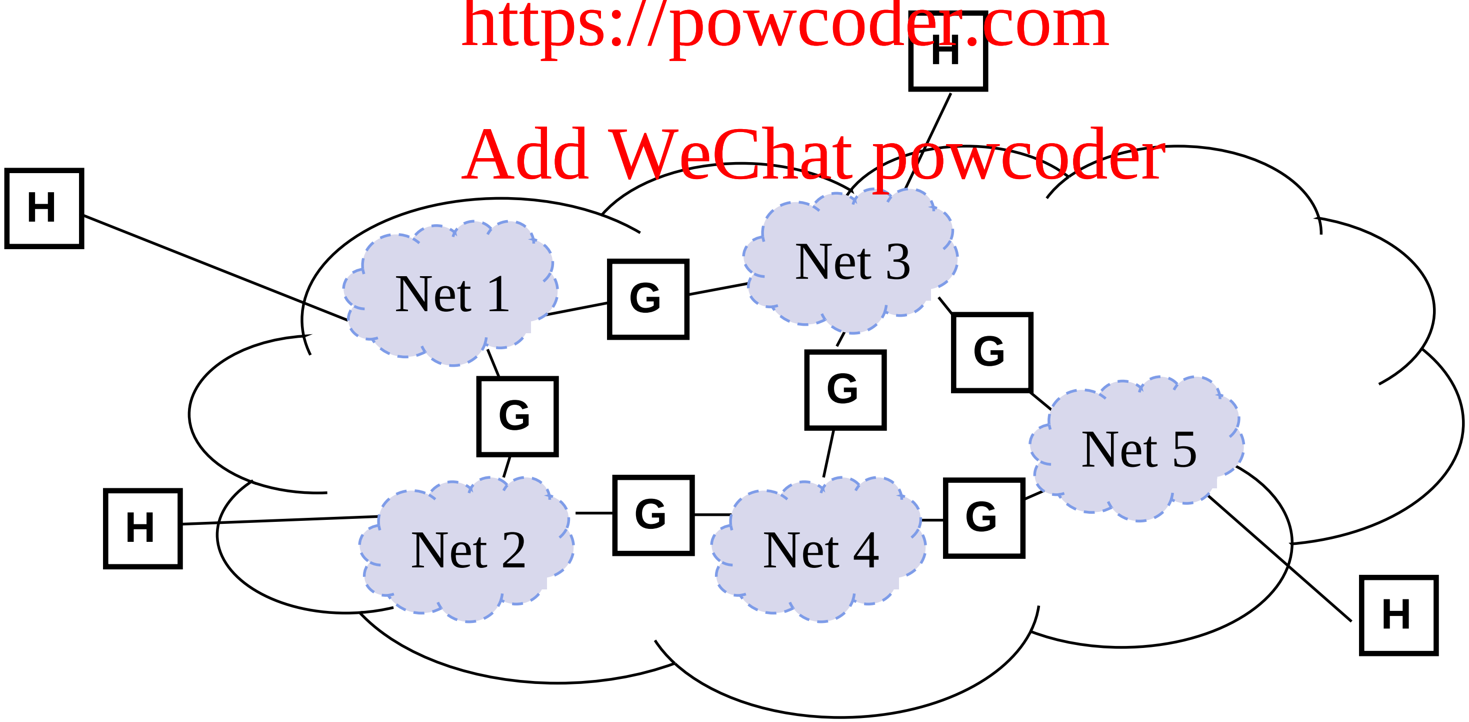
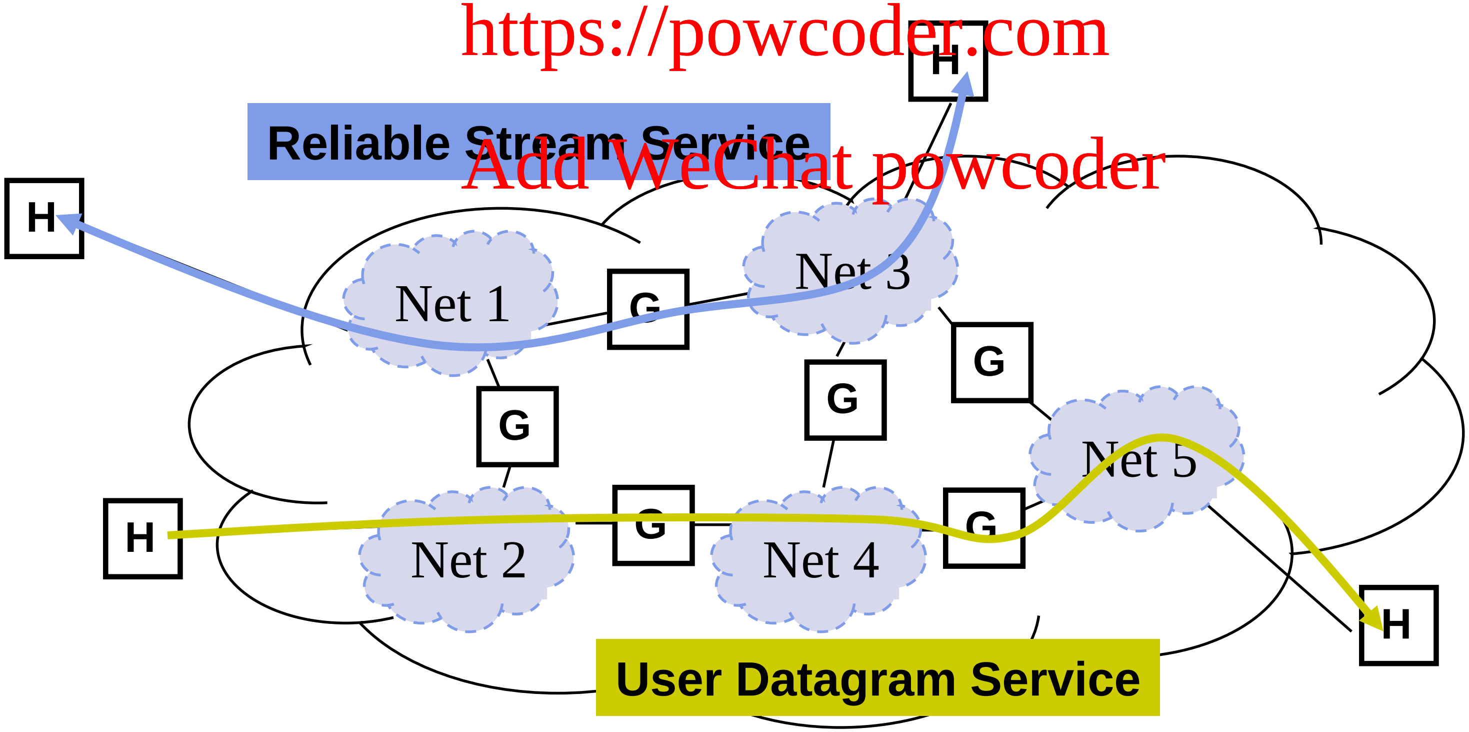
Why Internetworking?

- To build a “network of networks” or internet
  - operating over multiple, coexisting, different network technologies
  - providing ubiquitous connectivity through IP packet transfer
  - achieving huge economies of scale



Why Internetworking?

- To provide *universal communication services*
  - independent of underlying network technologies
  - providing common interface to user applications



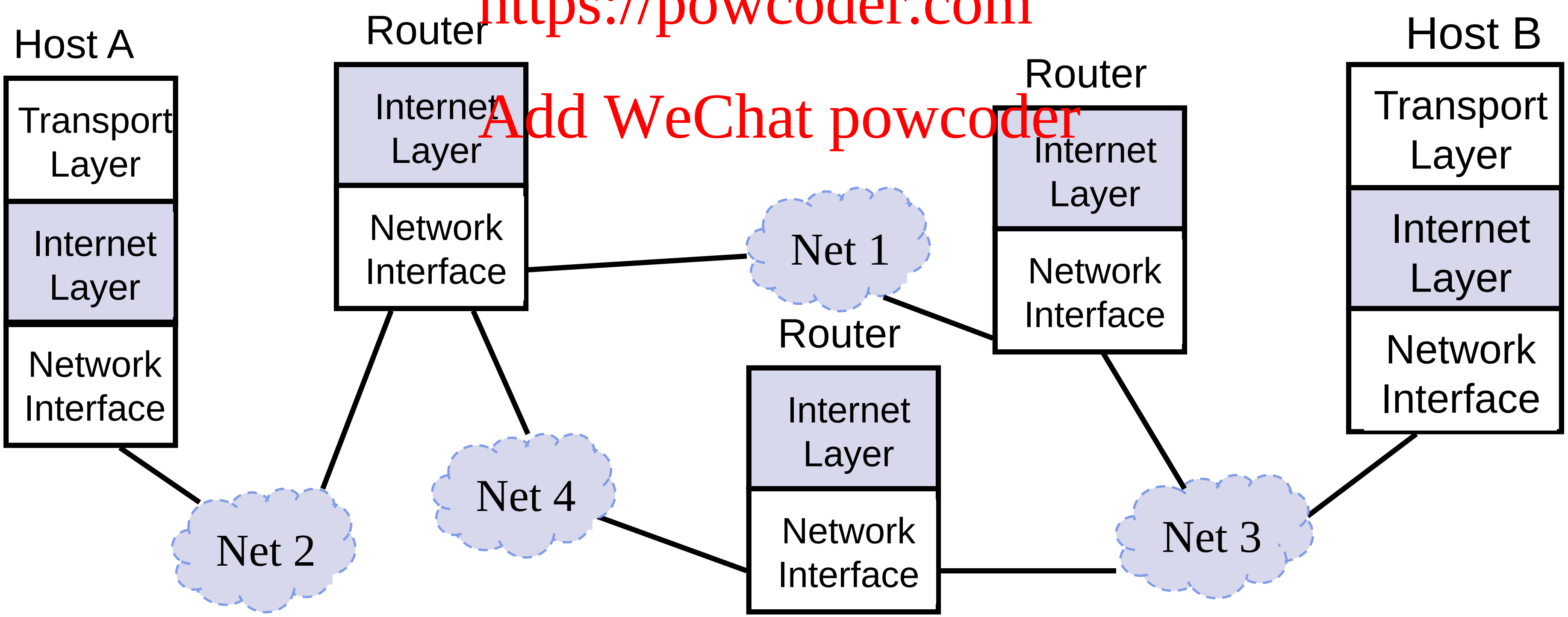
Why Internetworking?

- To provide *distributed applications*
  - Any application designed to operate based on Internet communication services immediately operates across the entire Internet
  - Rapid deployment of new applications
    - Email, WWW, Peer-to-peer
  - Applications independent of network technology
    - New networks can be introduced below
    - Old network technologies can be retired

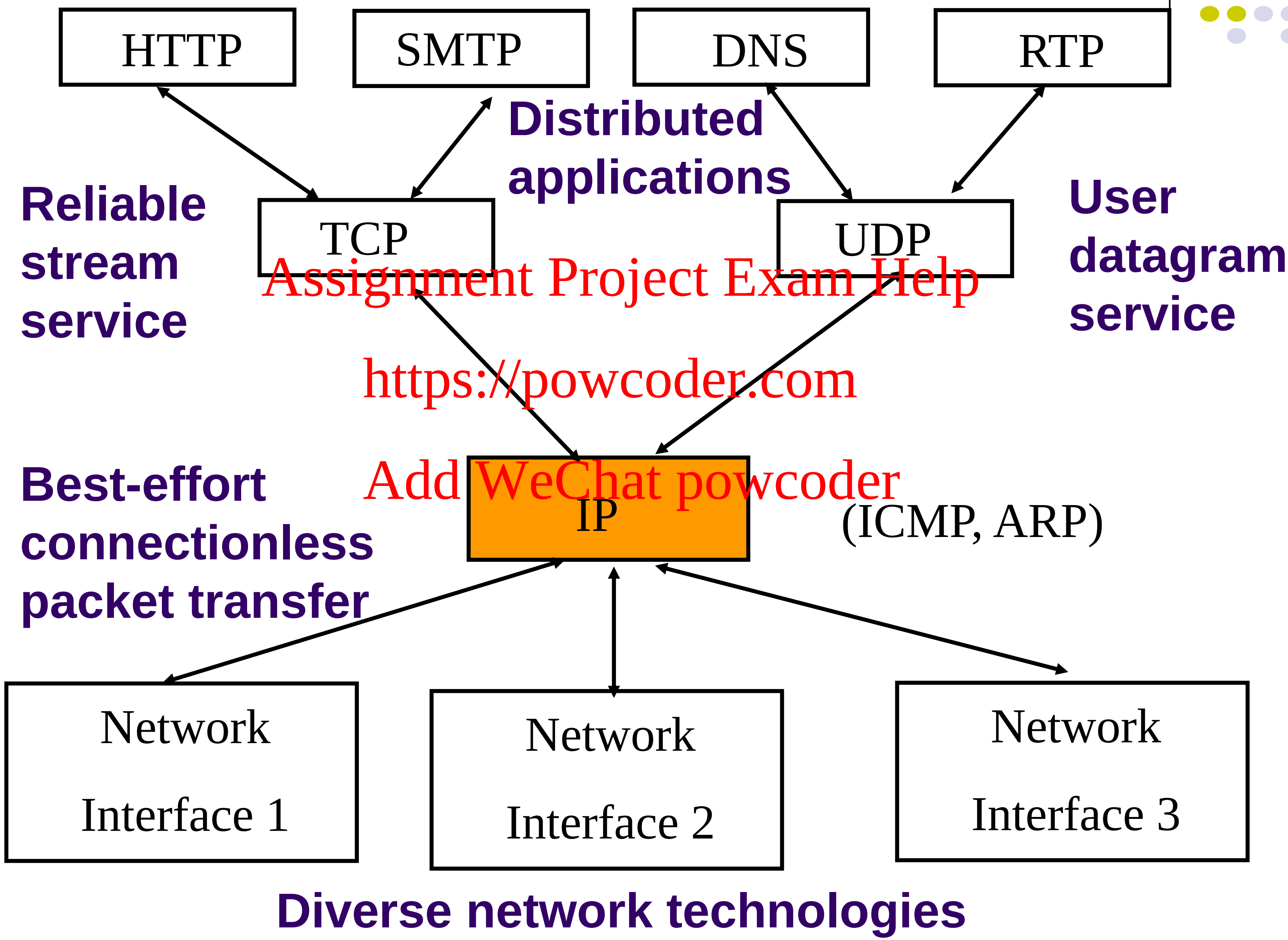
Internet Protocol Approach

- IP packets transfer information across Internet

  Host A IP → router→ router…→ router→ Host B IP
- IP layer in each router determines next hop (router)
- Network interfaces transfer IP packets across networks



TCP/IP Protocol Suite



Internet Names & Addresses

Internet Names

- Each host has a unique name
  - Independent of physical location
  - Facilitate memorization by humans
  - Domain Name
  - Organization under single administrative unit
- Host Name
  - Name given to host computer
- User Name
  - Name assigned to user

Internet Addresses

- Each host has globally unique *logical* 32 bit IP address
- Separate address for each physical connection to a network
- Routing decision is done based on destination IP address
- IP address has two parts:
  - *netid* and *hostid*
  - *netid* unique
  - *netid* facilitates routing
- Dotted Decimal Notation:

  int1.int2.int3.int4

  (intj = jth octet)

  128.100.10.13

DNS resolves IP name to IP address

leongarcia@comm.utoronto.ca

Physical Addresses

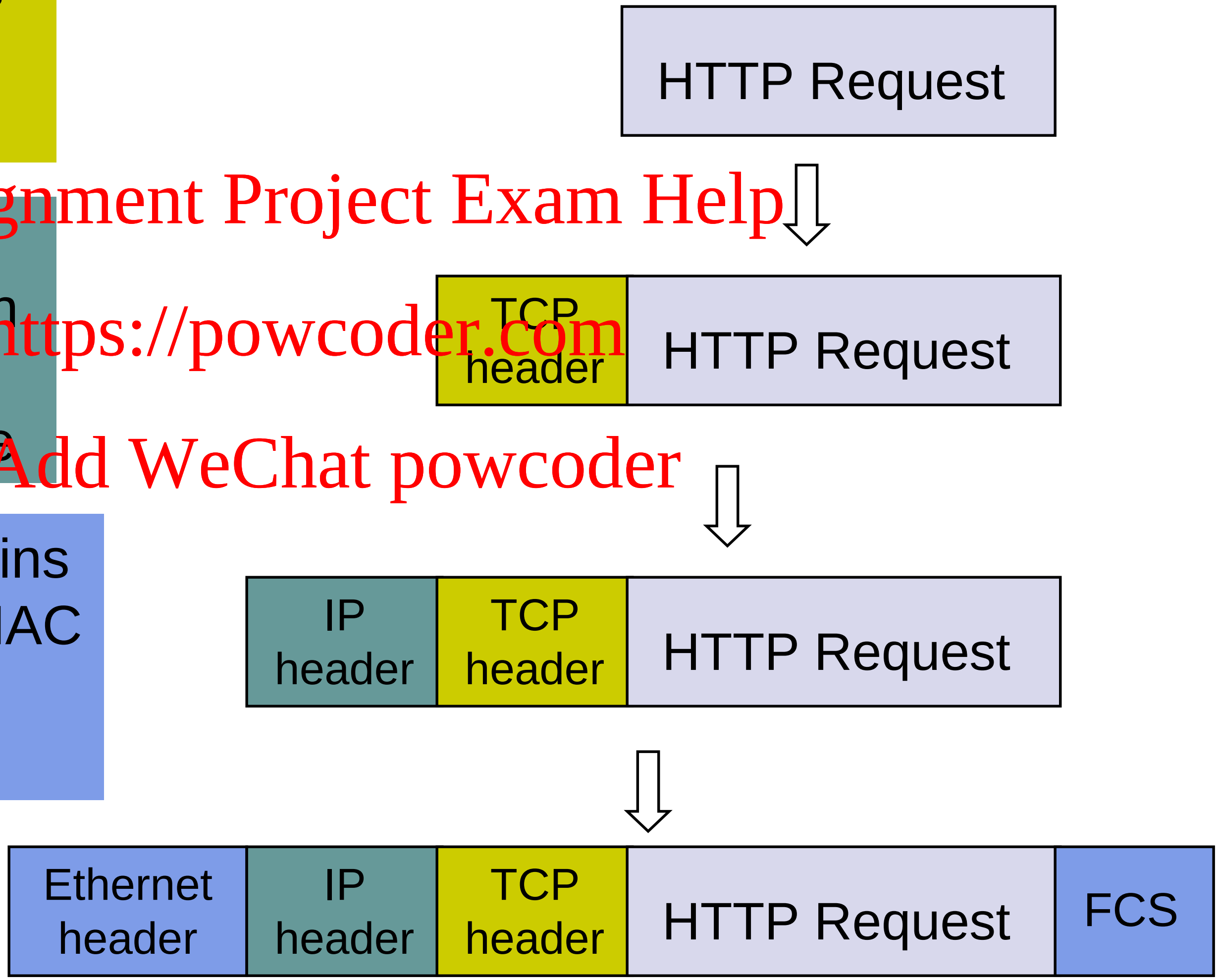
- LANs (and other networks) assign physical addresses to the physical attachment to the network
- The network uses its own address to transfer packets or frames to the appropriate destination
- IP address needs to be resolved to physical address at each IP network interface
- Example: Ethernet uses 48-bit addresses
  - Each Ethernet network interface card (NIC) has globally unique Medium Access Control (MAC) or physical address
  - First 24 bits identify NIC manufacturer; second 24 bits are serial number
  - $\underbrace{\text{00:90:27}}_{\text{Intel}}$:96:68:07   12 hex numbers

Encapsulation

TCP Header contains source & destination port numbers

IP Header contains source and destination IP addresses; transport protocol type

Ethernet Header contains source & destination MAC addresses;
network protocol type



Internet Protocol

- Provides best effort, connectionless packet delivery
  - motivated by need to keep routers simple and by adaptibility to failure of network elements
  - packets may be lost, out of order, or even duplicated
  - higher layer protocols must deal with these, if necessary
- RFCs 791, 950, 919, 922, and 2474.
- Internet Control Message Protocol (ICMP), RFC 792
- Internet Group Management Protocol (IGMP), RFC 1112

IP Packet Header

0 4 8 16 19 24 31

Version	IHL	Type of Service	Total Length	
Identification			Flags	Fragment Offset
Time to Live		Protocol	Header Checksum	
Source IP Address				
Destination IP Address				
Options				Padding

- Minimum 20 bytes
- Up to 40 bytes in options fields

IP Packet Header

0	4	8	16	19	24	31
Version	IHL	Type of Service	Total Length			
Identification			Flags	Fragment Offset		
Time to Live		Protocol	Header Checksum			
Source IP Address						
Destination IP Address						
Options					Padding	

Version: current IP version is 4.

Internet header length (IHL): length of the header in 32-bit words.

Type of service (TOS): traditionally priority of packet at each router. Recent Differentiated Services redefines TOS field to include other services besides best effort.

IP Packet Header

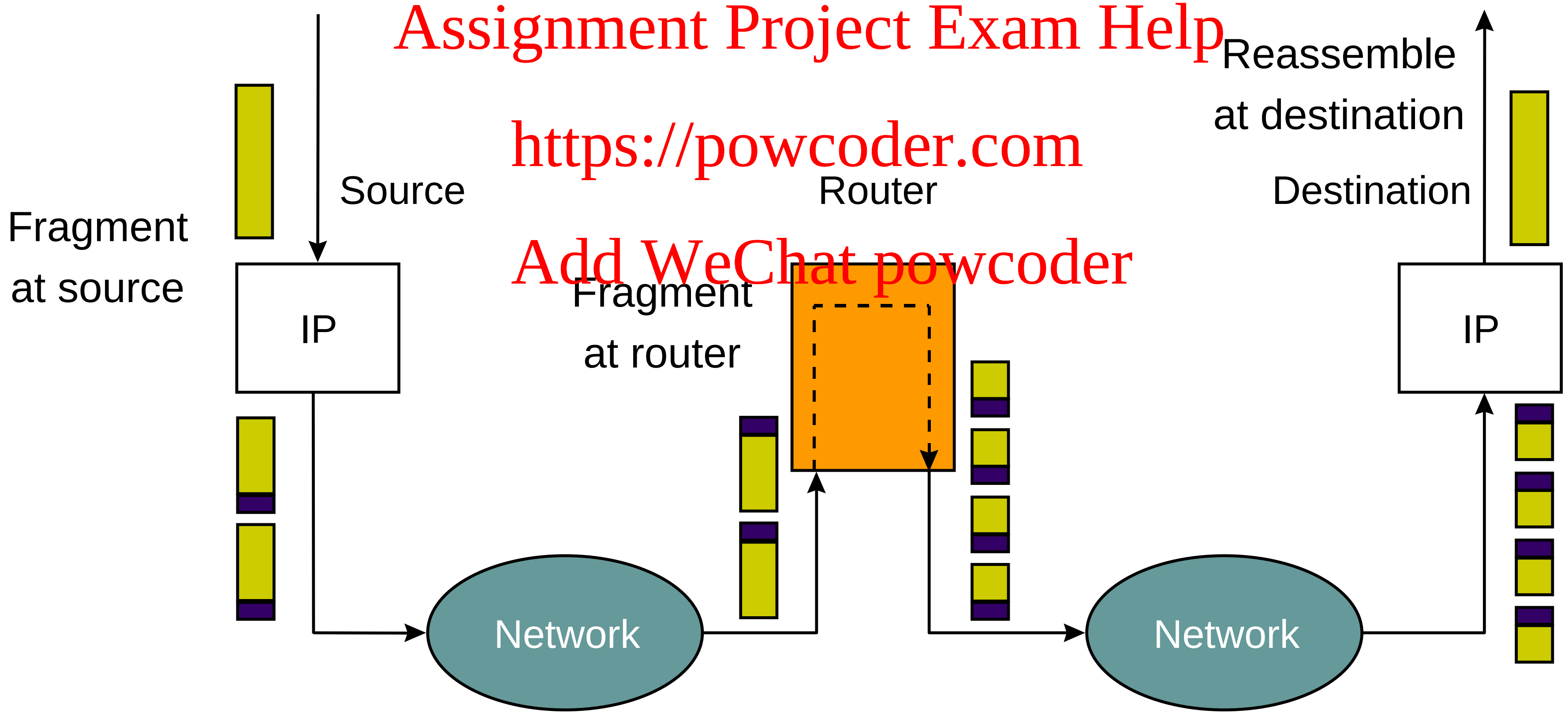
0	4	8	16	19	24	31
Version	IHL	Type of Service	Total Length			
Identification			Flags	Fragment Offset		
Time to Live		Protocol	Header Checksum			
Source IP Address						
Destination IP Address						
Options					Padding	

Total length: number of bytes of the IP packet including header and data, maximum length is 65535 bytes.

Identification, Flags, and Fragment Offset: used for fragmentation and reassembly (More on this shortly).

Fragmentation and Reassembly

- Identification identifies a particular packet
- Flags = (unused, don’t fragment/DF, more fragment/MF)
- Fragment offset identifies the location of a fragment within a packet



Example: Fragmenting a Packet

- A packet is to be forwarded to a network with MTU of 576 bytes. The packet has an IP header of 20 bytes and a data part of 1484 bytes. and of each fragment.
- Maximum data length per fragment = 576 – 20 = 556 bytes.
- We set maximum data length to 552 bytes to get multiple of 8.

	Total Length	Id	MF	Fragment Offset
Original packet	1504	x	0	0
Fragment 1	572	x	1	0
Fragment 2	572	x	1	69
Fragment 3	400	x	0	138

IP Packet Header

0 4 8 16 19 24 31

<table>
<tr><td>Version</td><td>IHL</td><td>Type of Service</td><td colspan="3">Total Length</td></tr>
<tr><td colspan="3">Identification</td><td>Flags</td><td colspan="2">Fragment Offset</td></tr>
<tr><td colspan="2">Time to Live</td><td>Protocol</td><td colspan="3">Header Checksum</td></tr>
<tr><td colspan="6">Source IP Address</td></tr>
<tr><td colspan="6">Destination IP Address</td></tr>
<tr><td colspan="5">Options</td><td>Padding</td></tr>
</table>

Time to live (TTL): number of hops packet is allowed to traverse in the network.

- Each router along the path to the destination decrements this value by one.
- If the value reaches zero before the packet reaches the destination, the router discards the packet and sends an error message back to the source.

IP Packet Header

0	4	8	16	19	24	31
Version	IHL	Type of Service	Total Length			
Identification			Flags	Fragment Offset		
Time to Live		Protocol	Header Checksum			
Source IP Address						
Destination IP Address						
Options					Padding	

Protocol: specifies upper-layer protocol that is to receive IP data at the destination. Examples include TCP (protocol = 6), UDP (protocol = 17), and ICMP (protocol = 1).

Header checksum: verifies the integrity of the IP header.

Source IP address and **destination IP address:** contain the addresses of the source and destination hosts.

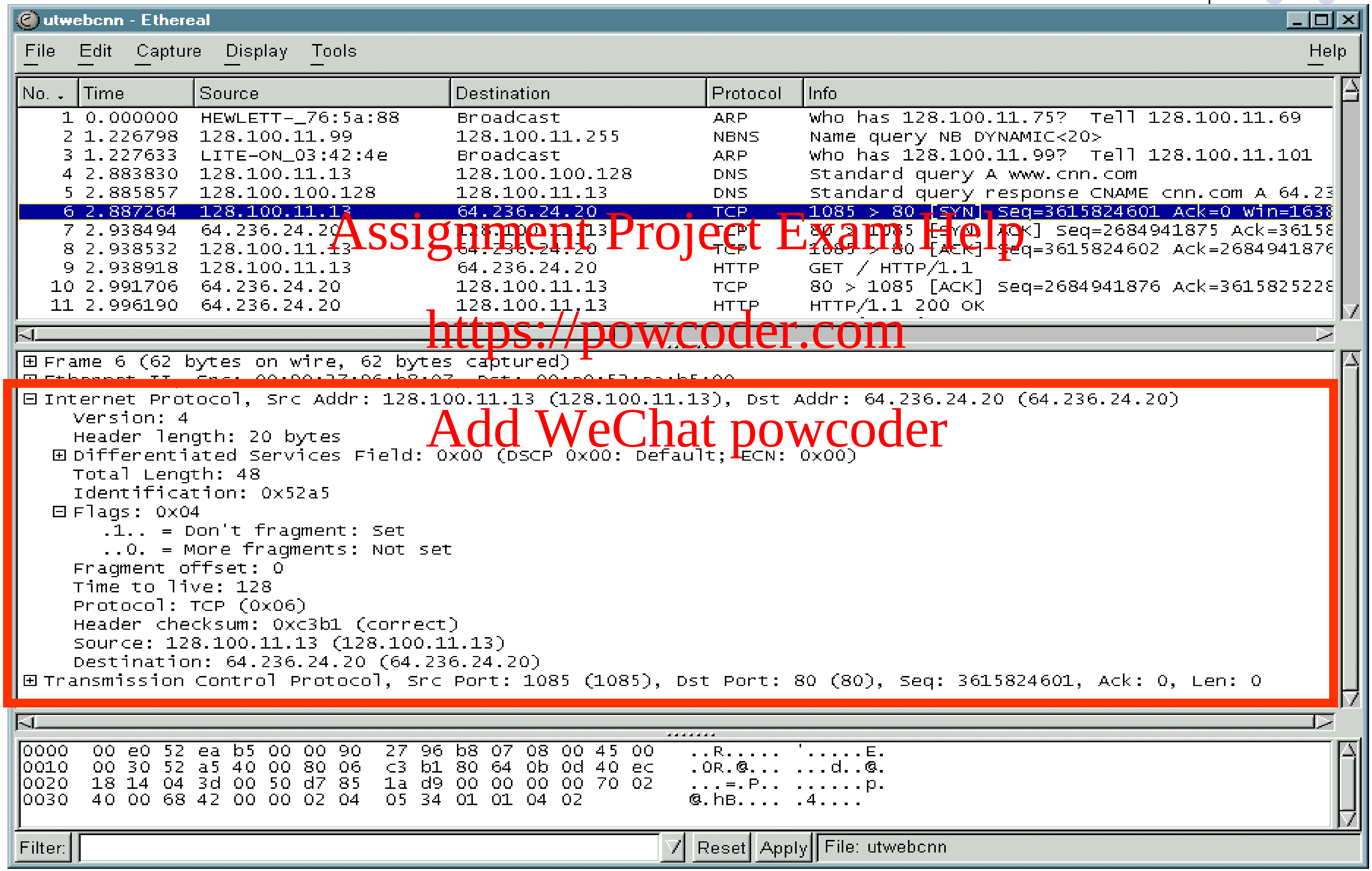
IP Packet Header

0	4	8	16	19	24	31
Version	IHL	Type of Service	Total Length			
Identification			Flags	Fragment Offset		
Time to Live		Protocol	Header Checksum			
Source IP Address						
Destination IP Address						
Options					Padding	

Options: Variable length field, allows packet to request special features such as security level, route to be taken by the packet, and timestamp at each router. Detailed descriptions of these options can be found in [RFC 791].

Padding: This field is used to make the header a multiple of 32-bit words.

Example of IP Header



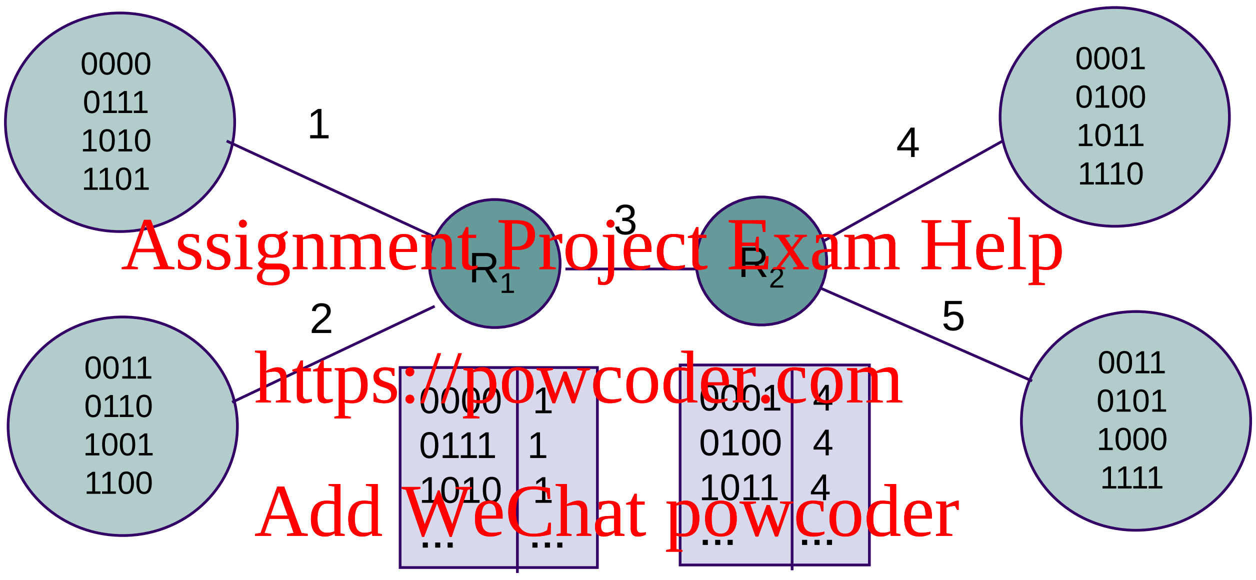
Header Checksum

- IP header uses check bits to detect errors in the ***header***
- A checksum is calculated for header contents
- Checksum recalculated at every router, so algorithm selected for ease of implementation in software

IP Header Processing

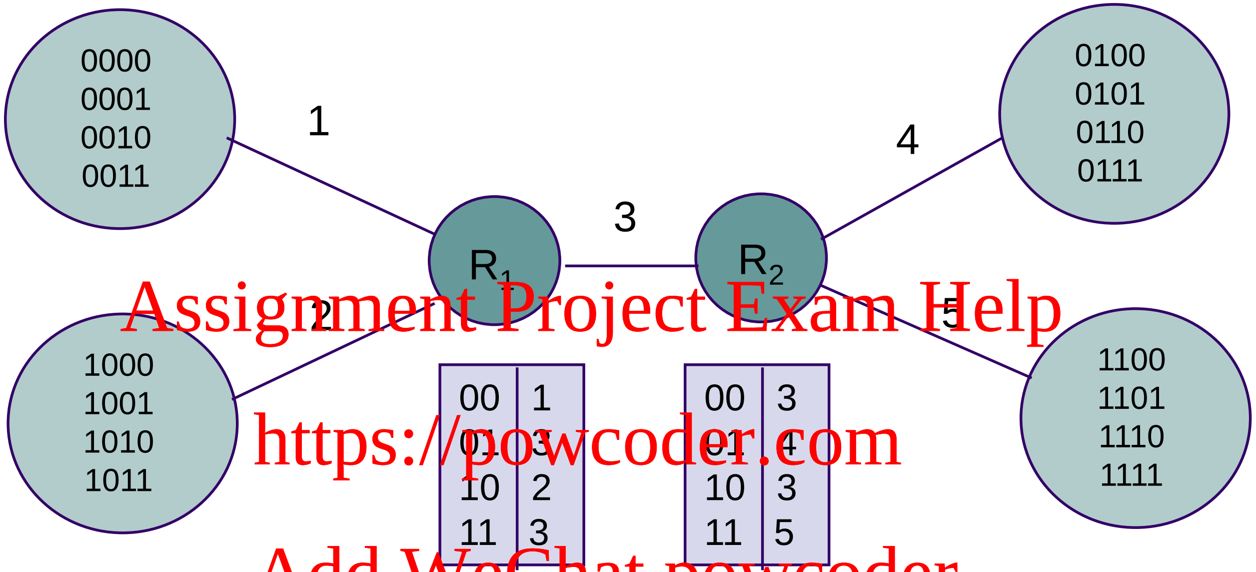
1. Compute header checksum for correctness and check that fields in header (e.g. version and total length) contain valid values
2. Consult routing table to determine next hop
3. Change fields that require updating (TTL, header checksum)

Non-Hierarchical Addresses and Routing



- No relationship between addresses & routing proximity
- Routing tables require 16 entries each

Hierarchical Addresses and Routing



- Prefix indicates network where host is attached
- Routing tables require 4 entries each

IP Addressing

- RFC 1166
- Each host on Internet has unique 32 bit IP address
- Each address has two parts: *netid* and *hostid*
- *netid* unique & administered by
  - American Registry for Internet Numbers (ARIN)
  - Reseaux IP Europeens (RIPE)
  - Asia Pacific Network Information Centre (APNIC)
- Facilitates routing
- A separate address is required for each physical connection of a host to a network; “multi-homed” hosts
- Dotted-Decimal Notation:

  int1.int2.int3.int4 where intj = integer value of jth octet

  IP address of 10000000 10000111 01000100 00000101

  is 128.135.68.5 in dotted-decimal notation

Classful Addresses

Class A

	7 bits	24 bits
0	netid	hostid

- 126 networks with up to 16 million hosts

1.0.0.0 to 127.255.255.255

Class B

		14 bits	16 bits
1	0	netid	hostid

- 16,382 networks with up to 64,000 hosts

128.0.0.0 to 191.255.255.255

Class C

			22 bits	8 bits
1	1	0	netid	hostid

- 2 million networks with up to 254 hosts

192.0.0.0 to 223.255.255.255

Class D

				28 bits
1	1	1	0	multicast address

224.0.0.0 to 239.255.255.255

- Up to 250 million multicast groups at the same time

- Permanent group addresses

  - All systems in LAN; All routers in LAN;

  - All OSPF routers on LAN; All designated OSPF routers on a LAN, etc.
- Temporary groups addresses created as needed

Reserved Host IDs (all 0s & 1s)

Internet address used to refer to network has hostid set to all 0s

0	0	0	0		0	0	this host (used when booting up)

0	0		0	**host**		a host in this network

Broadcast address has hostid set to all 1s

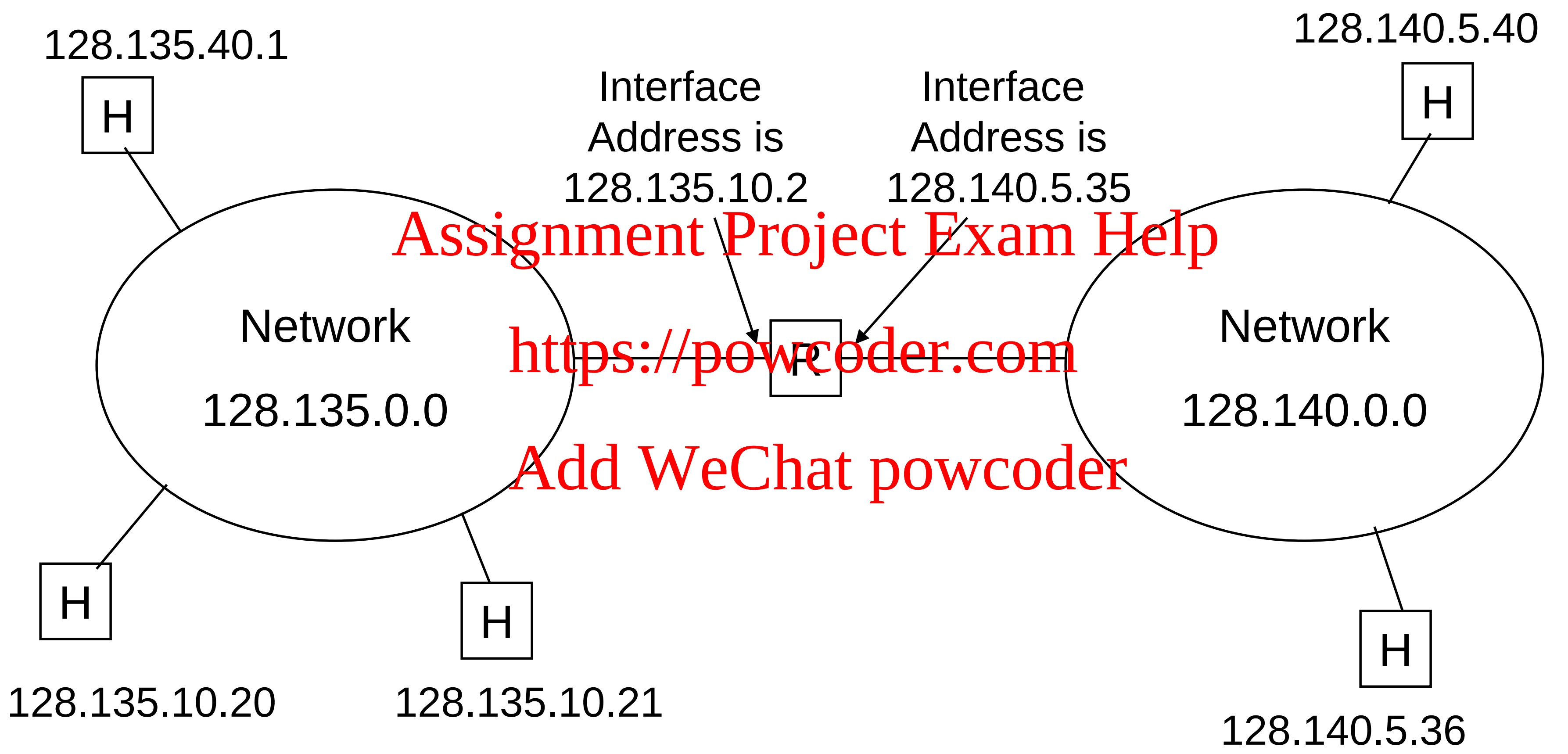
1	1	1	1		1	1	broadcast on local network

netid	1	1	1		1	1	1	1		broadcast on distant network

Private IP Addresses

- Specific ranges of IP addresses set aside for use in private networks (RFC 1918)
- Use restricted to private internets; routers in public Internet discard packets with these addresses
- Range 1: 10.0.0.0 to 10.255.255.255
- Range 2: 172.16.0.0 to 172.31.255.255
- Range 3: 192.168.0.0 to 192.168.255.255
- Network Address Translation (NAT) used to convert between private & global IP addresses

Example of IP Addressing



Address with host ID=all 0s refers to the network

Address with host ID=all 1s refers to a broadcast packet

R = router

H = host

Subnet Addressing

- Subnet addressing introduces another hierarchical level
- Transparent to remote networks
- Simplifies management of multiplicity of LANs
- Masking used to find subnet number

Original address	1	0	Net ID	Host ID

Subnetted address	1	0	Net ID	Subnet ID	Host ID

Subnetting Example

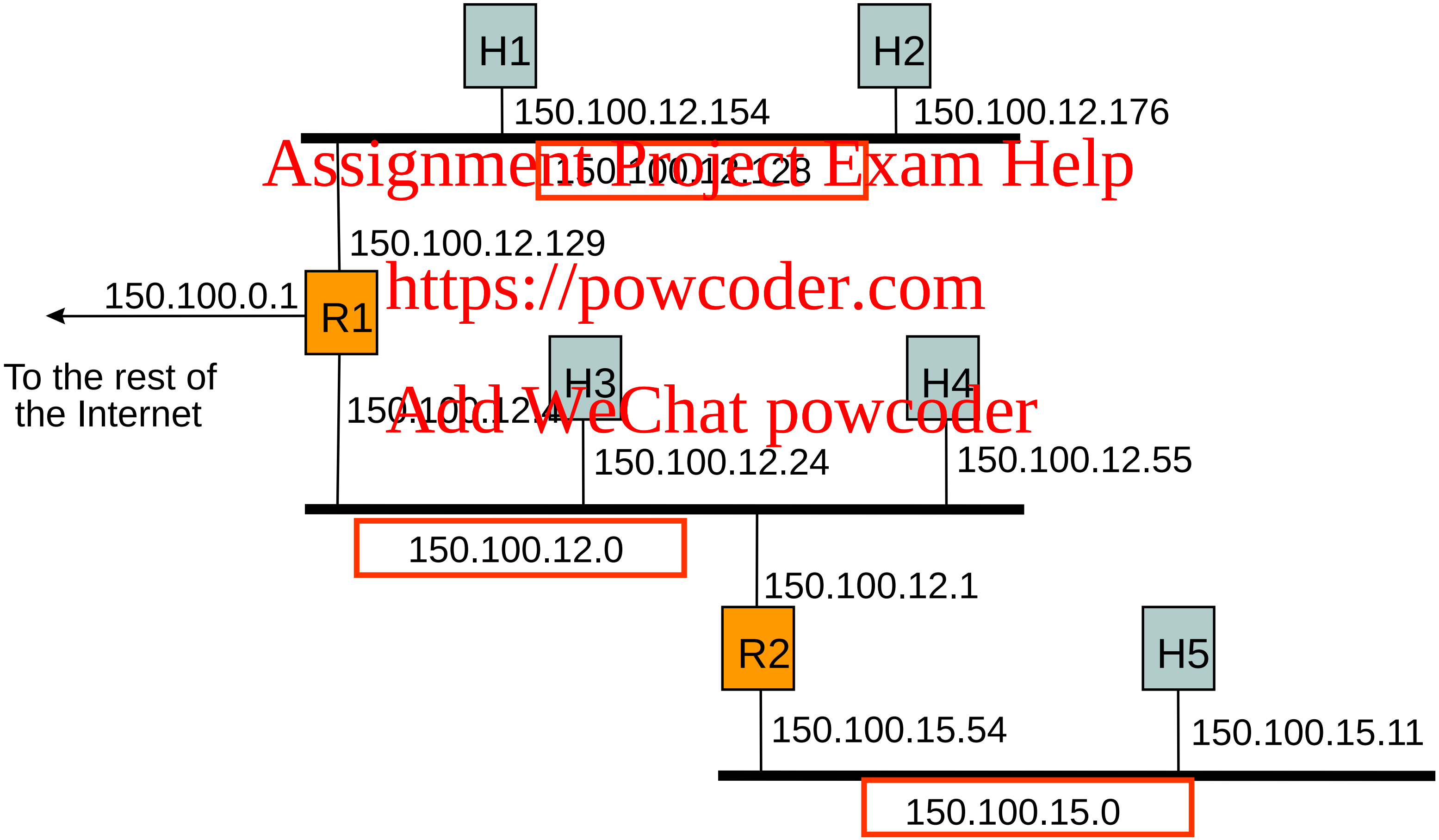
- Organization has Class B address (16 host ID bits) with network ID: 150.100.0.0
- Create subnets with up to 100 hosts each
  - 7 bits sufficient for each subnet

  - 16-7=9 bits for subnet ID

- Apply subnet mask to IP addresses to find corresponding subnet

  - Example: Find subnet for 150.100.12.176
  - IP add = 10010110 01100100 00001100 10110000
  - Mask = 11111111 11111111 11111111 10000000
  - AND = 10010110 01100100 00001100 10000000
  - Subnet = 150.100.12.128
  - Subnet address used by routers within organization

Subnet Example



Routing with Subnetworks

- IP layer in hosts and routers maintain a routing table
- Originating host: To send an IP packet, consult routing table

  - If destination host is in same network, send packet *directly* using appropriate network interface

  - Otherwise, send packet indirectly; typically, routing table indicates a default router

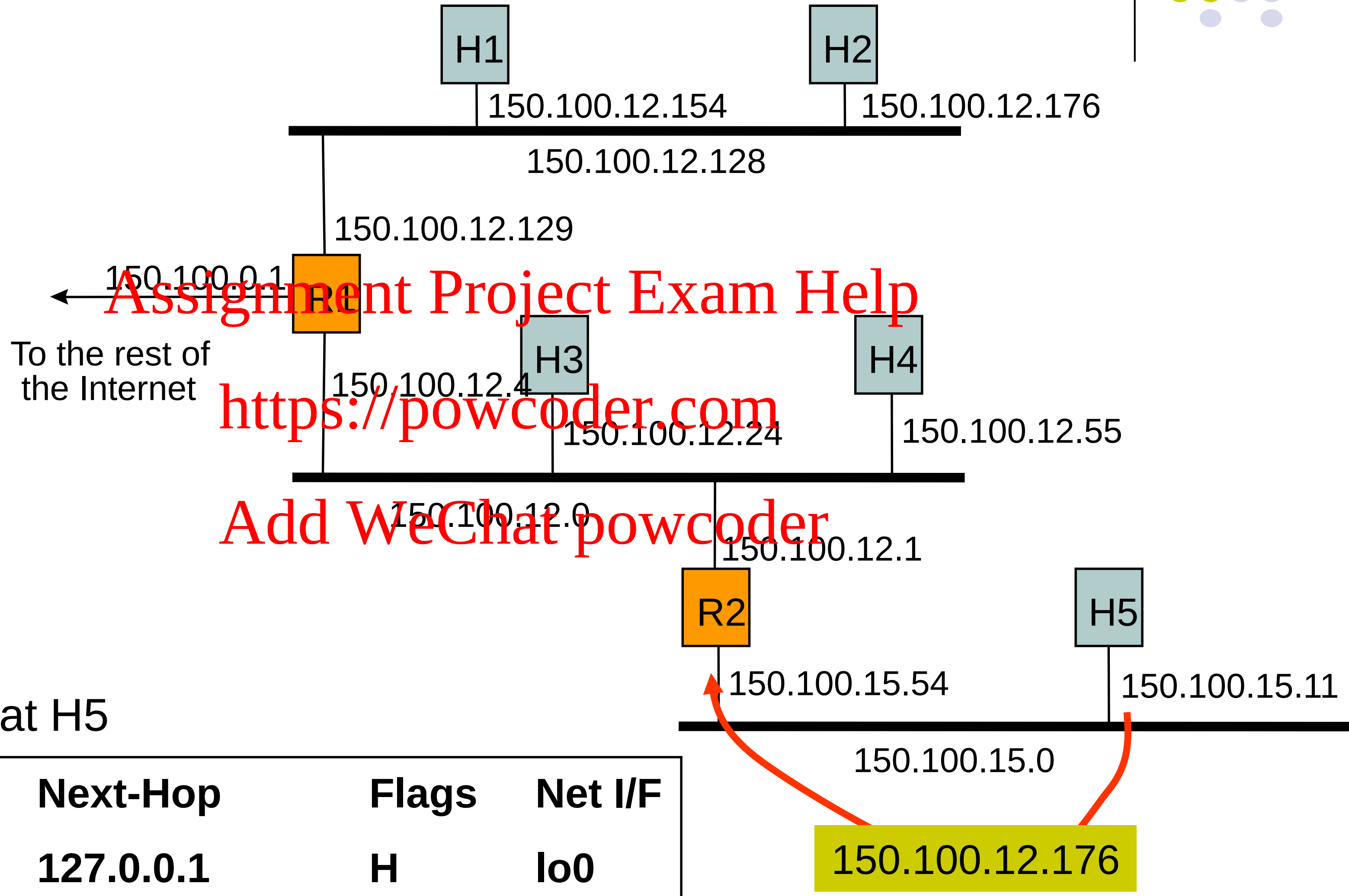
- Router: Examine IP destination address in arriving packet
  - If dest IP address not own, router consults routing table to determine next-hop and associated network interface & forwards packet

Routing Table

- Each row in routing table contains:
  - Destination IP address
  - IP address of next-hop router
  - Physical address
  - Statistics information
  - Flags
    - H=1 (0) indicates route is to a host (network)
    - G=1 (0) indicates route is to a router (directly connected destination)

- Routing table search order & action
  - Complete destination address; send as per next-hop & G flag
  - Destination network ID; send as per next-hop & G flag
  - Default router entry; send as per next-hop
  - Declare packet undeliverable; send ICMP “host unreachable error” packet to originating host

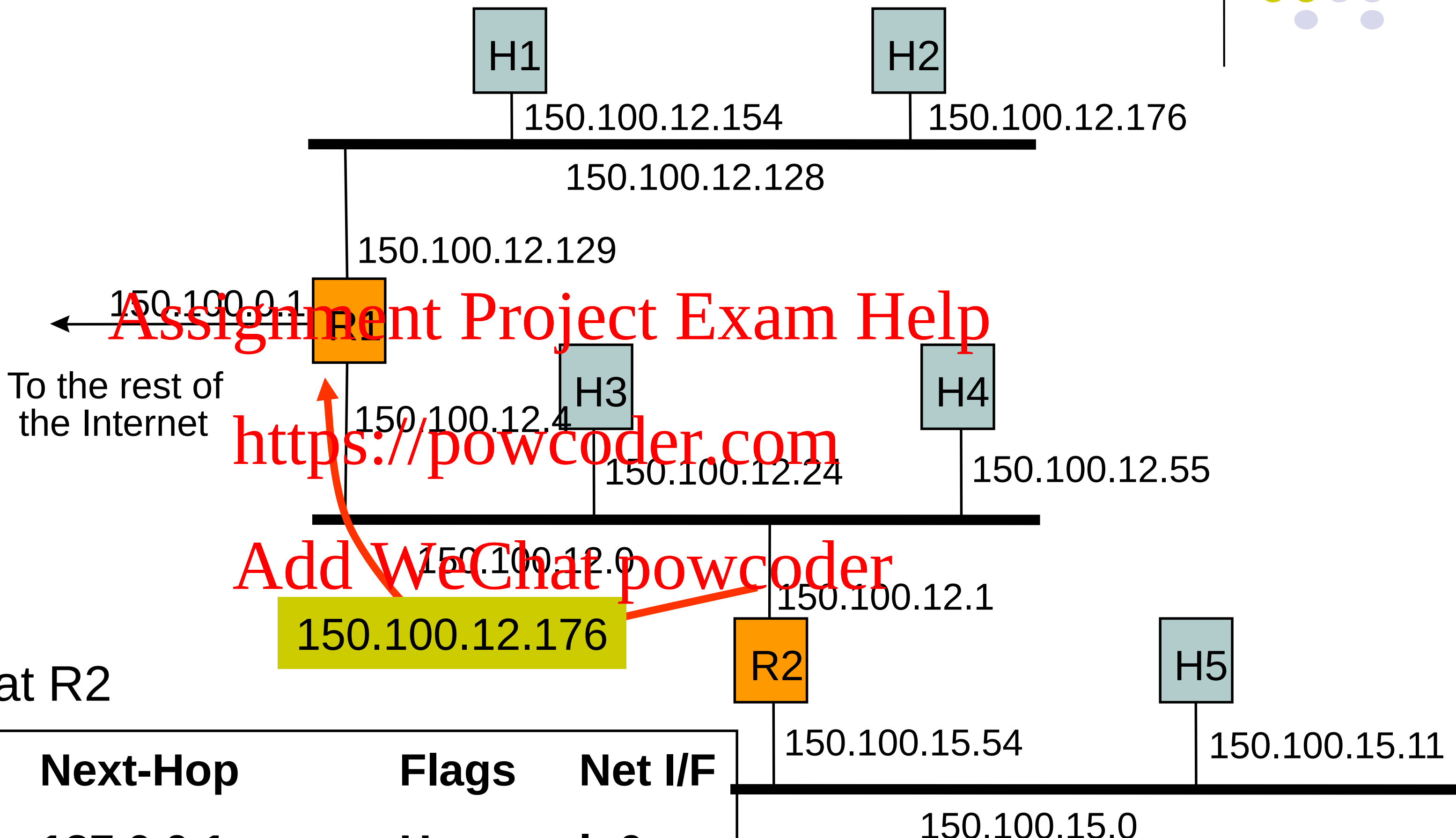
Example: Host H5 sends packet to host H2



Routing Table at H5

Destination	**Next-Hop**	**Flags**	**Net I/F**
127.0.0.1	**127.0.0.1**	**H**	**lo0**
default	**150.100.15.54**	**G**	**emd0**
150.100.15.0	**150.100.15.11**		**emd0**

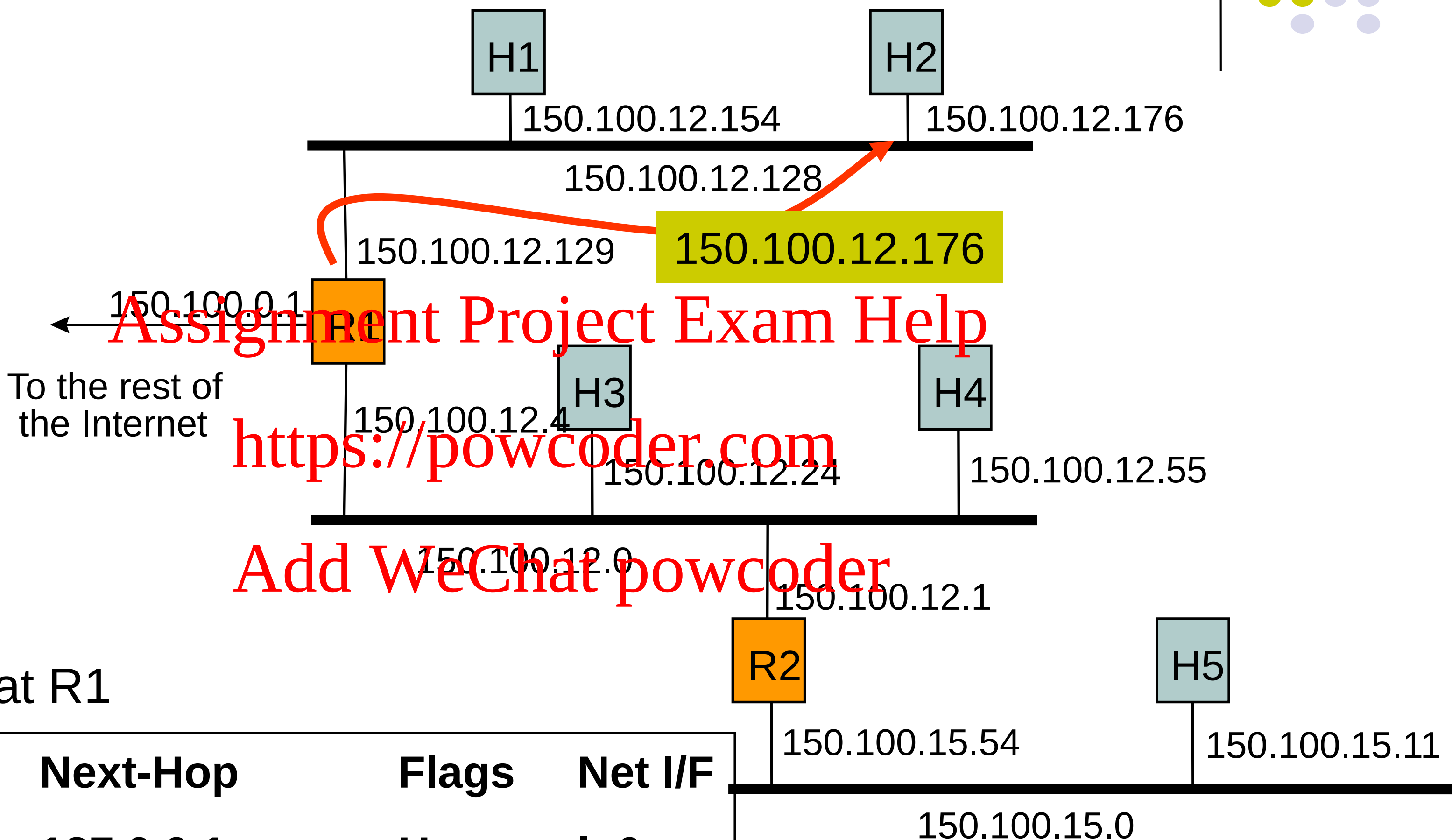
Example: Host H5 sends packet to host H2



Routing Table at R2

Destination	Next-Hop	Flags	Net I/F
127.0.0.1	127.0.0.1	H	lo0
default	150.100.12.4	G	emd0
150.100.15.0	150.100.15.54		emd1
150.100.12.0	150.100.12.1		emd0

Example: Host H5 sends packet to host H2



Routing Table at R1

Destination	Next-Hop	Flags	Net I/F
127.0.0.1	127.0.0.1	H	lo0
150.100.12.176	150.100.12.176		emd0
150.100.12.0	150.100.12.4		emd1
150.100.15.0	150.100.12.1	G	emd1

IP Address Problems

- In the 1990, two problems became apparent
  - IP addresses were being exhausted
  - IP routing tables were growing very large
- IP Address Exhaustion
  - Class A, B, and C address structure inefficient
    - Class B too large for most organizations, but future proof
    - Class C too small
    - Rate of class B allocation implied exhaustion by 1994
- IP routing table size
  - Growth in number of networks in Internet reflected in # of table entries
    - From 1991 to 1995, routing tables doubled in size every 10 months
    - Stress on router processing power and memory allocation
- Short-term solution:
- Classless Interdomain Routing (CIDR), RFC 1518
- New allocation policy (RFC 2050)
- Private IP Addresses set aside for intranets
- Long-term solution: IPv6 with much bigger address space

New Address Allocation Policy

- Class A & B assigned only for clearly demonstrated need
- *Consecutive* blocks of class C assigned (up to 64 blocks)
  - All IP addresses in the range have a common **prefix**, and every address with that prefix is within the range
  - Arbitrary prefix length for network ID improves efficiency

Address Requirement	Address Allocation
<256	1 Class C
256<,<512	2 Class C
512<,<1024	4 Class C
1024<,<2048	8 Class C
2048<,<4096	16 Class C
4096<,<8192	32 Class C
8192<,<16384	64 Class C

Supernetting

- Summarize a contiguous group of class C addresses using variable-length mask
- Example: 150.158.16.0/20

  - IP Address (150.158.16.0) & mask length (20)

  - IP add = 10010110 10011110 00010000 00000000
  - Mask = 11111111 11111111 11110000 00000000

  - Contains 16 Class C blocks:
  - From 10010110 10011110 00010000 00000000
  - i.e. 150.158.16.0
  - Up to 10010110 10011110 00011111 00000000
  - i.e. 150.158.31.0

Classless Inter-Domain Routing

- CIDR deals with Routing Table Explosion Problem
  - Networks represented by prefix and mask
  - Pre-CIDR: Network with range of 16 contiguous class C blocks requires 16 entries
  - Post-CIDR: Network with range of 16 contiguous class C blocks requires 1 entry
- Solution: *Route according to prefix of address*, not class
  - Routing table entry has <IP address, network mask>
  - Example: 192.32.136.0/21
  - 11000000 00100000 10001000 00000001 min address
  - 11111111 11111111 11111--- -------- mask
  - 11000000 00100000 10001--- -------- IP prefix
  - 11000000 00100000 10001111 11111110 max address
  - 11111111 11111111 11111--- -------- mask
  - 11000000 00100000 10001--- -------- same IP prefix

Longest Prefix Match

- CIDR impacts routing & forwarding
- Routing tables and routing protocols must carry IP address and mask

- Multiple entries may match a given IP destination address

- Example: Routing table may contain

  - 205.100.0.0/22 which corresponds to a given supernet
  - 205.100.0.0/20 which results from aggregation of a larger number of destinations into a supernet
  - Packet must be routed using the *more specific route*, that is, the longest prefix match
- Several fast longest-prefix matching algorithms are available

Address Resolution Protocol

Although IP address identifies a host, the packet is physically delivered by an underlying network (e.g., Ethernet) which uses its own *physical address* (MAC address in Ethernet). How to map an IP address to a physical address?

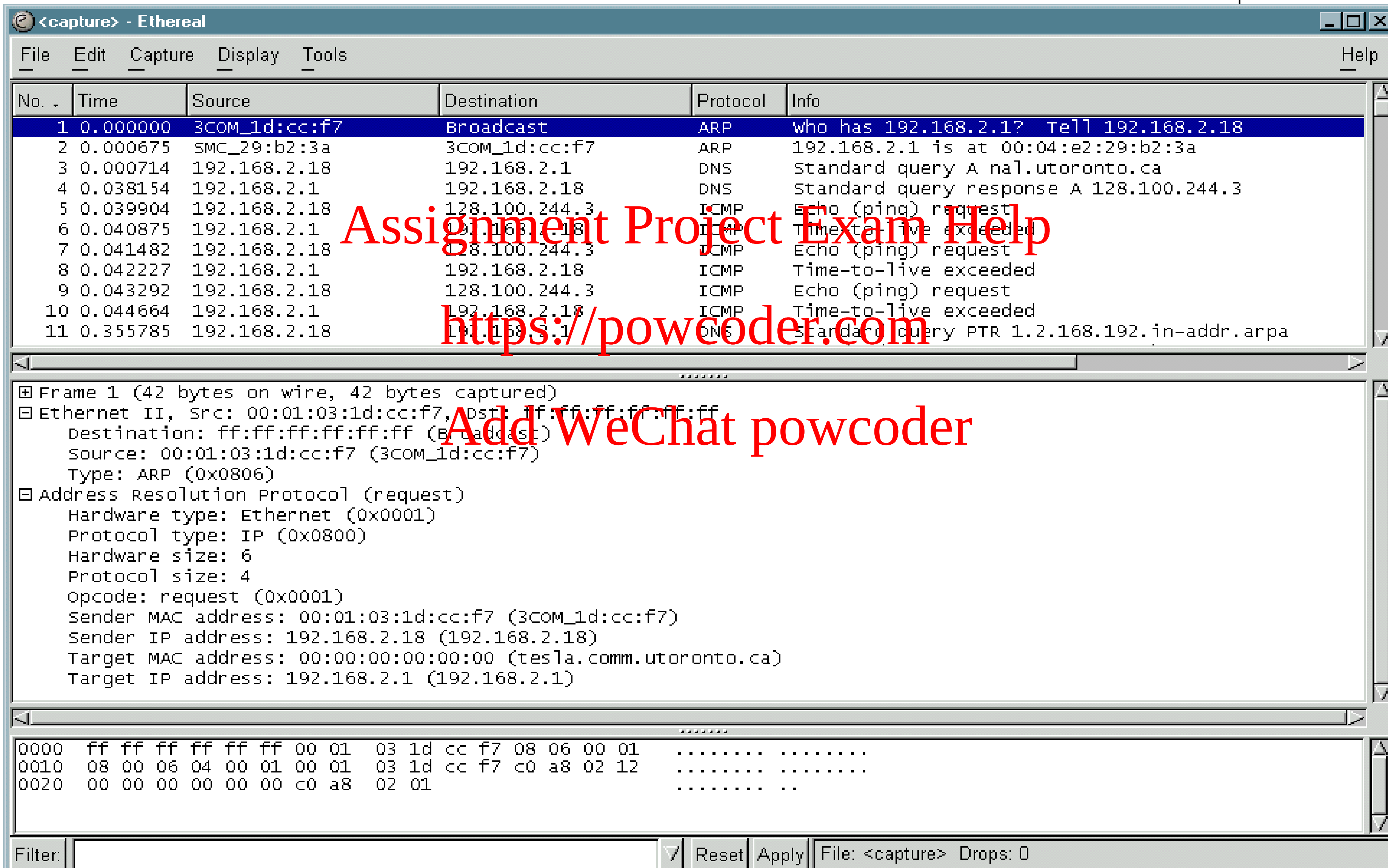
H1 wants to learn physical address of H3 -> broadcasts an ARP request

ARP request (what is the MAC address of 150.100.76.22?)

Every host receives the request, but only H3 reply with its physical address

ARP response (my MAC address is 08:00:5a:3b:94)

Example of ARP



Internet Control Message Protocol (ICMP)

- RFC 792; Encapsulated in IP packet (protocol type = 1)
- Handles error and control messages
- If router cannot deliver or forward a packet, it sends an ICMP “host unreachable” message to the source
- If router receives packet that should have been sent to another router, it sends an ICMP “redirect” message to the sender; Sender modifies its routing table
- ICMP “router discovery” messages allow host to learn about routers in its network and to initialize and update its routing tables
- ICMP echo request and reply facilitate diagnostic and used in “ping”

UDP

- Best effort datagram service
- Multiplexing enables sharing of IP datagram service
- Simple transmitter & receiver
  - Connectionless: no handshaking & no connection state
  - Low header overhead
  - No flow control, no error control, no congestion control
  - UDP datagrams can be lost or out-of-order
- Applications
  - multimedia (e.g. RTP)
  - network services (e.g. DNS, RIP, SNMP)

UDP Datagram

0	16 31
Source Port	Destination Port
UDP Length	UDP Checksum
Data	

- Source and destination port numbers
  - Client ports are short-lived
  - Server ports are well-known
  - Max number is 65,535
- UDP length
  - Total number of bytes in datagram (including header)
  - 8 bytes ≤ length ≤ 65,535
- UDP Checksum
  - Optionally detects errors in UDP datagram

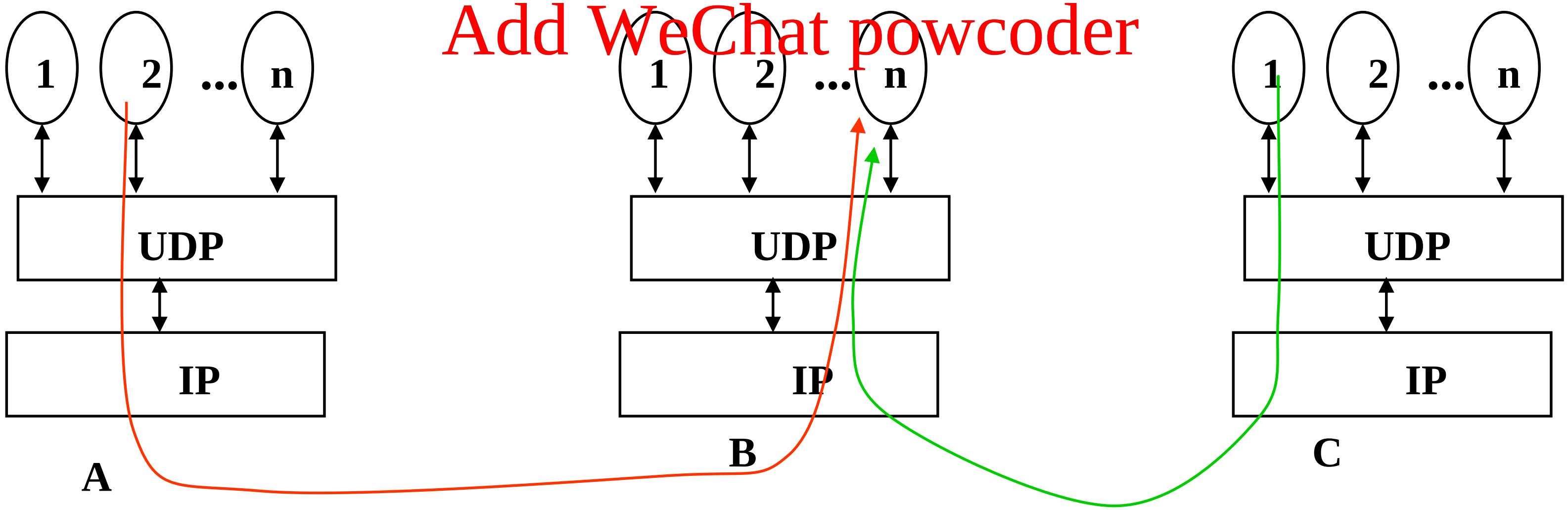
0-255
- Well-known ports

256-1023
- Less well-known ports

1024-65536
- Client ports

UDP Multiplexing

- All UDP datagrams arriving to IP address B and destination port number *n* are delivered to the same process
- Source port number is not used in multiplexing

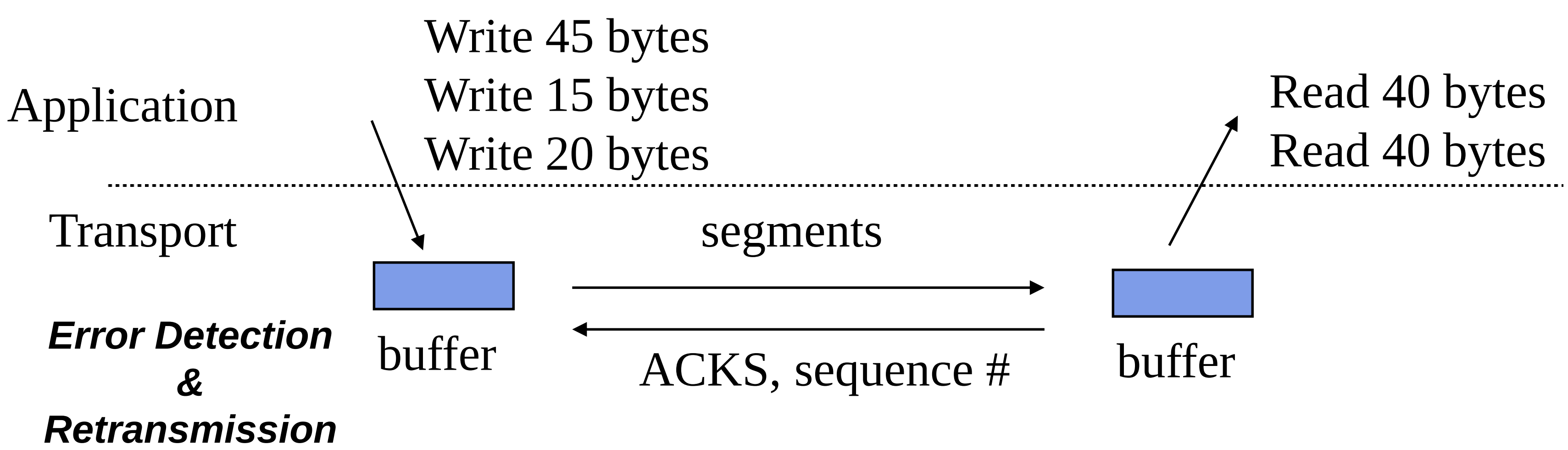


TCP

- Reliable byte-stream service
- More complex transmitter & receiver
  - Connection-oriented: full-duplex unicast connection between client & server processes
  - Connection setup, connection state, connection release
  - Higher header overhead
  - Error control, flow control, and congestion control
  - Higher delay than UDP
- Most applications use TCP
  - HTTP, SMTP, FTP, TELNET, POP3, …

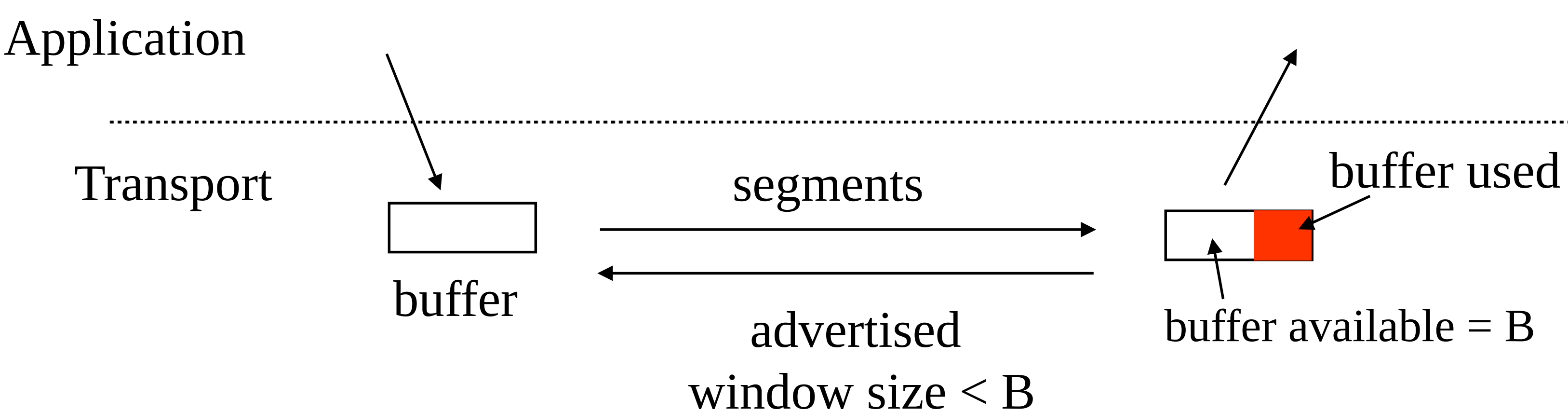
Reliable Byte-Stream Service

- Stream Data Transfer
  - transfers a contiguous stream of bytes across the network, with no indication of boundaries
  - groups bytes into segments
  - transmits segments as convenient (Push function defined)
- Reliability
  - error control mechanism to deal with IP transfer impairments



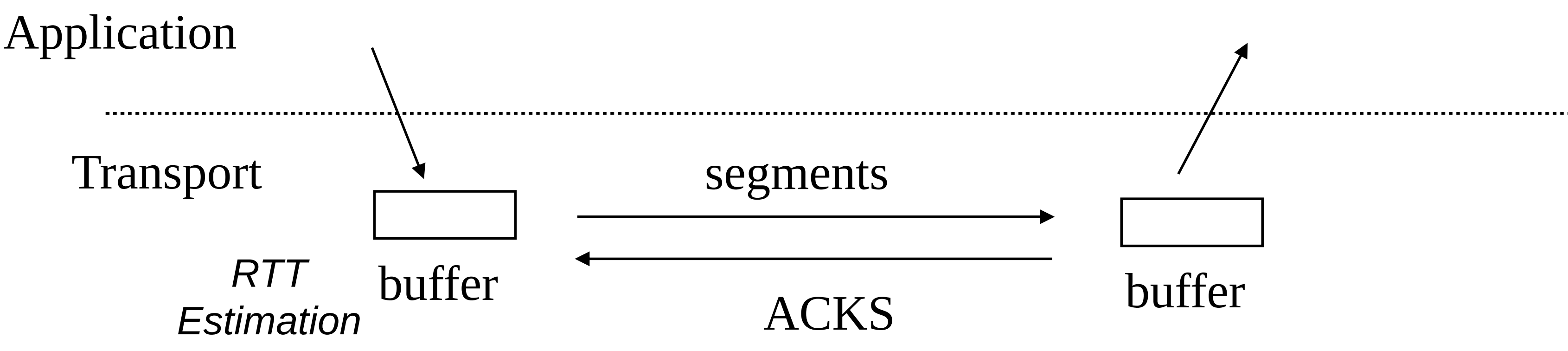
Flow Control

- Buffer limitations & speed mismatch can result in loss of data that arrives at destination
- Receiver controls rate at which sender transmits to prevent buffer overflow



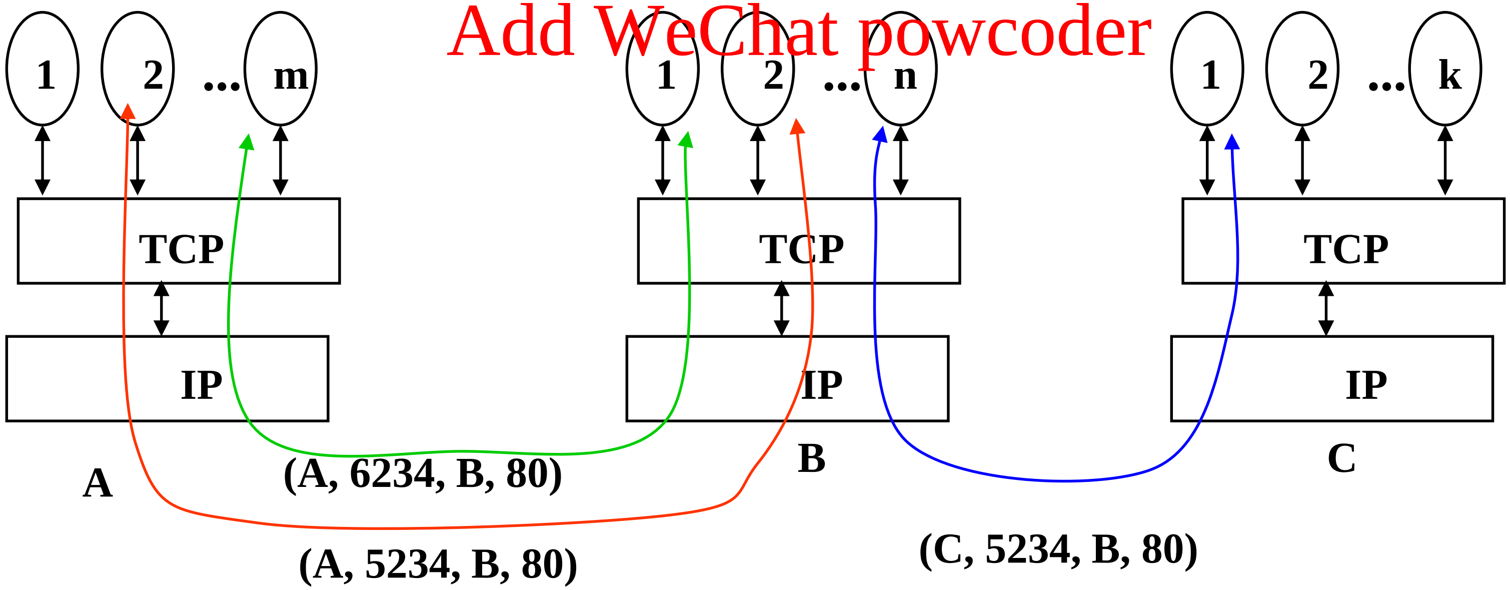
Congestion Control

- Available bandwidth to destination varies with activity of other users
- Transmitter dynamically adjusts transmission rate according to network congestion as indicated by RTT (round trip time) & ACKs
- Elastic utilization of network bandwidth

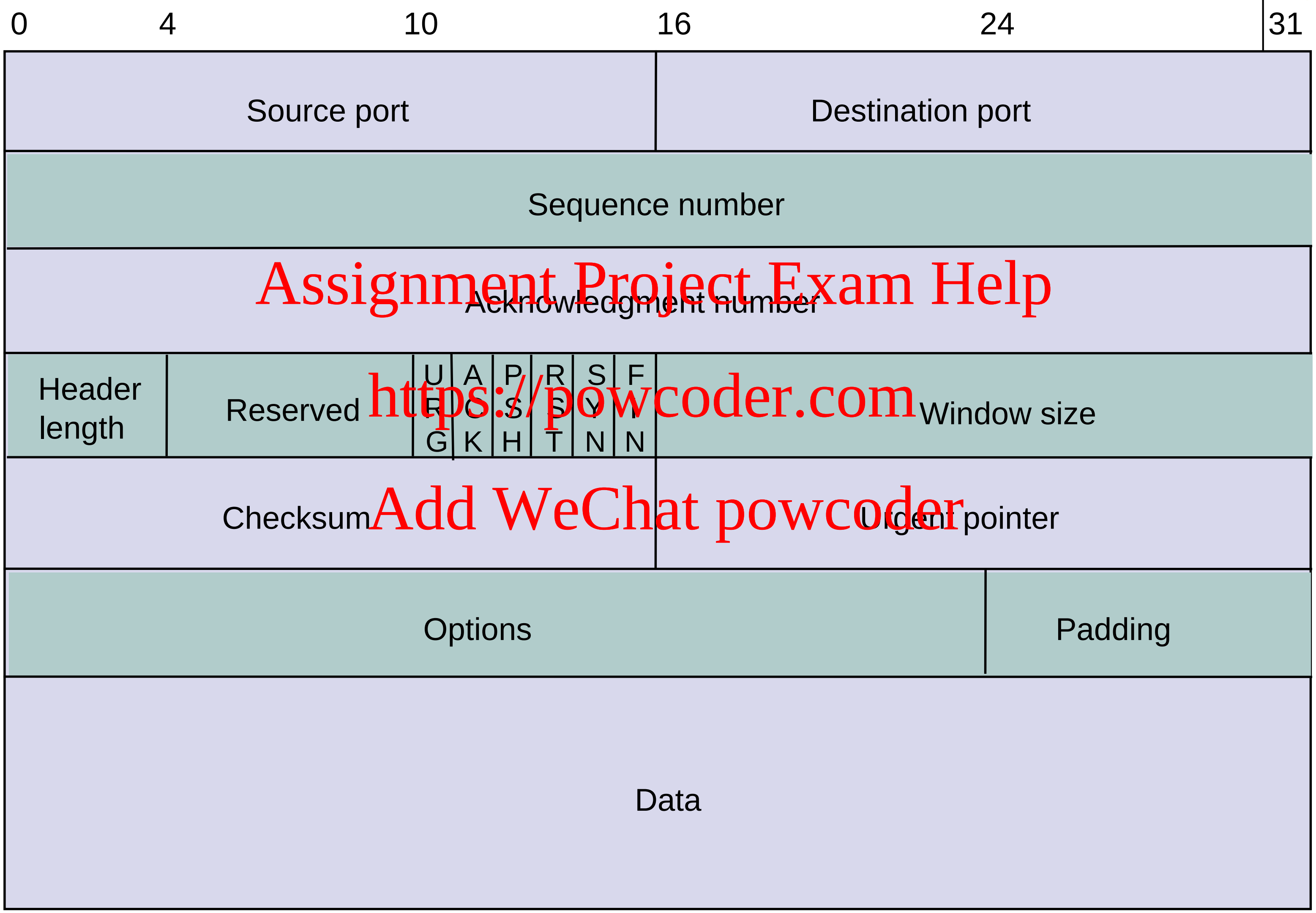


TCP Multiplexing

- A *TCP connection* is specified by a *4-tuple*
  - (source IP address, source port, destination IP address, destination port)
- TCP allows multiplexing of multiple connections between end systems to support multiple applications simultaneously
- Arriving segment directed according to connection 4-tuple



TCP Segment Format



- Each TCP segment has header of 20 or more bytes + 0 or more bytes of data

TCP Header

Port Numbers

- A socket identifies a connection endpoint
  - IP address + port
- A connection specified by a *socket pair*
- Well-known ports
  - FTP 20
  - Telnet 23
  - DNS 53
  - HTTP 80

Sequence Number

- Byte count
- First byte in segment
- 32 bits long
- $0 \leq SN \leq 2^{32}-1$
- Initial sequence number selected during connection setup

TCP Header

Acknowledgement Number

- SN of next byte expected by receiver
- Acknowledges that all prior bytes in stream have been received correctly
- Valid if ACK flag is set

Header length

- 4 bits
- Length of header in multiples of 32-bit words
- Minimum header length is 20 bytes
- Maximum header length is 60 bytes

TCP Header

Reserved

- 6 bits

Control

- 6 bits
- URG: urgent pointer flag
  - Urgent message end = SN + **urgent pointer**
- ACK: ACK packet flag
- PSH: override TCP buffering
- RST: reset connection
  - Upon receipt of RST, connection is terminated and application layer notified
- SYN: establish connection
- FIN: close connection

TCP Header

Window Size

- 16 bits to advertise window size
- Used for flow control
- Sender will accept bytes with SN from ACK to ACK + window
- Maximum window size is 65535 bytes

TCP Checksum

- Internet checksum method

TCP Header

Options

- Variable length
- NOP (No Operation) option is used to pad TCP header to multiple of 32 bits
- Time stamp option is used for round trip measurements

Options

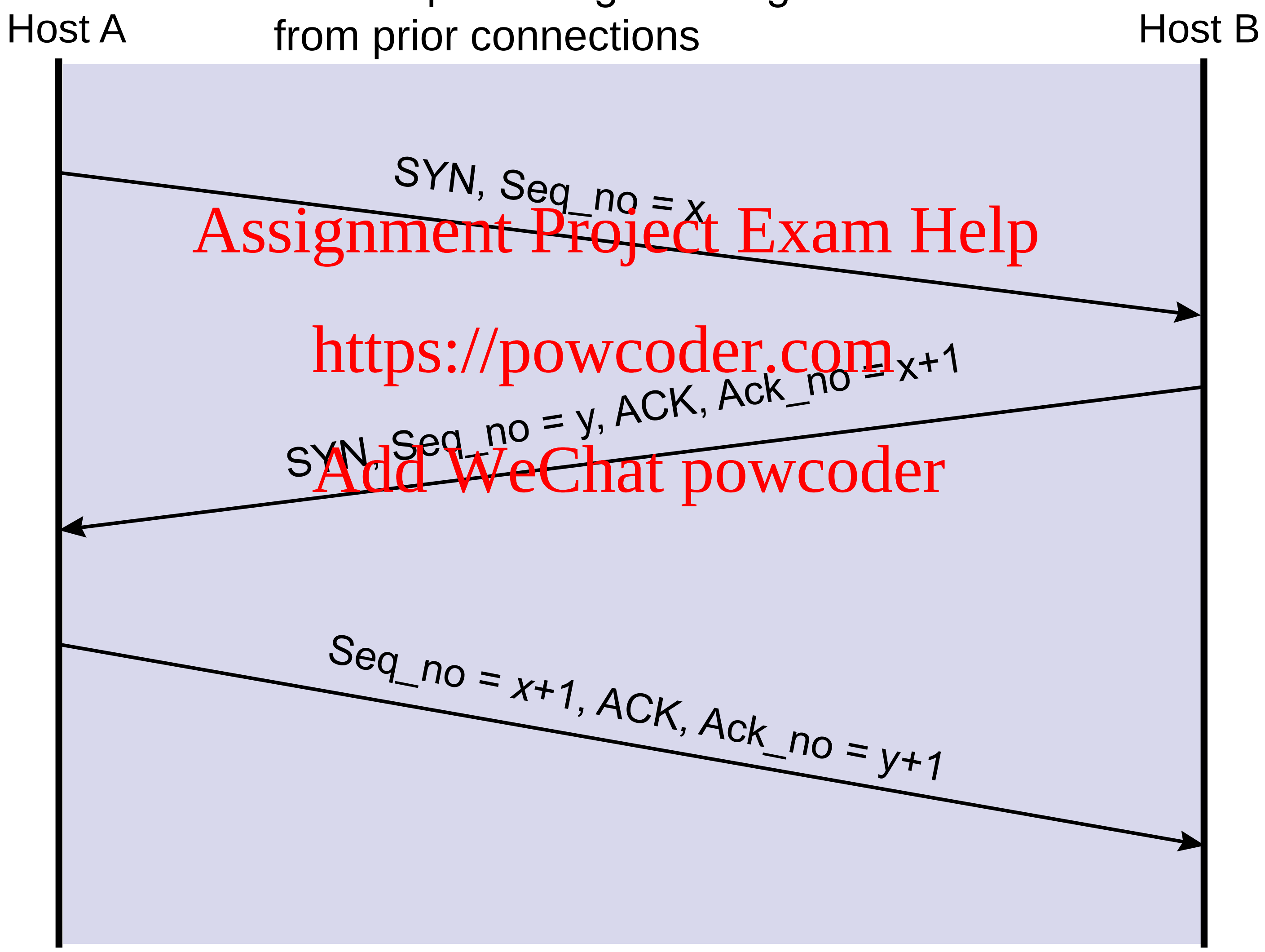
- Maximum Segment Size (MSS) option specifies largest segment a receiver wants to receive
- Window Scale option increases TCP window from 16 to 32 bits

Initial Sequence Number

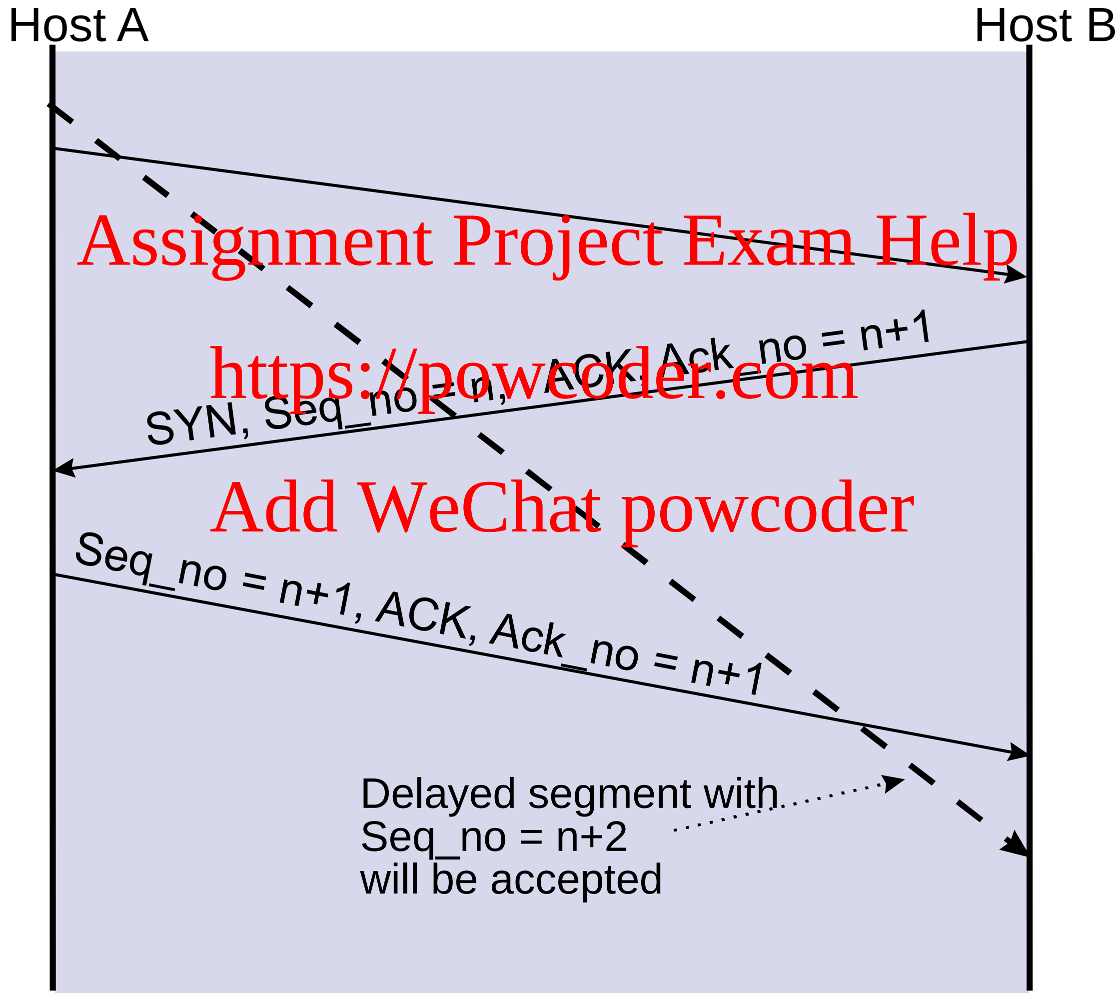
- Select initial sequence numbers (ISN) to protect against segments from prior connections (that may circulate in the network and arrive at a much later time)
- Select ISN to avoid overlap with sequence numbers of prior connections
- Use local clock to select ISN sequence number
- Time for clock to go through a full cycle should be greater than the maximum lifetime of a segment (MSL); Typically MSL=120 seconds
- High bandwidth connections pose a problem
- 32bit SN wraps around after $2^{32} = 4.29 \times 10^9$ bytes = 34.3×10^9 bits have been sent
  - At 1 Gbps, sequence number wraparound in 34.3 seconds.

TCP Connection Establishment

- "Three-way Handshake"
- ISN's protect against segments from prior connections



If host always uses the same ISN

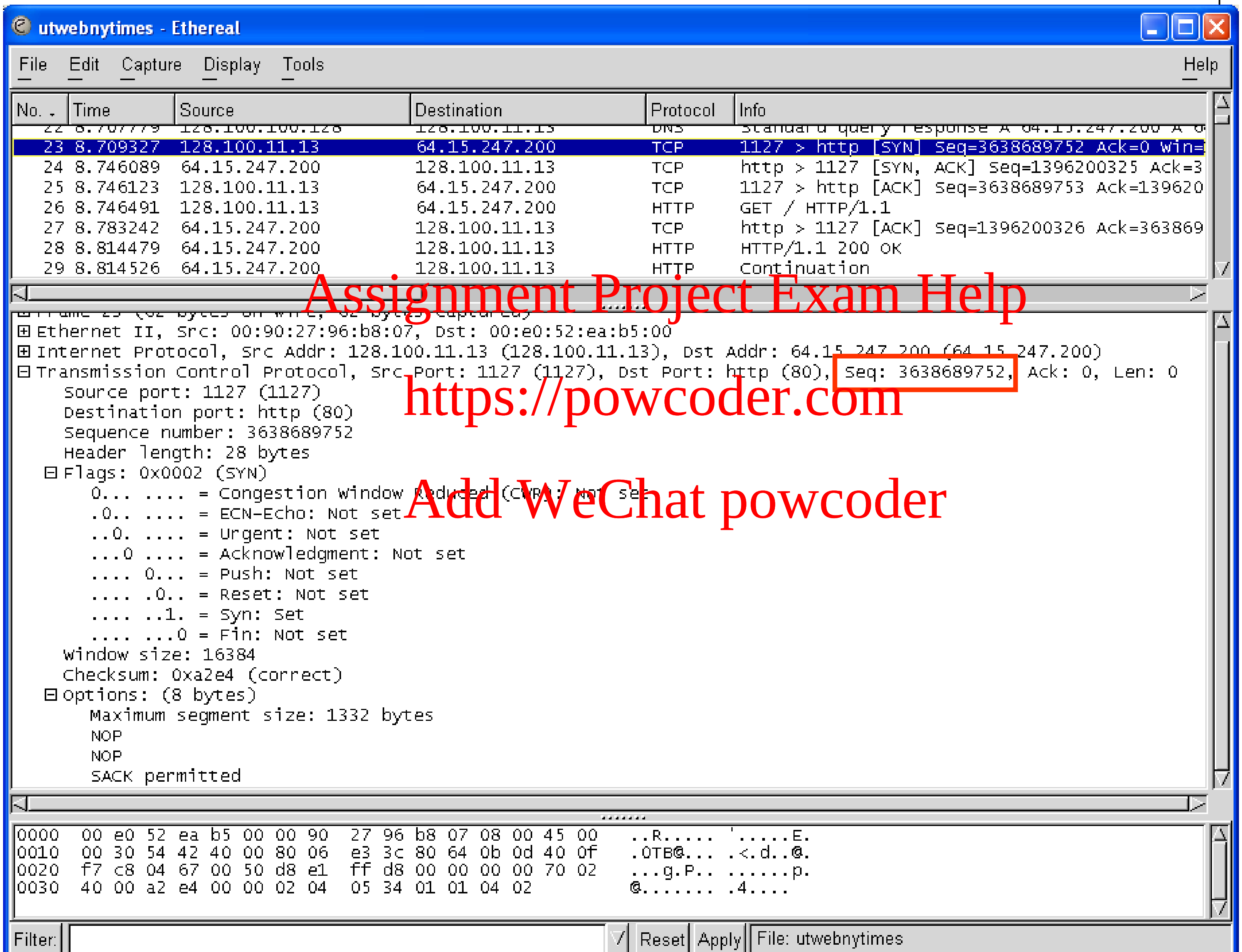


Maximum Segment Size

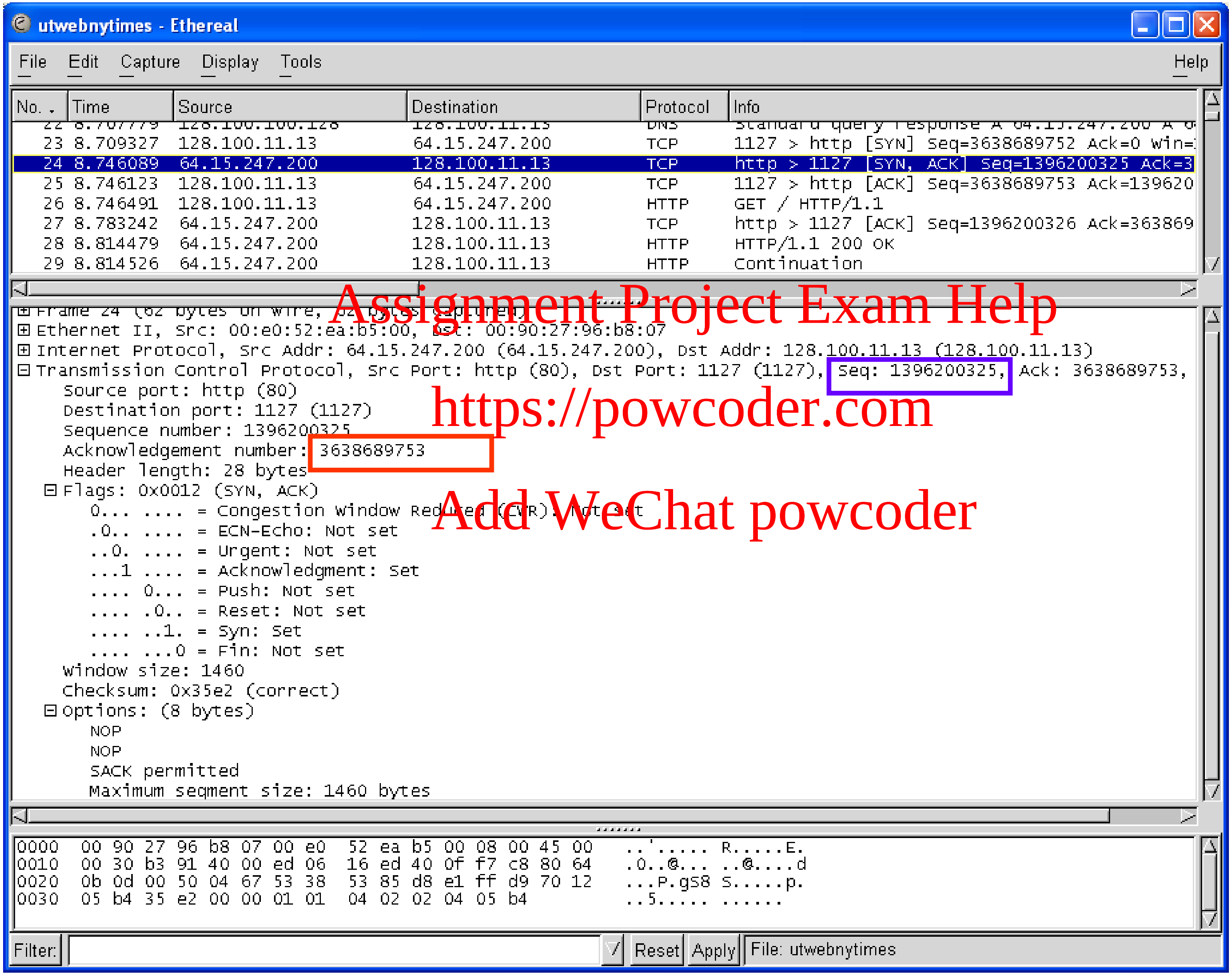
- Maximum Segment Size
  - largest block of data that TCP sends to other end
- Each end can announce its MSS during connection establishment

- Default is 576 bytes including 20 bytes for IP header and 20 bytes for TCP header
- Ethernet implies MSS of 1460 bytes
- IEEE 802.3 implies 1452

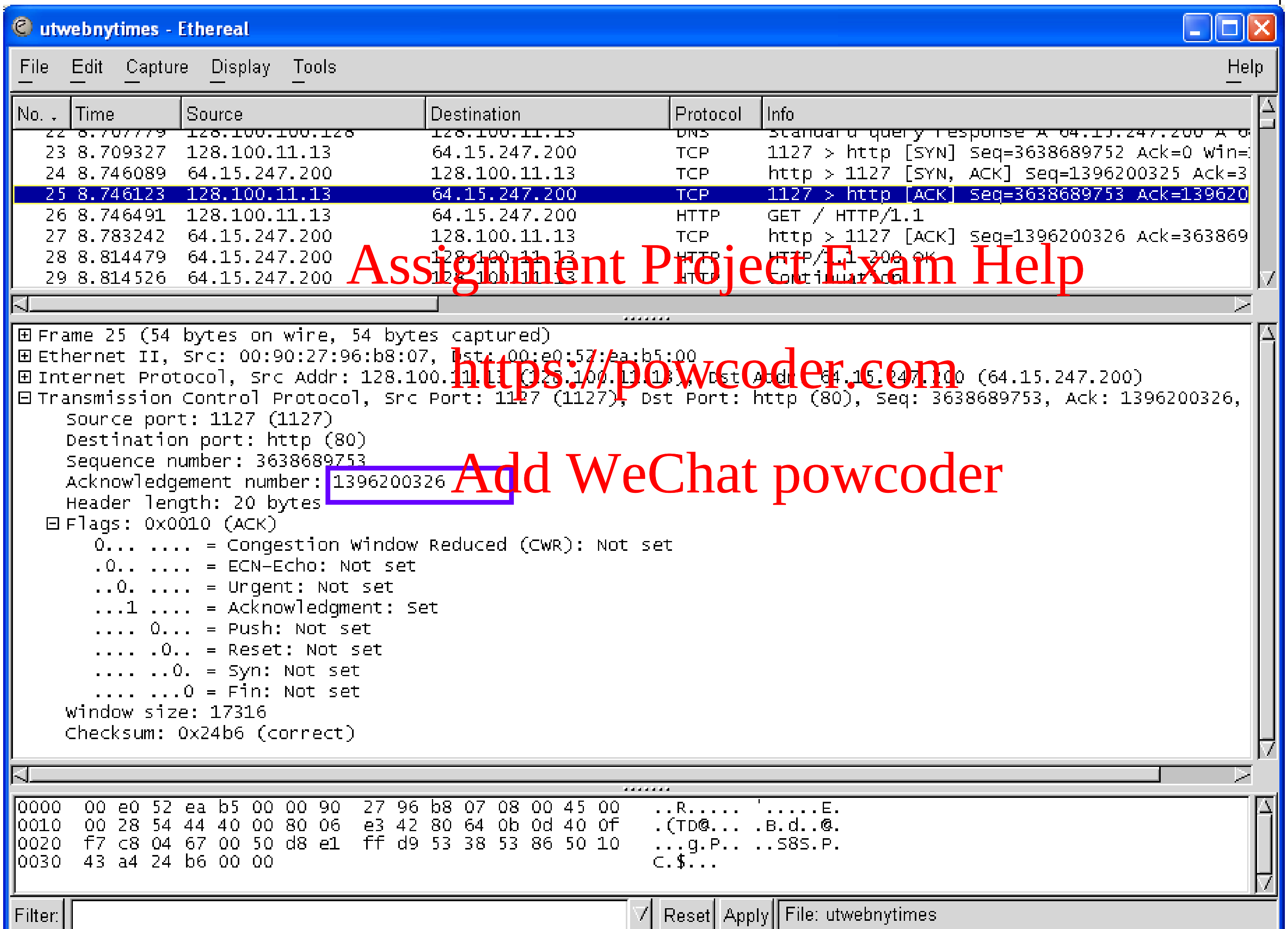
Near End: Connection Request



Far End: Ack and Request

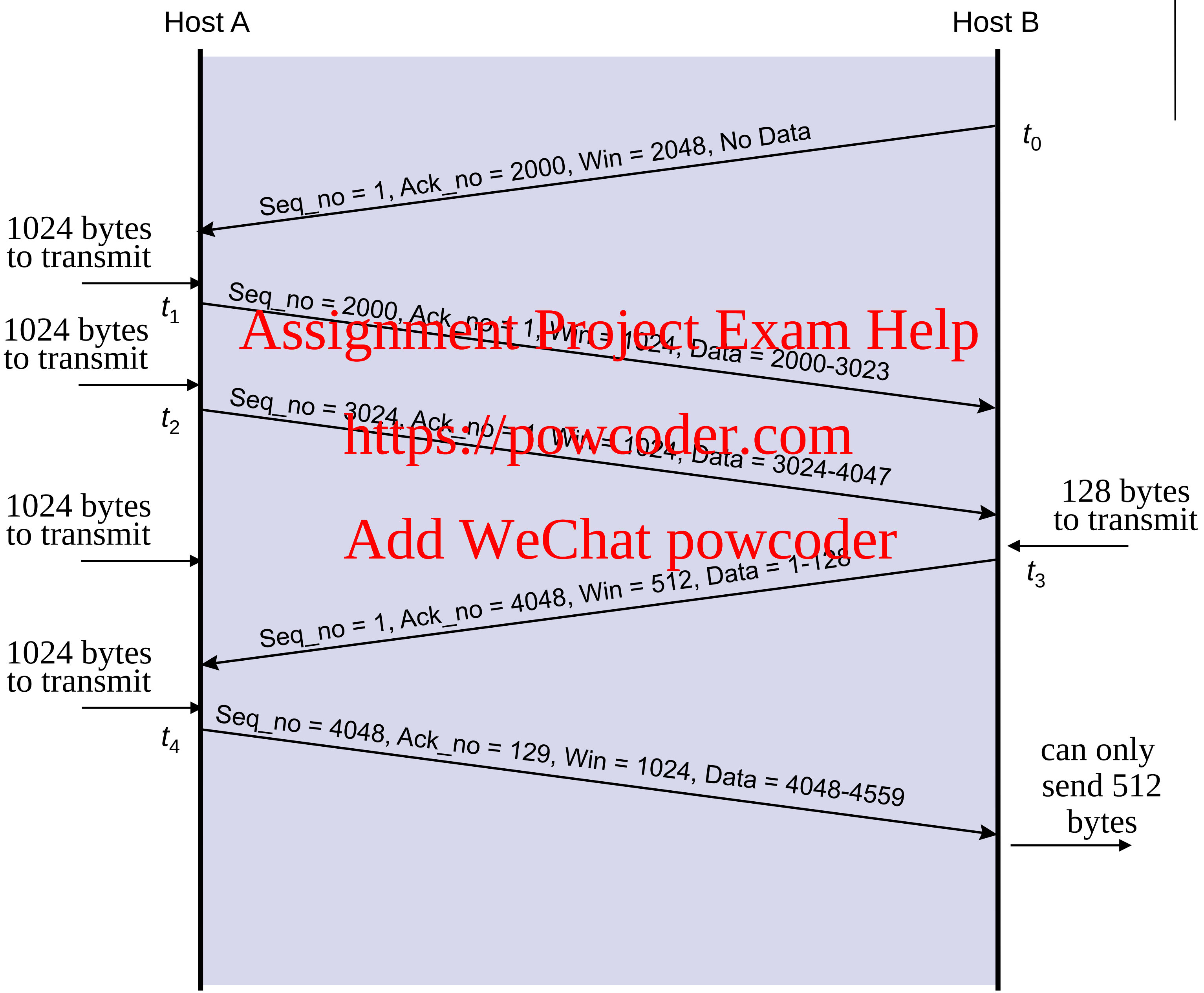


Near End: Ack



utwebnytimes - Ethereal
File Edit Capture Display Tools Help
No. . Time Source Destination Protocol Info
23 8.709327 128.100.11.13 64.15.247.200 TCP 1127 > http [SYN] Seq=3638689752 Ack=0 Win=
24 8.746089 64.15.247.200 128.100.11.13 TCP http > 1127 [SYN, ACK] Seq=1396200325 Ack=3
25 8.746123 128.100.11.13 64.15.247.200 TCP 1127 > http [ACK] Seq=3638689753 Ack=139620
26 8.746491 128.100.11.13 64.15.247.200 HTTP GET / HTTP/1.1
27 8.783242 64.15.247.200 128.100.11.13 TCP http > 1127 [ACK] Seq=1396200326 Ack=363869
28 8.814479 64.15.247.200
29 8.814526 64.15.247.200
Frame 25 (54 bytes on wire, 54 bytes captured)
Ethernet II, Src: 00:90:27:96:b8:07, Dst: 00:e0:52:ea:b5:00
Internet Protocol, Src Addr: 128.100.11.13 (128.100.11.13), Dst Addr: 64.15.247.200 (64.15.247.200)
Transmission Control Protocol, Src Port: 1127 (1127), Dst Port: http (80), Seq: 3638689753, Ack: 1396200326,
Source port: 1127 (1127)
Destination port: http (80)
Sequence number: 3638689753
Acknowledgement number: 1396200326
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
....0 = Fin: Not set
Window size: 17316
Checksum: 0x24b6 (correct)
0000 00 e0 52 ea b5 00 00 90 27 96 b8 07 08 00 45 00 ..R..... '.....E.
0010 00 28 54 44 40 00 80 06 e3 42 80 64 0b 0d 40 0f .(TD@... .B.d..@.
0020 f7 c8 04 67 00 50 d8 e1 ff d9 53 38 53 86 50 10 ...g.P.. ..S8S.P.
0030 43 a4 24 b6 00 00 C.$...
Filter: Reset Apply File: utwebnytimes

TCP Window Flow Control

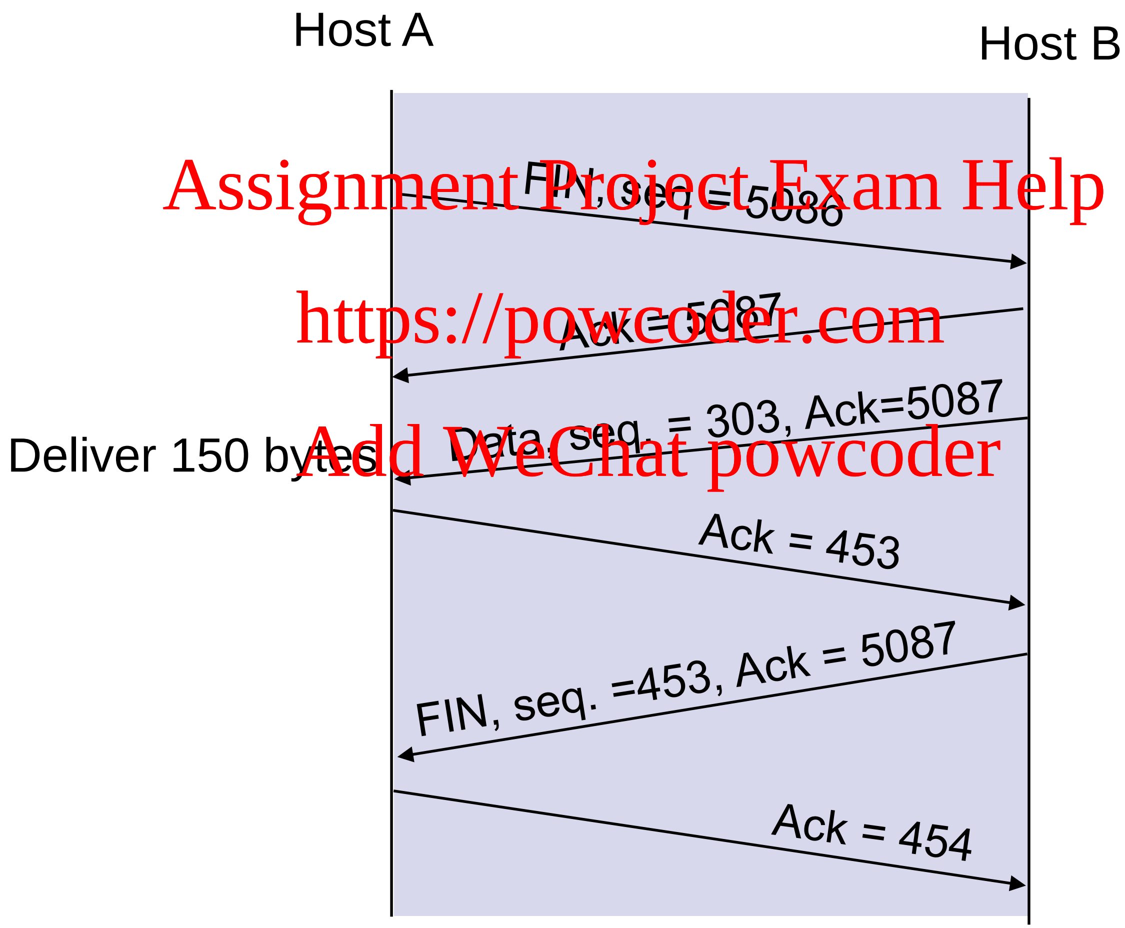


Delay-BW Product & Advertised Window Size

- Suppose RTT=100 ms, R=2.4 Gbps
  - # bits in pipe = 3 Mbytes
- If single TCP process occupies pipe, then required advertised window size is
  - RTT x Bit rate = 3 Mbytes
  - Normal maximum window size is 65535 bytes
- Solution: Window Scale Option
  - Window size up to 65535 x 2^{14} = 1 Gbyte allowed
  - Requested in SYN segment

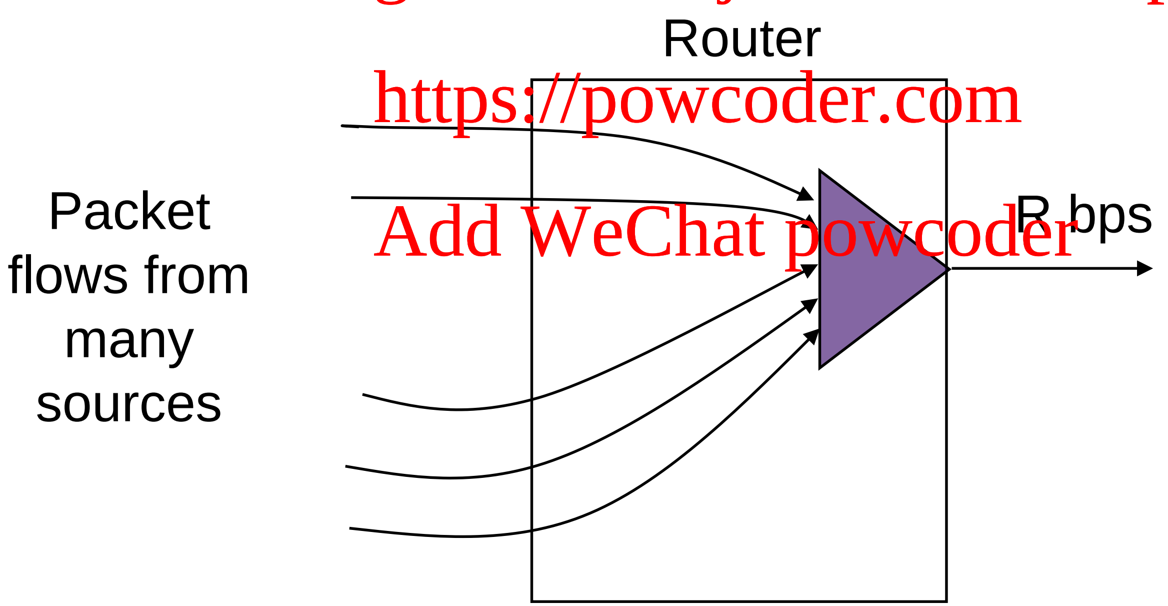
TCP Connection Closing

"Graceful Close"



TCP Congestion Control

- *Advertised window* size is used to ensure that receiver's buffer will not overflow
- However, buffers at intermediate routers between source and destination may overflow



- Congestion occurs when total arrival rate from all packet flows exceeds R over a sustained period of time
- Buffers at multiplexer will fill and packets will be lost