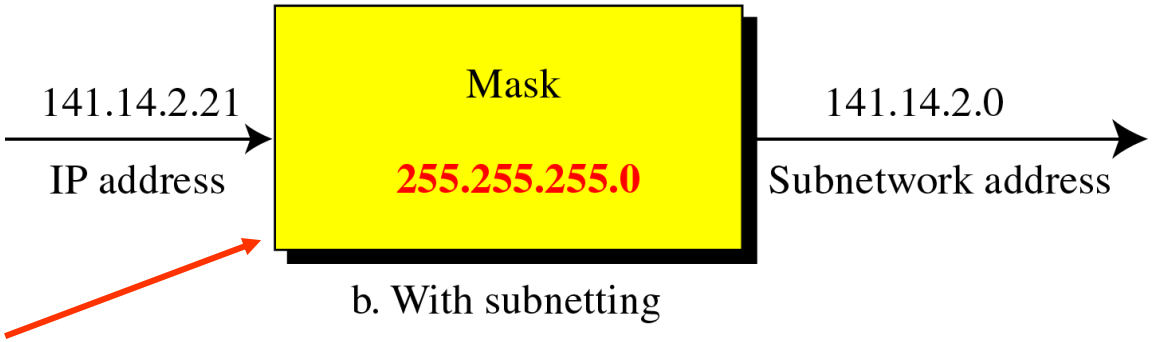
IP Subnetting: Masking

- masking means taking ***bit-by-bit AND*** of IP address and mask (255 ≡ all 1's, 0 ≡ all 0's)



a. Without subnetting

extracts network address from an IP address



b. With subnetting

extracts subnetwork address from an IP address

- mask address fields don't have to be 255 or 0
  - e.g. could have a mask = 255.255.192.0

Classless InterDomain Routing (CIDR)

- original class-based IP addressing scheme is inefficient
  - by 1996, more than 50% of all Class B networks had < 50 hosts
- could give organisations which are large enough for a Class B address a number of Class C addresses instead
  - then every Internet backbone router would need an entry in its routing table for each of these Class C network numbers
- CIDR: medium-large organisations get ***contiguous blocks*** of Class C addresses (in powers of 2)
  - example: company needs 8,000 IP addresses ⇒ gets 32 contiguous blocks of Class C addresses ≡ 8,192 addresses
    - e.g. if given Class C network numbers 192.4.0 – 192.4.31, then first 19 bits of all addresses in company's networks are the same *(19-bit network prefix):*

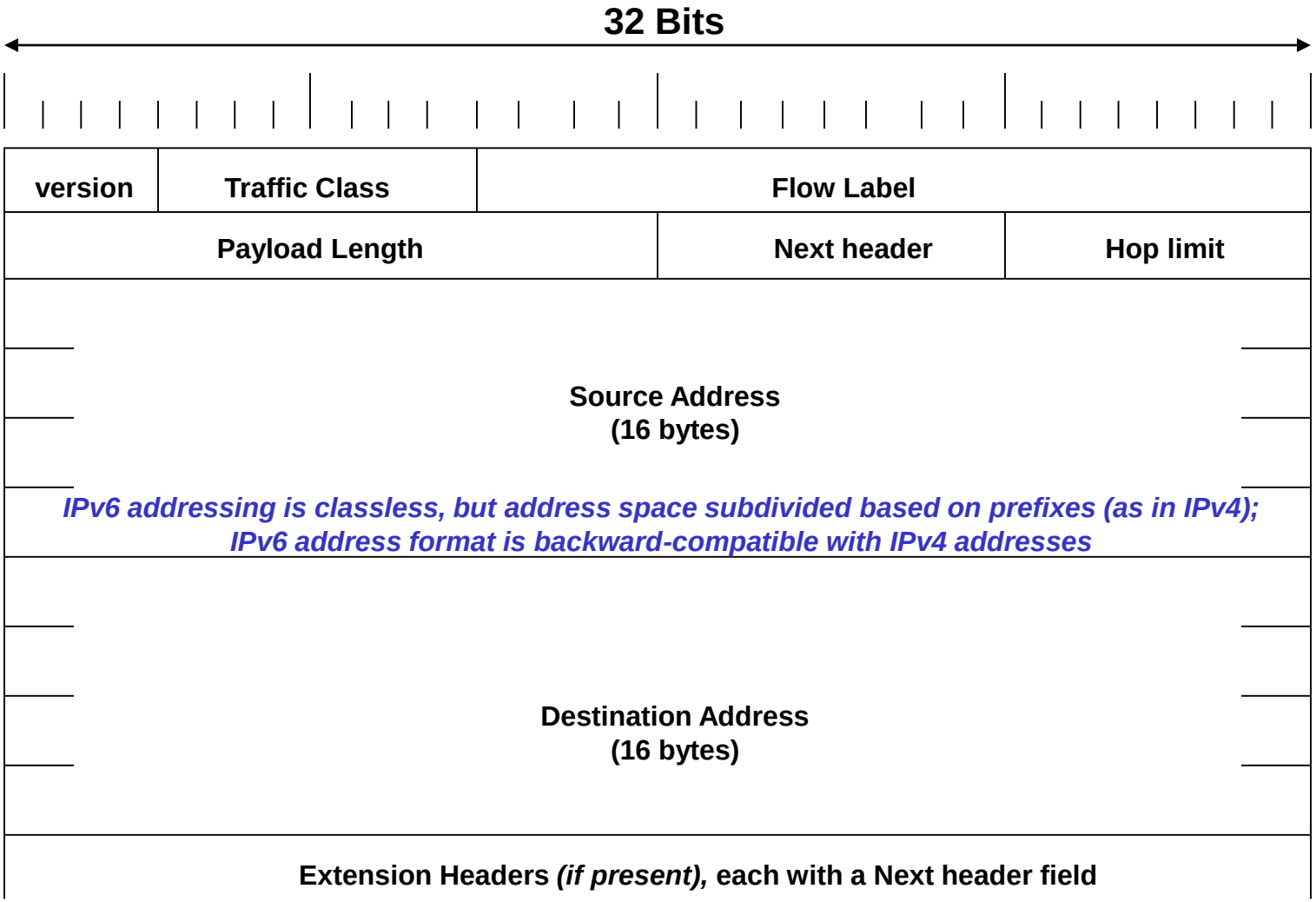
      11000000 00000100 000 ⇒ first 19 bits of network mask are 1, rest are 0
    - this company's IP addresses are now written (e.g.) 192.4.19.252/19
- same idea is now applied to all addresses, not just Class C addresses

subnetting: ***share one network IP address among multiple physical networks***
CIDR: ***collapse multiple adjacent IP addresses onto one network prefix***

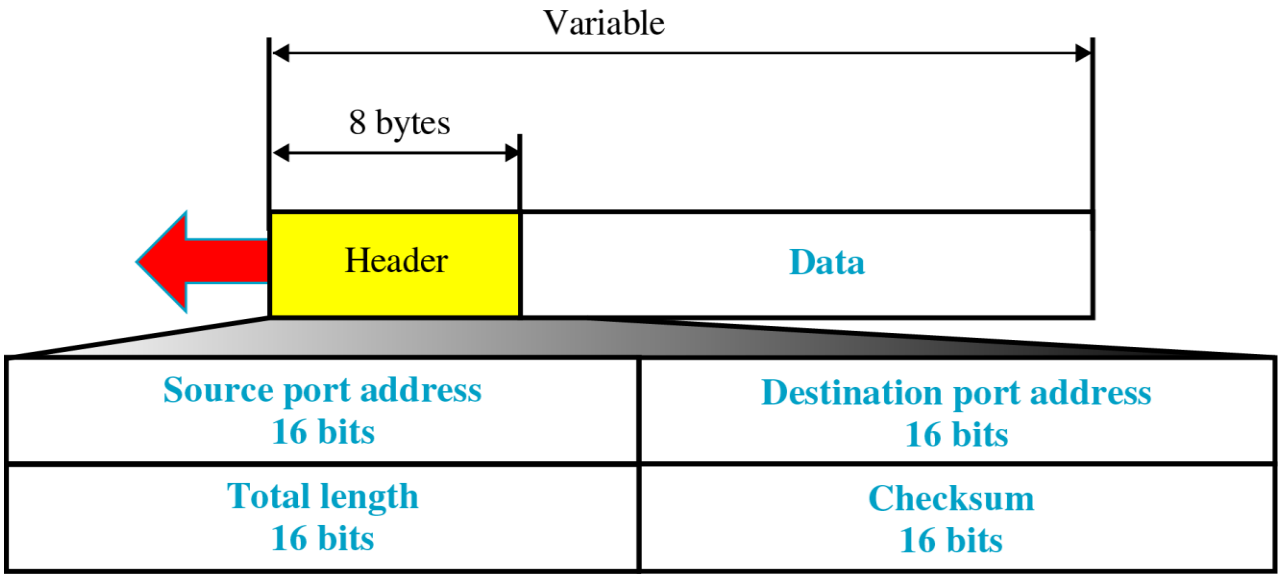
IPv6: Next Generation IP

- in early 90s, IETF held an open design process to design the next version of IP: IPv6
- motivated by problems with IPv4 (the current version of IP)
  - still have limited address space, even with subnetting & CIDR
  - didn't handle real-time "flows": routers ignored the IPv4 Type Of Service field
  - at the time, IPv4 security features were not widely used
- IPv6 is ***not*** compatible with IPv4, but it ***is*** compatible with TCP/UDP/OSPF/BGP/DNS…
- IPv4 and IPv6 will co-exist for a (possibly long) time
  - IPv6 implementations exist, but not many products so far
- major features of IPv6:
  - address space problems taken care of by using ***128-bit addresses***
    - there are a maximum of 2^{128} IPv6 addresses (approximately 3.4×10^{38})
    - even the most pessimistic expectations of IPv6 addressing efficiency predict 16,000 IPv6 addresses per square metre of the earth's surface…
  - simplified packet header functions
    - fragmentation at the source only; no header checksum
  - better support for options
    - e.g. hop-by-hop options, routing options, fragmentation options
    - *extension headers* are present when corresponding options used
  - support for ***per-flow handling*** and ***traffic classes***
    - flow designated by source & destination addresses and flow number
  - support for authentication and security

The IPv6 Header



UDP: User Datagram Protocol

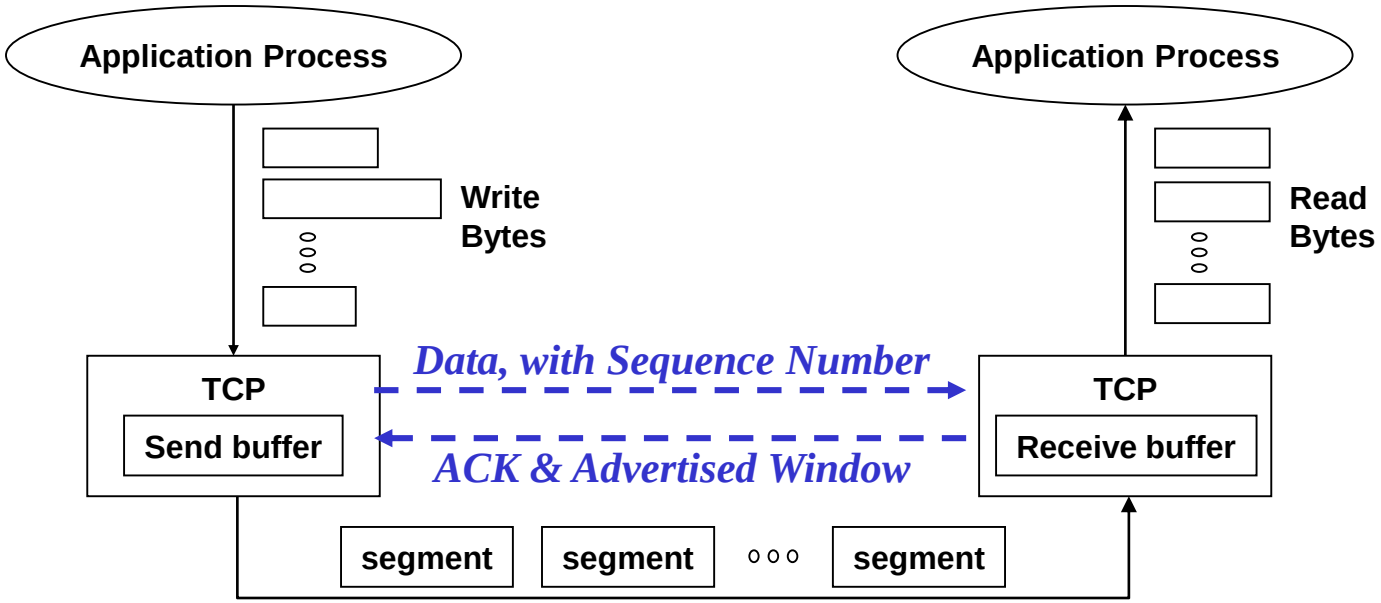


- ***UDP is a connectionless transport protocol – extends IP's host-to-host delivery service into a process-to-process communication service***
  - ***can have multiple application processes on a single host, each with their own port number***
  - ***a process is uniquely addressed by a < port, host > pair***
  - ***common services are available at well-known (and reserved) ports on each host; user applications must choose their ports from the set of non-reserved ports***

- ***UDP doesn't support flow control or reliable/in-order delivery, but it does support error detection by computing an "optional" checksum over the UDP header, UDP data, and IP pseudoheader (includes source and destination address fields from the IP header)***

- ***new: Reliable UDP – provides reliable in-order delivery (up to a maximum number of retransmissions), with simple window flow control, for virtual connections***

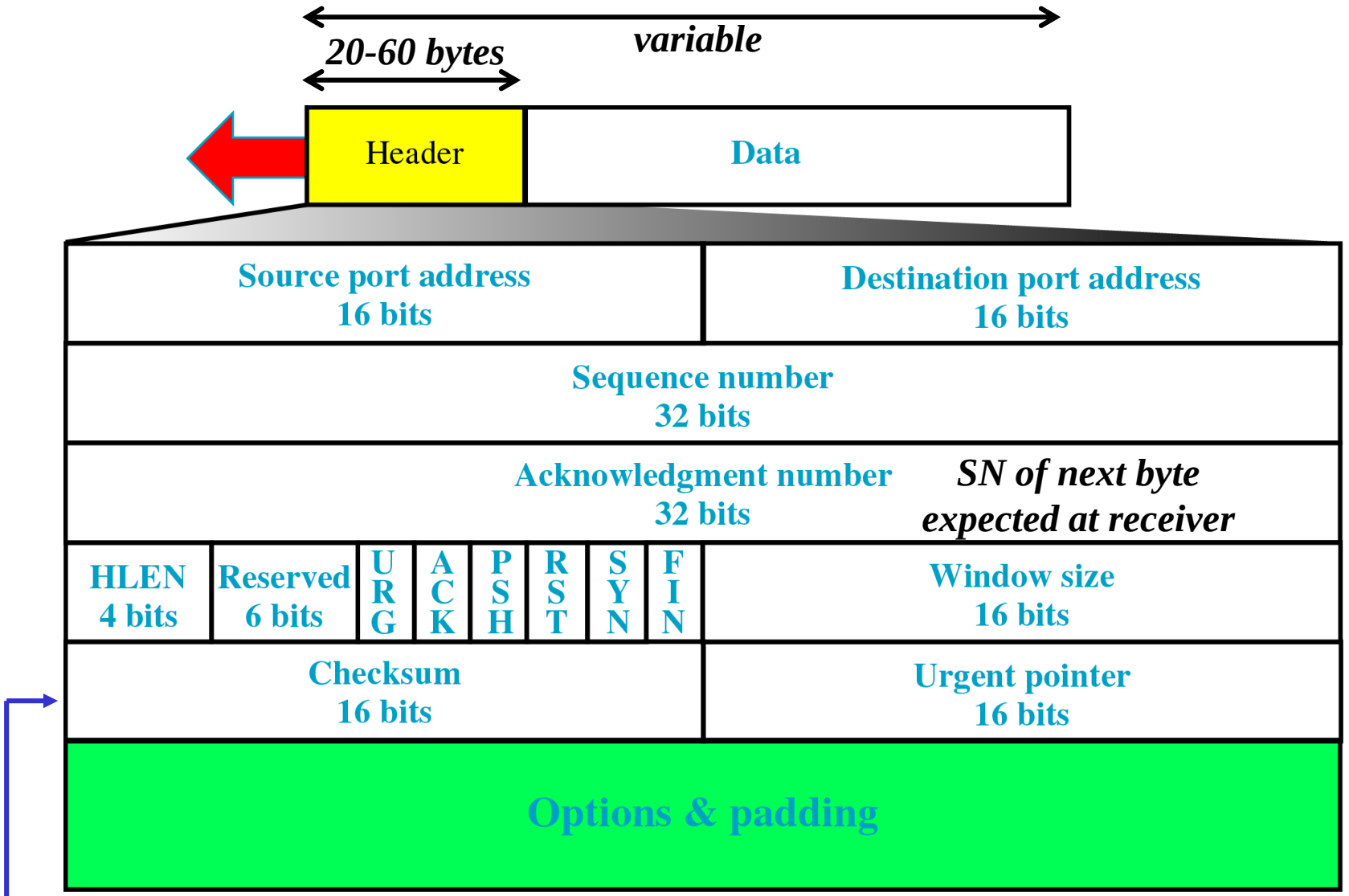
TCP: Transmission Control Protocol



For simplicity, only one direction of data flow shown; in general, a single TCP connection supports byte streams flowing in both directions

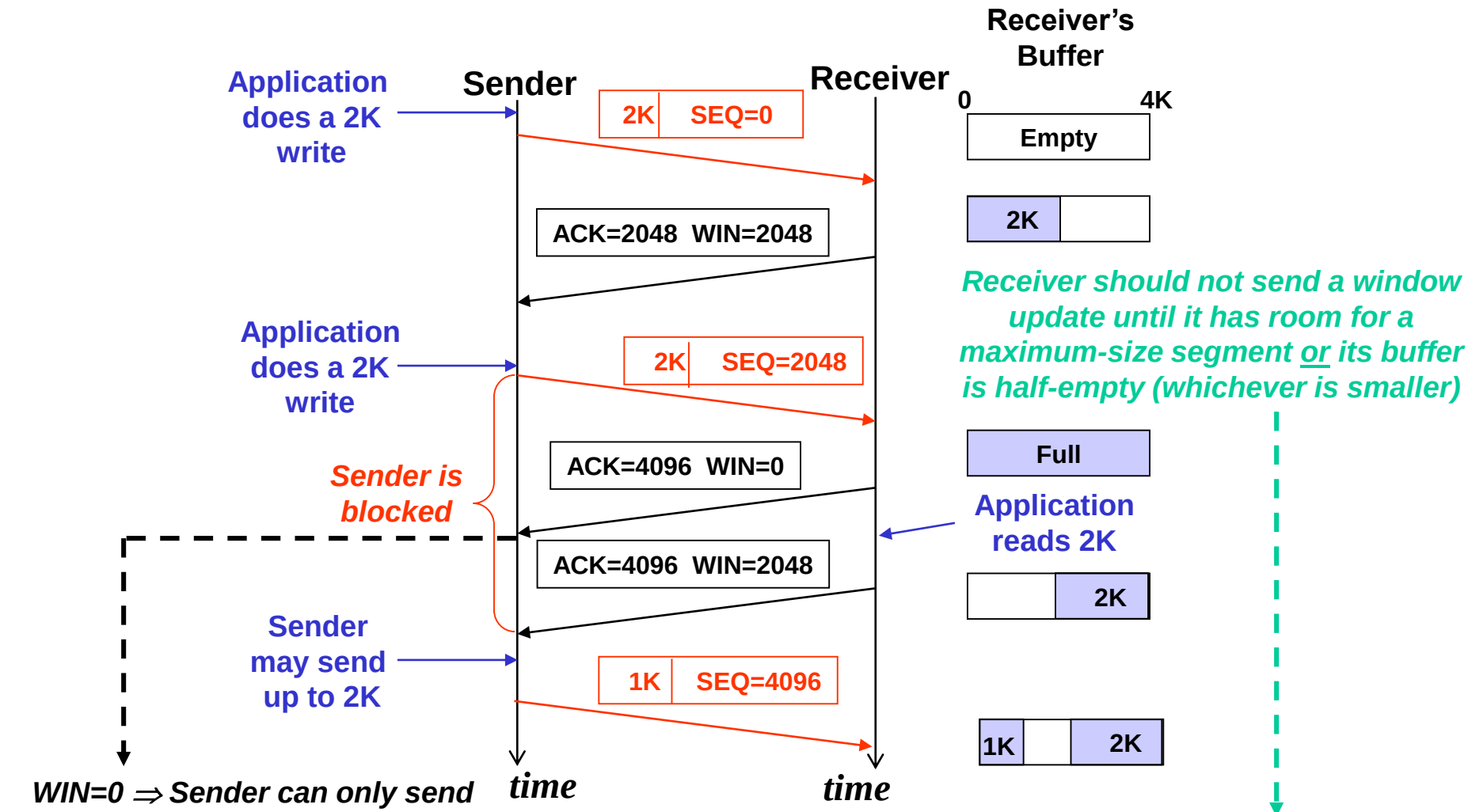
- a TCP connection is full-duplex point-to-point, and ***does not preserve message boundaries***
  - ➢ TCP is only concerned with end-to-end delivery of a byte stream
- each byte of data gets its own sequence number (SN)
  - ➢ the SN in a TCP segment is the SN of its first byte

The TCP Header



computed over the TCP header, TCP data, and IP pseudoheader
(includes source and destination address fields from the IP header)

TCP Window Management Example

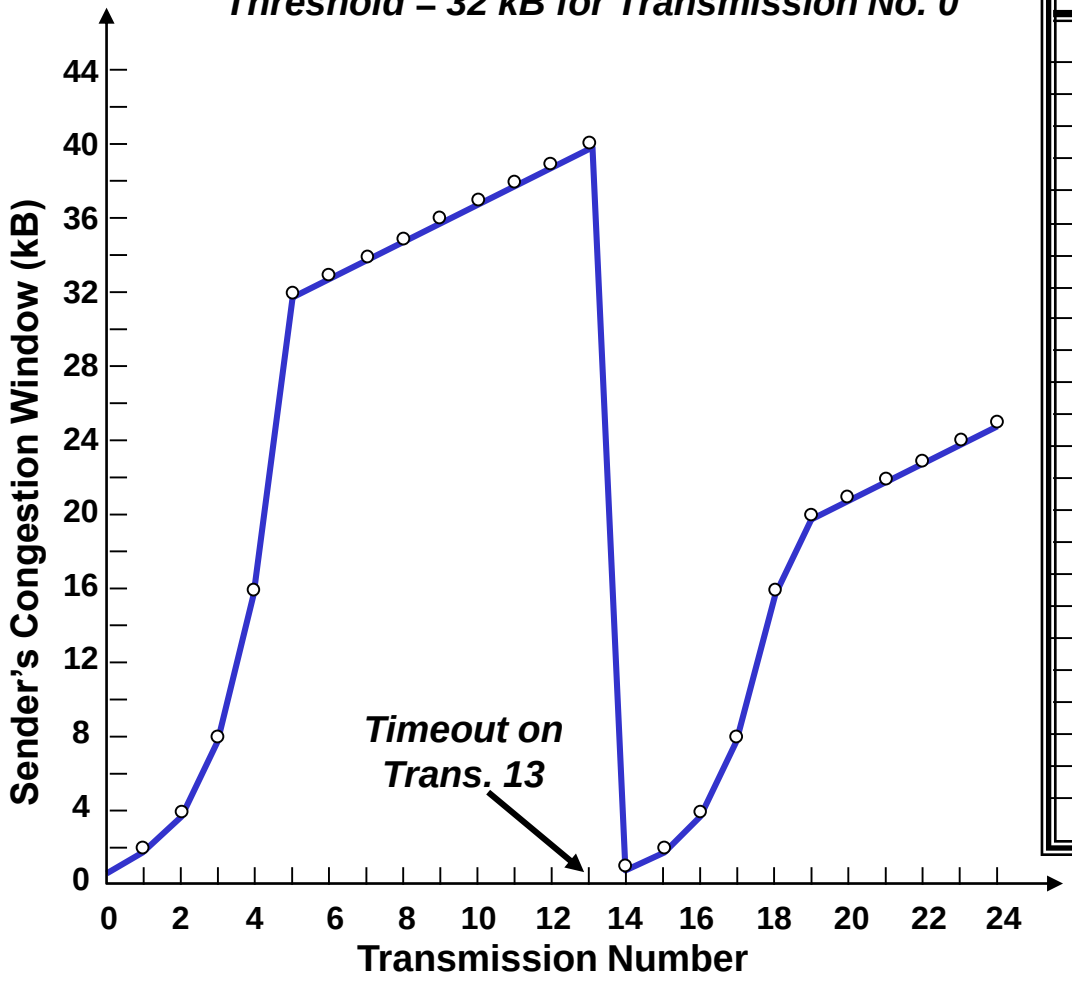


TCP Congestion Control

- assume: Timeout ⇒ packet loss due to congestion
  - ➢ packet loss due to transmission errors is assumed to be relatively rare (?)
- arrival of ACK at Sender ⇒ TCP segment has left network at Receiver, so can send another TCP segment without adding to network congestion
  - ➢ trying to maintain "conservation of segments" by using ACKs to pace the transmission of data ⇒ TCP is said to be *self-clocking*
- Sender sends the minimum of what Receiver and network can take
  - ➢ when connection initialised, Sender initialises ***congestion window*** to maximum TCP segment size (MSS) and sends one MSS
  - ➢ ACK received in time ⇒ add one MSS to congestion window, and send two MSS's; their ACKs received in time ⇒ add two MSS's to congestion window, and send four MSS's…
  - ➢ congestion window grows exponentially until Timeout occurs or Receiver's advertised window reached: this is **slow-start**
  - ➢ Timeout ⇒ reset ***Threshold*** (initially 64 kB) to 1/2 of current congestion window, then reset congestion window to one MSS; slow-start used until Threshold value reached, after which each successful transmission grows the congestion window by one MSS: this is **linear increase**

TCP Congestion Control: Example 1

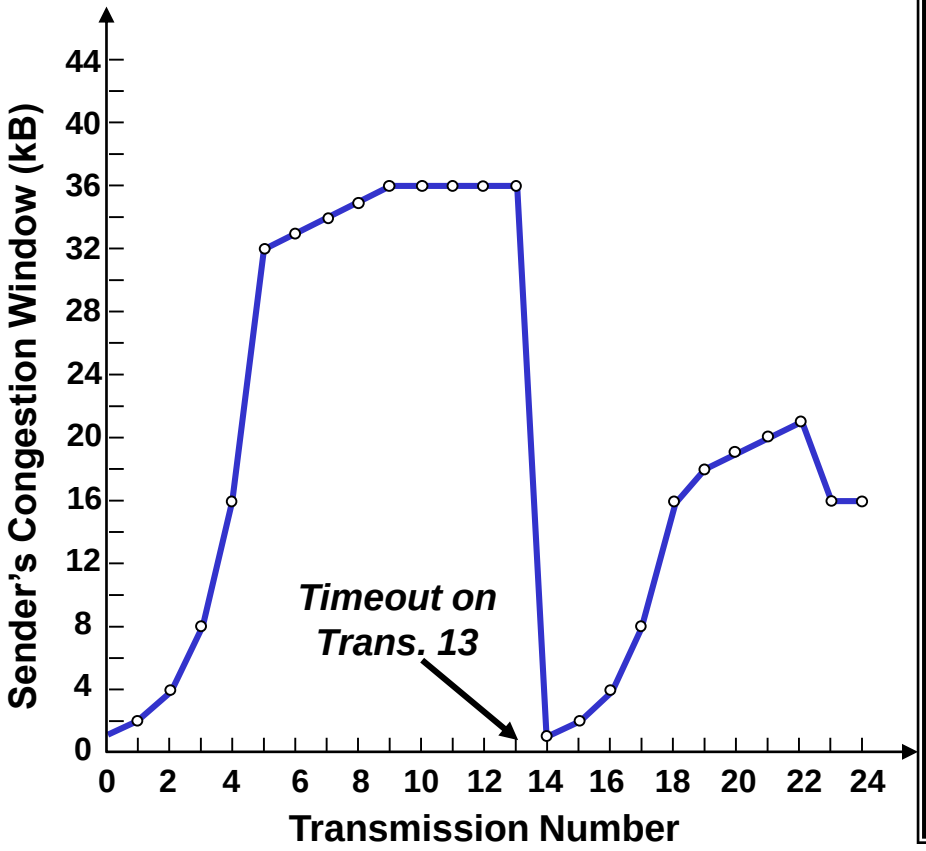
MSS = 1 kB
Receiver's advertised window = 64 kB always
Threshold = 32 kB for Transmission No. 0



Transmission Number	Congestion Window (kB)	Threshold (kB)
0	1	32
1	2	32
2	4	32
3	8	32
4	16	32
5	32	32
6	33	32
7	34	32
8	35	32
9	36	32
10	37	32
11	38	32
12	39	32
13	40	32
14	1	20
15	2	20
16	4	20
17	8	20
18	16	20
19	20	20
20	21	20
21	22	20
22	23	20
23	24	20
24	25	20

TCP Congestion Control: Example 2

MSS = 1kB
Receiver's advertised window = 36 kB initially
Threshold = 32 kB for Transmission No. 0



Transmission Number	Congestion Window (kB)	Threshold (kB)
0	1	32
1	2	32
2	4	32
3	8	32
4	16	32
5	32	32
6	33	32
7	34	32
8	35	32
9	36	32
10	36	32
11	36	32
12	36	32
13	36	32
14	1	18
15	2	18
16	4	18
17	8	18
18	16	18
19	18	18
20	19	18
21	20	18
22	21	18
23	16	18
24	16	18

Receiver's advertised window = 36 kB

Receiver's advertised window = 16 kB

In TCP, TIMEOUT value is updated dynamically, based on measurements of the connection's round-trip time (RTT) and the variation in RTT