NTNU | Kunnskap for en bedre verden

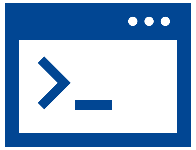
Teknostart – Week 37

Networking III

Routing and DNS

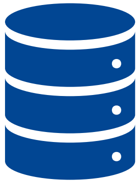
Goals

Recognize the role of routing in networking



Use ip route for managing routes

Retrieve basic DNS information



Deploy simple network services

Recap of Preparation Material

Readings

Routing and DNS

Web servers

Videos

Routing and DNS

Docker compose (optional)

DNS: Domain Name System

People: many identifiers
- SSN, name, passport #

Internet hosts, routers:
- IP address (32 bit) - used for addressing datagrams
- "name", e.g., cs.umass.edu - used by humans

Q: how to map between IP address and name, and vice versa ?

Domain Name System (DNS)

- *Distributed database* implemented in hierarchy of many *name servers*
- *Application-layer protocol:* hosts, DNS servers communicate to *resolve* names (address/name translation)
  - Core Internet function, implemented as application-layer protocol
  - Complexity at network's edge

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

Thinking About the DNS

Humongous distributed database

- ~ billion records, each simple

Handles many *trillions* of queries/day

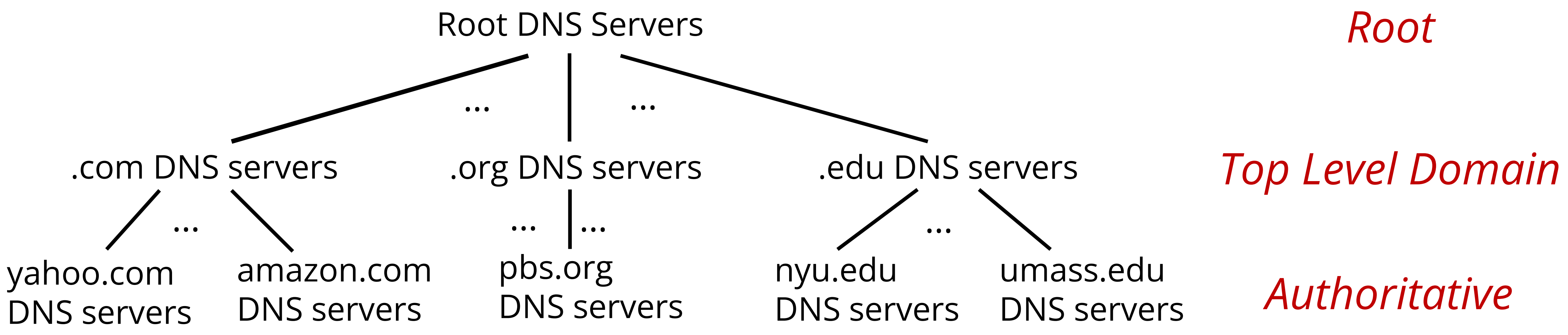
- *Many* more reads than writes
- *Performance matters:* almost every Internet transaction interacts with DNS - msecs count!

Organizationally, physically decentralized

- millions of different organizations responsible for their records

"Bulletproof": reliability, security

DNS - A Distributed, Hierarchical Database

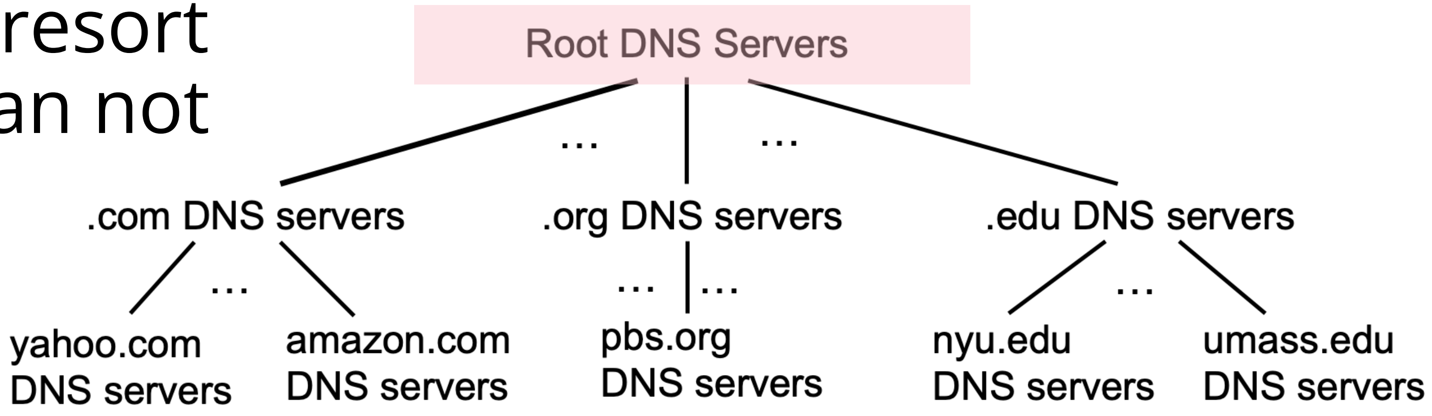


Client wants IP address for www.amazon.com; 1st approximation

- 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

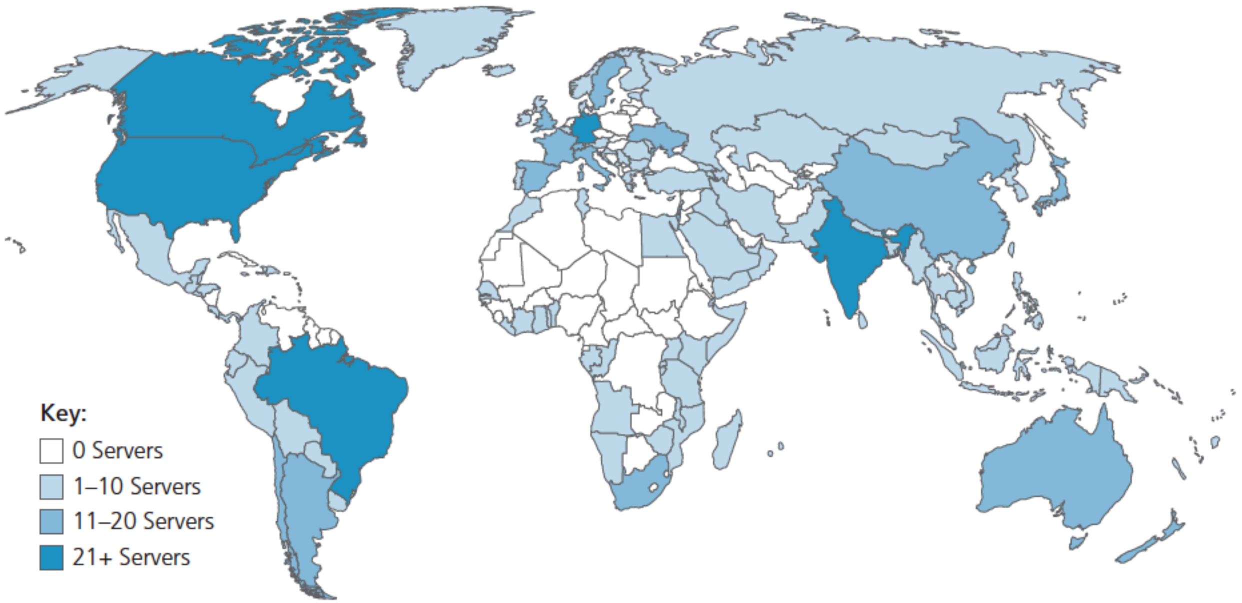
- Official, contact-of-last-resort by name servers that can not resolve name



DNS – Root Name Servers

- Official, contact-of-last-resort by name servers that can not resolve name
- *Incredibly important* Internet function
  - Internet couldn't function without it!
  - DNSSEC – provides security (authentication, message integrity)
- ICANN (Internet Corporation for Assigned Names and Numbers) manages root DNS domain

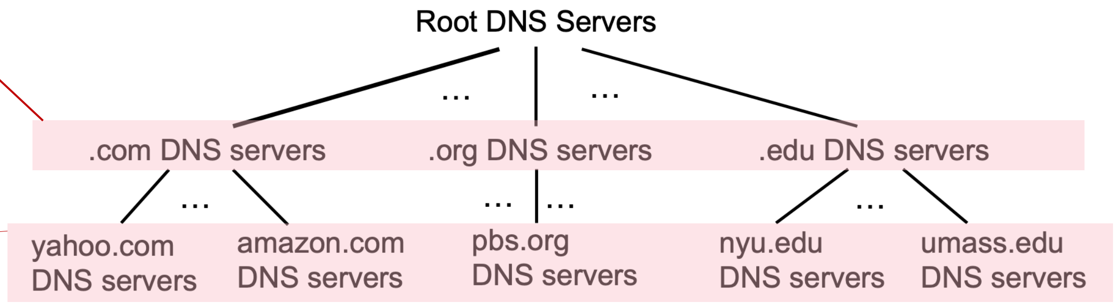
13 logical root name "servers" worldwide each "server" replicated many times (~200 servers in US)



Top-Level Domain and Authoritative Servers

Top-Level Domain (TLD) servers:

- Responsible for .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

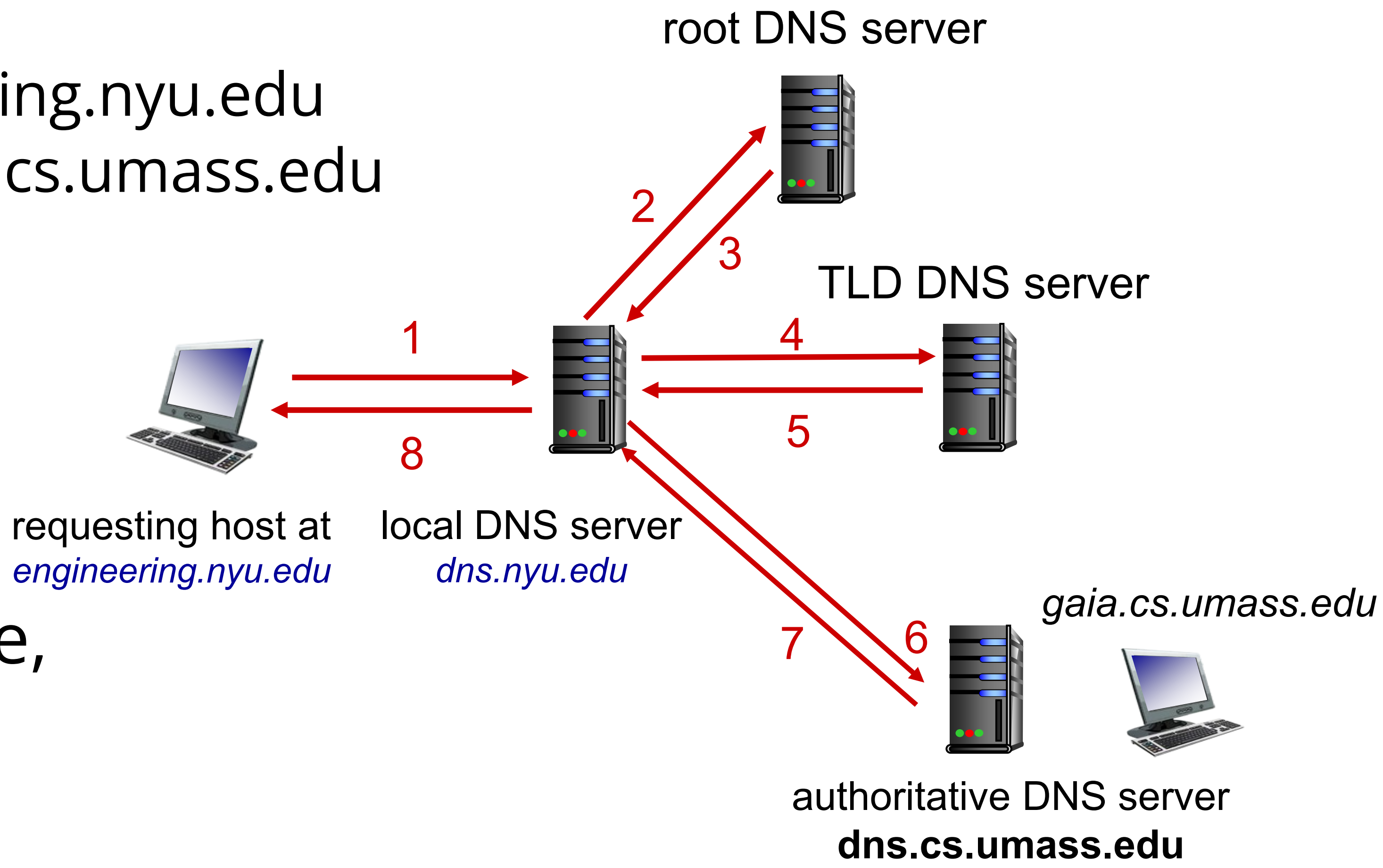
Local DNS Name Servers

- 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 (possibly out of date!)
    - Forwarding request into DNS hierarchy for resolution
  - 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 – Iterated Query

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

Iterated query
- Contacted server replies with name of server to contact
- “I don’t know this name, but ask this server”

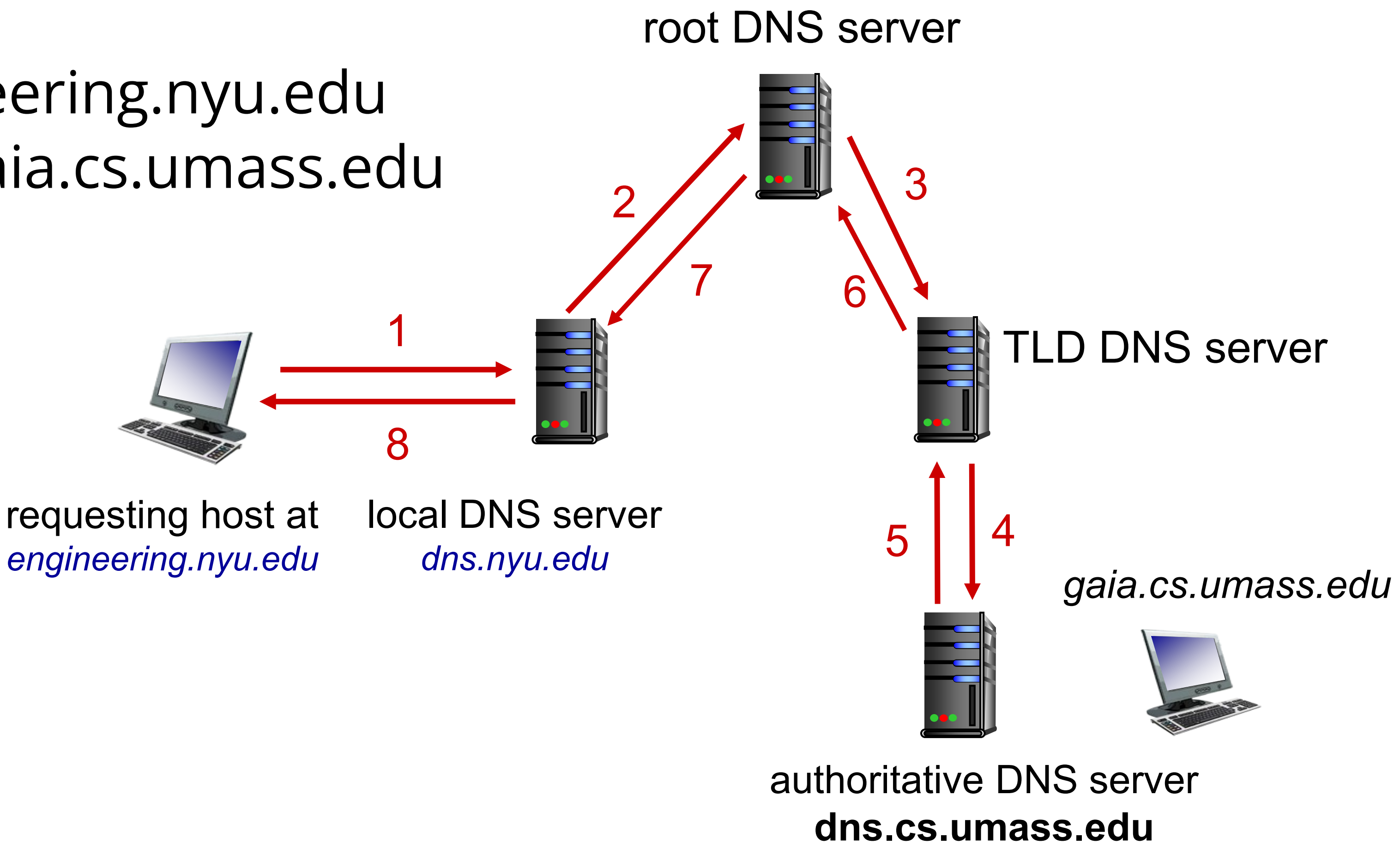


DNS Name Resolution – Recursive Query

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



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!*

IP and DNS – Useful Tools

- Checking your own IP address
  - Private: `ifconfig / ip / ipconfig`
  - Public: https://www.showmyip.com/

- Resolving IP address of a remote target
  - Operating system tools: `nslookup / dig / host`
  - Online tools: https://www.nslookup.io/

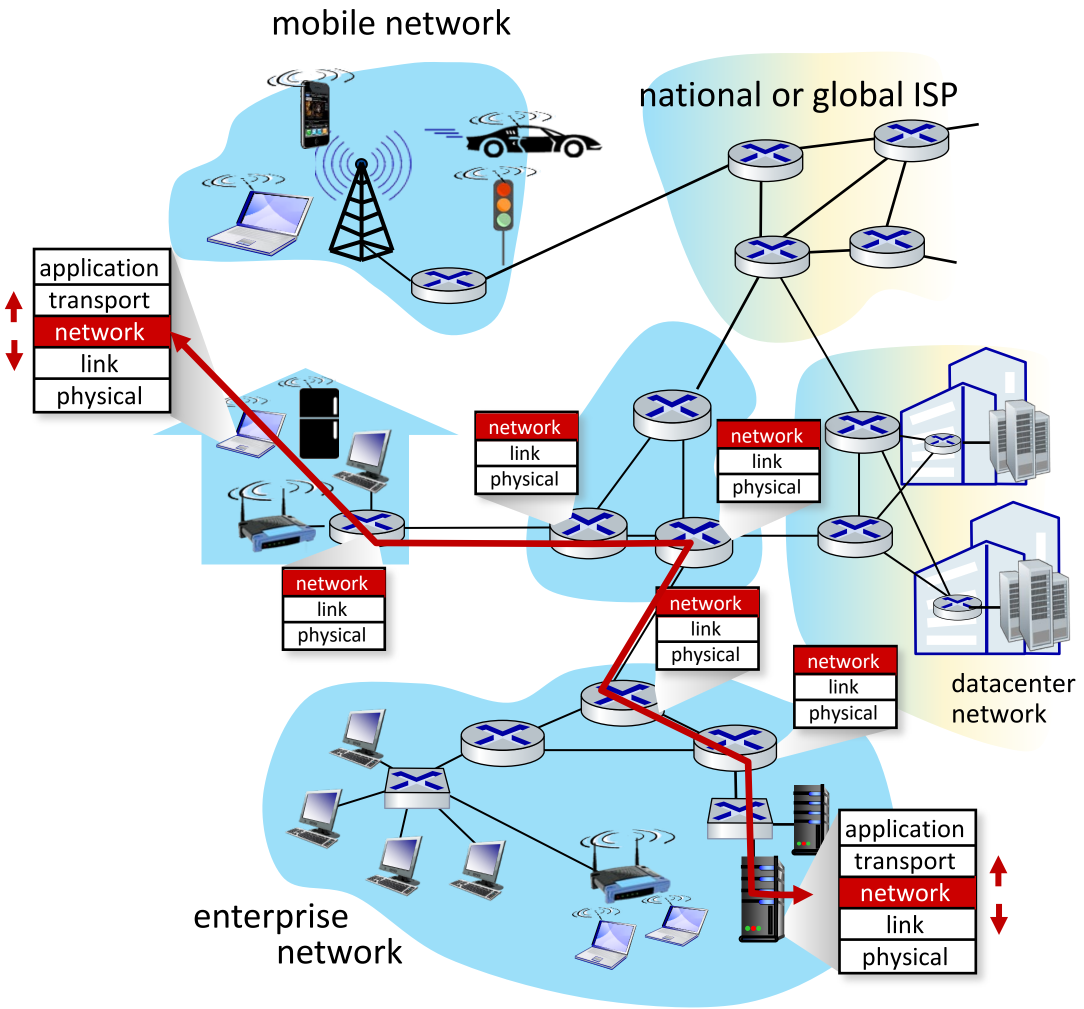
IP and DNS – Exercise

- Find your private IP address and compare with your team members. Do you notice a pattern?
- Find your public IP address and do the same
- When using your local DNS tools, which name server is used? Who owns it?
- Try different DNS servers at nslookup.io – do you notice something when comparing the results for large services like netflix.com?

12:00

Network-Layer Services and Protocols

- Transport segment from sending to receiving host
  - Sender: encapsulates segments into datagrams, passes to link layer
  - Receiver: delivers segments to transport layer protocol
- Network layer protocols in *every Internet device*: hosts, routers
- Routers
  - Examine header fields in all IP datagrams passing through it
  - Move datagrams from input ports to output ports to transfer datagrams along end-end path



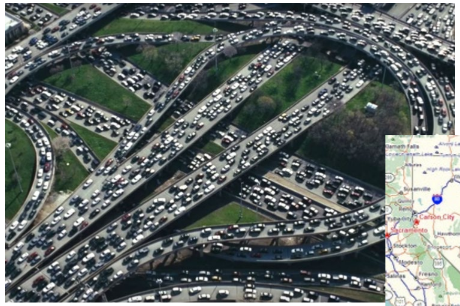
Two Key Network-Layer Functions

Network-layer functions

- *Forwarding:* move packets from a router’s input link to appropriate router output link
- *Routing:* determine route taken by packets from source to destination
  - *Routing algorithms*

Analogy: taking a trip

- *Forwarding:* process of getting through single interchange
- *Routing:* process of planning trip from source to destination



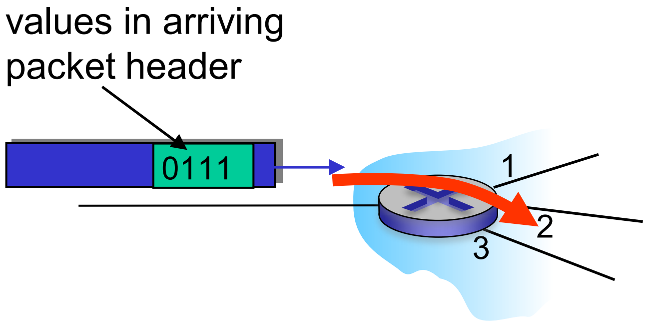
Forwarding

Routing

Network Layer – Data and Control Plane

Data plane

- *Local*, per-router function
- Determines how datagram arriving on router input port is forwarded to router output port

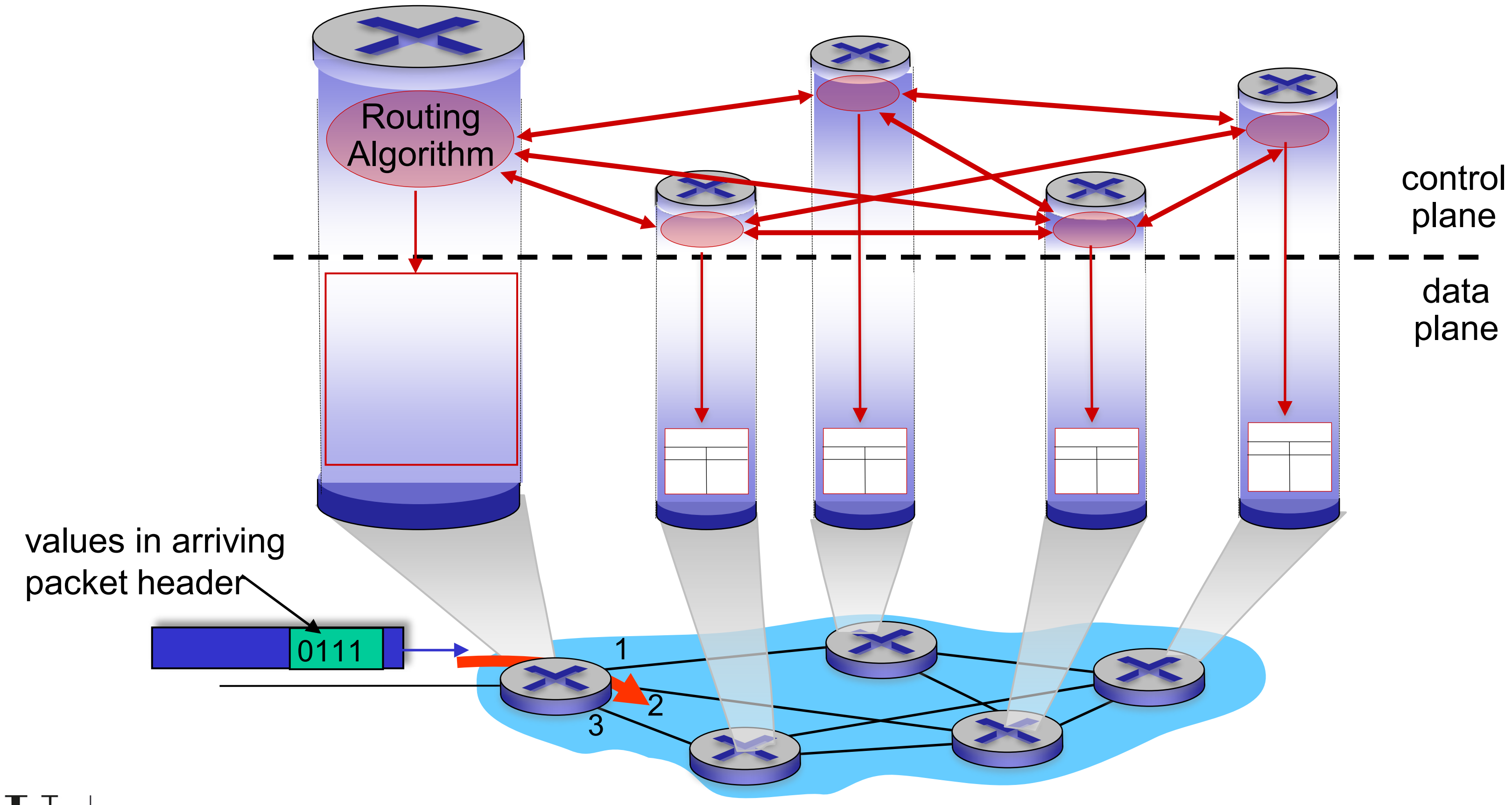


Control plane

- *Network-wide* logic
- Determines how datagram is routed among routers along end-end path from source host to destination host

Per-Router Control Plane

Individual routing algorithm components *in each router* interact in the control plane



Destination-Based Forwarding

forwarding table

Destination Address Range	Link Interface
11001000 00010111 00010000 00000000 through	0
11001000 00010111 00010000 00000100 through 11001000 00010111 00010000 00000111	3
11001000 00010111 00011000 11111111	
11001000 00010111 00011001 00000000 through 11001000 00010111 00011111 11111111	2
otherwise	3

Q: but what happens if ranges don't divide up so nicely?

Longest Prefix Matching

Longest prefix match

When looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Destination Address Range				Link interface
11001000	00010111	00010***	********	0
11001000	00010111	00011000	********	1
11001000	00010111	00011***	********	2
otherwise				3

examples:

11001000 00010111 00010110 10100001 which interface?

11001000 00010111 00011000 10101010 which interface?

Longest Prefix Matching

Longest prefix match

When looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Destination Address Range	Link interface
11001000 00010111 00010*** ********	0
11001000 00010111 00011000 ********	1
11001000 00010111 00011*** ********	2
otherwise	3

match!

examples:

11001000 00010111 00010110 10100001 which interface?

11001000 00010111 00011000 10101010 which interface?

Longest Prefix Matching

Longest prefix match

When looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Destination Address Range	Link interface
11001000 00010111 00010*** ********	0
11001000 00010111 00011000 ********	1
11001000 00010111 00011*** ********	2
otherwise	3

match!

examples:

11001000 00010111 00010110 10100001 which interface?

11001000 00010111 00011000 10101010 which interface?

Longest Prefix Matching

Longest prefix match

When looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Destination Address Range				Link interface
11001000	00010111	00010***	********	0
11001000	00010111	00011000	********	1
11001000	00010111	00011***	********	2
otherwise				3

match!

examples:

11001000	00010111	00010110	10100001	which interface?
11001000	00010111	00011000	10101010	which interface?

IP Prefixes, Subnet Masks, Headers

IP Prefixes

- Example: 10.240.1.0/24
  - Network address with prefix length 24
    - First 24 bits specify network address
    - $\boxed{00001010 . 11110000 . 00000001}$. 00000000

    Prefix
  - Allows routers to determine **interface towards next hop** on the way to a packet's destination in an **aggregated** way
    - Longest prefix match: compare destination IP of packet against **all** entries, return the one with the **longest** match
    - No need to create forwarding table entries for each IP address

Subnet Masks

- 32-bit number used to extract network part from IP address
- /24 mask = 11111111 . 11111111 . 11111111 . 00000000 = 255.255.255.0
- Applying mask to any address from 10.240.1.0/24 yields network
  - 10.240.1.23 → `00001010.11110000.00000001.00010111`
  - Bit-wise AND `11111111.11111111.11111111.00000000`
  - 10.240.1.0 ← `00001010.11110000.00000001.00000000`
- Used by hosts to determine reachability of destinations
  - Same subnet → reachable locally → send directly via layer 2
  - Other subnet → send to gateway (typically a router)

IP Prefixes, Subnet Masks, Headers

- Prefix: substring of specific length
  - Example: 00001010 11110000 00000001
  - Used by routers to perform longest prefix matching

Datagram

IP header
    src IP: 10.0.0.1
    dst IP: 10.240.1.23 =
    00001010 . 11110000 . 00000001 . 00010111
Ethernet header
    src / dst MAC address

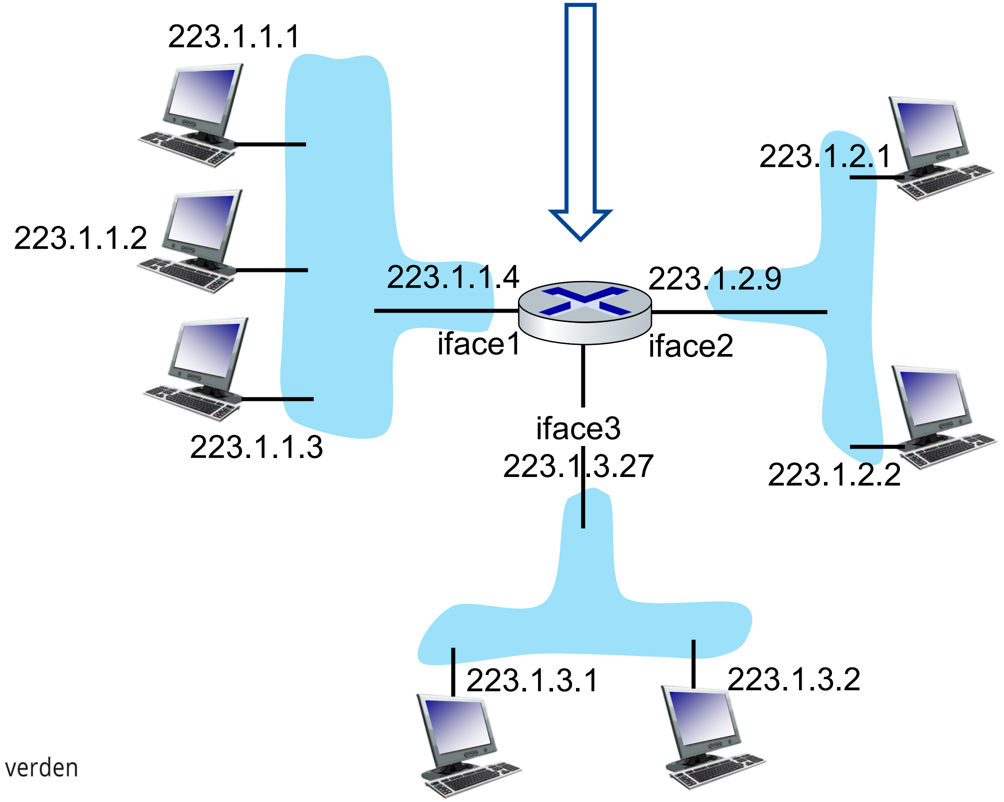
Router with forwarding table entries

```
00001010 11110000 00000001 ******** -> eth0
00001010 11110000 101000** ******** -> eth1
                   ...
```

- Subnet mask: bit mask to extract network part
  - Example: 11111111 . 11111111 . 11111111 . 00000000
  - Used by hosts to decide whether packets' destinations are reachable locally or require gateway involvement

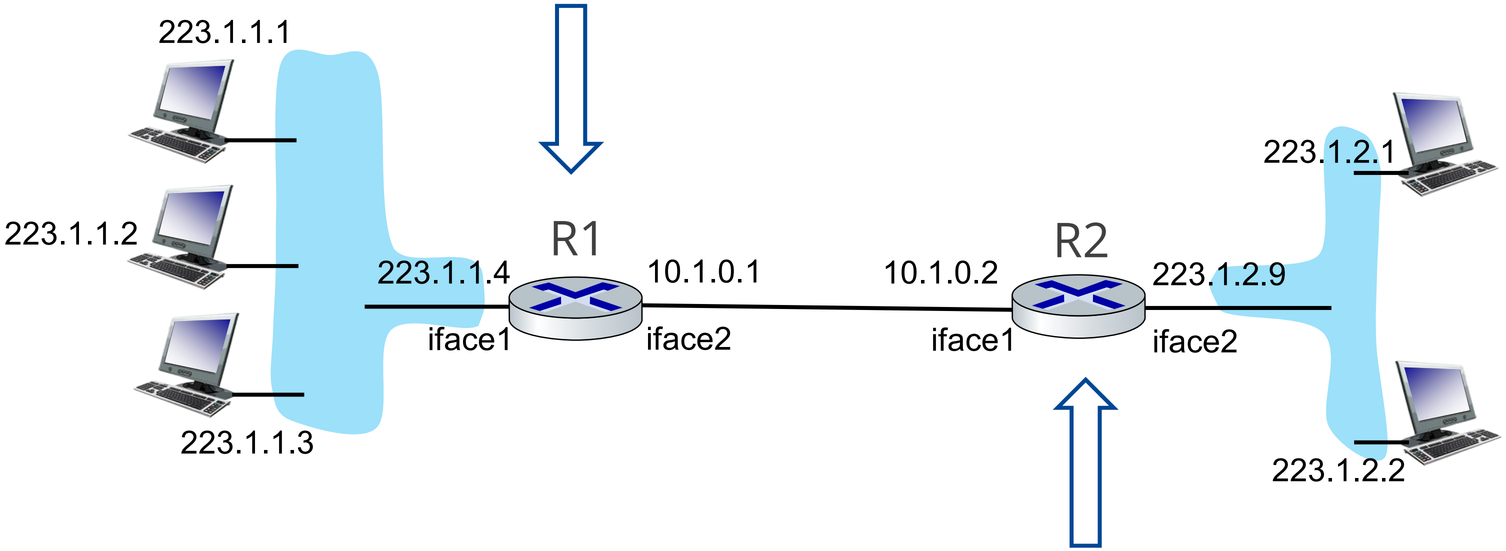
Router Configuration – Examples

Prefix	Next-hop IP	Interface
223.1.1.0/24	- (directly conn.)	1
223.1.2.0/24	- (directly conn.)	2
223.1.3.0/24	- (directly conn.)	3



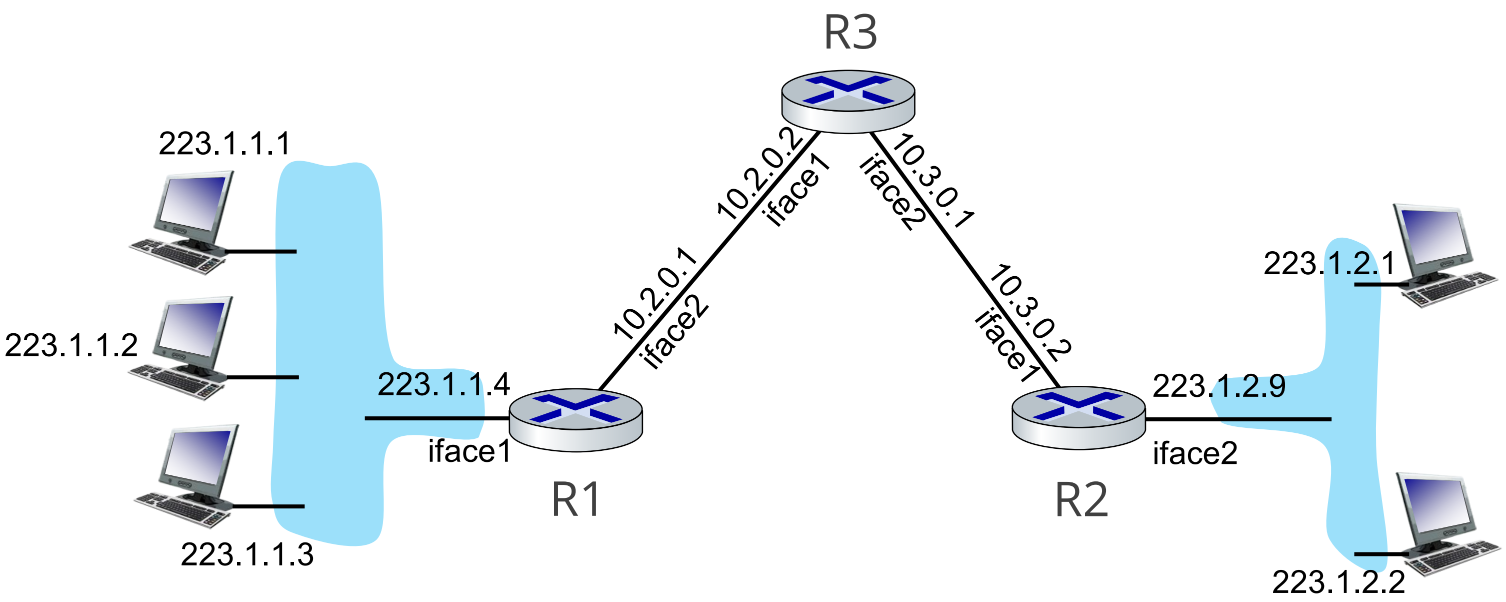
Router Configuration – Examples

Prefix	Next-hop IP	Interface
223.1.1.0/24	- (directly conn.)	1
10.1.0.0/30	- (directly conn.)	2
223.1.2.0/24	10.1.0.2	2



Prefix	Next-hop IP	Interface
223.1.2.0/24	- (directly conn.)	2
10.1.0.0/30	- (directly conn.)	1
223.1.1.0/24	10.1.0.1	1

Router Configuration – Exercise



- R1

Prefix	Next-hop IP	Int.
223.1.1.0/24	-	1
10.2.0.0/30	-	2
223.1.2.0/24	10.2.0.2	2
10.3.0.0/30	10.2.0.2	2

➔ Configure R2 and R3 to allow host-host connectivity

12:00

Lab Program Today

- Navigate complex networks
- Determine packet paths
- Adjust routing
- Modify DNS behavior
- Consolidate networking knowledge