UNIVERSITY OF WATERLOO

CS 456/656
Computer Networks

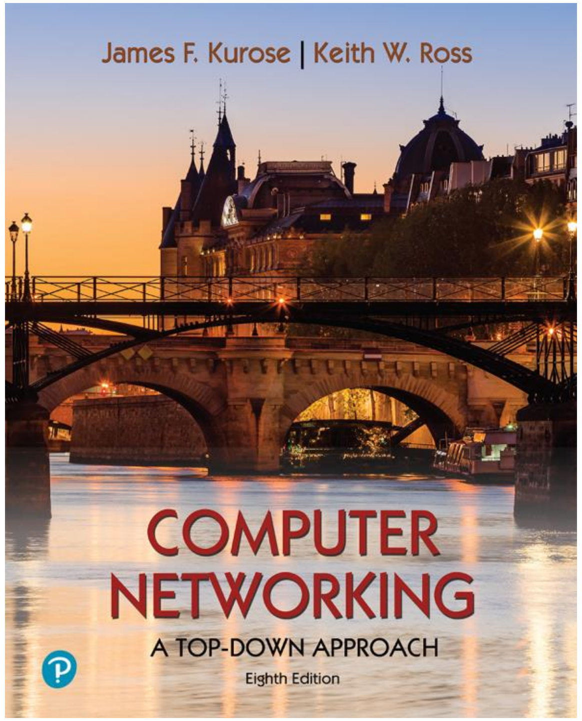
Lecture 17: Naming and Addressing

Mina Tahmasbi Arashloo and Bo Sun

Fall 2024

A note on the slides

Adapted from the slides that accompany this book.

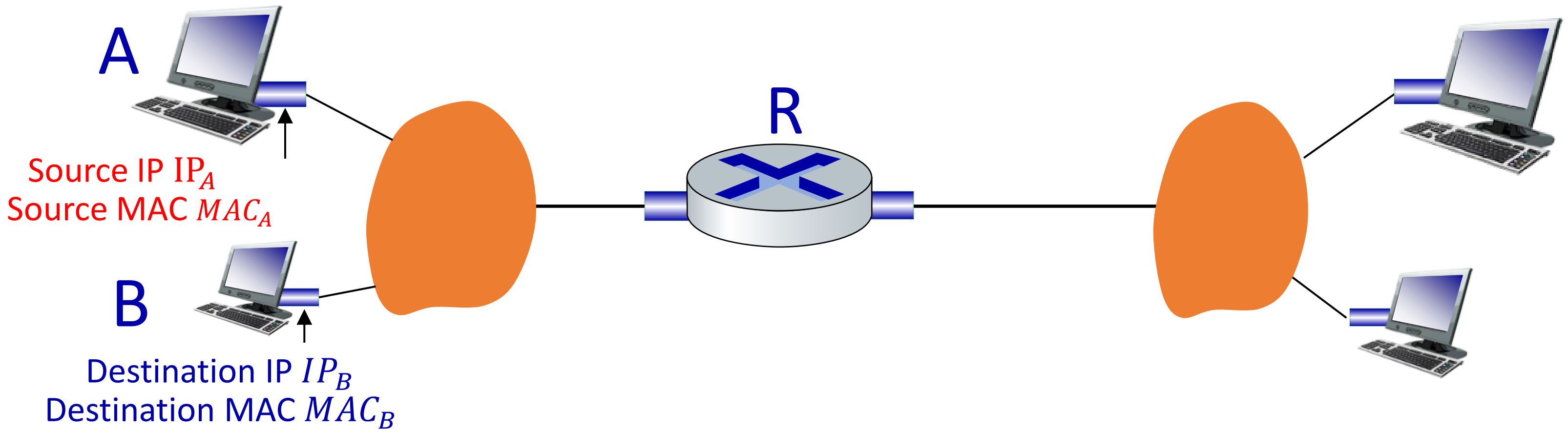


Computer Networking: A Top-Down Approach
8th edition
Jim Kurose, Keith Ross
Pearson, 2020

Naming and Addressing

If host A wants to communicate with host B

- A and B each should have an interface connecting them to their LAN.
- Each interface should have an IP address and a MAC address.
- To send data to B, A should create a packet with the following addresses:

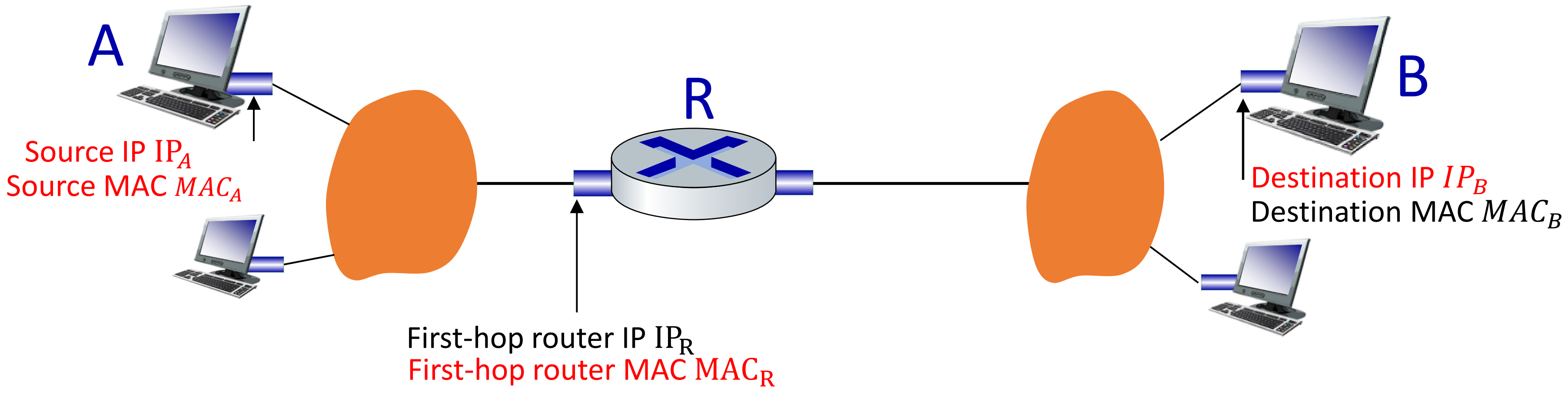


A and B are in the same LAN

Naming and Addressing

If host A wants to communicate with host B

- A and B each should have an interface connecting them to their LAN.
- Each interface should have an IP address and a MAC address.
- To send data to B, A should create a packet with the following addresses:



A and B are in different LANs

Naming and Addressing

If host A wants to communicate with host B

- A and B each should have an interface connecting them to their LAN.
- Each interface should have an IP address and a MAC address.
- To send data to B, A should create a packet with the following addresses:
  - source IP address: IP_A
  - destination IP address: IP_B
  - source MAC address: MAC_A
  - destination MAC address: MAC_B if they are in the same LAN. Otherwise, MAC_R, where R is the interface of the gateway (first-hop) router in A's LAN.

Naming and Addressing

If host A wants to communicate with host B

- A and B each should have an interface connecting them to their LAN.
- Each interface should have an IP address and a MAC address.
- To send data to B, A should create a packet with the following addresses:
  - source IP address: IP_A
  - destination IP address: IP_B
  - source MAC address: MAC_A
  - destination MAC address: MAC_B if they are in the same LAN. Otherwise, MAC_R, where R is the interface of the gateway (first-hop) router in A's LAN.

This Lecture: How does A find these addresses?

Naming and Addressing

If host A wants to communicate with host B

- A and B each should have an interface connecting them to their LAN.
- Each interface should have an IP address and a MAC address.
- To send data to B, A should create a packet with the following addresses:
  - source IP address: IP_A
  - destination IP address: IP_B
  - source MAC address: MAC_A Comes with the interface
  - destination MAC address: MAC_B if they are in the same LAN. Otherwise, MAC_R, where R is the interface of the gateway (first-hop) router in A's LAN.

Naming and Addressing

If host A wants to communicate with host B

- A and B each should have an interface connecting them to their LAN.
- Each interface should have an IP address and a MAC address.
- To send data to B, A should create a packet with the following addresses:
  - source IP address: IP_A
  - destination IP address: IP_B
  - source MAC address: MAC_A
  - destination MAC address: MAC_B if they are in the same LAN. Otherwise, MAC_R, where R is the interface of the gateway (first-hop) router in A's LAN.

Comes with the interface

Through ARP. But has to know B's (or R's) IP address.

Naming and Addressing

If host A wants to communicate with host B

- A and B each should have an interface connecting them to their LAN.
- Each interface should hav… C address.
- To send data to B, A should create a packet with the following addresses:
  - source IP address: IP_A
  - destination IP address: IP_B
  - source MAC address: MAC_A
  - destination MAC address: MAC_B if they are in the same LAN. Otherwise, MAC_R, where R is the interface of the gateway (first-hop) router in A's LAN.

How does A get an IP address ??

Comes with the interface

Through ARP. But has to know B's (or R's) IP address.

Naming and Addressing

If host A wants to communicate with host B

- A and B each should have an interface connecting them to their LAN.
- Each interface should hav … C address.
- To send data to B, A should create a packet with the following addresses:
  - source IP address: IP_A
  - destination IP address: IP_B
  - source MAC address: MAC_A
  - destination MAC address: MAC_B if they are in the same LAN. Otherwise, MAC_R, where R is the interface of the gateway (first-hop) router in A's LAN.

How does A get an IP address ??

How does A find B's (or R's) IP address?

Comes with the interface

Through ARP. But has to know B's (or R's) IP address.

Naming and Addressing: roadmap

- DHCP
  - Helps A find an IP address
  - Helps A find the IP address of the gateway router
- DNS
  - Helps A find B’s IP address

Naming and Addressing: roadmap

- DHCP
  - Helps A find an IP address
  - Helps A find the IP address of the gateway router
- DNS
  - Helps A find B's IP address

DHCP: Dynamic Host Configuration Protocol

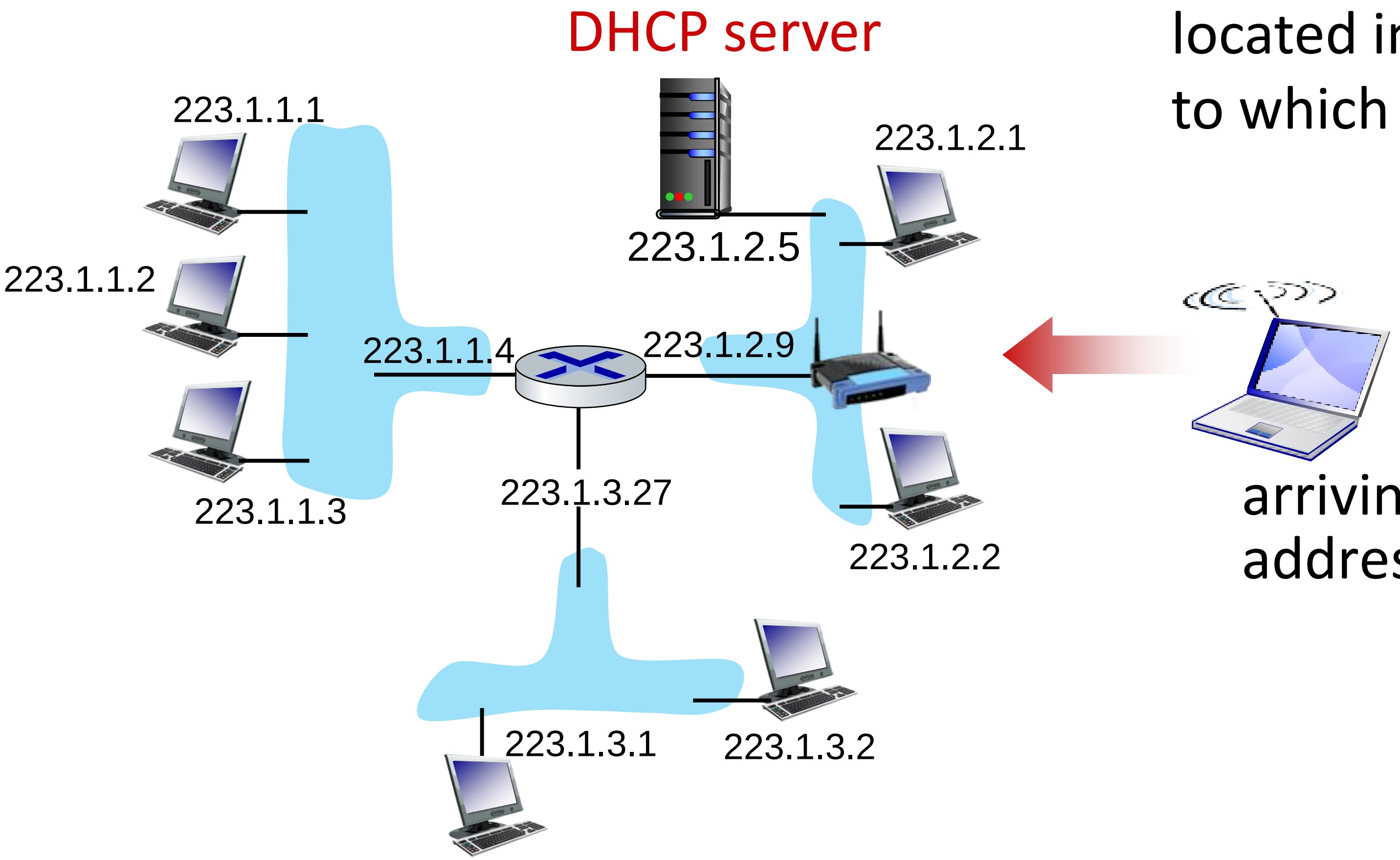
goal: host *dynamically* obtains IP address from network server when it "joins" network

- can renew its lease on address in use
- allows reuse of addresses (only hold address while connected/on)
- support for mobile users who join/leave network

DHCP overview:

- host broadcasts DHCP discover msg [optional]
- DHCP server responds with DHCP offer msg [optional]
- host requests IP address: DHCP request msg
- DHCP server sends address: DHCP ack msg

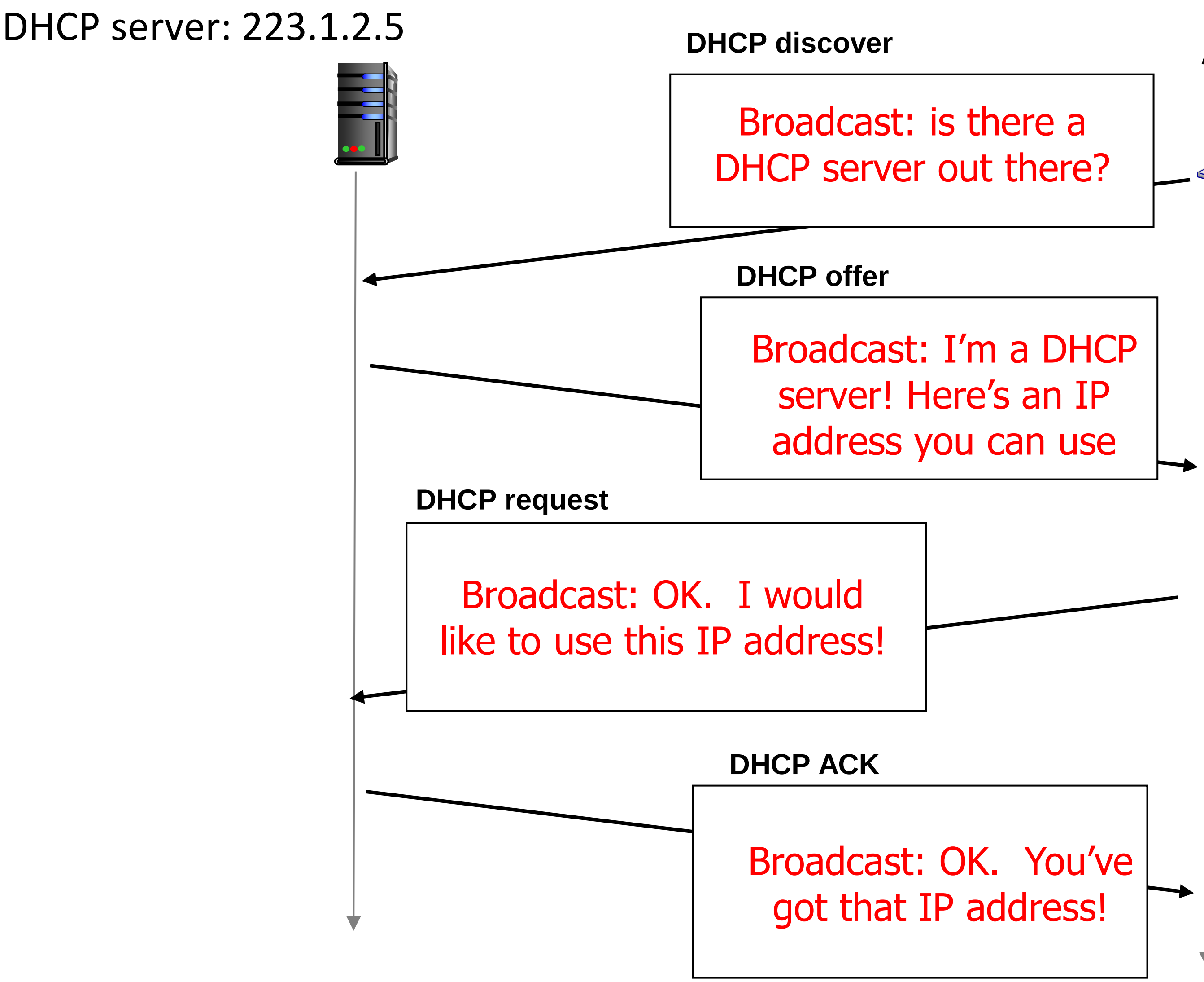
DHCP client-server scenario



Typically, DHCP server will be co-located in router, serving all subnets to which router is attached

arriving DHCP client needs address in this network

DHCP client-server scenario



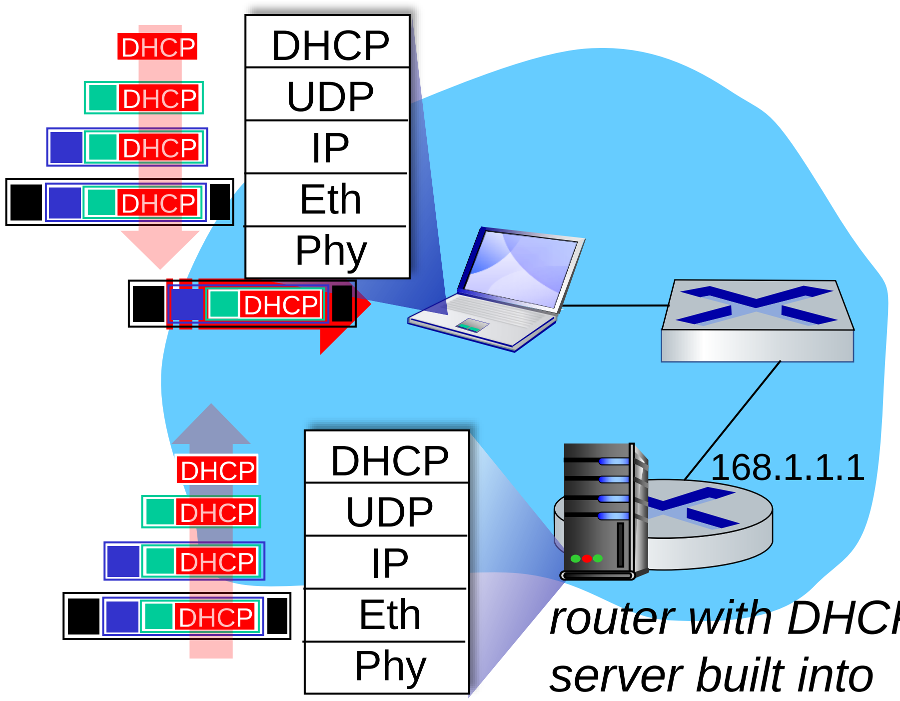
The two steps above can be skipped "if a client remembers and wishes to reuse a previously allocated network address" [RFC 2131]

DHCP: more than IP addresses

DHCP can return more than just allocated IP address on subnet:

- address of first-hop router for client
- name and IP address of DNS sever
  - Which we will learn about next
- network mask (indicating network versus host portion of address)

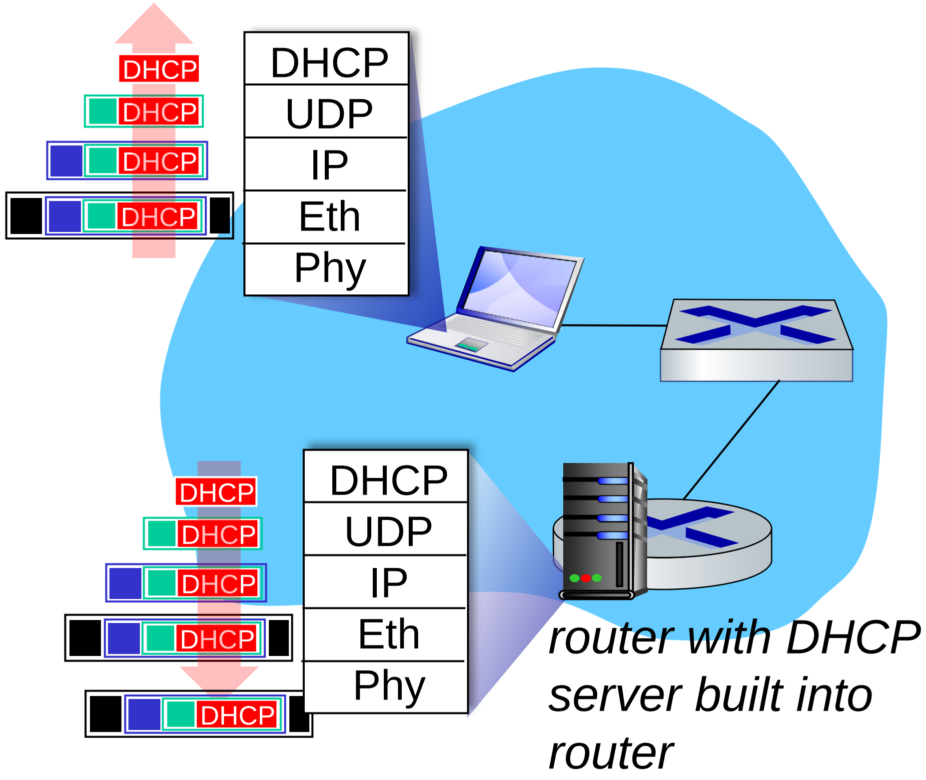
DHCP: example



router with DHCP server built into router

- Connecting laptop will use DHCP to get IP address, address of first-hop router, address of DNS server.
- DHCP REQUEST message encapsulated in UDP, encapsulated in IP, encapsulated in Ethernet
- Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server
- Ethernet de-mux'ed to IP de-mux'ed, UDP de-mux'ed to DHCP

DHCP: example



- DHCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server
- encapsulated DHCP server reply forwarded to client, de-muxing up to DHCP at client
- client now knows its IP address, name and IP address of DNS server, IP address of its first-hop router

Naming and Addressing

If host A wants to communicate with host B

- A and B each should have an interface connecting them to their LAN.
- Each interface should hav … C address.
- To send data to B, A should create a packet with the following addresses:
  - source IP address: IP_A
  - destination IP address: IP_B
  - source MAC address: MAC_A
  - destination MAC address: MAC_B if they are in the same LAN. Otherwise, MAC_R, where R is the interface of the gateway router in A's LAN.

How does A get an IP address ??

How does A find B's (or R's) IP address?

Comes with the interface

Through ARP. But has to know B's (or R's) IP address.

Naming and Addressing

If host A wants to communicate with host B

- A and B each should have an interface connecting them to their LAN.
- Each interface should hav ess.
- To send data to B, A should create a packet with the following addresses:
  - source IP address: IP_A
  - destination IP address: IP_B
  - source MAC address: MAC_A
  - destination MAC address: MAC_B if they are in the same LAN. Otherwise, MAC_R, where R is the interface of the gateway router in A's LAN.

How does A get an IP address ?? **DHCP**

How does A find B's (or R's) IP address?

Comes with the interface

Through ARP. But has to know B's (or R's) IP address.

What you need to know about DHCP

- The purpose of the protocol
- The protocol's basic operations, i.e., the messages and information exchanged when a host wants to acquire an IP address.
  - E.g., given a scenario where a host (re)joins a LAN, you should be able to fill out the basic information inside the exchanged messages.
- The extra information that DHCP provide for hosts to bootstrap their network configurations.

Naming and Addressing: roadmap

- DHCP
  - Helps A find an IP address
  - Helps A find the IP address of the gateway router
- <u>DNS</u>
  - Helps A find B’s IP address

DNS: Domain Name System

people: many identifiers:

- SIN, name, passport #

Internet hosts, routers:

- IP address (32 bit) - used for addressing datagrams in the network layer.
- "name", e.g., cs.umass.edu - used by humans

DNS: Domain Name System

people: many identifiers:

- SIN, name, passport #

Internet hosts, routers:

- IP address (32 bit) - used for addressing datagrams in the network layer.
- “name”, e.g., cs.umass.edu - used by humans

Human readable
Much easier to remember
IP addresses may change but this doesn’t have to change

DNS: Domain Name System

people: many identifiers:

- SIN, name, passport #

Internet hosts, routers:

- IP address (32 bit) - used for addressing datagrams in the network layer.
- "name", e.g., cs.umass.edu - used by humans

Q: how to map from a name to an IP address?

DNS: Domain Name System

people: many identifiers:

- SIN, name, passport #

Internet hosts, routers:

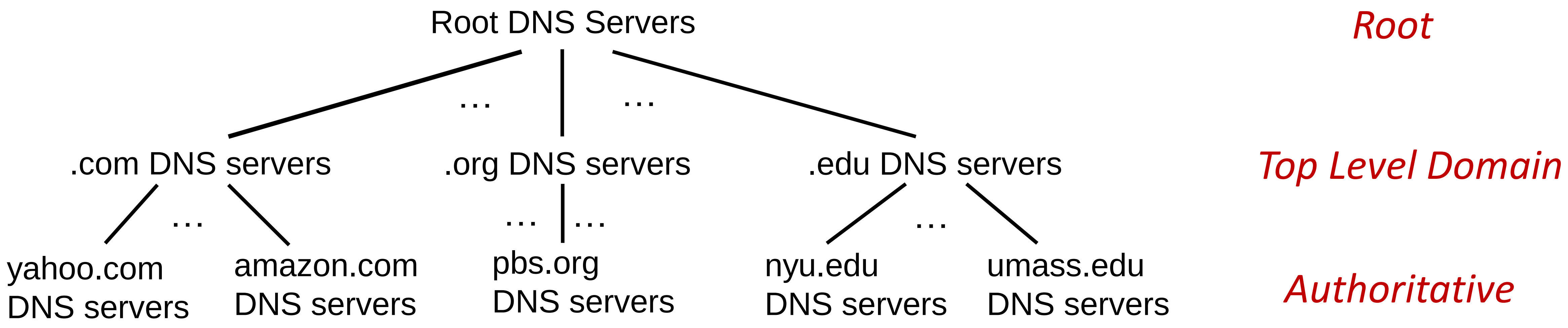
- IP address (32 bit) - used for addressing datagrams in the network layer.
- "name", e.g., cs.umass.edu - used by humans

Q: how to map from a name to an IP address?

Domain Name System (DNS):

- *Database* that keeps the mapping between names and IP addresses
- Implemented in a *distributed* manner over a hierarchy of *name servers*
- *application-layer protocol:* hosts and DNS servers communicate to *resolve* names (address/name translation)
  - DNS is core Internet function, implemented as application-layer protocol

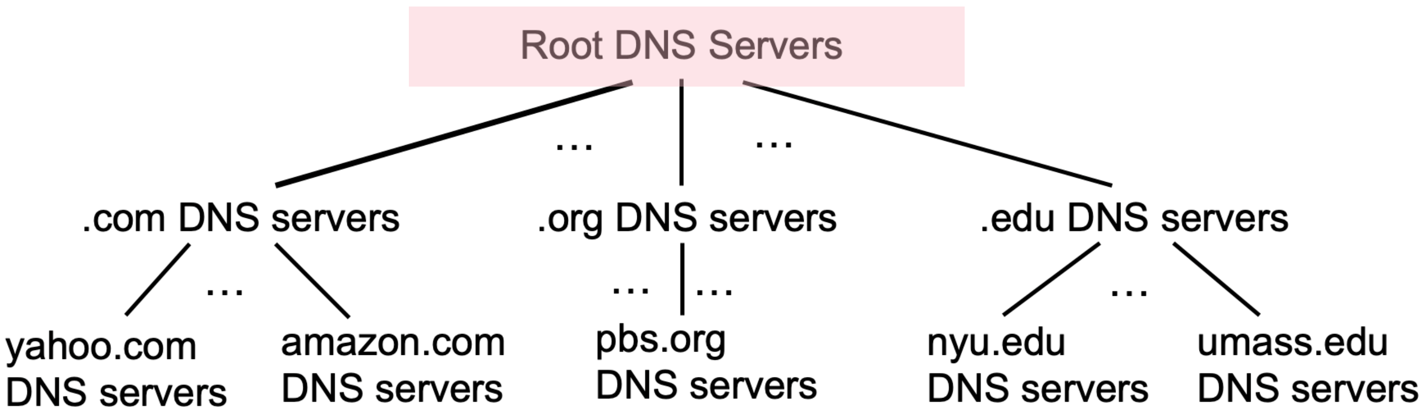
DNS: a distributed, hierarchical database



Client wants IP address for www.amazon.com:

- client queries root server to find .com DNS server
- client queries .com DNS server to get amazon.com DNS server
- client queries amazon.com DNS server to get IP address for www.amazon.com

DNS: root name servers

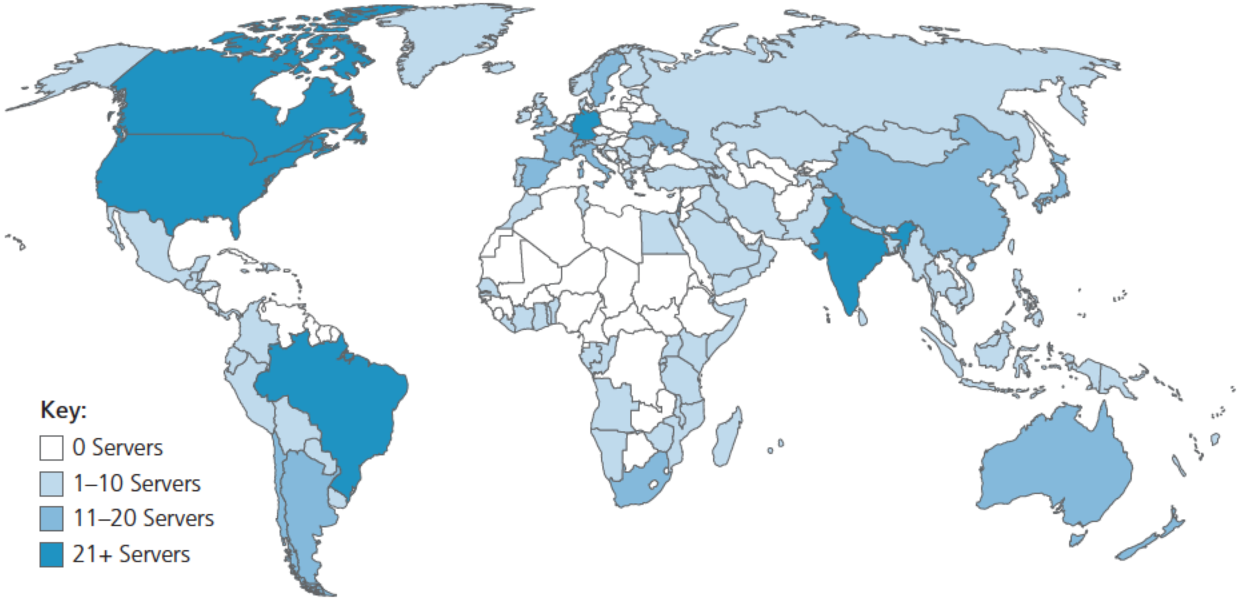


- Provides the IP address of Top-Level-Domain (TLD) servers
  - Right below it in the tree
- They are where one goes to get the translations started.

DNS: root name servers

- *incredibly important* Internet function
  - Internet couldn't function without it!
- ICANN (Internet Corporation for Assigned Names and Numbers) manages root DNS domain

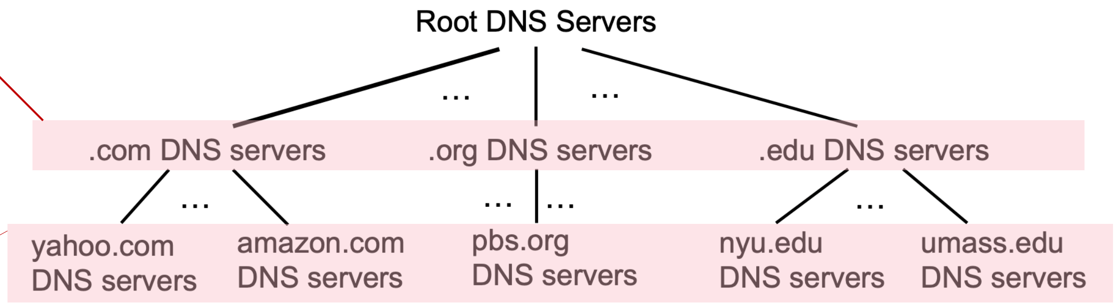
13 logical root name "servers" worldwide each "server" replicated many times (~200 servers in US)



Top-Level Domain, and authoritative servers

Top-Level Domain (TLD) servers:

- responsible for names under .com, .org, .net, .edu, .aero, .jobs, .museums, and all top-level country domains, e.g.: .cn, .uk, .fr, .ca, .jp
- Network Solutions: authoritative registry for .com, .net TLD
- Educause: .edu TLD

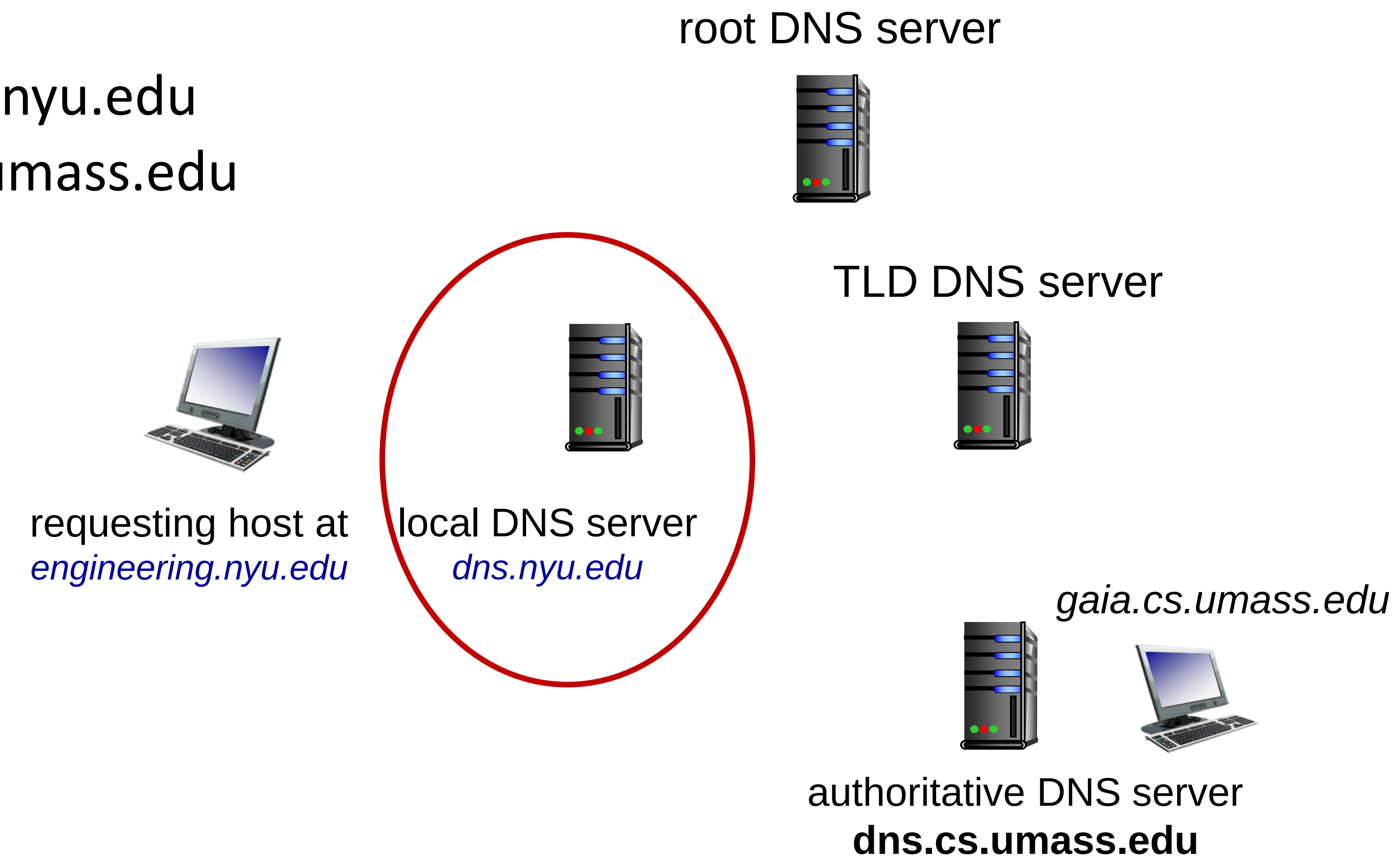


authoritative DNS servers:

- organization's own DNS server(s), providing authoritative hostname to IP mappings for organization's named hosts
- can be maintained by organization or service provider

DNS name resolution

Example: host at engineering.nyu.edu wants IP address for gaia.cs.umass.edu

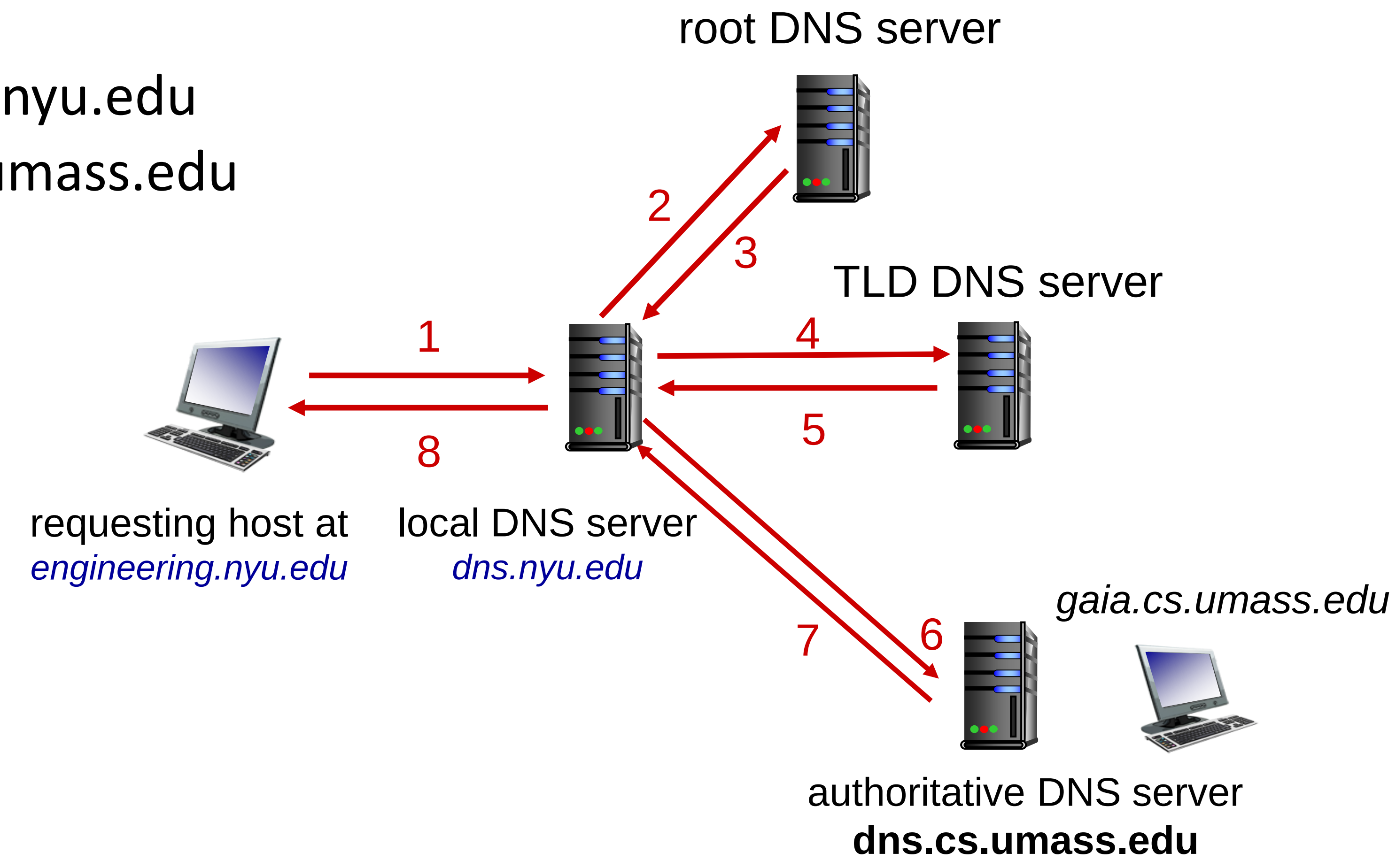


Local DNS name servers

- when host makes DNS query, it is sent to its *local* DNS server
  - Local DNS server returns reply, answering:
    - from its local cache of recent name-to-address translation pairs
    - forwarding request into DNS hierarchy for resolution if not in cache
  - each ISP has local DNS name server; to find yours:
    - MacOS: `% scutil --dns`
    - Windows: `>ipconfig /all`
- local DNS server doesn’t strictly belong to hierarchy

DNS name resolution: iterative query

Example: host at engineering.nyu.edu wants IP address for gaia.cs.umass.edu



Iterative query:

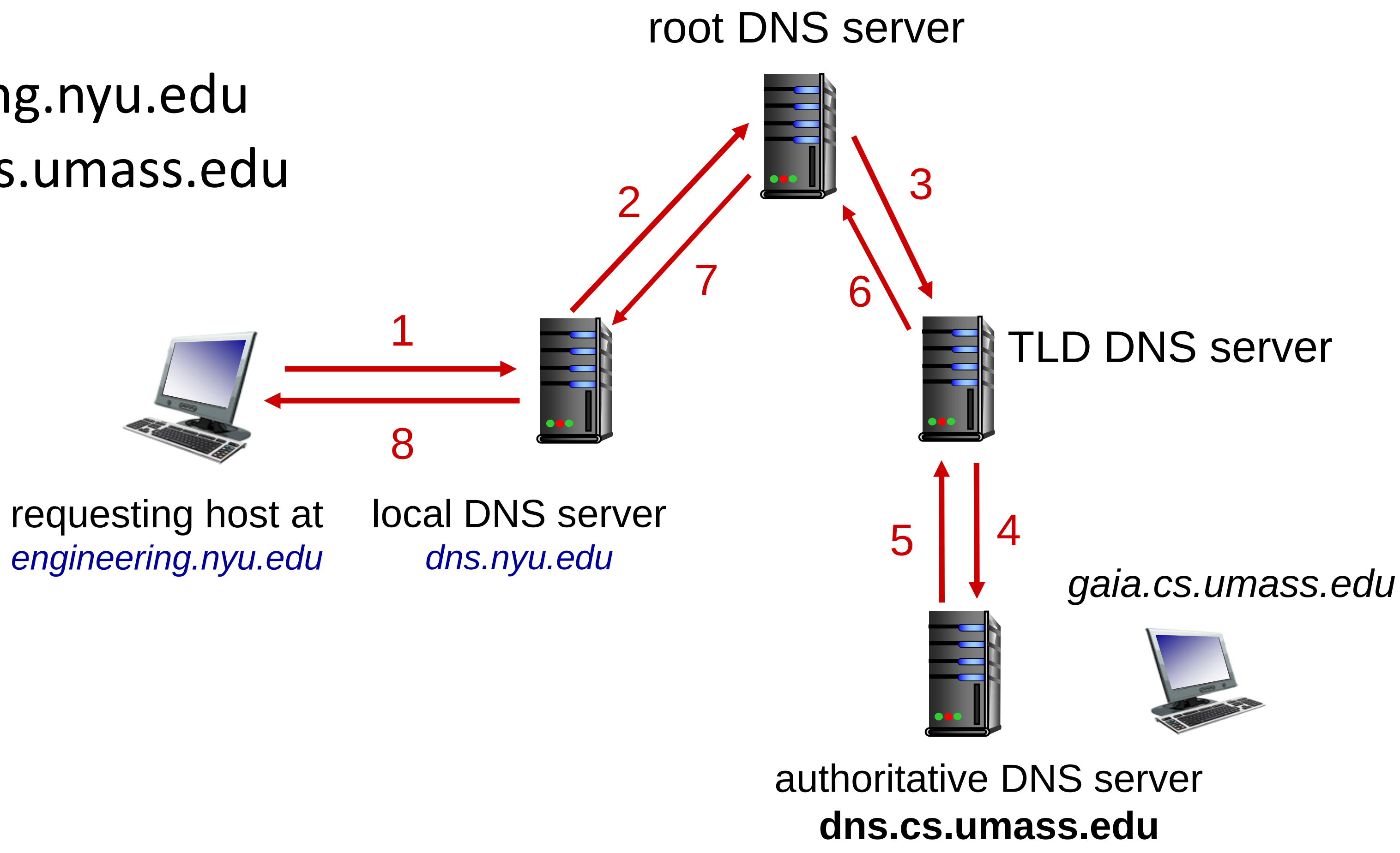
- contacted server replies with name of server to contact
- "I don't know this name, but ask this server"

DNS name resolution: recursive query

Example: host at engineering.nyu.edu wants IP address for gaia.cs.umass.edu

Recursive query:

- puts burden of name resolution on contacted name server
- heavy load at upper levels of hierarchy?



DNS records

DNS: distributed database storing resource records **(RR)**

RR format: (`name, value, type, ttl`)

type=A

- `name` is hostname
- `value` is IP address

type=NS

- `name` is domain (e.g., foo.com)
- `Value` is hostname of authoritative name server for this domain or a name server for a domain that includes the hostname

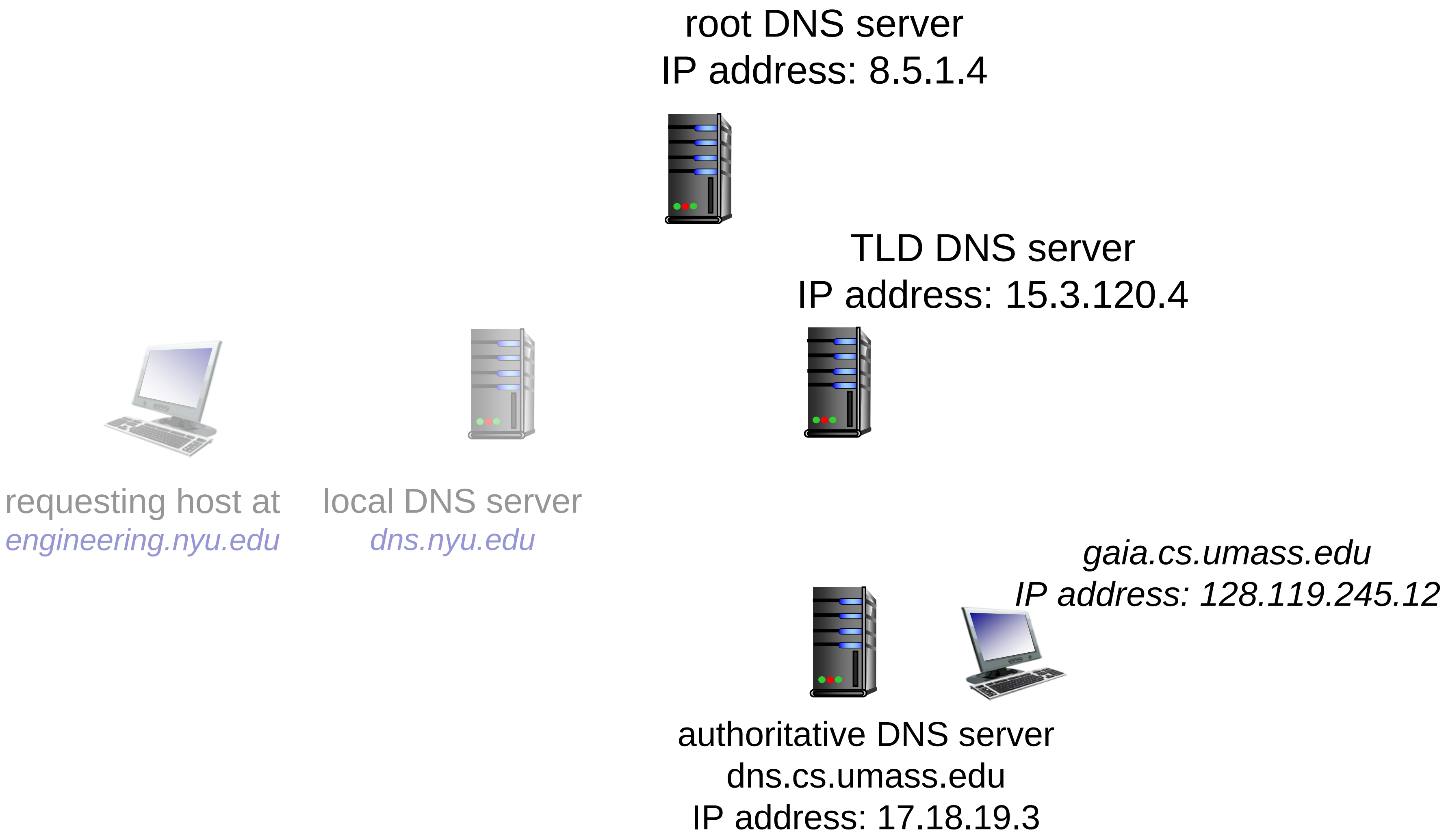
type=CNAME

- `name` is alias name for some “canonical” (the real) name
- www.ibm.com is really servereast.backup2.ibm.com
- `value` is canonical name

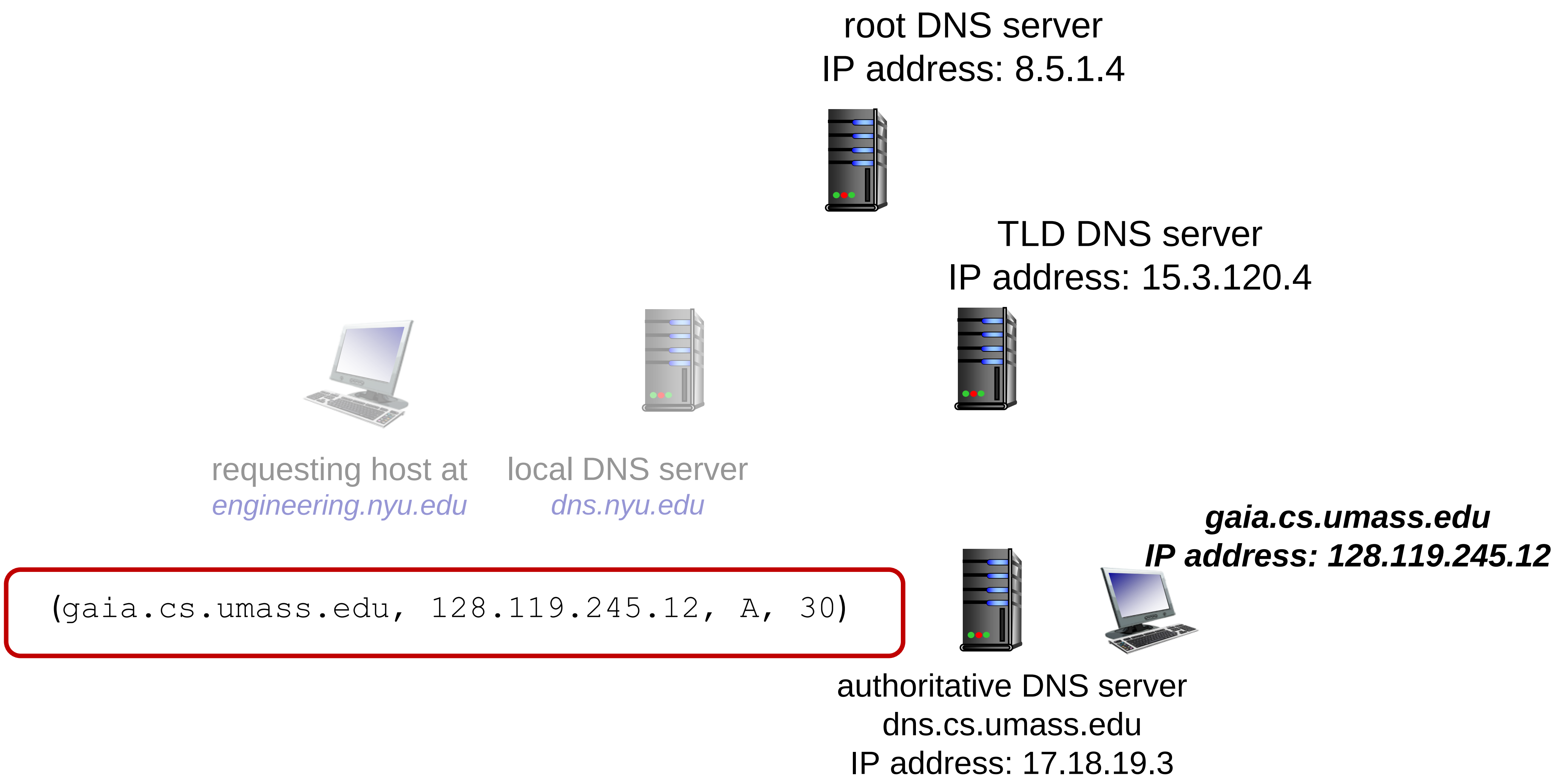
type=MX

- `value` is name of SMTP mail server associated with `name`

Example of records



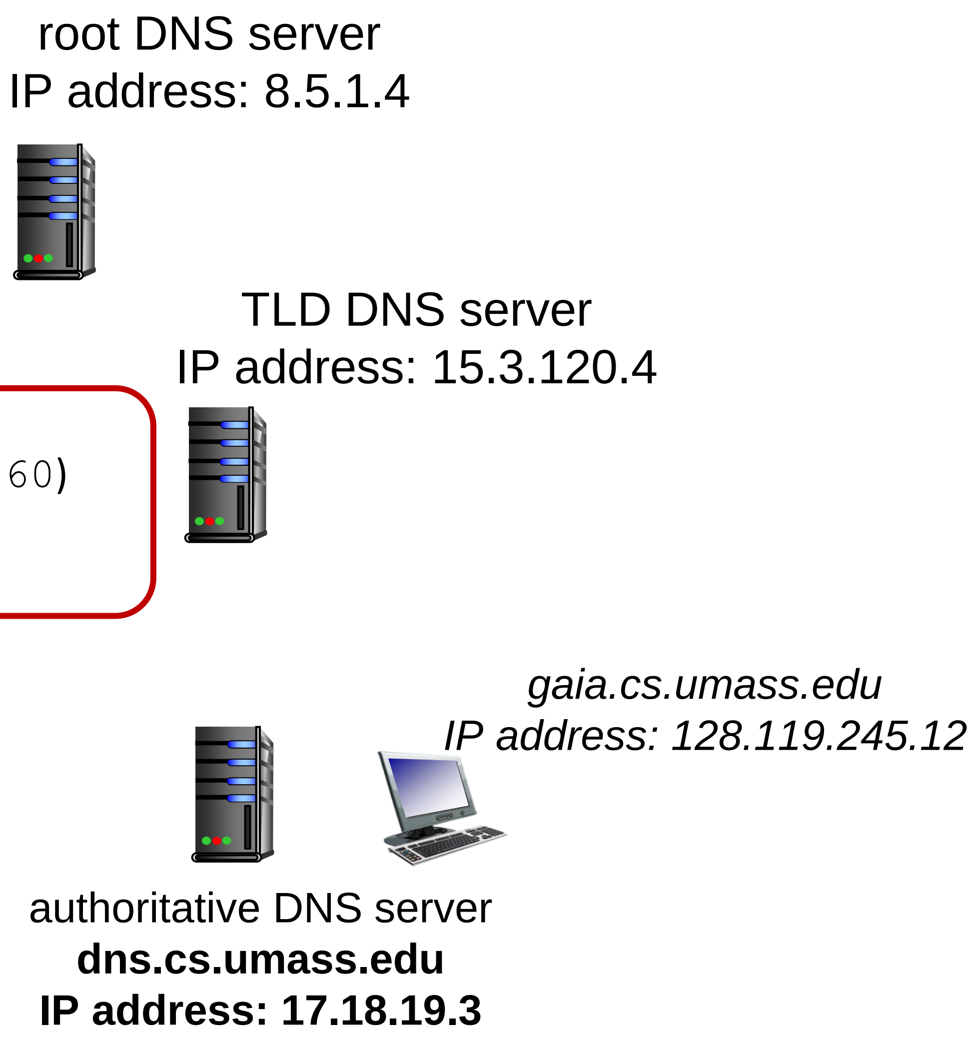
Example of records



Example of records

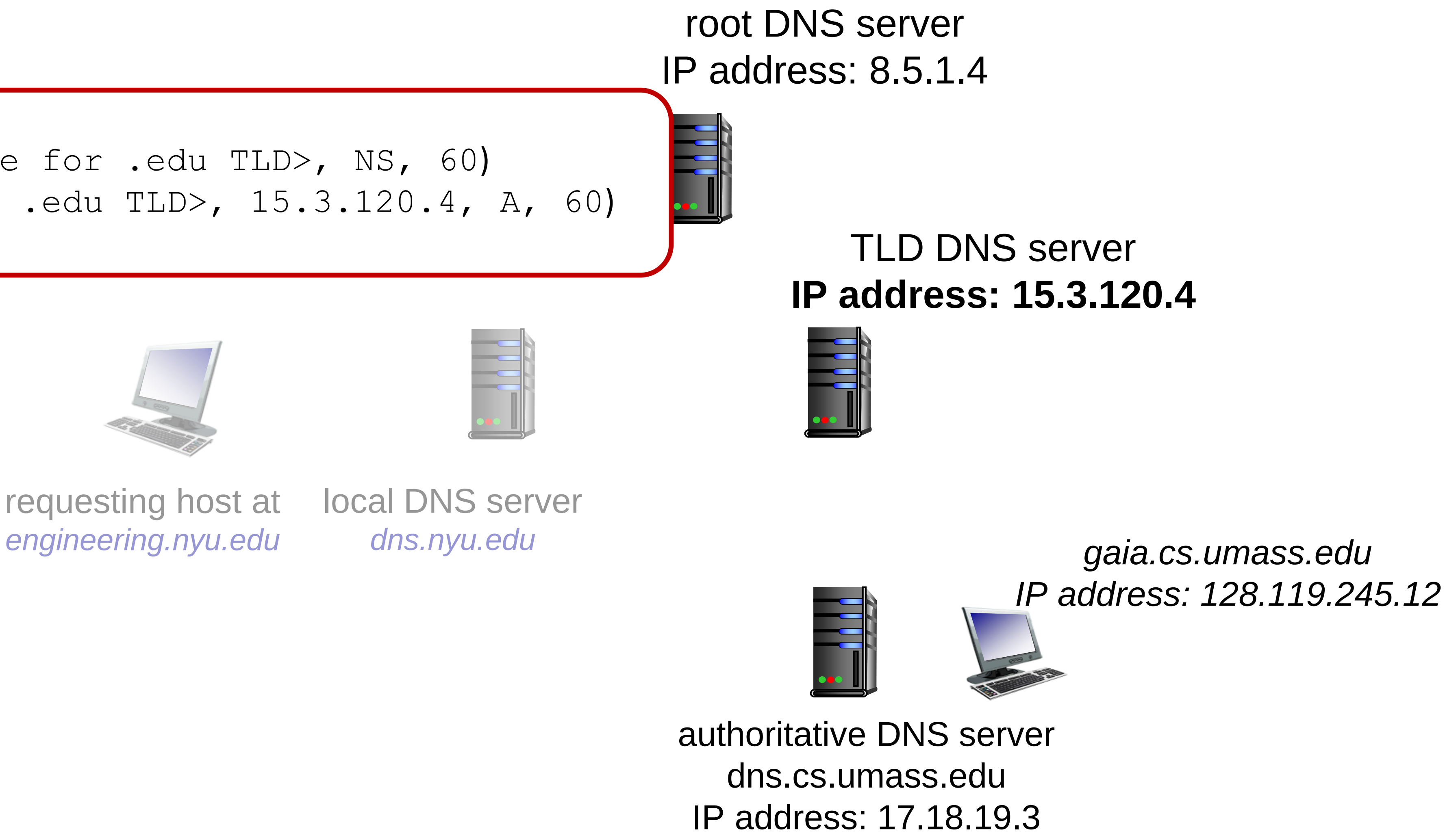
```
(cs.umass.edu, dns.cs.umass.edu, NS, 60)
(dns.cs.umass.edu, 17.18.19.3, A, 60)
```

engineering.nyu.edu *dns.nyu.edu*



Example of records

```
(edu, <hostname for .edu TLD>, NS, 60)
(<hostname for .edu TLD>, 15.3.120.4, A, 60)
```



DNS protocol messages

DNS *query* and *reply* messages, both have same *format:*

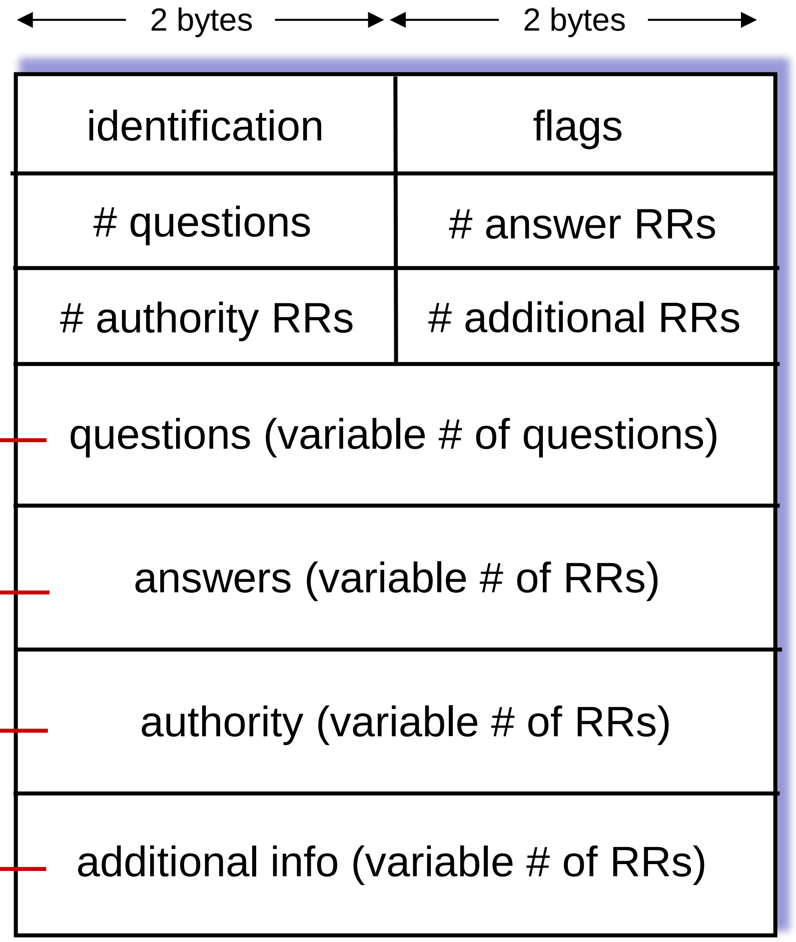
message header:

- identification: 16 bit # for query, reply to query uses same #
- flags:
  - query or reply
  - recursion desired
  - recursion available
  - reply is authoritative

2 bytes	2 bytes
identification	flags
# questions	# answer RRs
# authority RRs	# additional RRs
questions (variable # of questions)	
answers (variable # of RRs)	
authority (variable # of RRs)	
additional info (variable # of RRs)	

DNS protocol messages

DNS *query* and *reply* messages, both have same *format:*



name, type fields for a query

RRs in response to query

records for authoritative servers

additional “ helpful” info that may be used

Caching DNS Information

- once (any) name server learns mapping, it *caches* mapping, and *immediately* returns a cached mapping in response to a query
  - caching improves response time
  - cache entries timeout (disappear) after some time (TTL)
  - TLD servers typically cached in local name servers
- cached entries may be *out-of-date*
  - if named host changes IP address, may not be known Internet-wide until all TTLs expire!
  - *best-effort name-to-address translation!*

Getting your info into the DNS

example: new startup "Network Utopia"

- register name networkuptopia.com at *DNS registrar* (e.g., Network Solutions)
  - provide names, IP addresses of authoritative name server (primary and secondary)
  - registrar inserts NS, A RRs into .com TLD server:

    `(networkutopia.com, dns1.networkutopia.com, NS)`

    `(dns1.networkutopia.com, 212.212.212.1, A)`
- create authoritative server locally with IP address `212.212.212.1`
  - type A record for www.networkuptopia.com
  - type MX record for networkutopia.com

DNS security

DDoS attacks

- bombard root servers with traffic
  - not successful to date
  - traffic filtering
  - local DNS servers cache IPs of TLD servers, allowing root server bypass
- bombard TLD servers
  - potentially more dangerous

Spoofing attacks

- intercept DNS queries, returning bogus replies
  - DNS cache poisoning
  - RFC 4033: DNSSEC authentication services

DNS: services, structure

DNS services:

- hostname-to-IP-address translation
- host aliasing
  - canonical, alias names
- mail server aliasing
- load distribution
  - replicated Web servers: many IP addresses correspond to one name

Q: Why not centralize DNS?

- single point of failure
- traffic volume
- distant centralized database
- maintenance

A: doesn't scale!

- Comcast DNS servers alone: 600B DNS queries/day
- Akamai DNS servers alone: 2.2T DNS queries/day

Final thoughts about the DNS

humongous distributed database:

- ~ billion records, each simple

handles many *trillions* of queries/day:

- *many* more reads than writes
- *performance matters:* almost every Internet transaction interacts with DNS - msecs count!

organizationally, physically decentralized:

- millions of different organizations responsible for their records

Needs to be reliable and secure

What you need to know about DNS

DNS is a very important protocol so almost everything we have discussed is important to know. This includes but is not limited to

- The purpose of DNS and the details of how it operates
- How DNS name servers are structured
- Various DNS records and what they are used for
- The difference between local DNS name servers and other name servers and how they interact
- Various ways for a DNS query to be resolved.
- Being able to fill out details the steps in resolving DNS queries (including the order in which they happen) and DNS records at various name servers
- …

Naming and Addressing

If host A wants to communicate with host B

- A and B each should have an interface connecting them to their LAN.
- Each interface should hav ess.
- To send data to B, A should c esses:
  - source IP address: IP_A
  - destination IP address: IP_B
  - source MAC address: MAC_A
  - destination MAC address: MAC_B <u>if they are in the same LAN</u>. Otherwise, MAC_R, where R is the interface of the gateway router in A's LAN.

How does A get an IP address ?? **DHCP**

How does A find B's (or R's) IP address?
B's IP address: **DNS**, R's IP address: **DHCP**

Comes with the interface

Through ARP. But has to know B's (or R's) IP address.

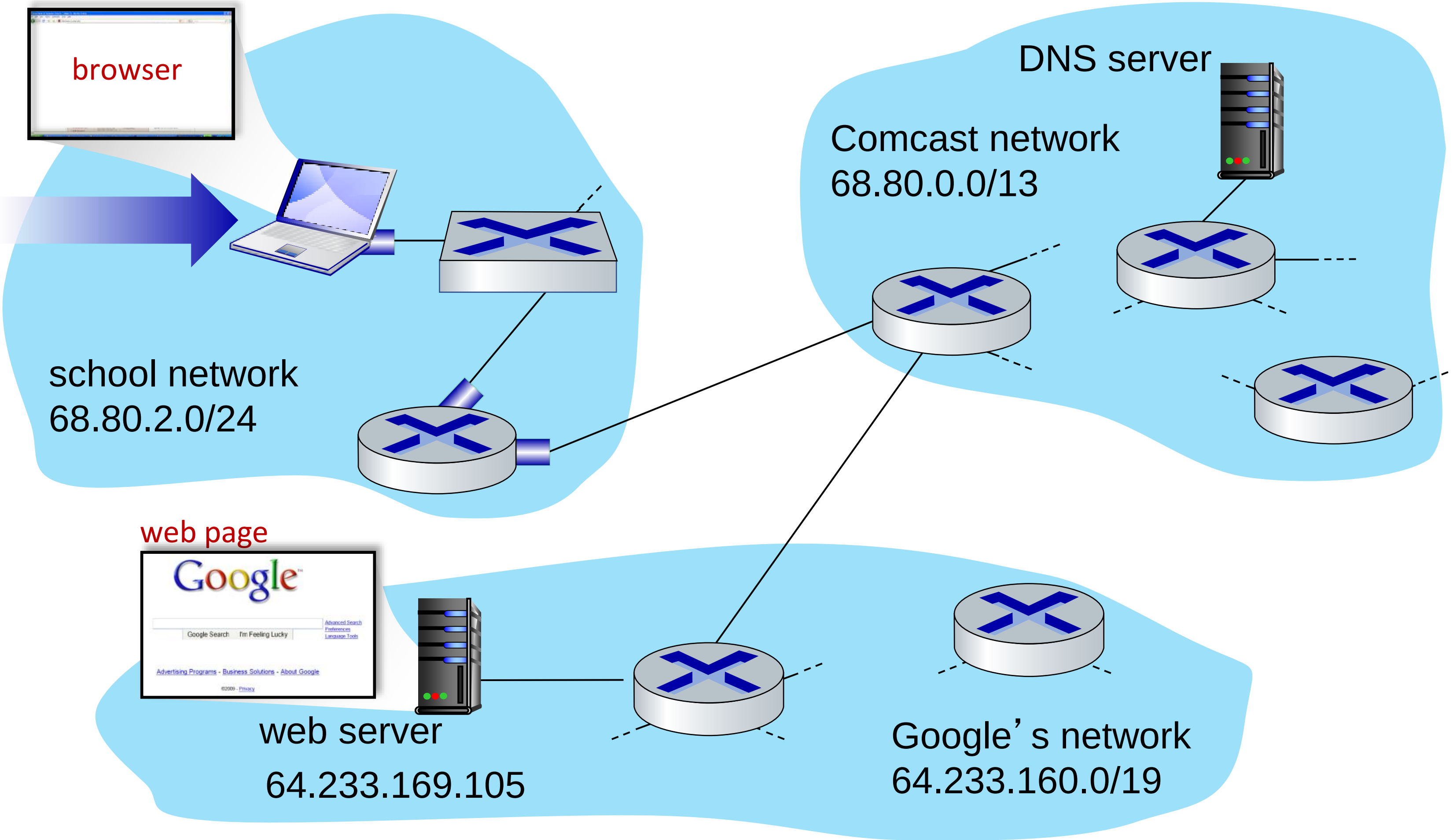
Naming and Addressing: roadmap

- DHCP
  - Helps A find an IP address
  - Helps A find the IP address of the gateway router
- <u>DNS</u>
  - Helps A find B’s IP address

Bringing it all together

- We have covered (almost) all the main layers of a network stack!
- We have also covered important protocols that help with naming and addressing.
- When you come to campus, open your laptop, connect to the network, and type “www.google.com” in the browser…
- All of these protocols have to work together under the hood before you can actually see google’s landing page ☺
- Let’s walk through a (very important) example

A day in the life: scenario

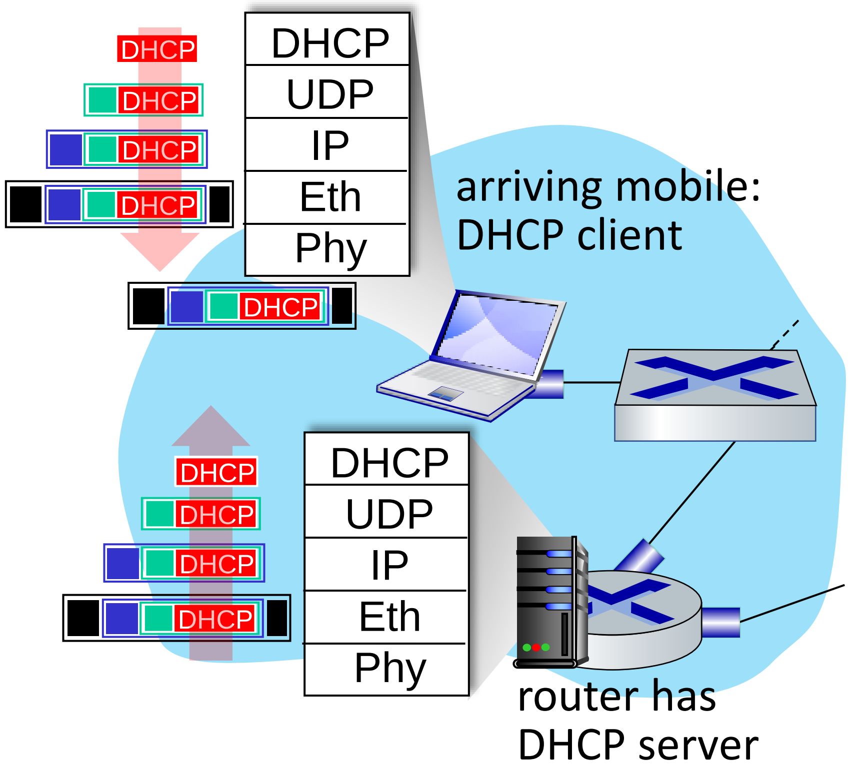


scenario:

- arriving mobile client attaches to network …
- requests web page: www.google.com

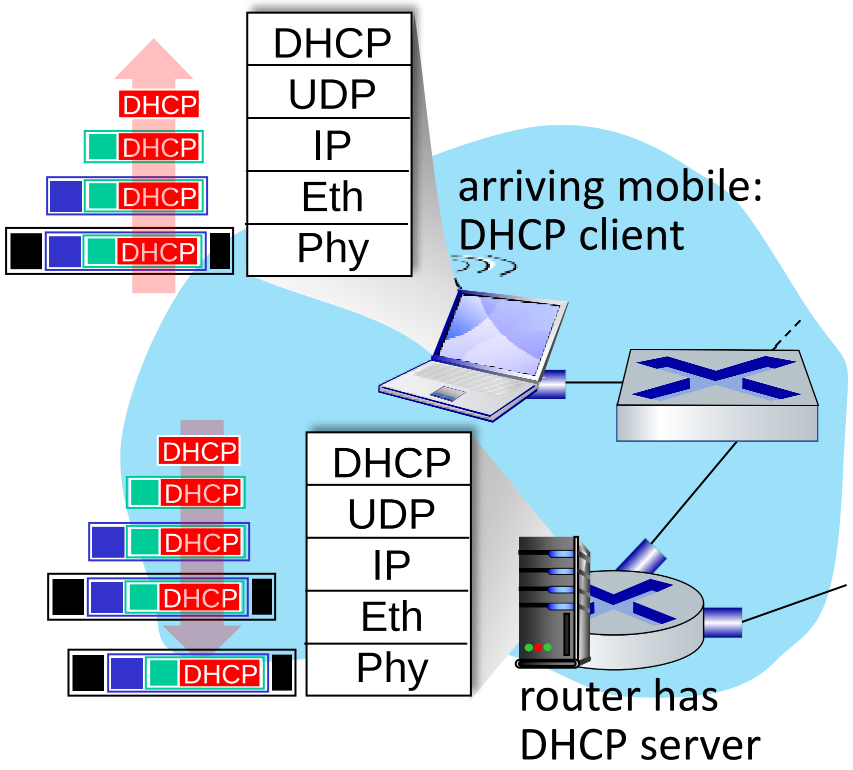
Sounds simple!

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 de-muxed to IP de-muxed, UDP de-muxed to DHCP

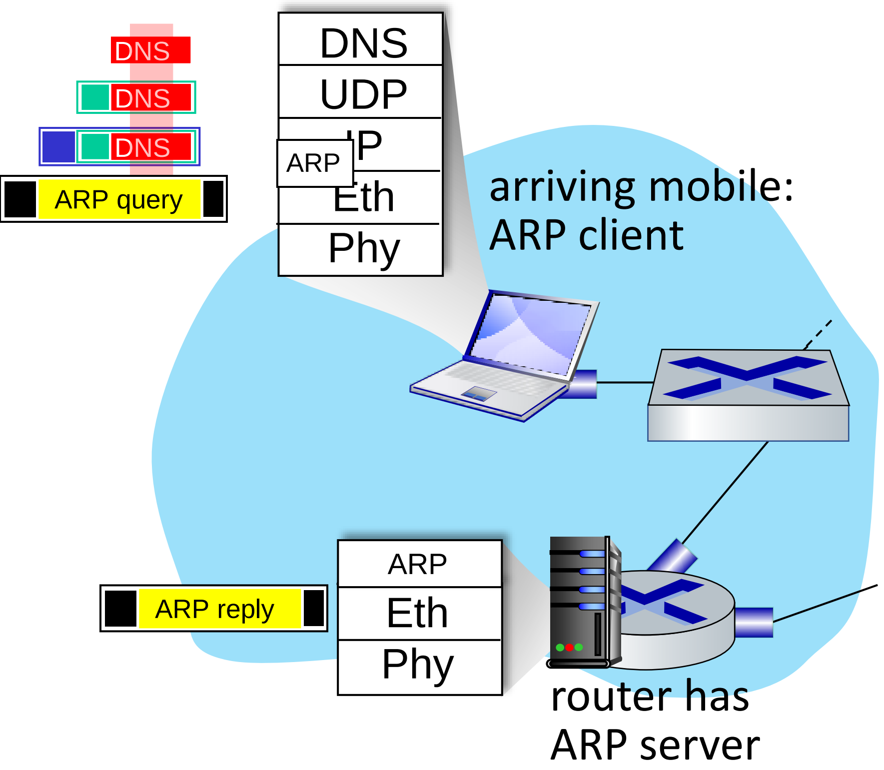
A day in the life: connecting to the Internet



- 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

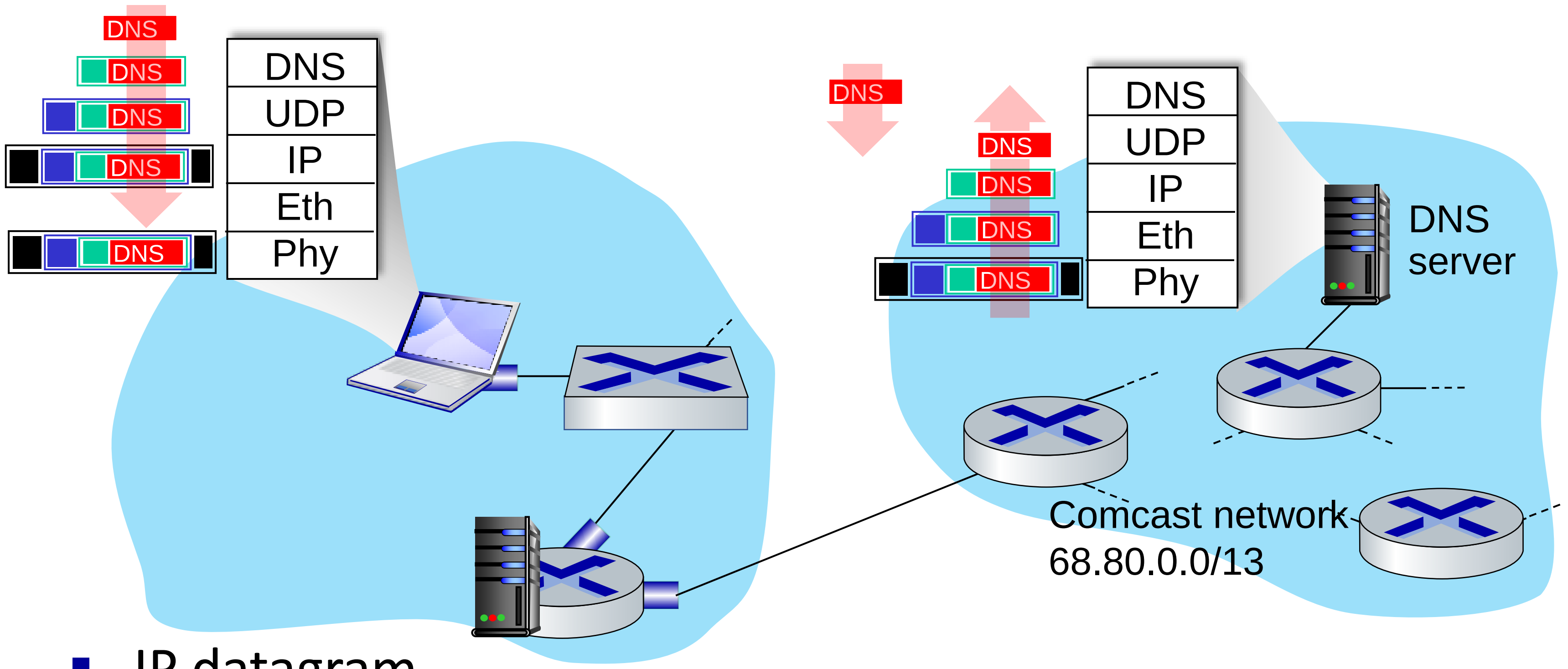
Client now has IP address, knows name & addr of DNS server, IP address of its first-hop router

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



- 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

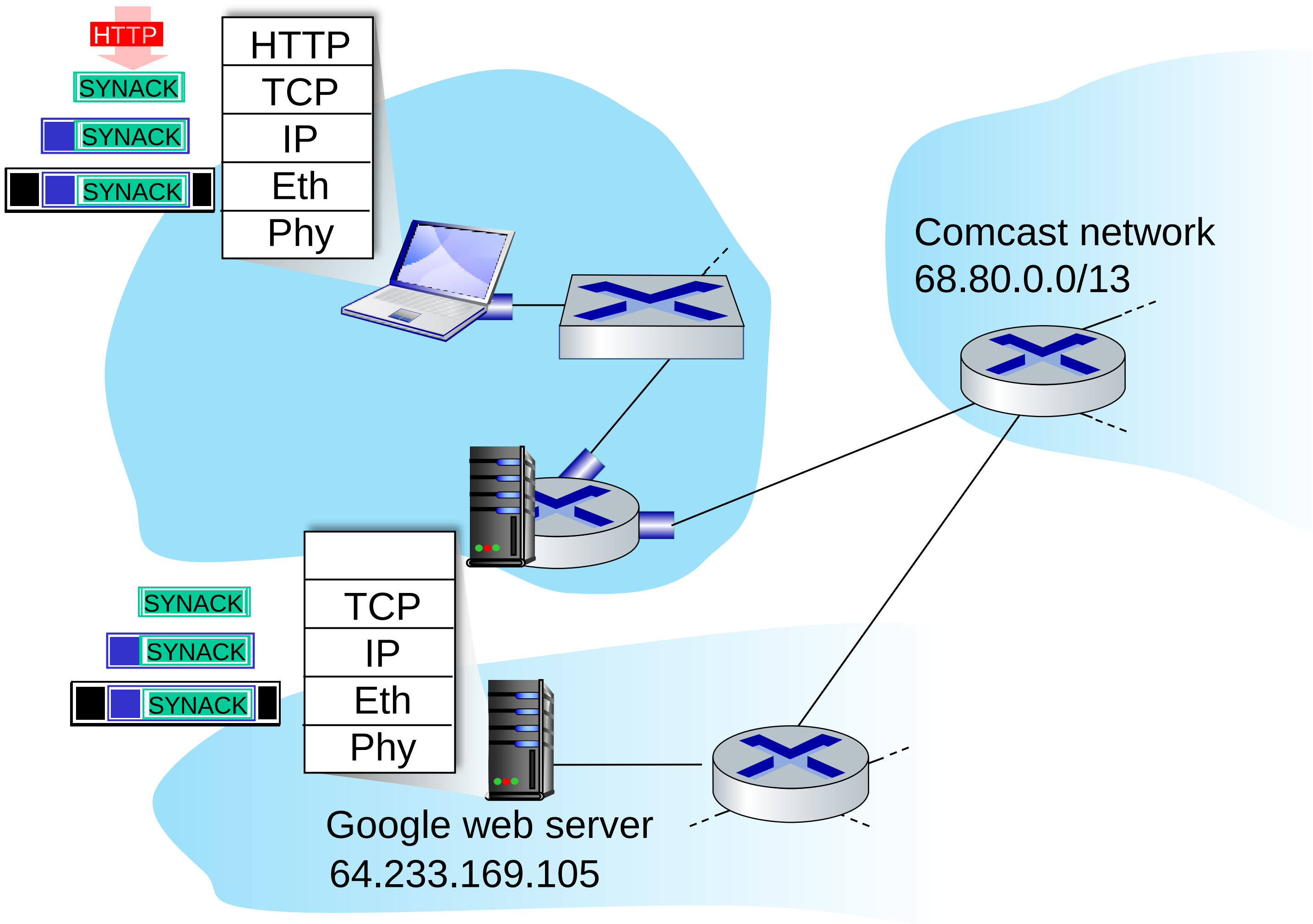


- de-muxed to DNS
- DNS replies to client with IP address of www.google.com

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

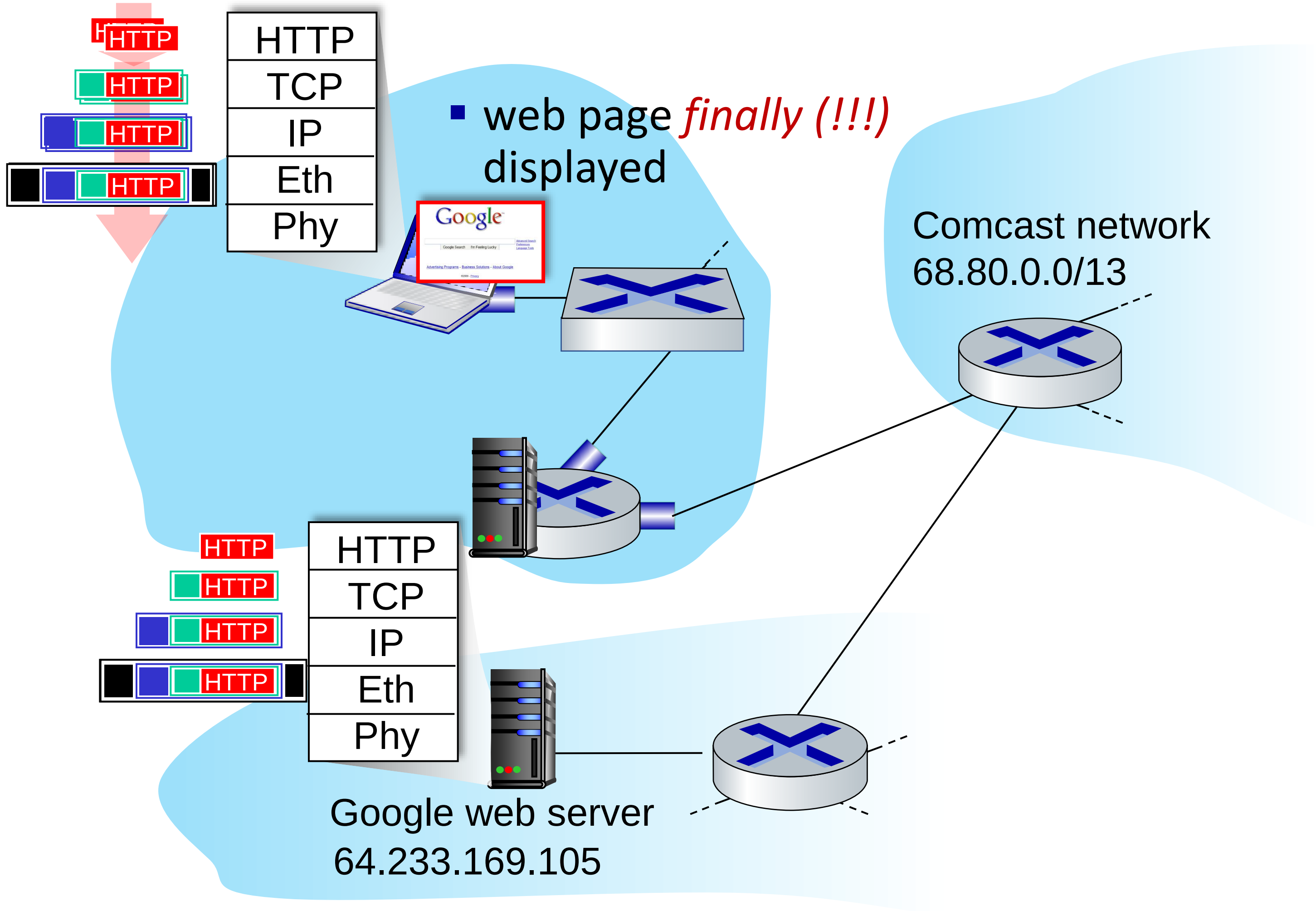
- IP datagram forwarded from campus network into Comcast network, routed (tables created by RIP, OSPF, IS-IS and/or BGP routing protocols) to DNS server

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 TCP 3-way handshake) inter-domain routed to web server
- web server responds with TCP SYNACK (step 2 in TCP 3-way handshake)
- TCP connection established!

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

What you need to know

- This example brings together a lot of what we have talked about in this course
- Make sure you are comfortable with the individual steps that are happening, and their order.
- E.g., given a similar scenario, you should be able to work out what happens under the hood from when a client arrives in a network and starts using some of the network applications we have talked about.