Domain Name System

ECE 50863 – Computer Network Systems

DNS: Mapping between Name and Address

Routers send packets to Internet hosts based on IP addresses (e.g., 216.58.192.196)

Why do we need names? (e.g., www.google.com)

How do we efficiently locate resources?

- DNS: name → IP address

www.google.com -> 216.58.192.196

Challenge

- How do we scale these to the wide area?

Why not centralize DNS?

Single point of failure

Traffic volume

Distant centralized database

Single point of update

Doesn’t scale!

Domain Name System Goals

A wide-area distributed database

Scalability

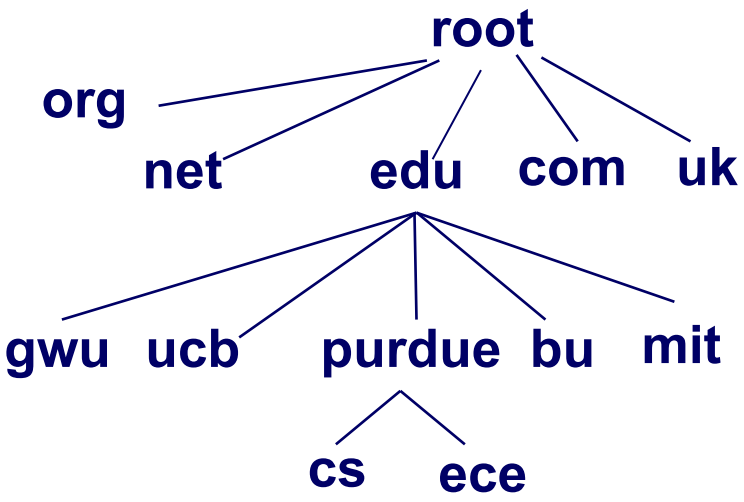
Decentralized maintenance

Robustness

Global scope

- Names mean the same thing everywhere

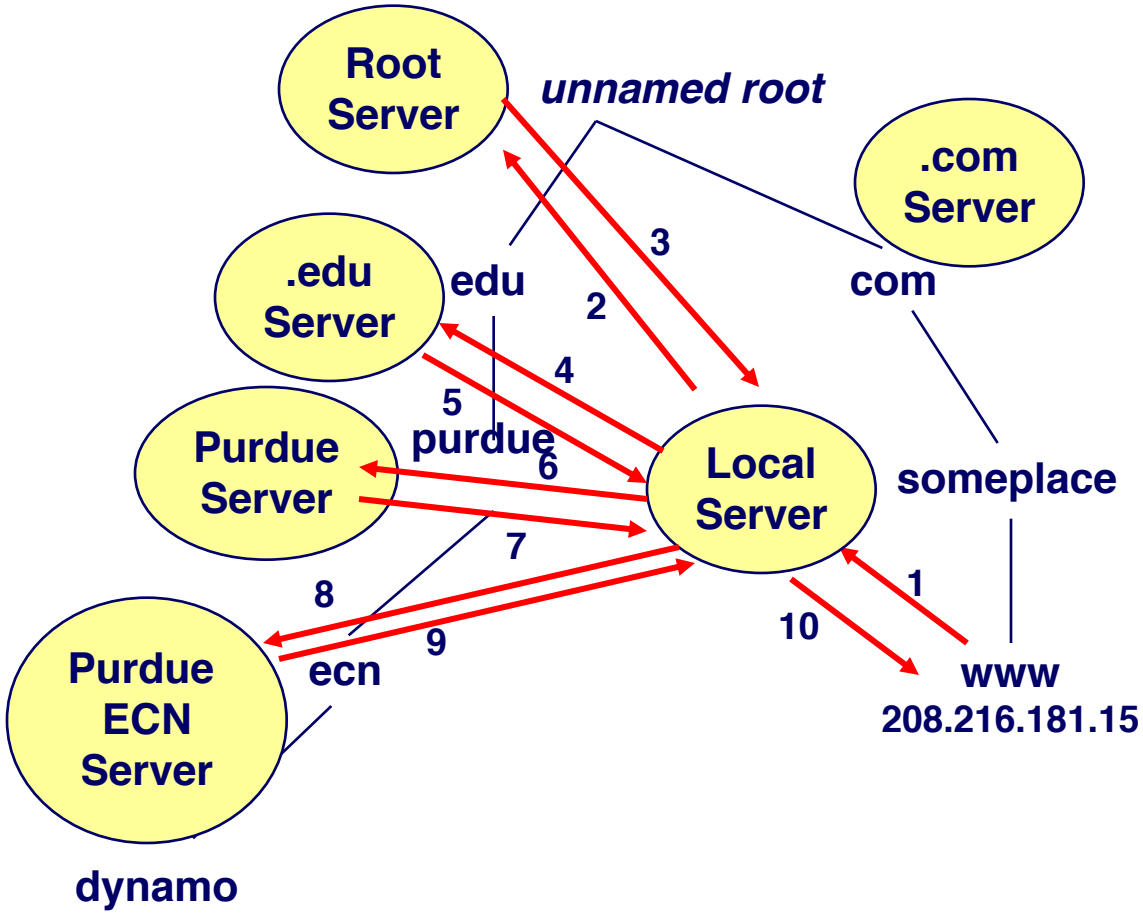
DNS Design: Hierarchy Definitions



- Hostnames reflect hierarchy
  - E.g., www.ece.purdue.edu
- Each node in hierarchy runs one or more “name servers”.
- Leaf level name servers hold name to IP bindings for hosts that end with same suffix
- Name servers at higher levels maintain pointers to name servers of their child zone

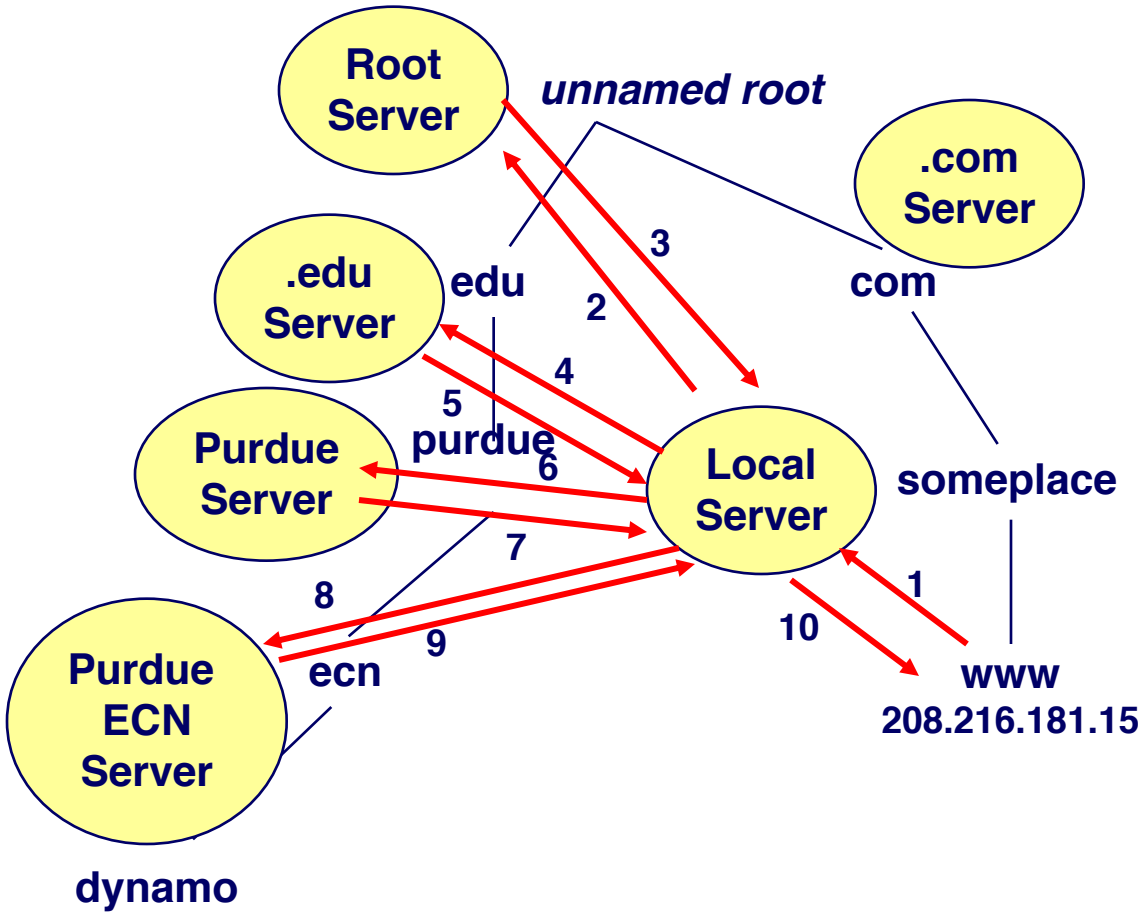
- Purdue ECE’s name server holds bindings for hostnames ending with .ece.purdue.edu
- .edu name server maintains name servers for purdue.edu, .mit.edu etc.
- Top level DNS servers maintained by organization called ICANN
  - Heavy replication to ensure robustness.

Iterative DNS Name Resolution



- Local servers do lookup of distant host names for local hosts
- At each step, server returns name of next server down
- Local server directly queries each successive server

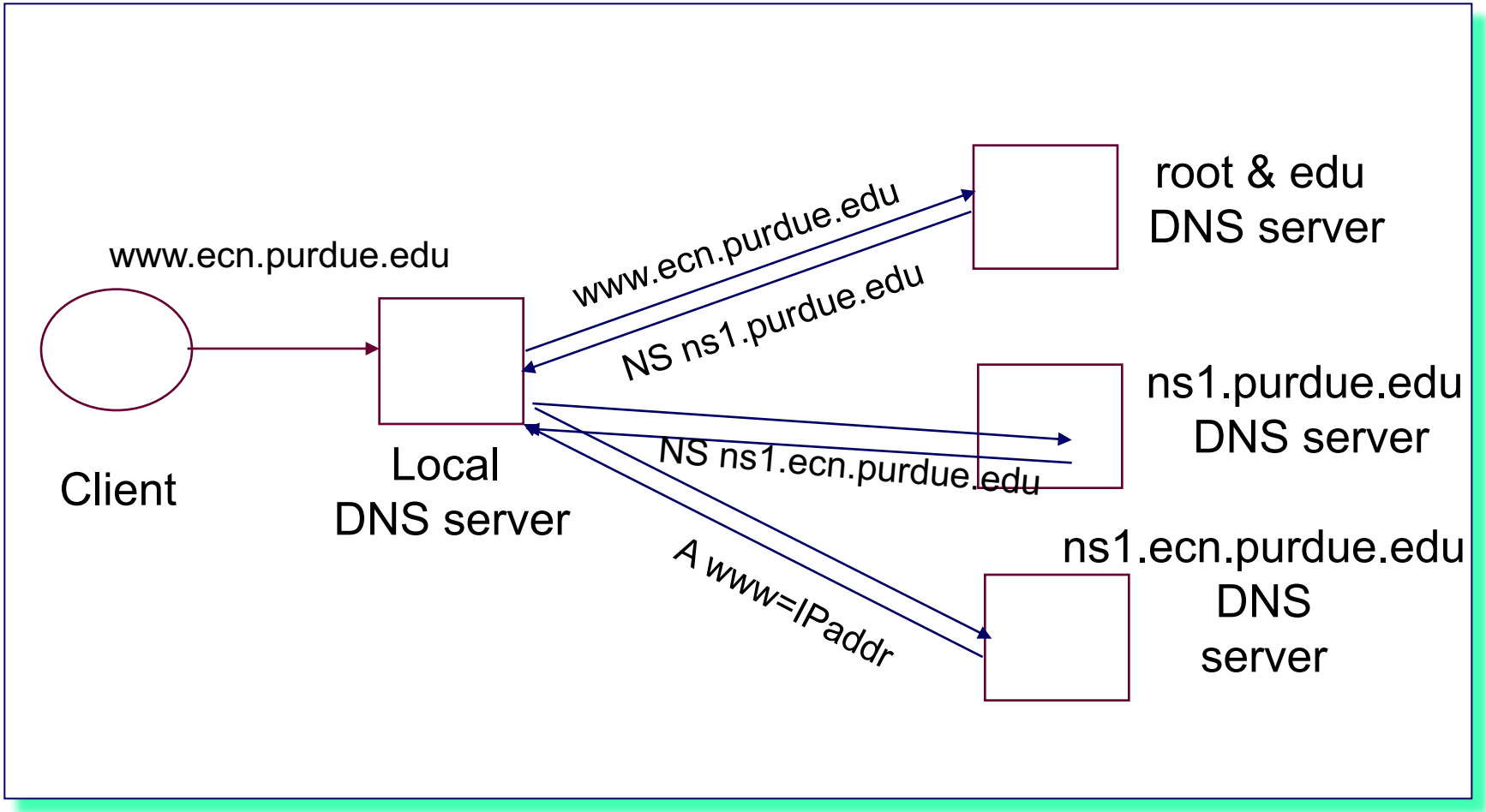
Caching



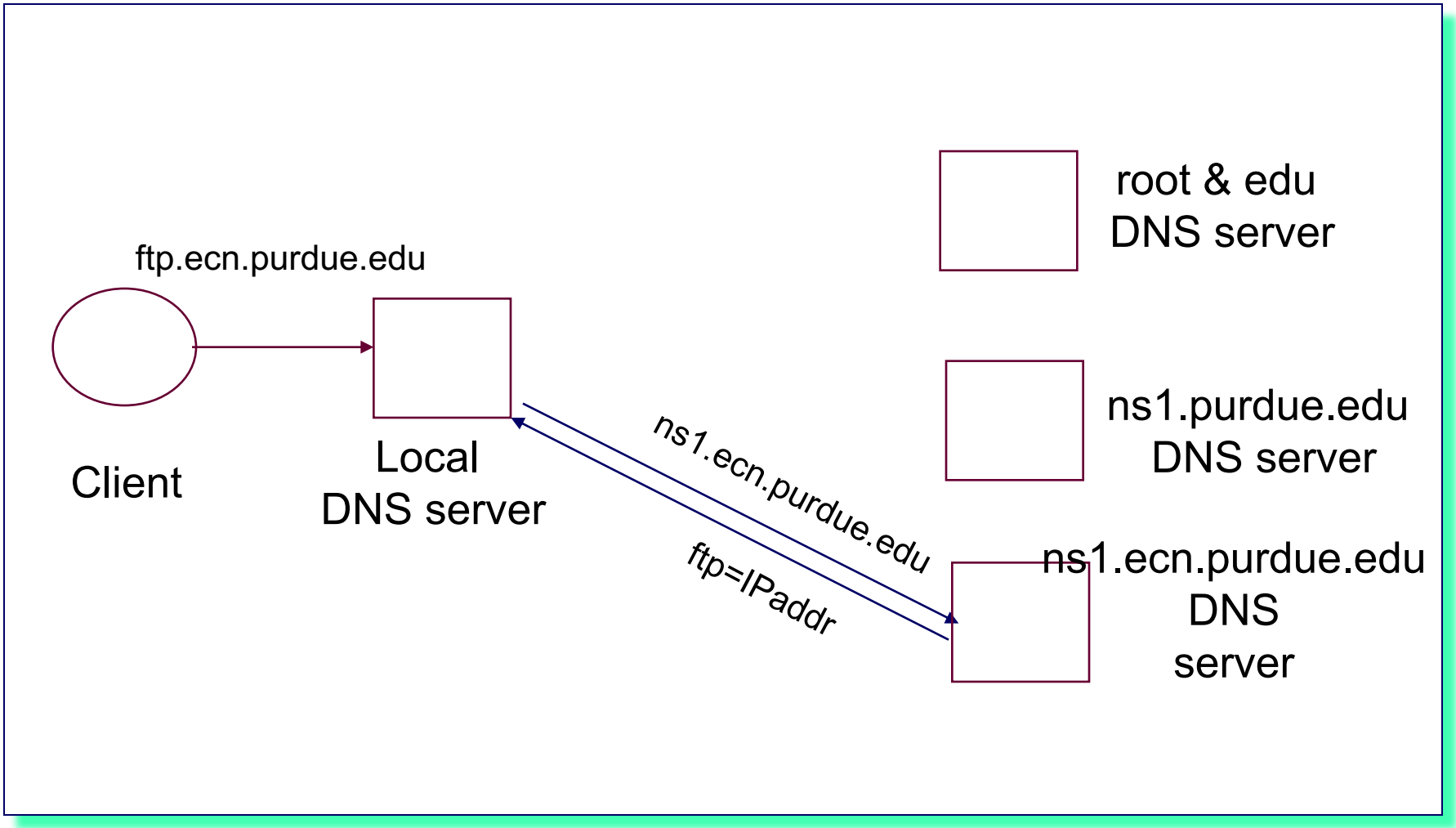
- Local server builds up cache of intermediate translations
- Helps in resolving names xxx.ecn.purdue.edu, yy.purdue.edu, and z.edu
- Cached data periodically times out

  - Lifetime (TTL) of data controlled by owner

Typical Resolution



Subsequent Lookup Example



DNS Records

RR format: **(class, name, value, type, ttl)**

DB contains tuples called resource records (RRs)

- **Classes = Internet (IN), Chaosnet (CH), etc.**
- **Each class defines value associated with type**
- **Each DNS record associated with its own TTL**

FOR IN class:

Type=A

- name **is hostname**
- value **is IP address**

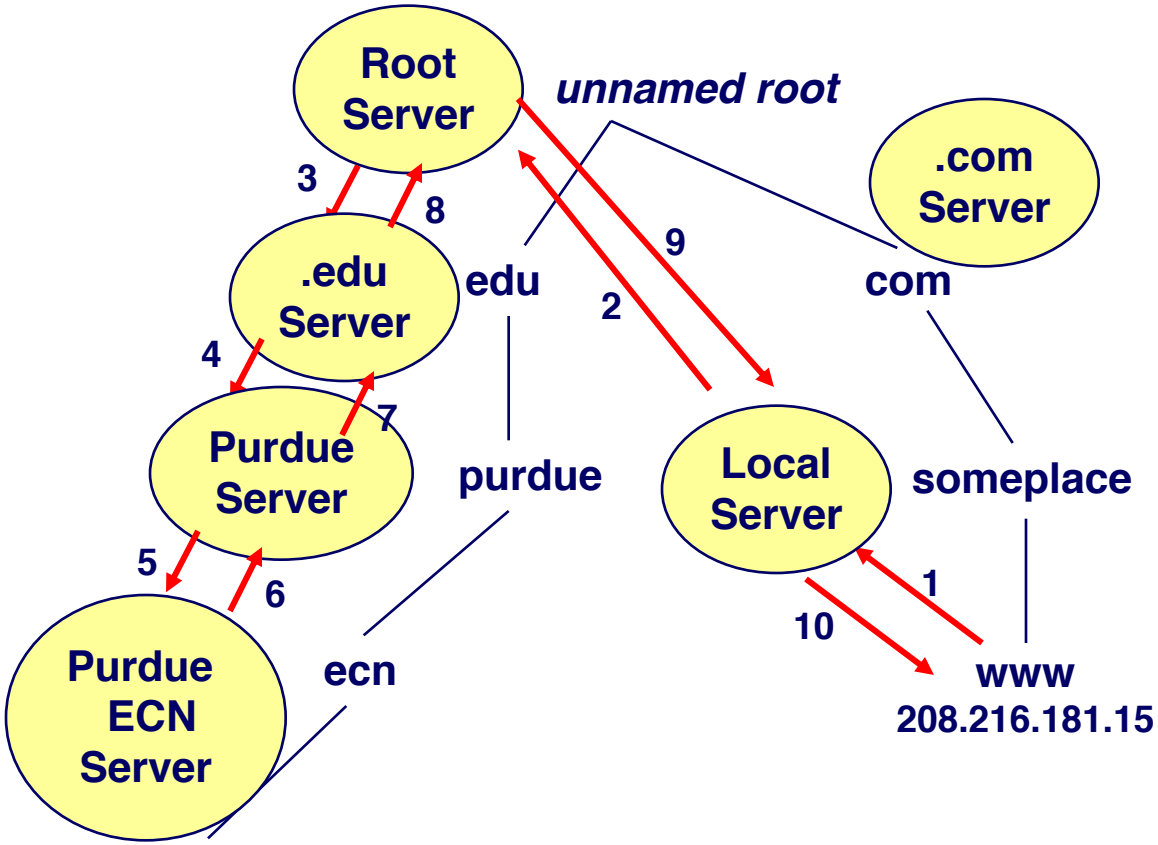
Type=NS

- name **is domain (e.g. foo.com)**
- value **is name of authoritative name server for this domain**

Type=CNAME

- name **is an alias name for some "canonical" (the real) name**
- value **is canonical name**

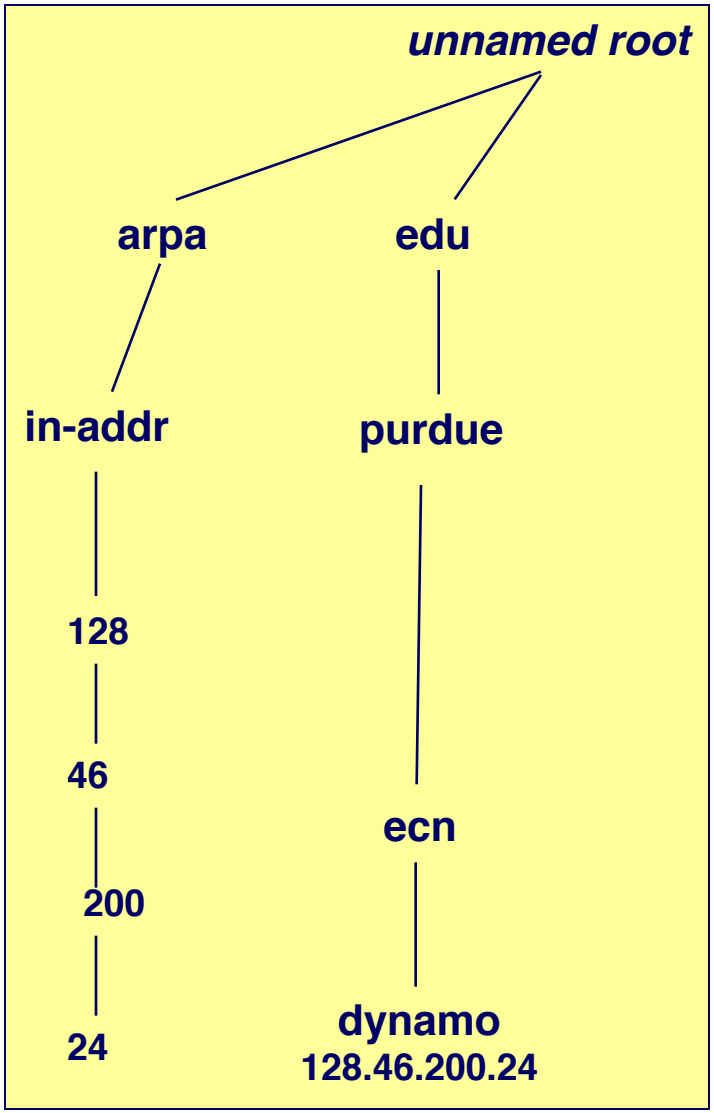
Recursive DNS Name Resolution



- Recursively from root server downward
- Results passed up

Exploited in security attacks.
Often disabled, especially at root servers

Reverse DNS



Task

- Given IP address, find its name

Method

- Maintain separate hierarchy based on IP names
- Write 128.46.200.24 as 24.200.46.128.in-addr.arpa

Managing

- Authority manages IP addresses assigned to it
- E.g., Purdue manages name space 46.128.in-addr.arpa

Content Delivery Networks and DNS

Content Delivery Networks (CDNs)

- Replicate content across geographically disperse servers
- Direct users to servers close to them
- First-party CDNs: Google, Netflix etc.
- Third-party CDNs: Akamai, Fastly, CloudFront etc.

How do CDNs direct clients to close locations?

- Exploit DNS

CDNs and DNS

Example redirection process