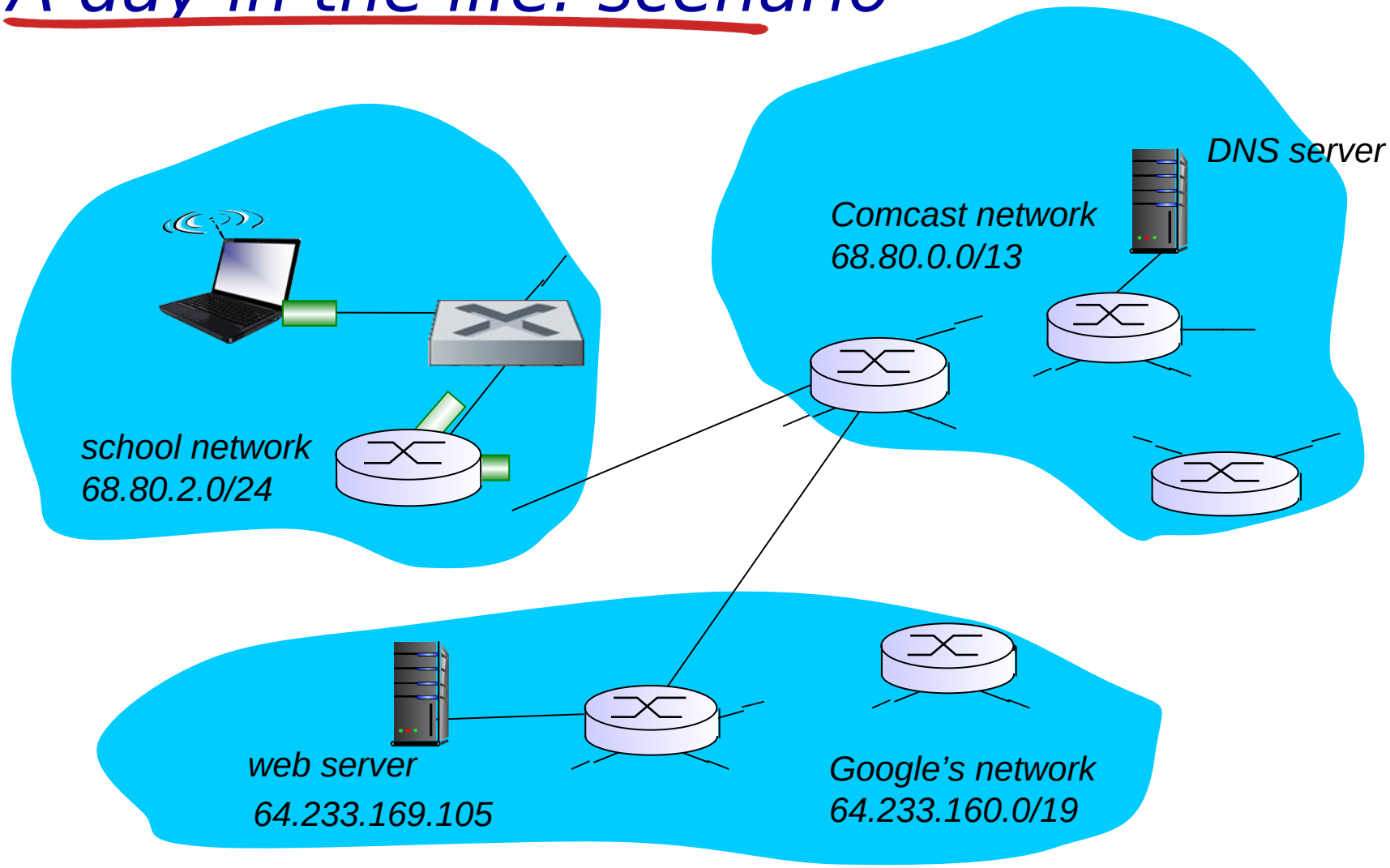
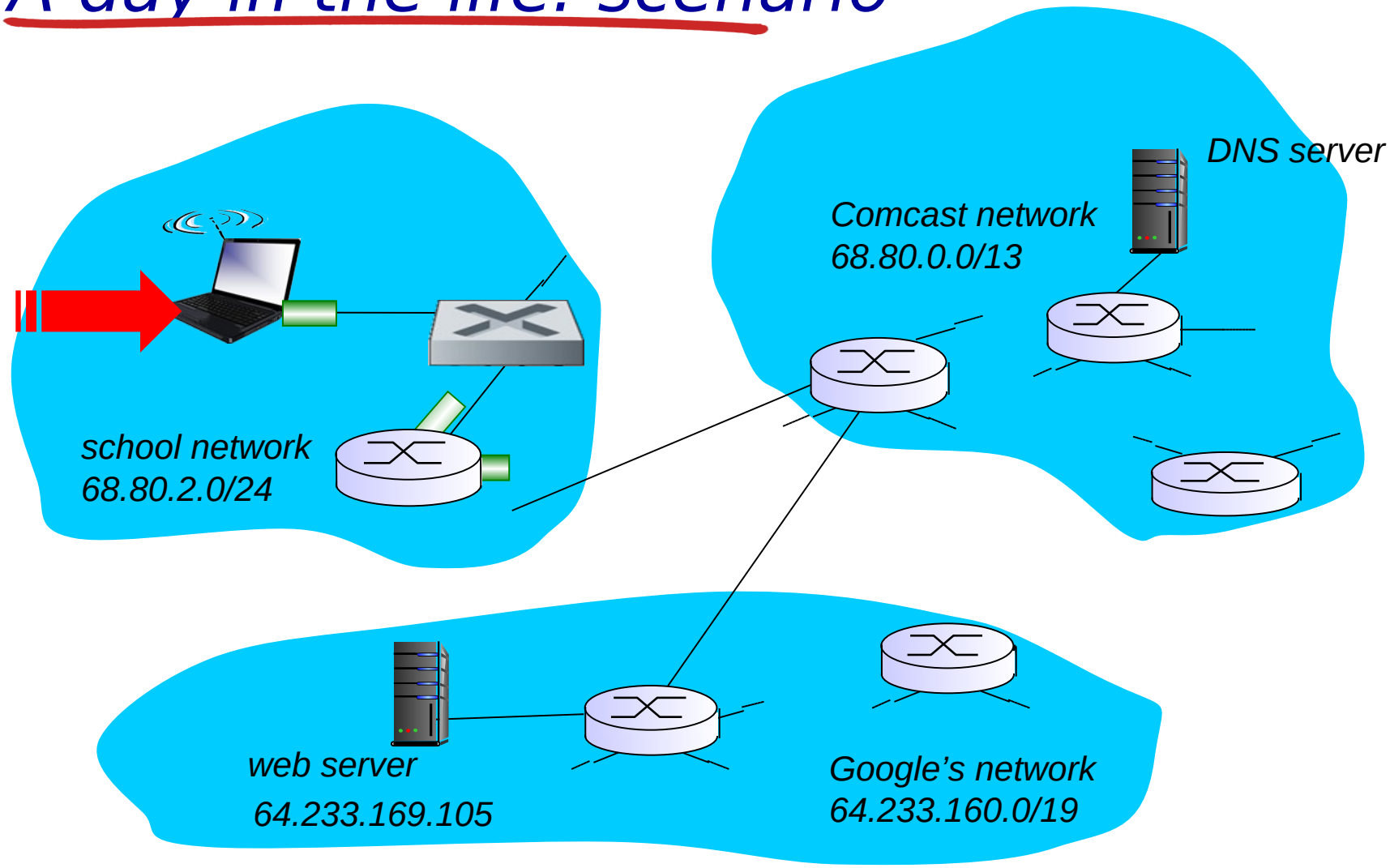
Synthesis: a day in the life of a web request

- *journey down protocol stack complete!*
  - *application, transport, network, link*
- *putting-it-all-together: synthesis!*
  - *goal: identify, review, understand protocols (at all layers) involved in seemingly simple scenario: requesting www page*
  - *scenario: student attaches laptop to campus network, requests/receives www.google.com*

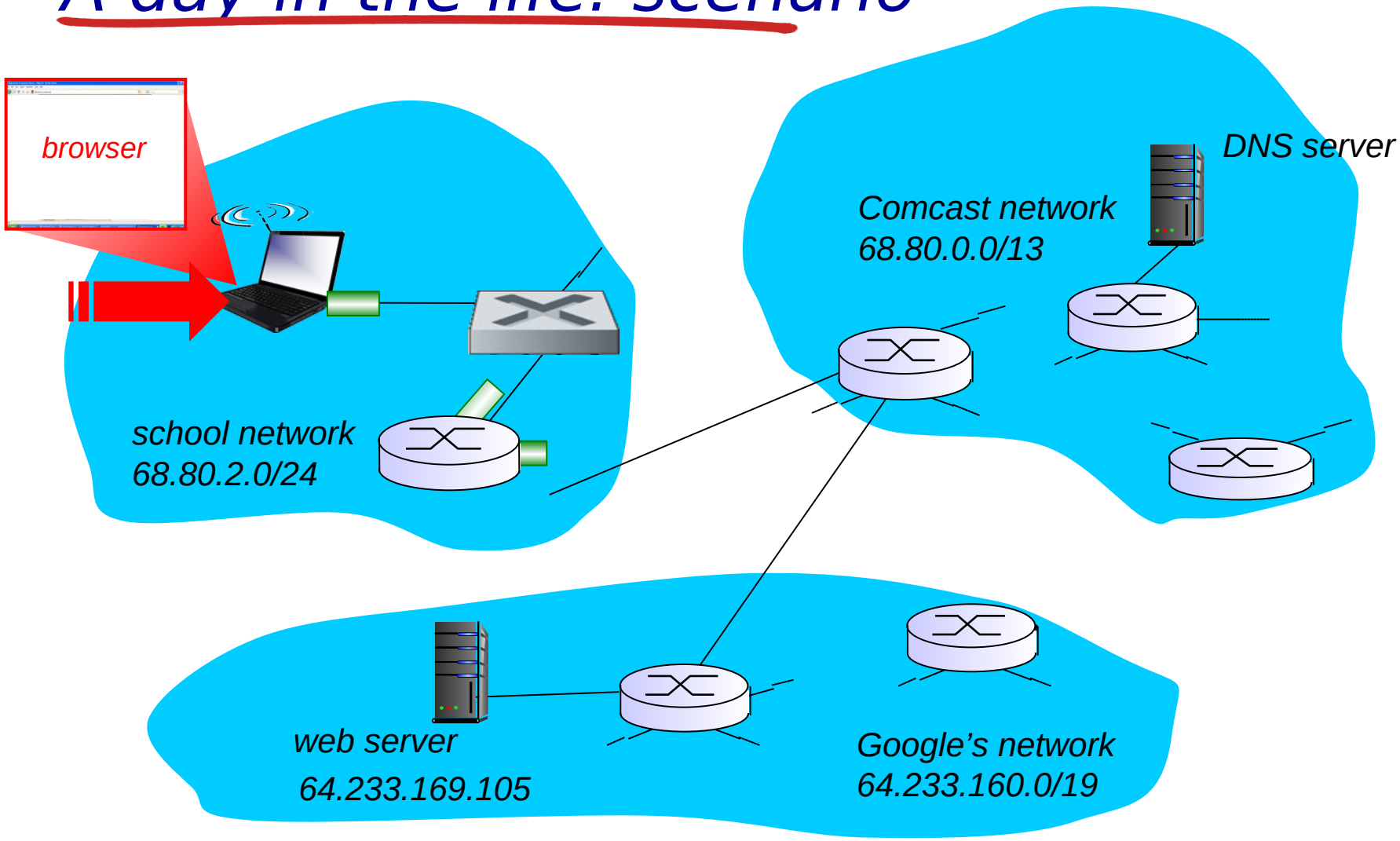
A day in the life: scenario



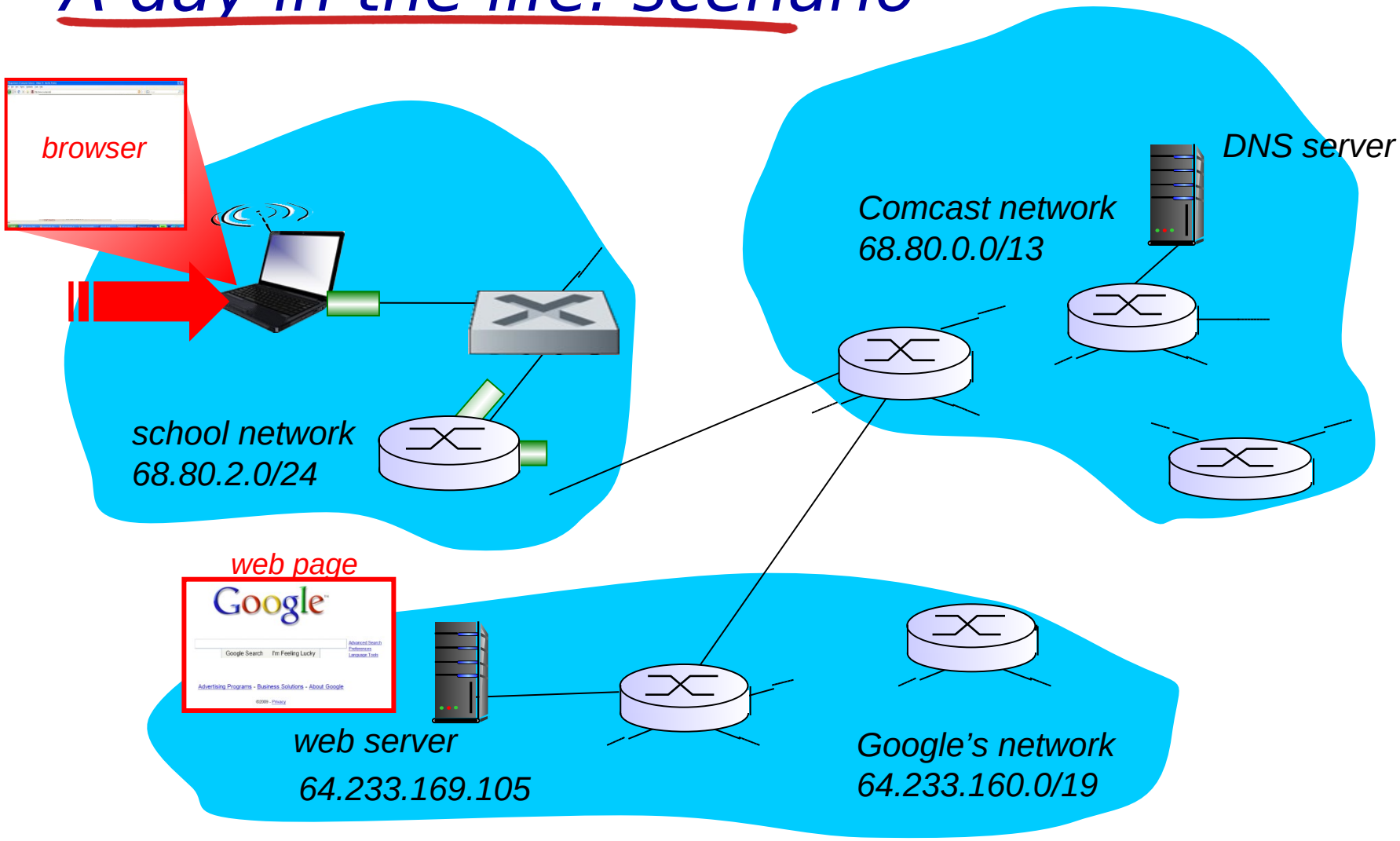
A day in the life: scenario



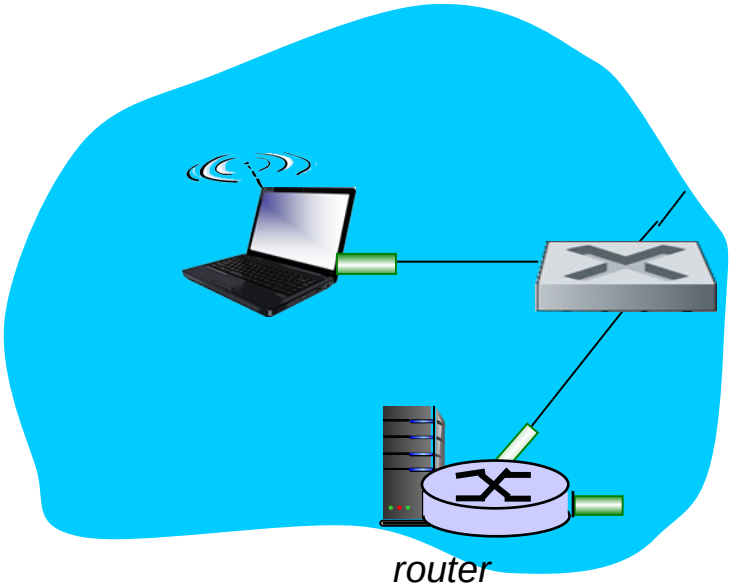
A day in the life: scenario



A day in the life: scenario

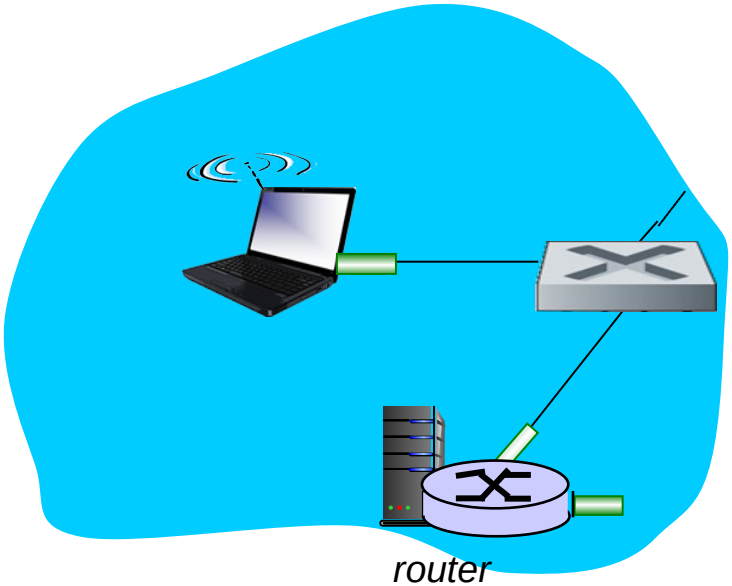


A day in the life... connecting to the Internet



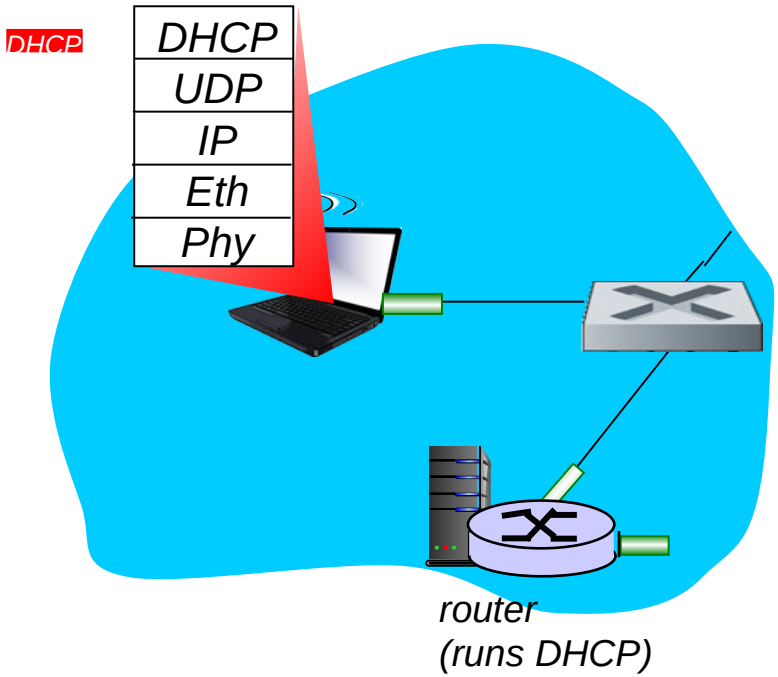
router
(runs DHCP)

A day in the life... connecting to the Internet



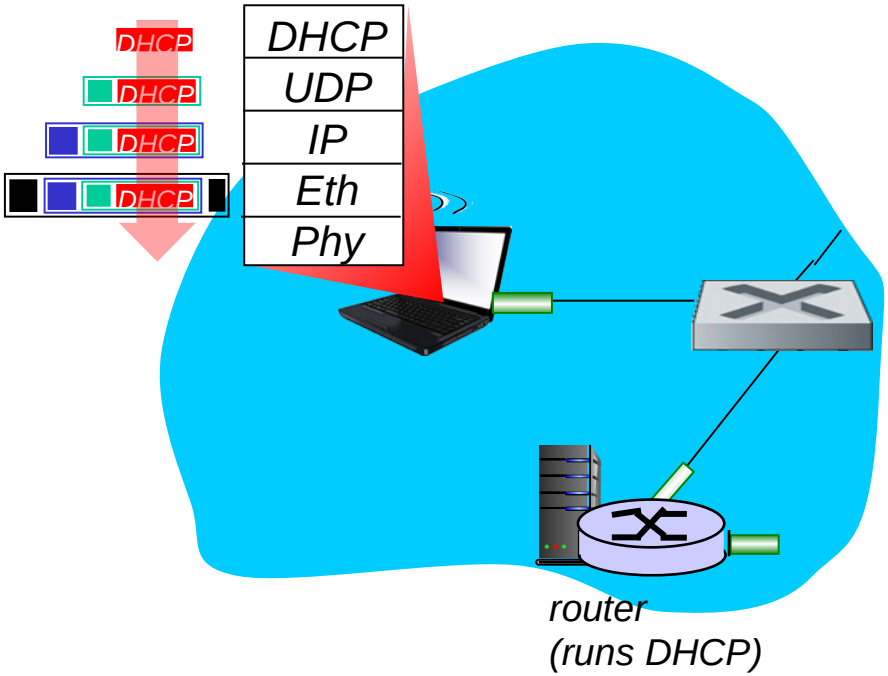
router
(runs DHCP)

A day in the life... connecting to the Internet



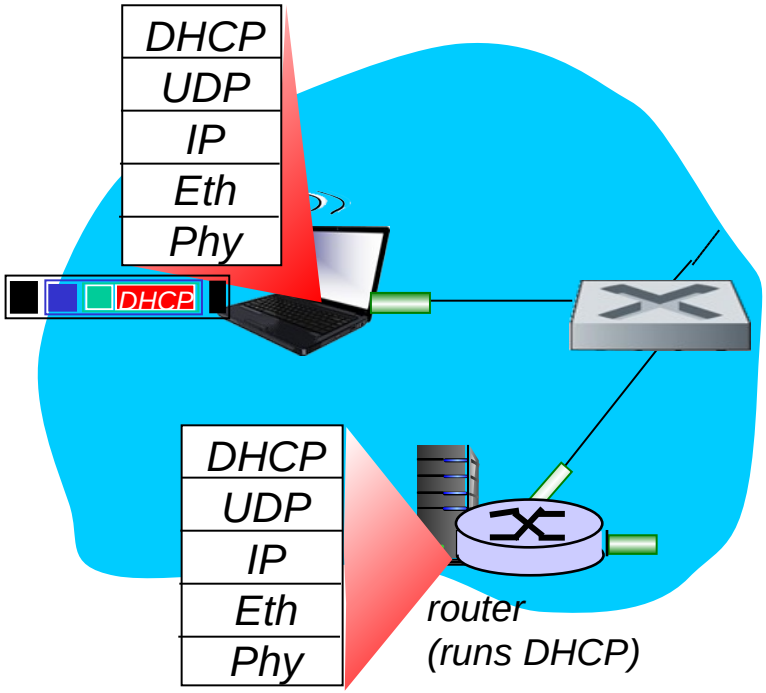
- *connecting laptop needs to get its own IP address, addr of first-hop router, addr of DNS server: use DHCP*

A day in the life... connecting to the Internet



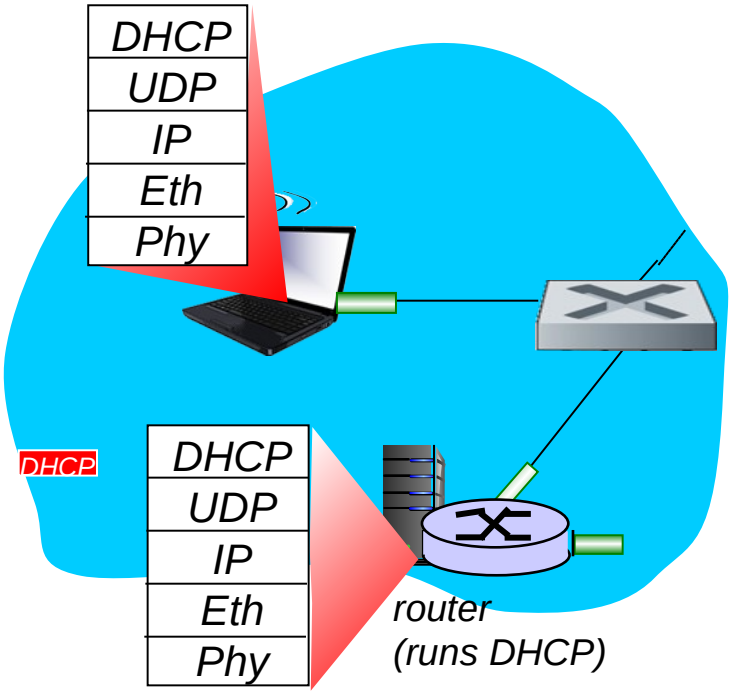
- *connecting laptop needs to get its own IP address, addr of first-hop router, addr of DNS server: use DHCP*
- *DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in 802.3 Ethernet*

A day in the life... connecting to the Internet



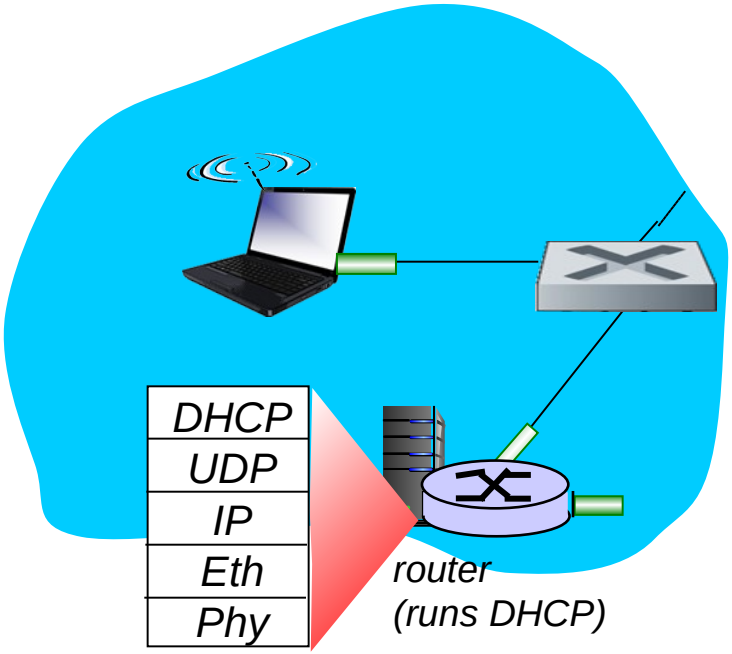
- *connecting laptop needs to get its own IP address, addr of first-hop router, addr of DNS server: use DHCP*
- *DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in 802.3 Ethernet*
- *Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server*

A day in the life... connecting to the Internet

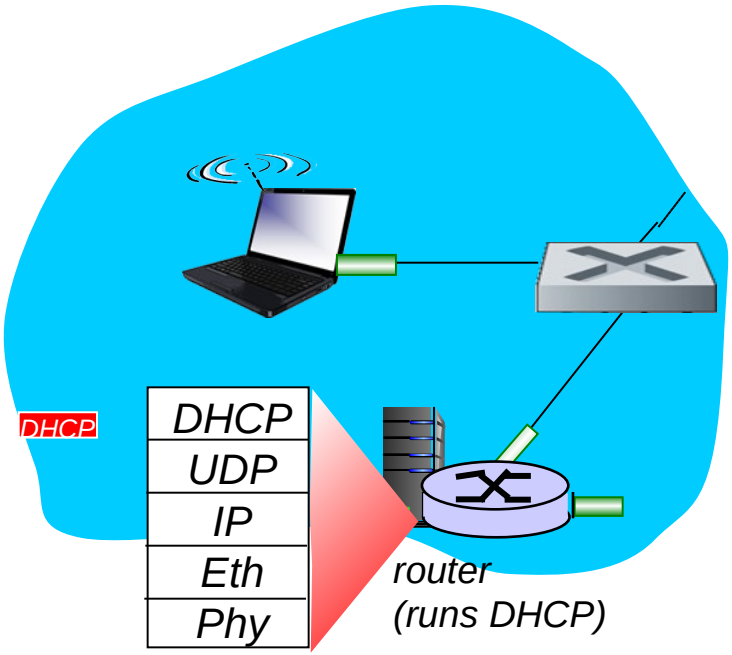


- *connecting laptop needs to get its own IP address, addr of first-hop router, addr of DNS server: use DHCP*
- *DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in 802.3 Ethernet*
- *Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server*
- *Ethernet demuxed to IP demuxed, UDP demuxed to DHCP*

A day in the life... connecting to the Internet

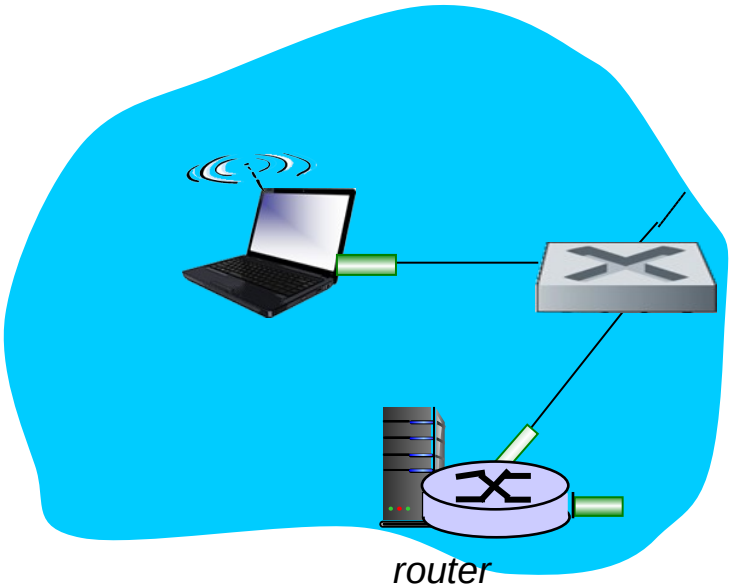


A day in the life... connecting to the Internet



- *DHCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server*

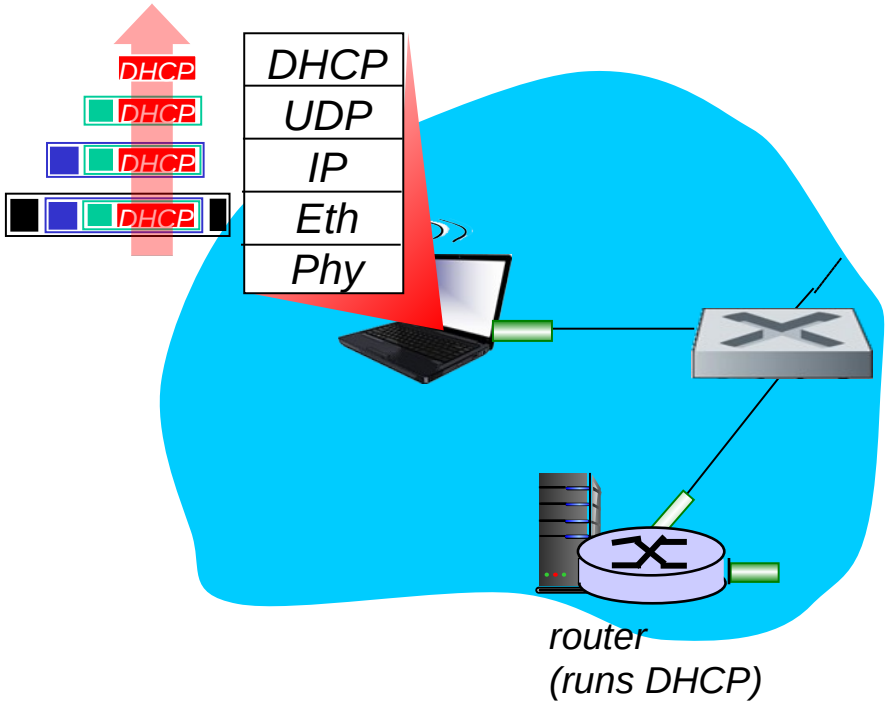
A day in the life... connecting to the Internet



router
(runs DHCP)

- DHCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server
- encapsulation at DHCP server, frame forwarded (switch learning) through LAN, demultiplexing at client

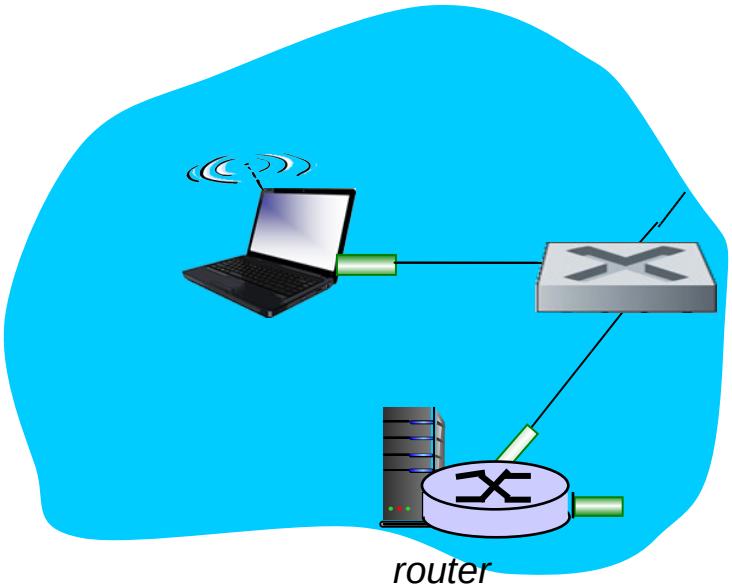
A day in the life... connecting to the Internet



- *DHCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server*
- *encapsulation at DHCP server, frame forwarded (switch learning) through LAN, demultiplexing at client*
- *DHCP client receives DHCP ACK reply*

Client now has IP address, knows name & addr of DNS server, IP address of its first-hop router

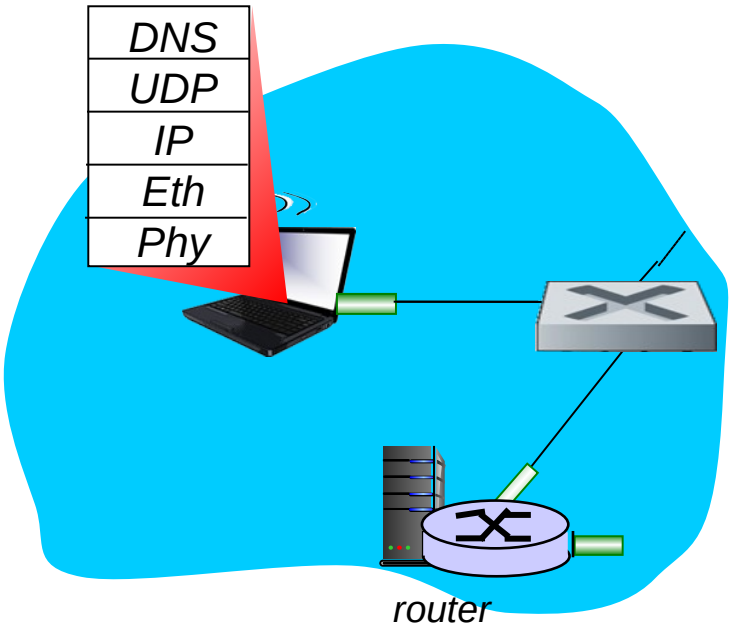
A day in the life... ARP (before DNS, before HTTP)



router
(runs DHCP)

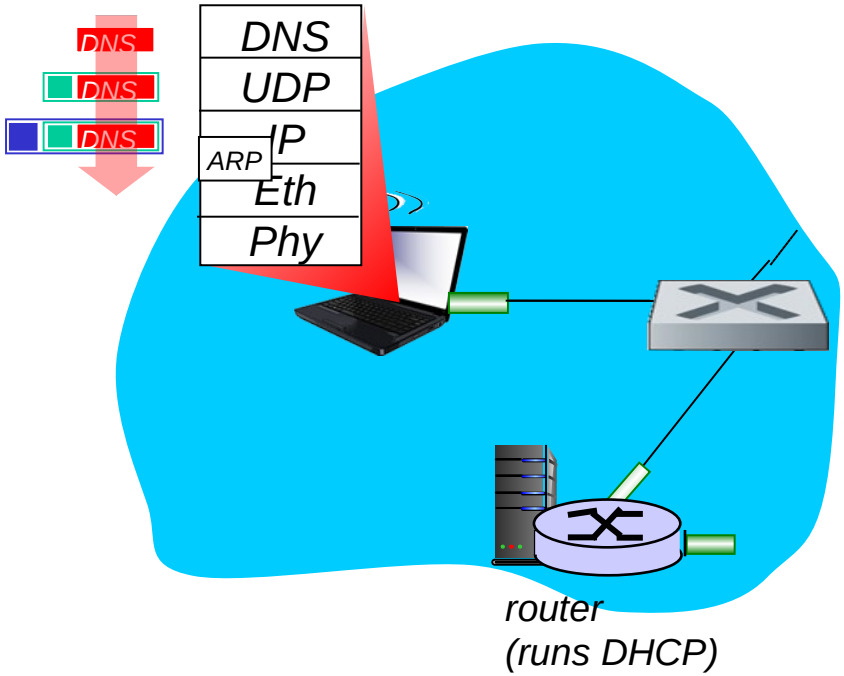
- before sending HTTP request, need IP address of www.google.com: DNS

A day in the life... ARP (before DNS, before HTTP)



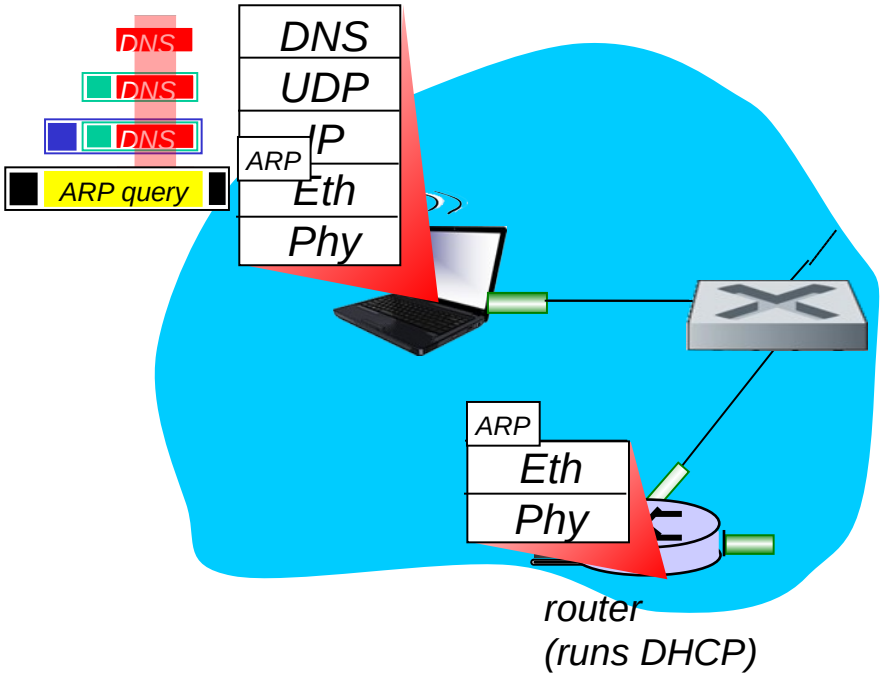
- *before sending HTTP request, need IP address of www.google.com: DNS*

A day in the life... ARP (before DNS, before HTTP)



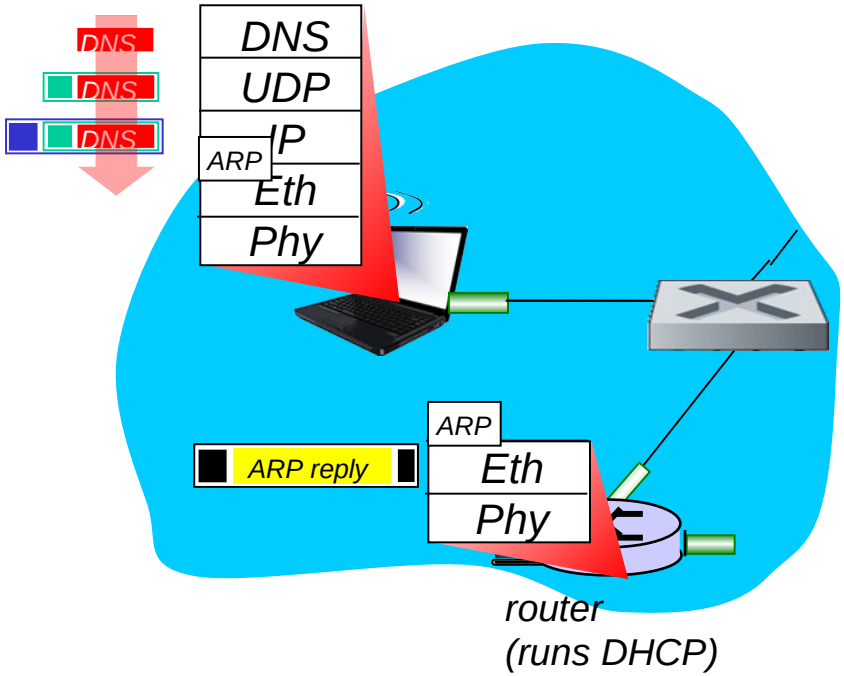
- *before sending HTTP request, need IP address of www.google.com: DNS*
- *DNS query created, encapsulated in UDP, encapsulated in IP, encapsulated in Eth. To send frame to router, need MAC address of router interface: ARP*

A day in the life... ARP (before DNS, before HTTP)



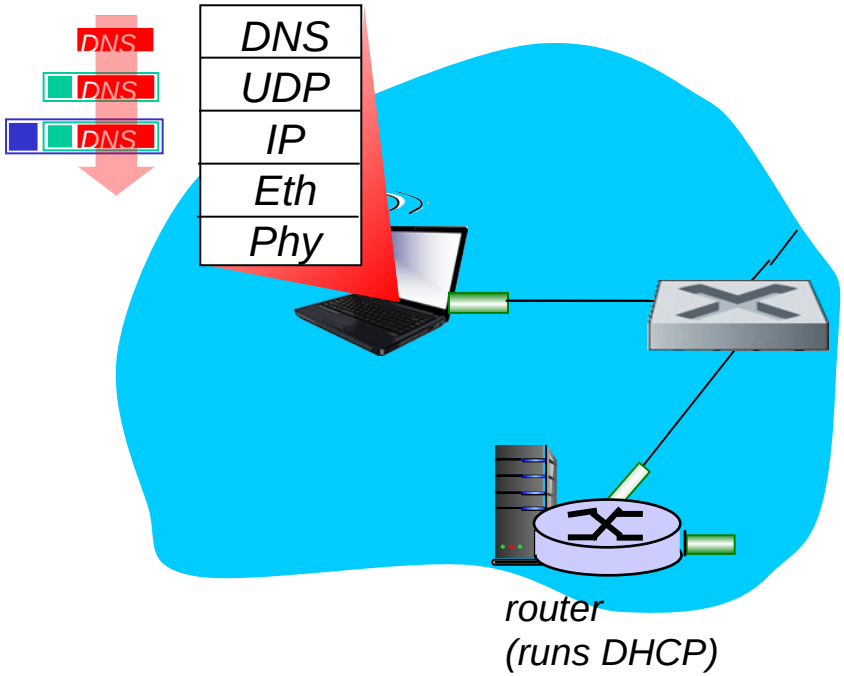
- *before sending HTTP request, need IP address of www.google.com: DNS*
- *DNS query created, encapsulated in UDP, encapsulated in IP, encapsulated in Eth. To send frame to router, need MAC address of router interface: ARP*
- *ARP query broadcast, received by router, which replies with ARP reply giving MAC address of router interface*

A day in the life... ARP (before DNS, before HTTP)



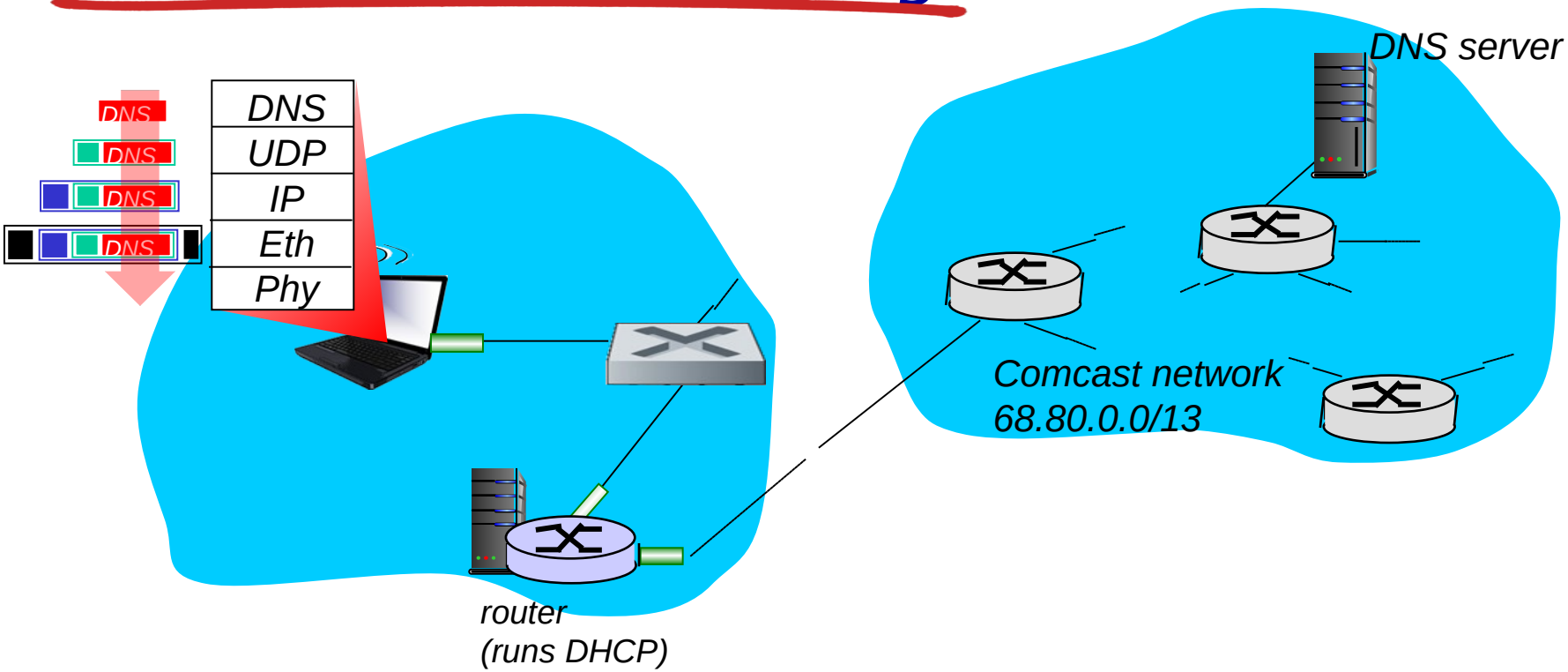
- *before sending HTTP request, need IP address of www.google.com: DNS*
- *DNS query created, encapsulated in UDP, encapsulated in IP, encapsulated in Eth. To send frame to router, need MAC address of router interface: ARP*
- *ARP query broadcast, received by router, which replies with ARP reply giving MAC address of router interface*

A day in the life... ARP (before DNS, before HTTP)

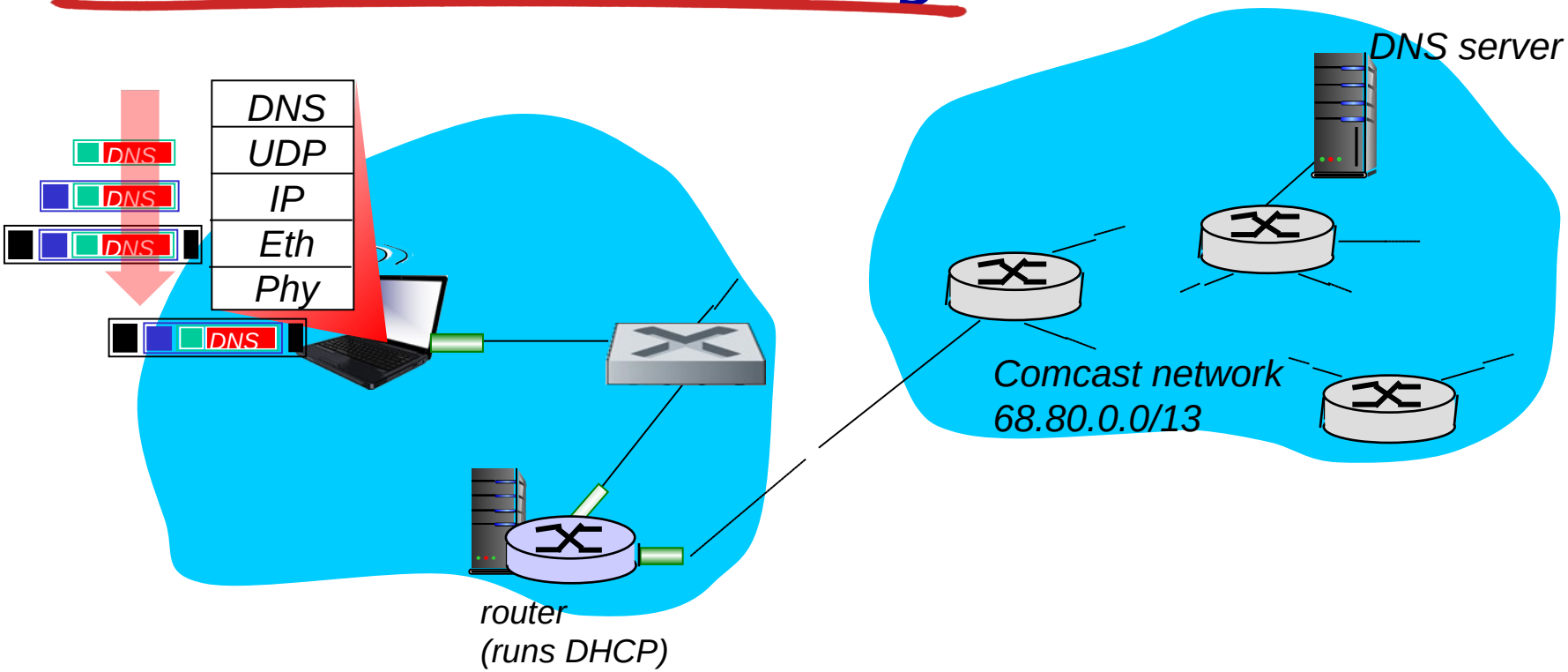


- *before sending HTTP request, need IP address of www.google.com: DNS*
- *DNS query created, encapsulated in UDP, encapsulated in IP, encapsulated in Eth. To send frame to router, need MAC address of router interface: ARP*
- *ARP query broadcast, received by router, which replies with ARP reply giving MAC address of router interface*
- *client now knows MAC address of first hop router, so can now send frame containing DNS query*

A day in the life... using DNS

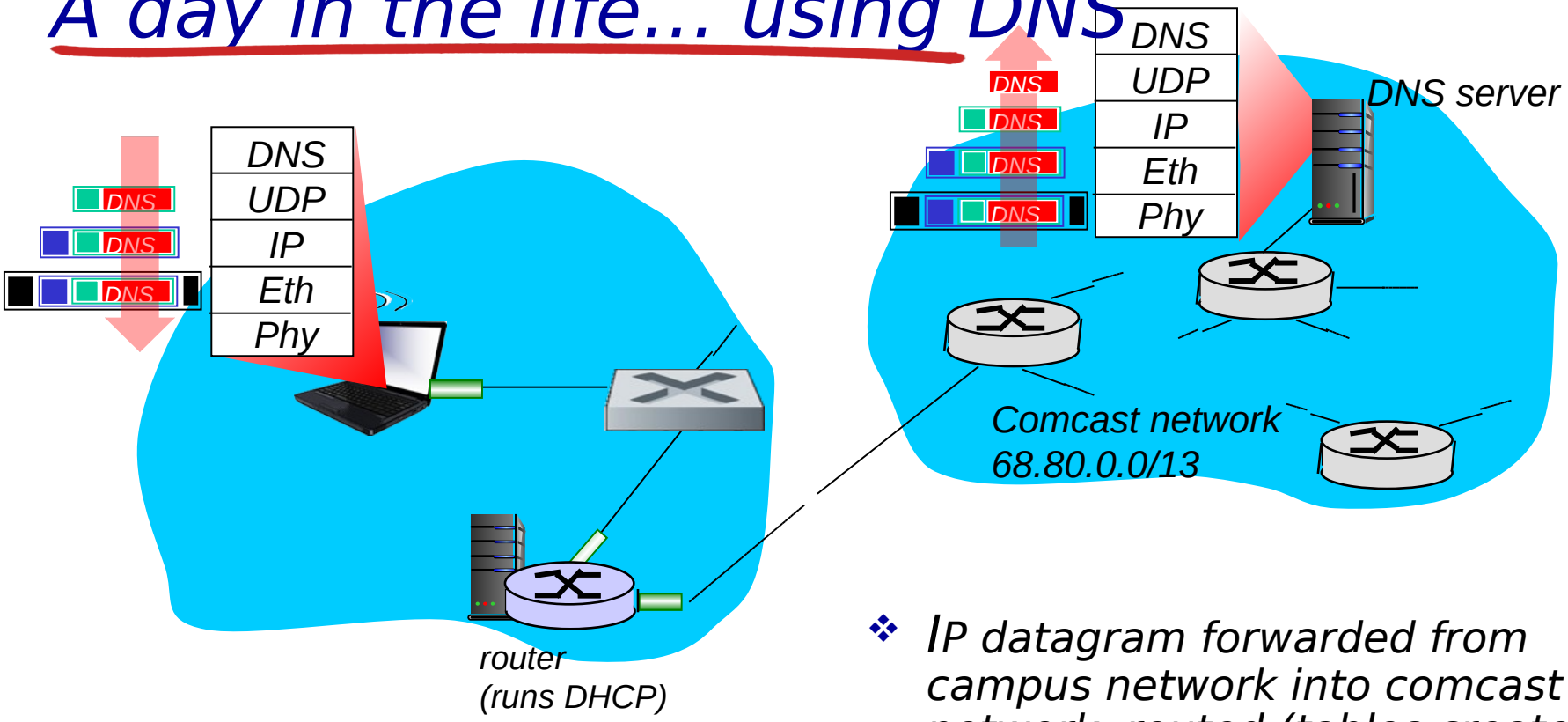


A day in the life... using DNS



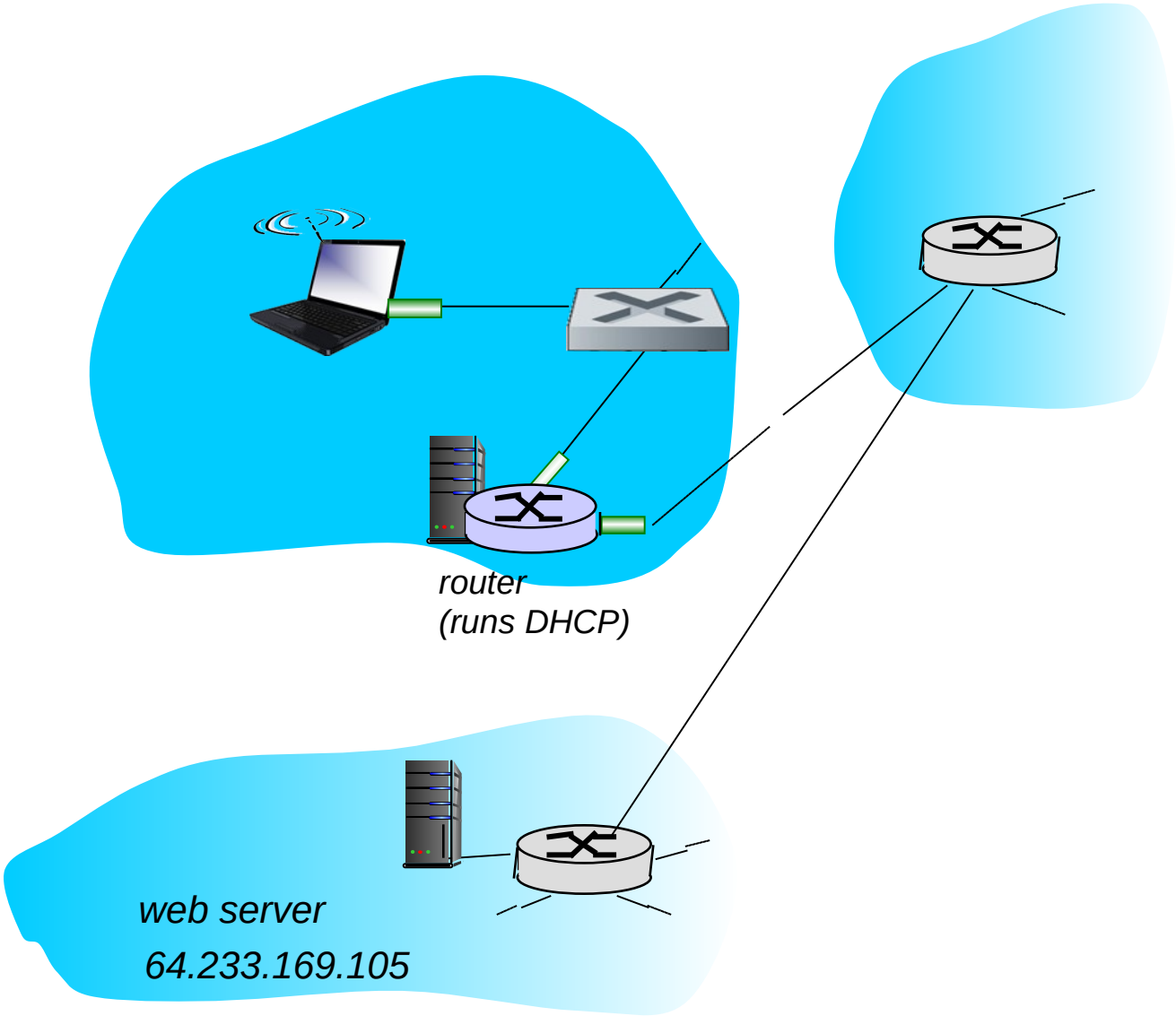
- *IP datagram containing DNS query forwarded via LAN switch from client to 1st hop router*

A day in the life... using DNS

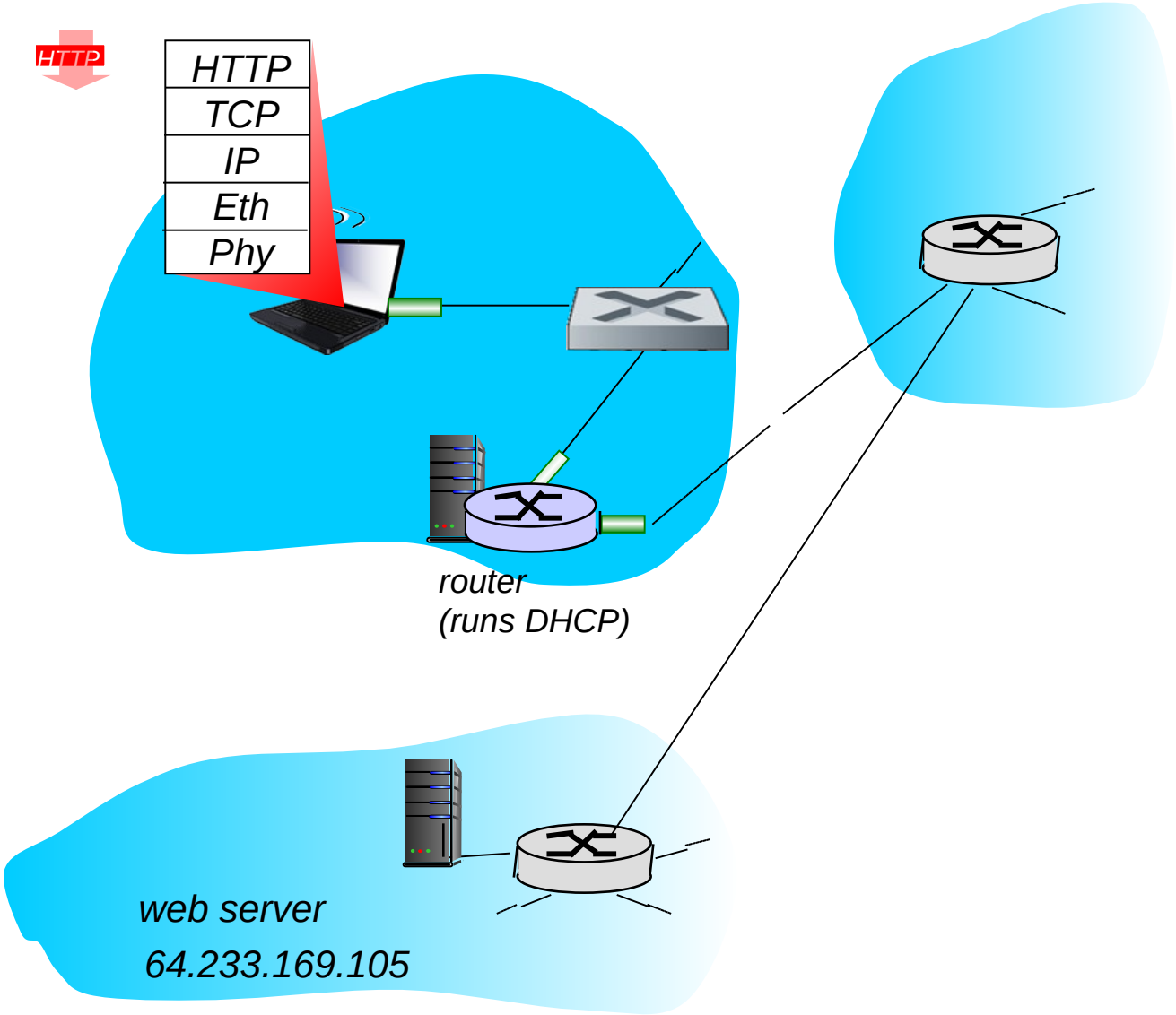


- *IP datagram containing DNS query forwarded via LAN switch from client to 1st hop router*
- *IP datagram forwarded from campus network into comcast network, routed (tables created by RIP, OSPF, IS-IS and/or BGP routing protocols) to DNS server*
- *demux'ed to DNS server*
- *DNS server replies to client with IP address of www.google.com*

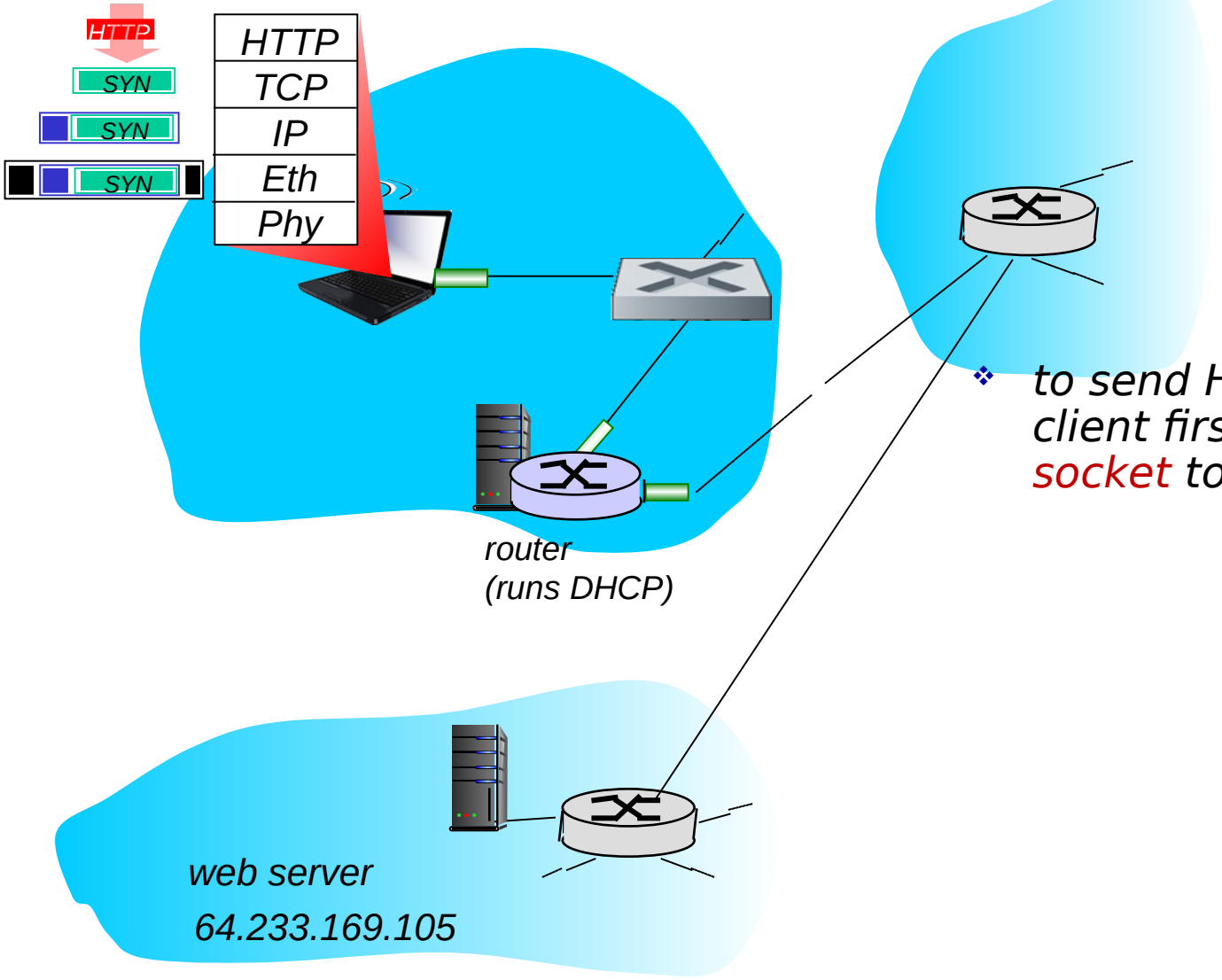
A day in the life...TCP connection carrying HTTP



A day in the life...TCP connection carrying HTTP

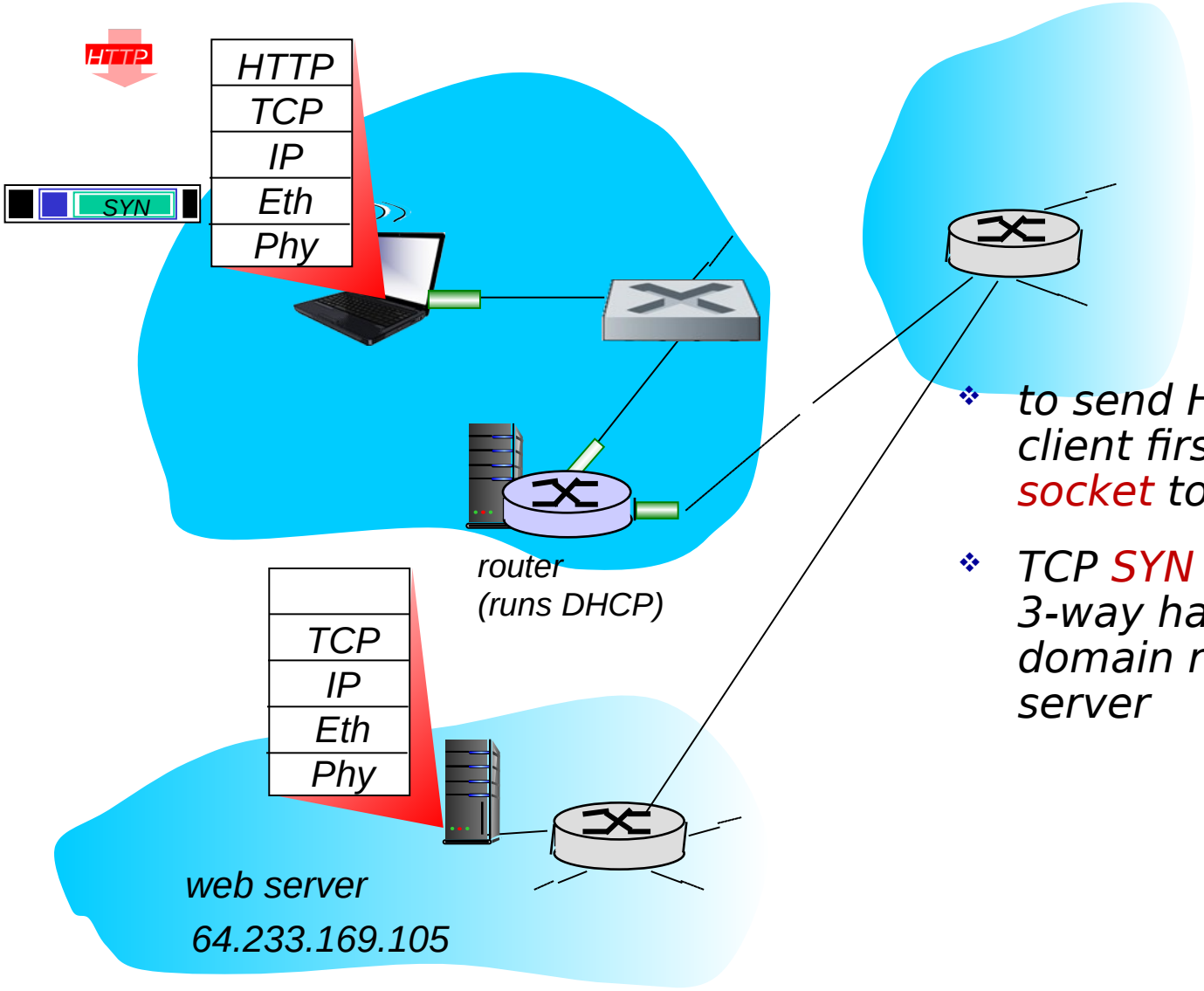


A day in the life...TCP connection carrying HTTP



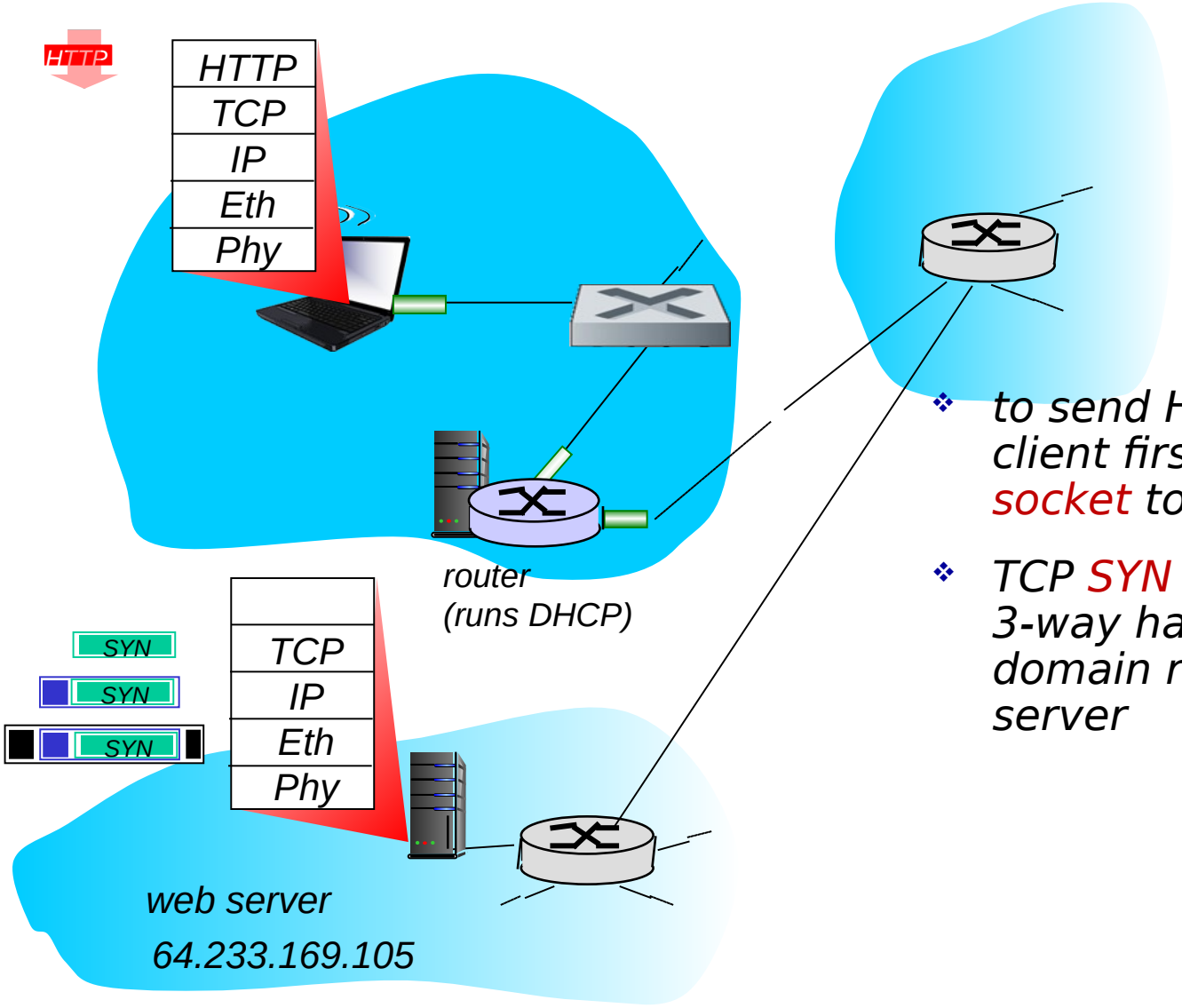
- to send HTTP request, client first opens TCP socket to web server

A day in the life...TCP connection carrying HTTP



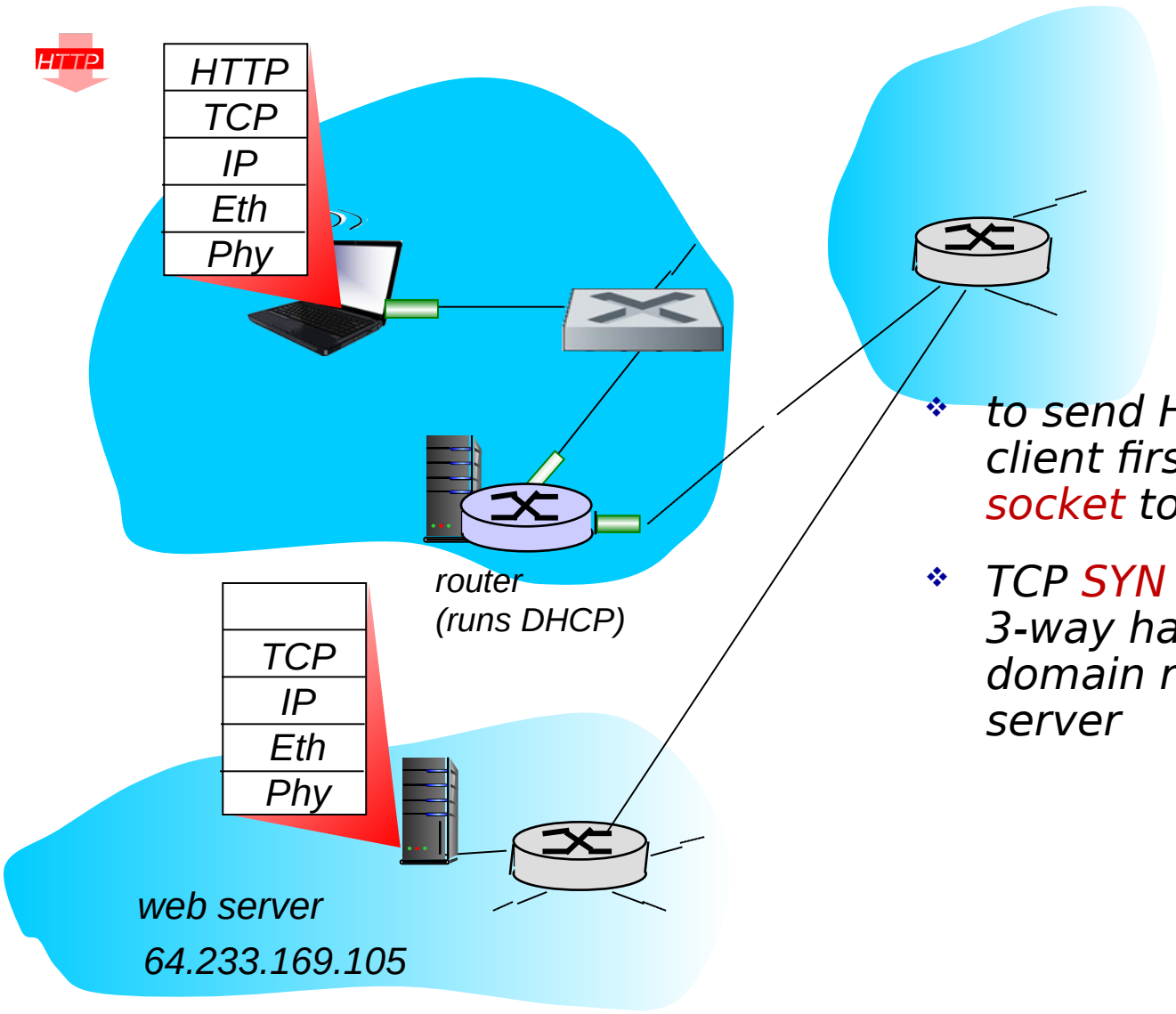
- *to send HTTP request, client first opens TCP socket to web server*
- *TCP SYN segment (step 1 in 3-way handshake) inter-domain routed to web server*

A day in the life...TCP connection carrying HTTP



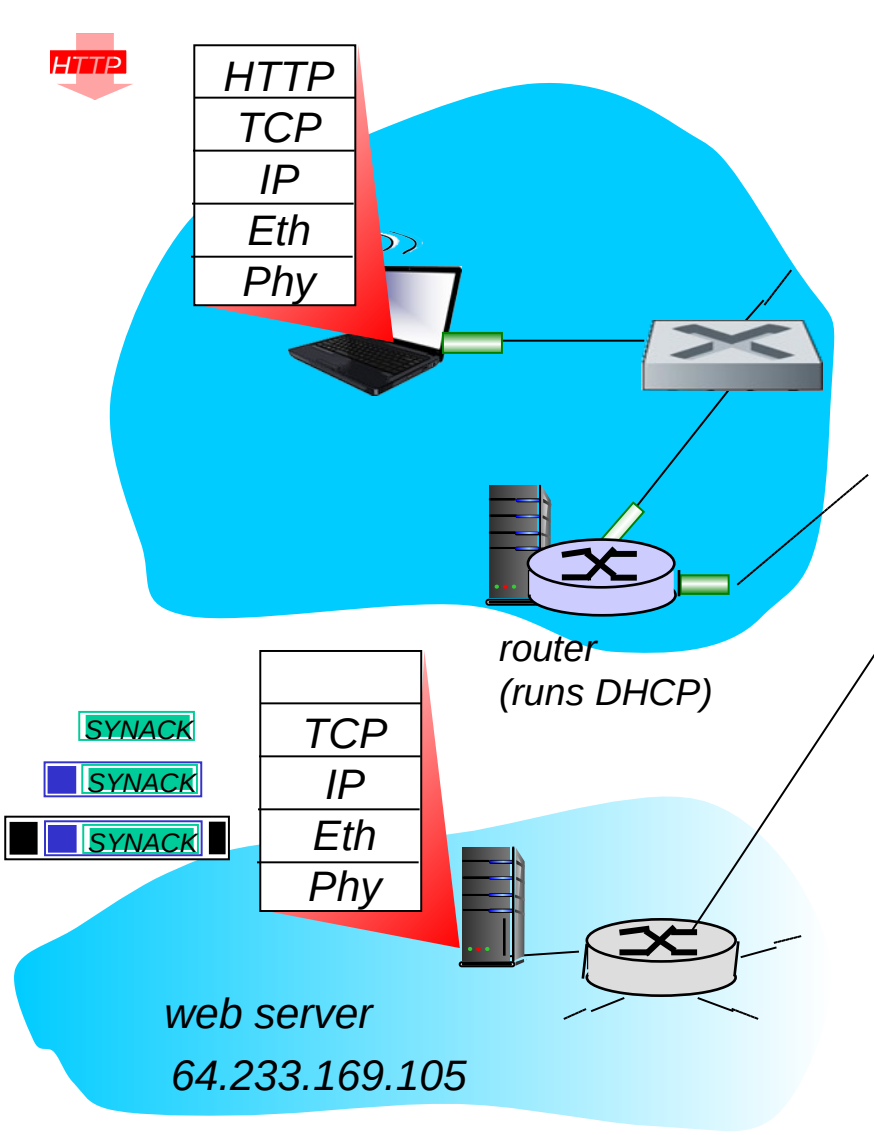
- *to send HTTP request, client first opens TCP socket to web server*
- *TCP SYN segment (step 1 in 3-way handshake) inter-domain routed to web server*

A day in the life...TCP connection carrying HTTP



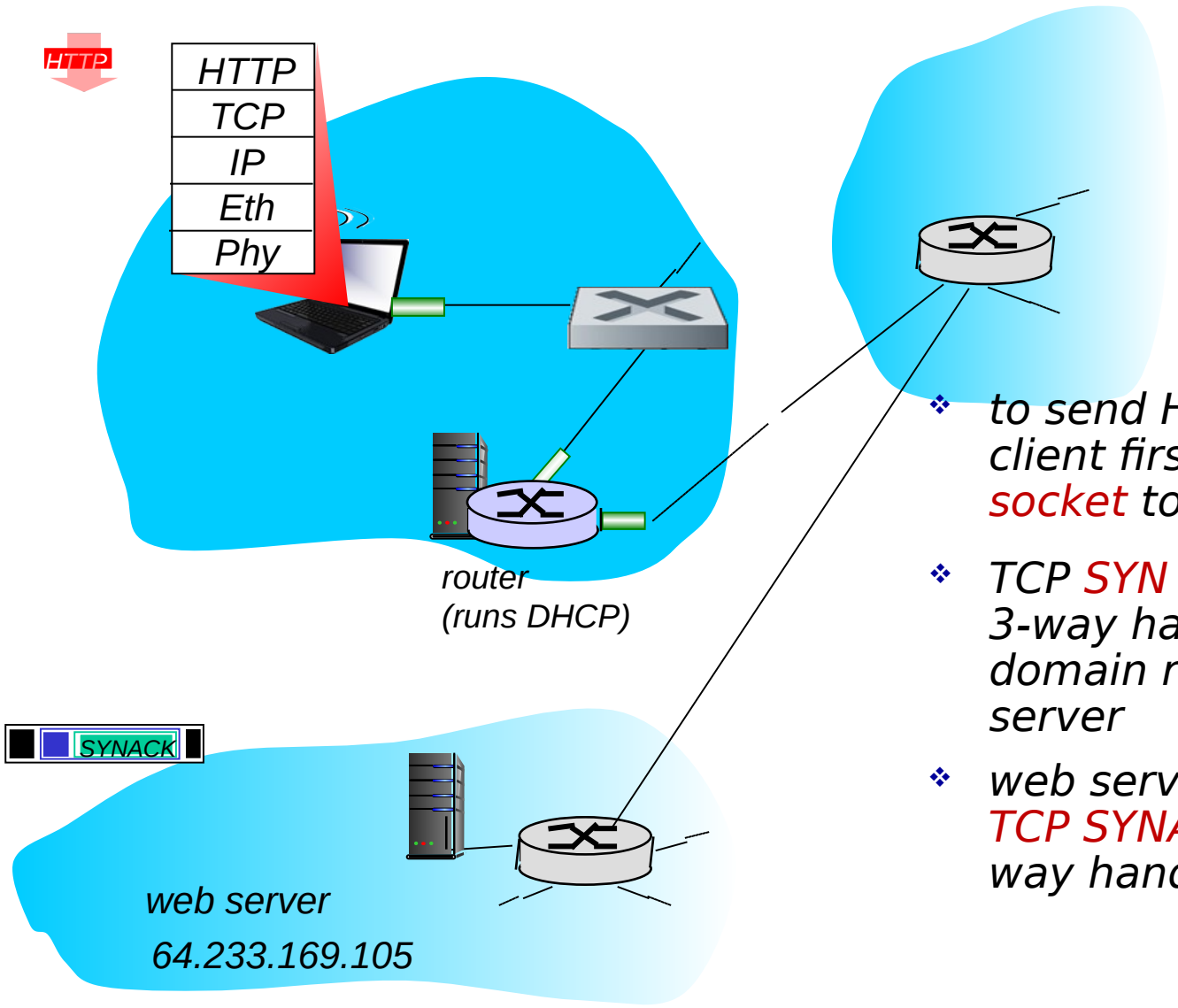
- *to send HTTP request, client first opens TCP socket to web server*
- *TCP SYN segment (step 1 in 3-way handshake) inter-domain routed to web server*

A day in the life...TCP connection carrying HTTP



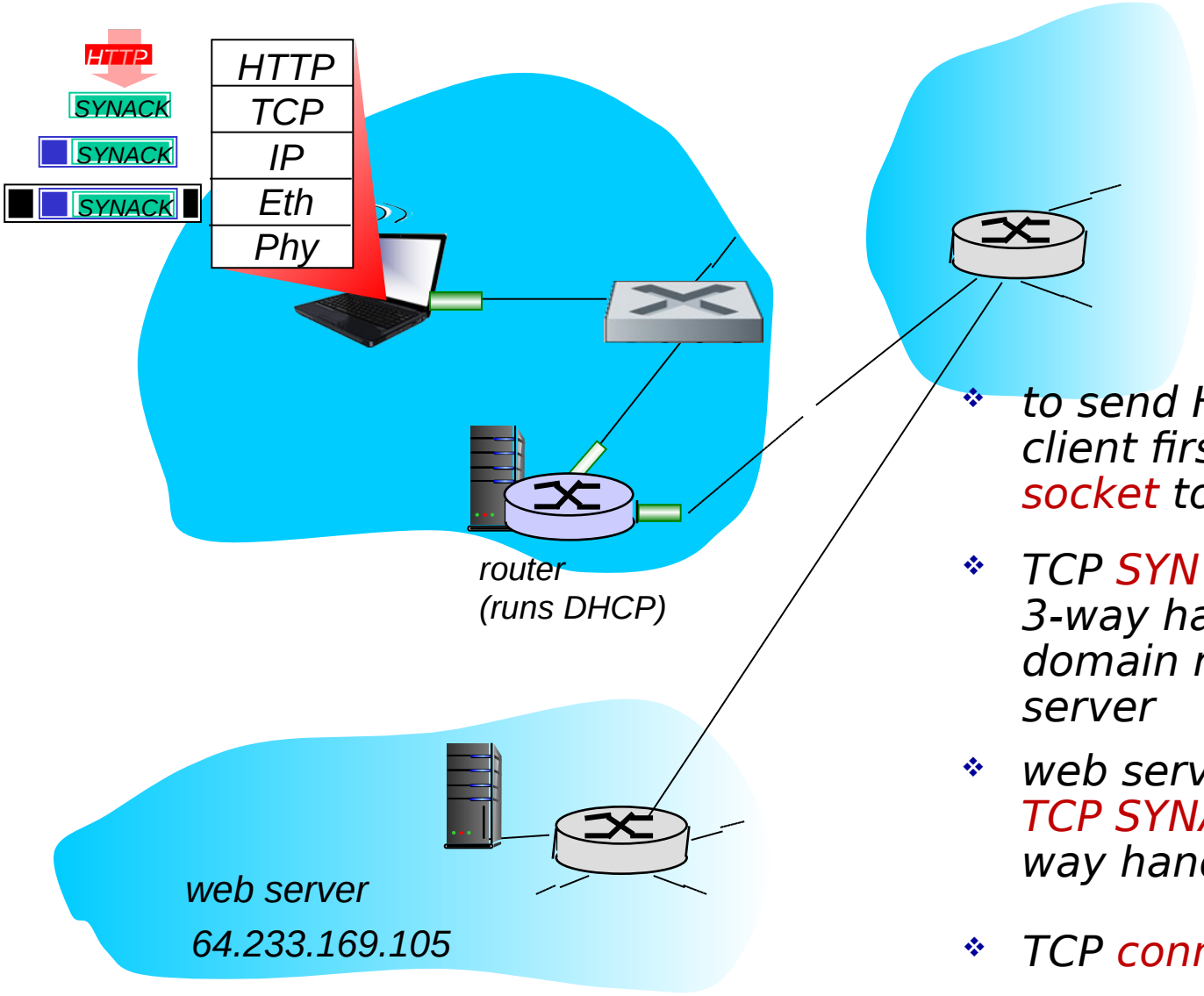
- *to send HTTP request, client first opens TCP socket to web server*
- *TCP SYN segment (step 1 in 3-way handshake) inter-domain routed to web server*
- *web server responds with TCP SYNACK (step 2 in 3-way handshake)*

A day in the life...TCP connection carrying HTTP



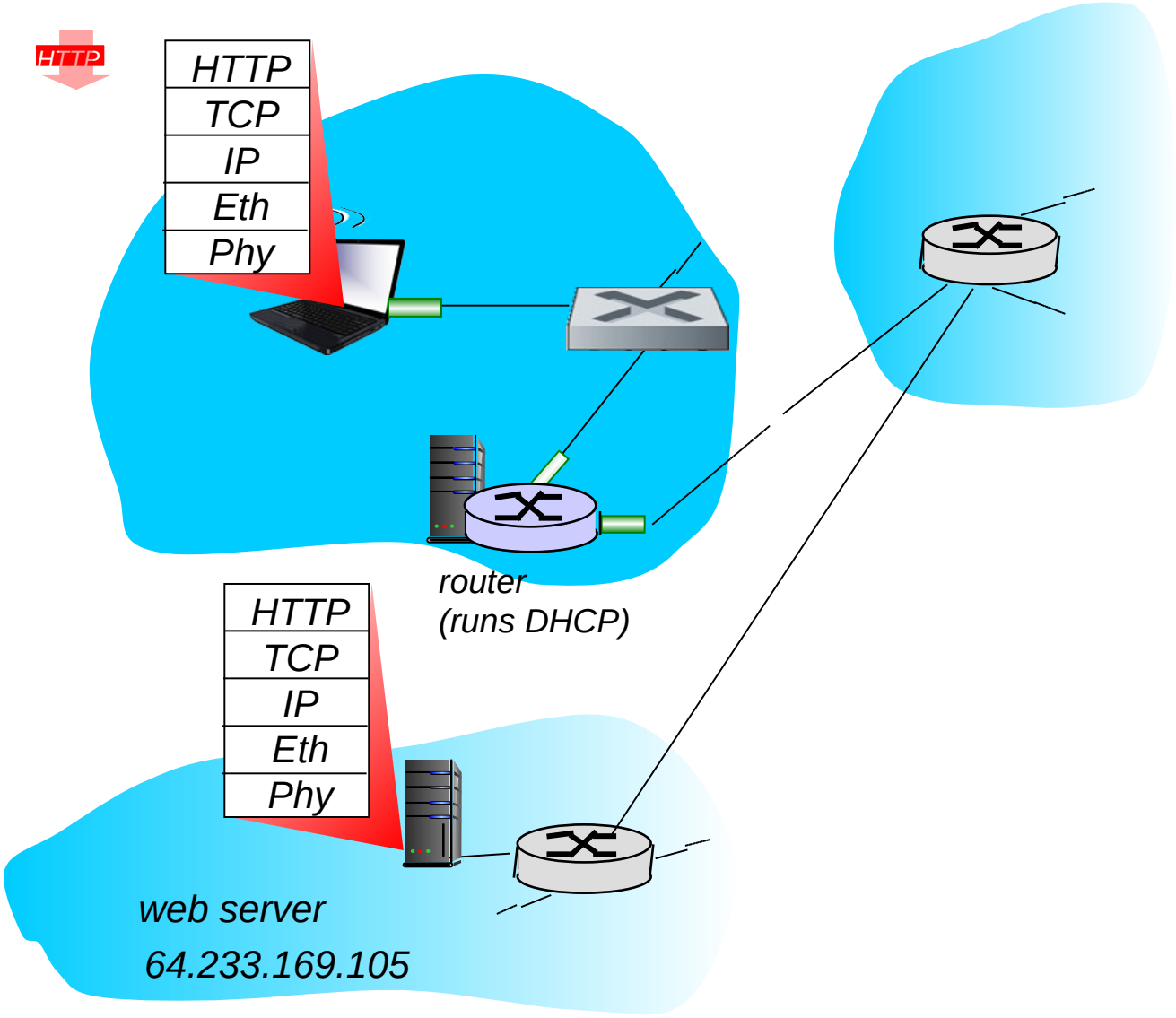
- *to send HTTP request, client first opens TCP socket to web server*
- *TCP SYN segment (step 1 in 3-way handshake) inter-domain routed to web server*
- *web server responds with TCP SYNACK (step 2 in 3-way handshake)*

A day in the life...TCP connection carrying HTTP

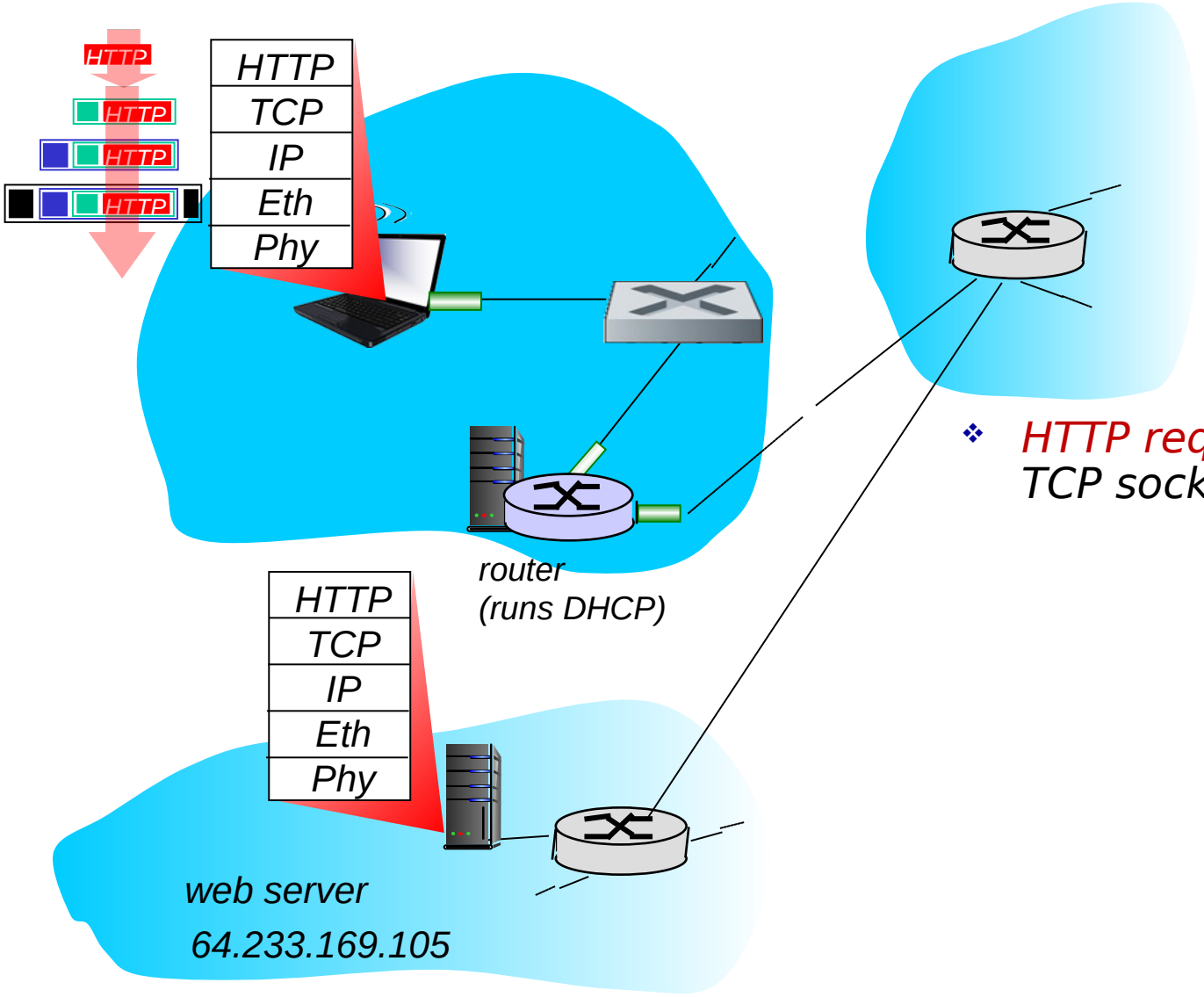


- to send HTTP request, client first opens TCP socket to web server
- TCP SYN segment (step 1 in 3-way handshake) inter-domain routed to web server
- web server responds with TCP SYNACK (step 2 in 3-way handshake)
- TCP connection established!

A day in the life... HTTP request/reply

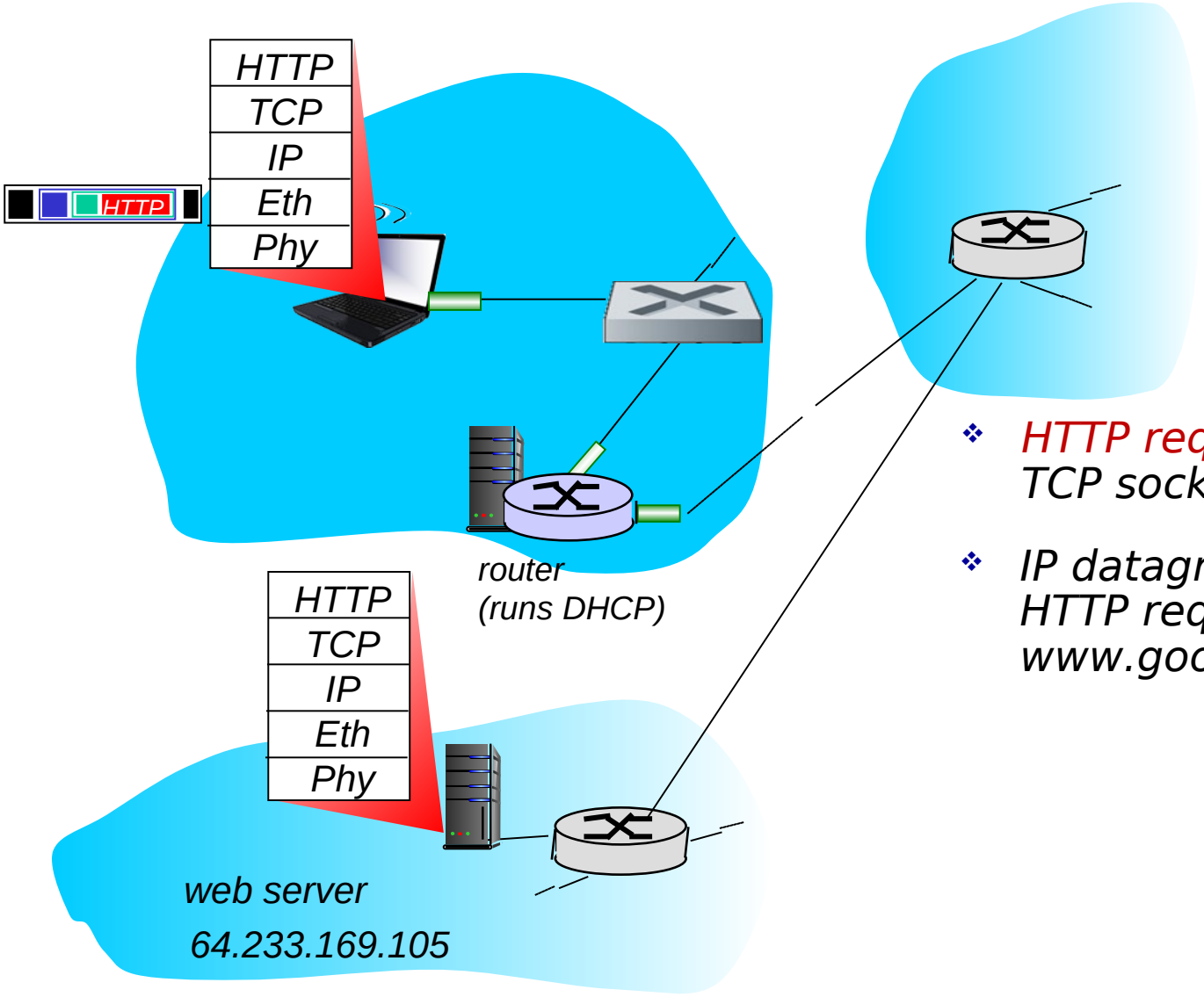


A day in the life... HTTP request/reply



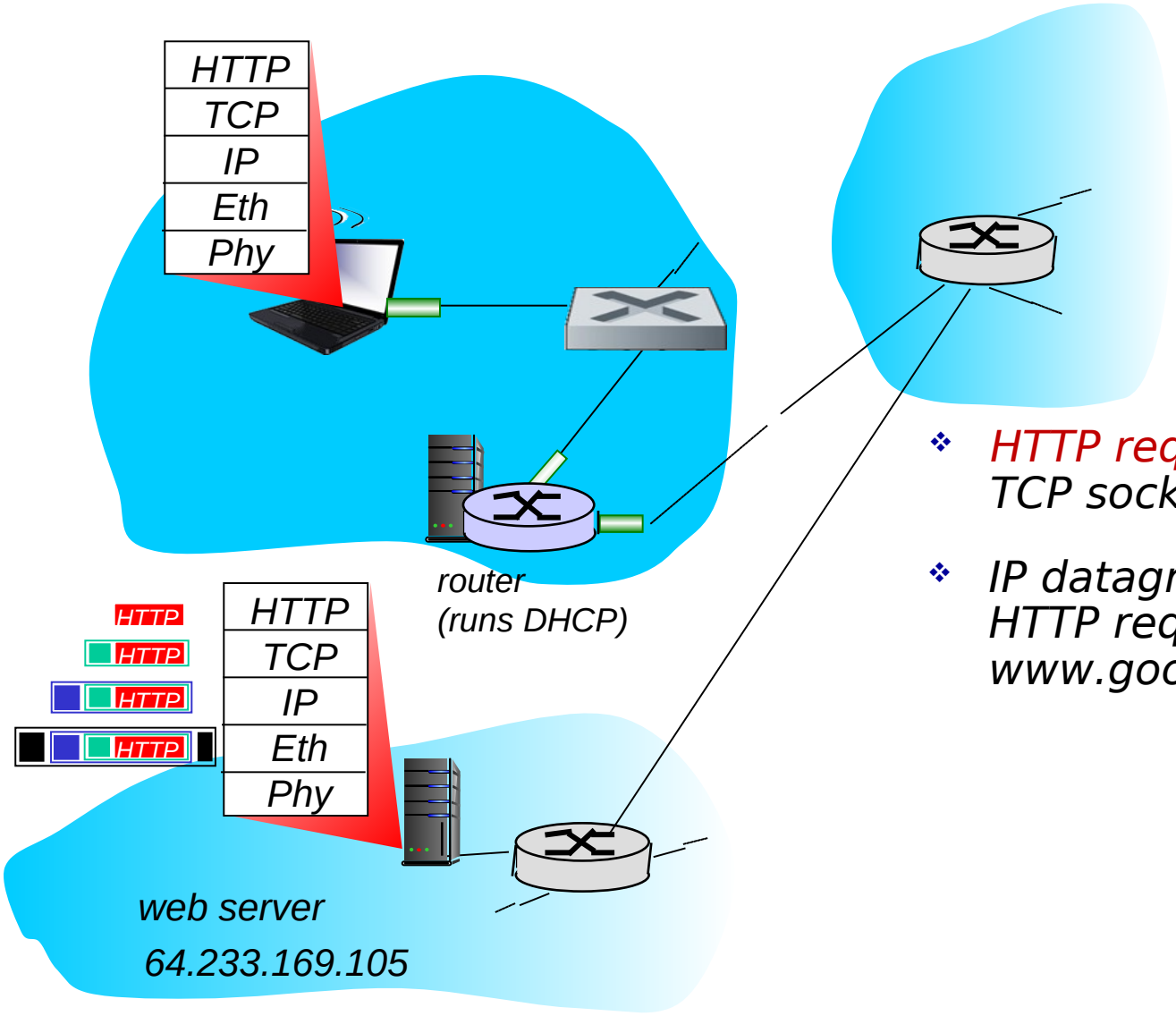
- *HTTP request sent into TCP socket*

A day in the life... HTTP request/reply



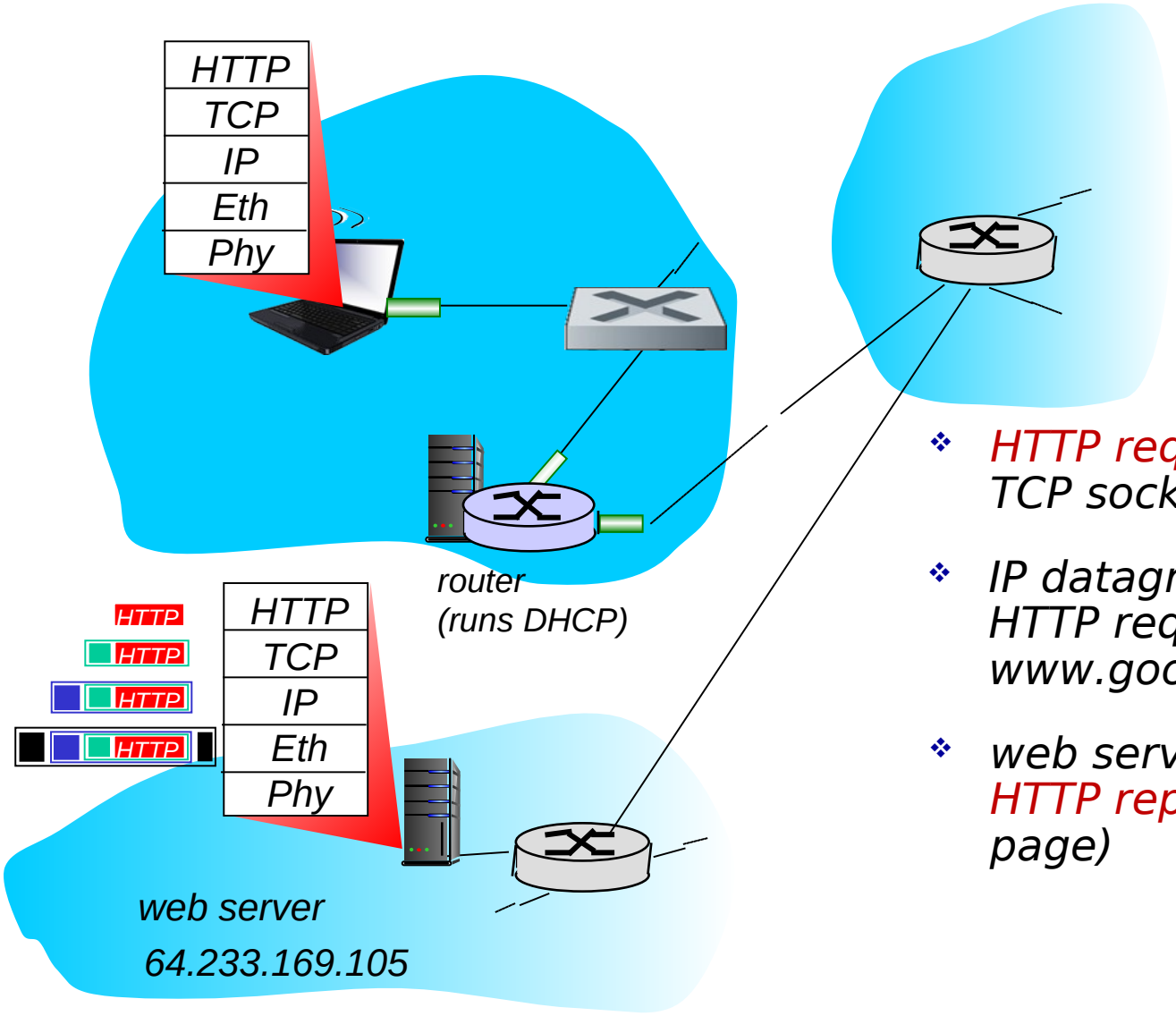
- *HTTP request sent into TCP socket*
- *IP datagram containing HTTP request routed to www.google.com*

A day in the life... HTTP request/reply



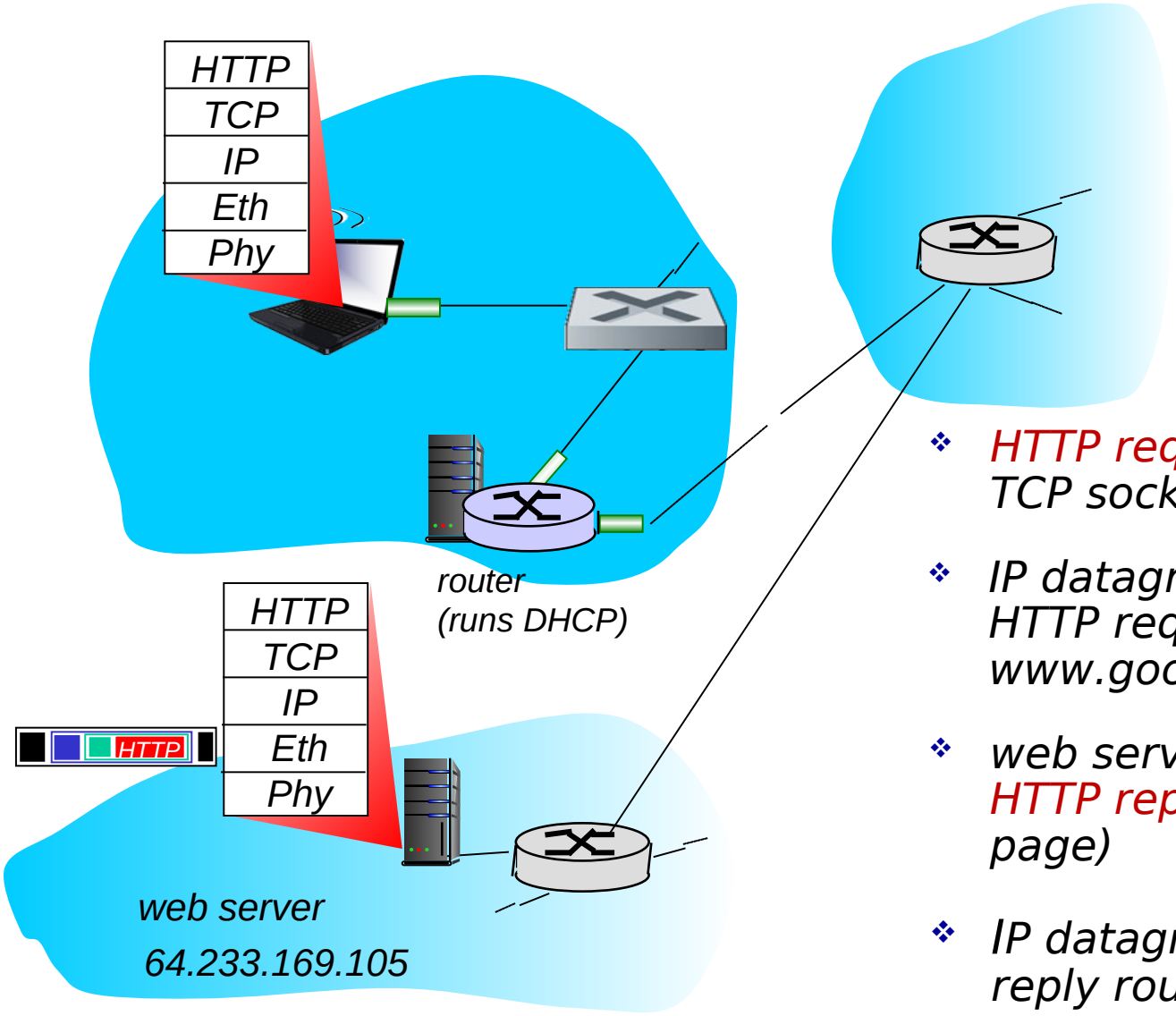
- *HTTP request sent into TCP socket*
- *IP datagram containing HTTP request routed to www.google.com*

A day in the life... HTTP request/reply



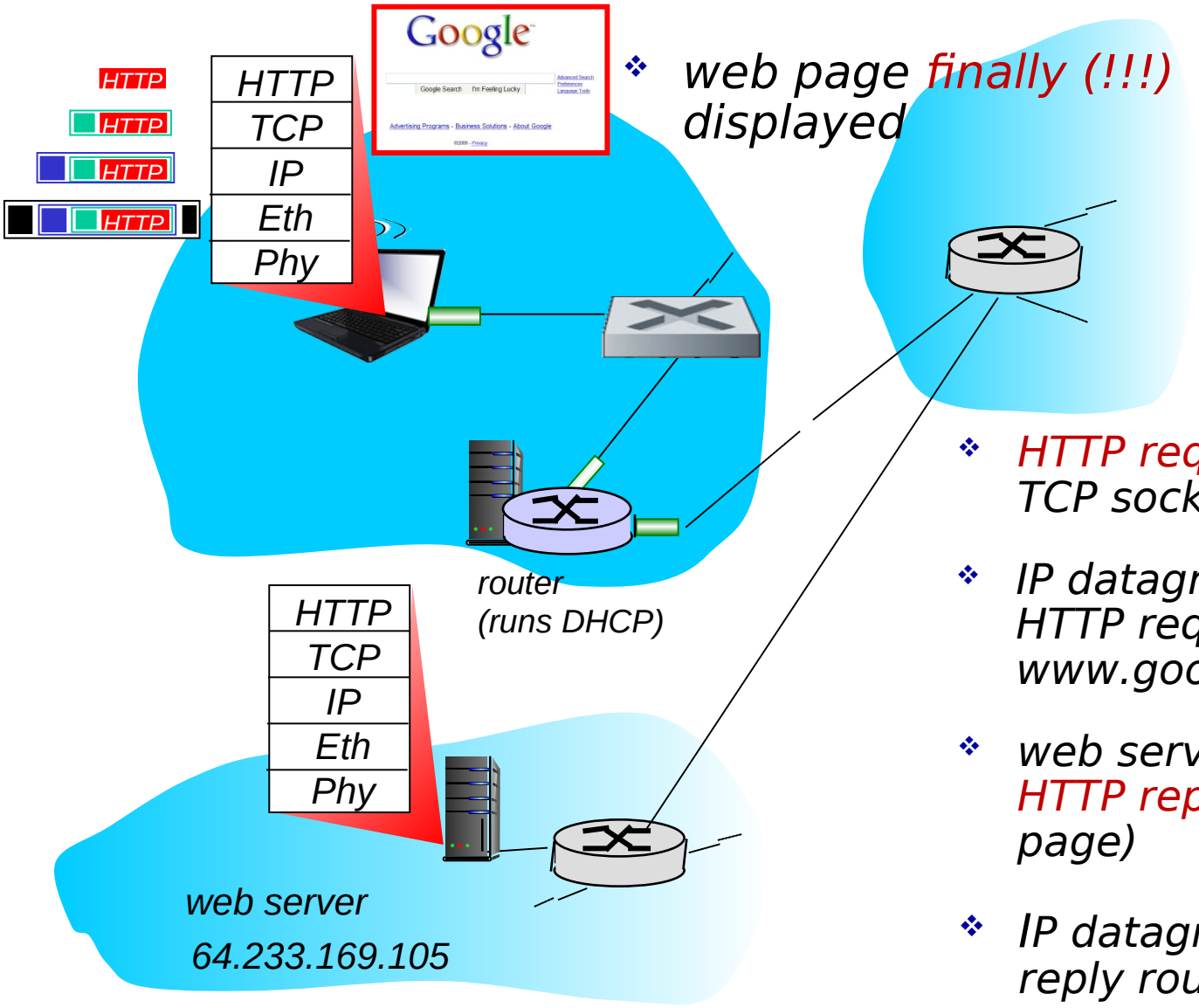
- *HTTP request sent into TCP socket*
- *IP datagram containing HTTP request routed to www.google.com*
- *web server responds with HTTP reply (containing web page)*

A day in the life... HTTP request/reply



- *HTTP request sent into TCP socket*
- *IP datagram containing HTTP request routed to www.google.com*
- *web server responds with HTTP reply (containing web page)*
- *IP datagram containing HTTP reply routed back to client*

A day in the life... HTTP request/reply



- *web page finally (!!!) displayed*

- *HTTP request sent into TCP socket*
- *IP datagram containing HTTP request routed to www.google.com*
- *web server responds with HTTP reply (containing web page)*
- *IP datagram containing HTTP reply routed back to client*