Naming in Distributed Systems

CSE 486/586: Distributed Systems

Ethan Blanton

Department of Computer Science and Engineering

University at Buffalo

Naming

In general, naming is a hard problem.

There are two hard problems in computer science: cache invalidation, naming things, and off-by-one errors.
— Phil Karlton, as modified by Leon Bambrick [3]

This is true of:

- Projects
- Variables and functions
- Files
- …

Distributed Naming

Distributed naming is even harder.

(It's kind of like naming *and* cache invalidation!)

It involves at least two hard problems:

- Authority: who can name
- Consistency: who knows the names

Typically naming includes mapping a name to a value.

What Do We Name?

When we say naming, what are we naming?

In a distributed system, this is likely to be:

- Hosts
- Processes
- Data
- Operations

We will look closely at hosts and data.

Who Can Create a Name?

We are going to look at three authority models.

- Centralized: Some entity creates and binds names
- Hierarchical: Different entities have authority over parts of a namespace
- Globally Unique: Names are guaranteed to be unique, so authority is not required

The entity may be a person, organization, or service.

Centralized Authority

With centralized naming, one authority maintains all names.

Examples:

- Early Internet naming; `hosts.txt` distributed by SRI [5]
- ISO two-letter country codes [1]
- TCP/UDP port number assignments [2]
- IPv4 addresses on a local network via DHCP [4]

This type of naming is relatively uninteresting to us.

Typically one simply fetches the registry periodically.

Hierarchical Authority

With hierarchical naming, authorities divide up the namespace.

Examples:

- The Internet Domain Name System [7]
- Go and Java package naming conventions
- ASN.1 Object Identifiers [8]
- MAC addresses

Authority is normally delegated in some structured fashion.

Each portion of the namespace is looked up with the appropriate authority.

Globally Unique Names

Globally unique names are sometimes used to avoid the authority problem.

Every name is different, so anyone can choose a name.

Examples:

- UUIDs [6]
- Distributed hash tables (more later!)
- Public keys (*e.g.*, your SSH key)

A particular use of these names is content-addressed storage.

Consistency

Name mappings can change over time.

Names can be:

- Added (*e.g.*, registering a domain)
- Removed ()
- Changed

Correct operation may require consistency in these mappings.

Performance may require caching of values.

Caching

Some names may be used frequently.

Looking them up for every use can be slow.

This is particularly true for centralized authorities.

Performance can be improved by caching mappings.

A cached mapping may be out of date.

This is normally handled with time-to-live (TTL) values.

A cached entry is discarded after its TTL expires.

Races

Even without caching, name lookups can be stale.

In an asynchronous system, delays are arbitrary.

The name may have changed since the lookup!

We won't deal with this sort of consistency …yet.

Domain Names

Domain Names map hostnames to addresses.

It is a hierarchical authority [7].

It is valuable because:

- IP addresses can be hard for humans to remember:
  www.cse.buffalo.edu *vs.* 128.205.32.52
  google.com *vs.* 2607:f8b0:4006:814::200e
- IP addresses can change when changing Internet providers
- Names can map to different addresses in different places/times
- One name can map to multiple addresses or vice-versa

Internet Addresses

IP addresses are essentially large integers.

IPv4 addresses are 32-bit, written as a dotted quad:
1.2.3.4 where each number is one byte in decimal (0-255)

IPv6 addresses are 128-bit, written in groups of four hex digits:
0011:2233:4455:6677:8899:aabb:ccdd:eeff

IP addresses are bound to a location in the network.

Recall from week 1 that their primary purpose is routing.

Domain Names

Domain names are human-readable names.

Everyone knows that buffalo.edu is:

- An American educational institution (.edu)
- Related somehow to “buffalo”

They form a hierarchy rooted at “”.
(A final . (often elided) indicates this!)

The last element is called the top-level domain (TLD).

The Root Zone

The domain “” is the root zone.

The root zone is a central authority.

The Internet Corporation for Assigned Names and Numbers (ICANN) runs the root zone.

The root zone delegates authority to top-level domains.

Top-level Domains

Top-level domains (TLDs) are autonomous authorities.

Every country in the world has its own two-letter TLD.

You probably recognize: .io, .ly, .uk, .cn, …

The US administers a bevy of three-letter TLDs.

You know these, too: .com, .org, .net, .gov, .int, …

There are also more recent generic TLDs (gTLDs):

.dev, .travel, .aero, .name, .рф

These authorities choose who may register under their TLD.

Further Delegation

TLDs and gTLDs must be delegated by ICANN.

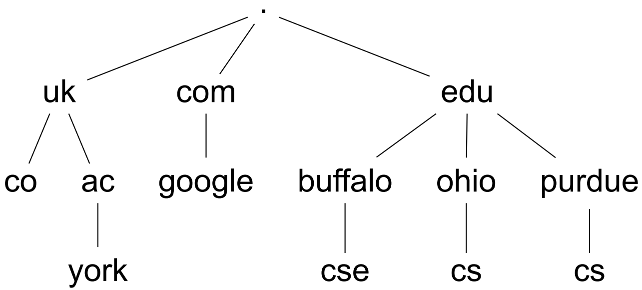
Those TLDs control their own delegation.

For example:

- ICANN delegates .edu to Educause in the US
- Educause delegates buffalo.edu to The University at Buffalo
- UB assigns cse.buffalo.edu to CSE
- CSE gave westruun.cse.buffalo.edu to me for my office

I certainly did not have to contact ICANN!

Delegation Examples



The Benefits of Hierarchy

This hierarchical delegation of authority is efficient.

The formation of new TLDs is rare:

- Recognized countries change very slowly
- gTLDs are intentionally expensive and require approval

Registration of individual domains is more common.

Creation of names within domains is even more common!

This pushes name management closer to the users.

The Domain Name System

The Domain Name System [7] is a distributed database.

It handles:

- Mapping domain names to IP addresses
- Delegating authority to the domain owners
- Controlling caching
- Load balancing

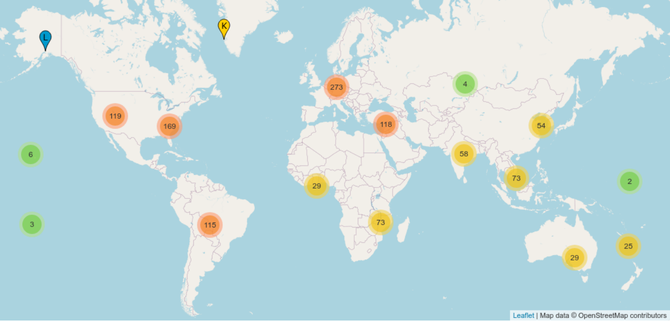
It has been running in a backward-compatible fashion for 30+ years!

Zones

Each delegation is called a zone.

We already talked about the root zone.

It is served by thirteen servers, named A-M.
Each one is a distributed cluster.



DNS Servers

The database is served by DNS servers around the Internet.

Each server can:

- Serve one or more zones authoritatively
- Serve zones or records non-authoritatively
- Cache DNS information

Authoritative servers are delegates for the zone.

Non-authoritative servers spread the load.

Zone Start of Authority

Each zone begins with a Start of Authority record.

The Start of Authority record defines:

- The serial number of the zone data
  This increases monotonically as the zone changes
- Cache timeouts and parameters
- The primary DNS server for the zone

SOA records are how delegation of zone management is handled.

Resource Records

The database is made up of resource records.

Resource records bind names to values.

Typically those values are addresses or other names.

Examples:

- A records bind names to IPv4 addresses
- NS records bind DNS servers to domain names
- MX records bind mail servers to domain names

Arbitrary information can be bound into the DNS system.

This is used for distributing public keys, fighting spam, …

Querying the Database

There are two kinds of DNS queries:

- Recursive: DNS servers walk the hierarchy to find a binding for a querent
- Iterative: A querent walks the hierarchy to find a binding.

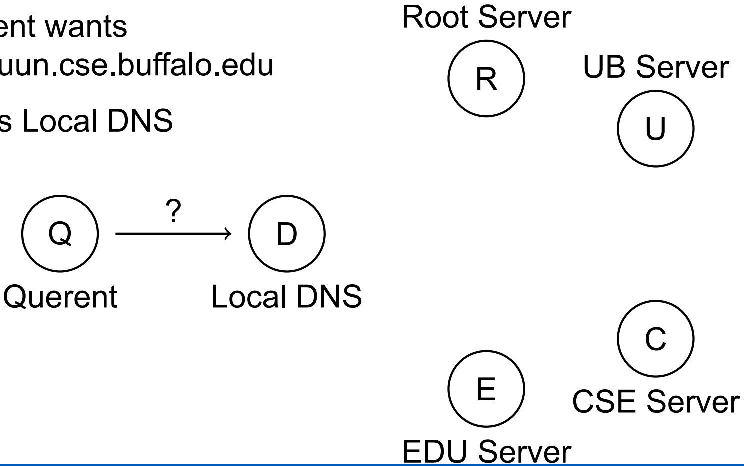
Typically end hosts perform recursive queries.

This maximizes caching benefits.

Full Query Example

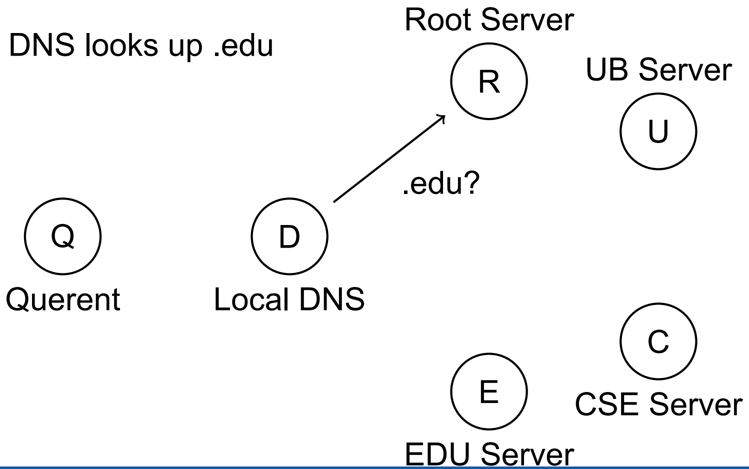
Querent wants westruun.cse.buffalo.edu

It asks Local DNS



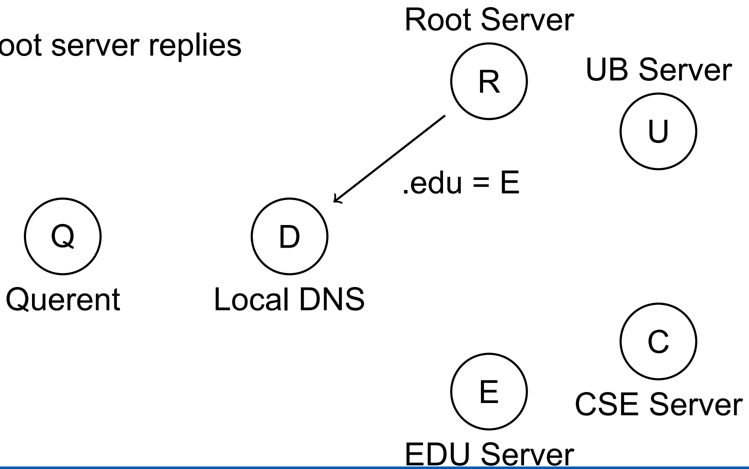
Full Query Example

Local DNS looks up .edu



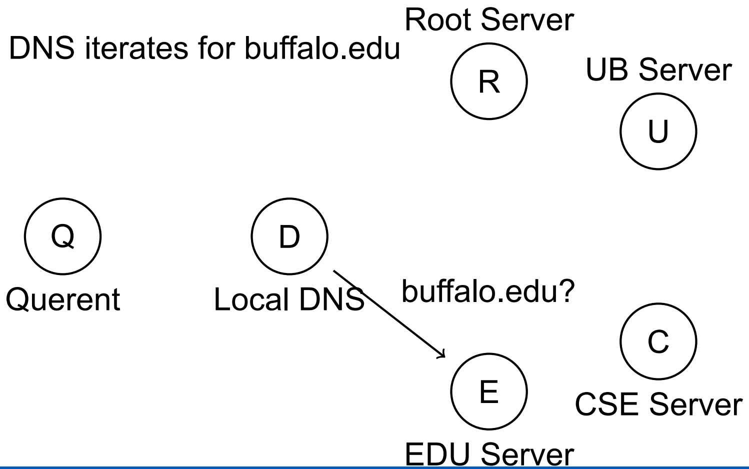
Full Query Example

The root server replies

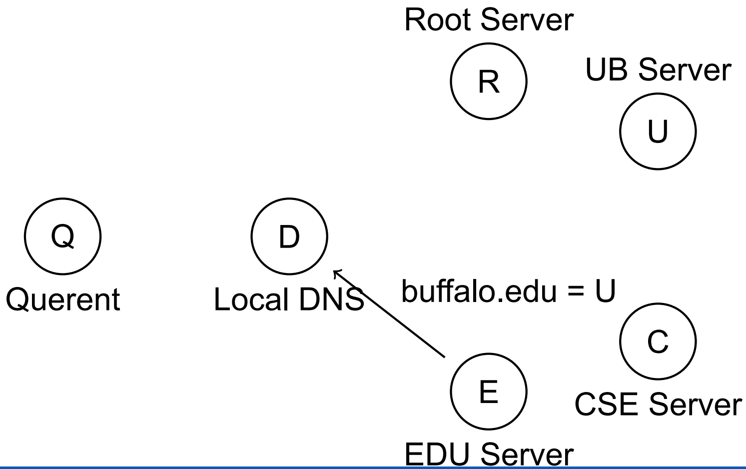


Full Query Example

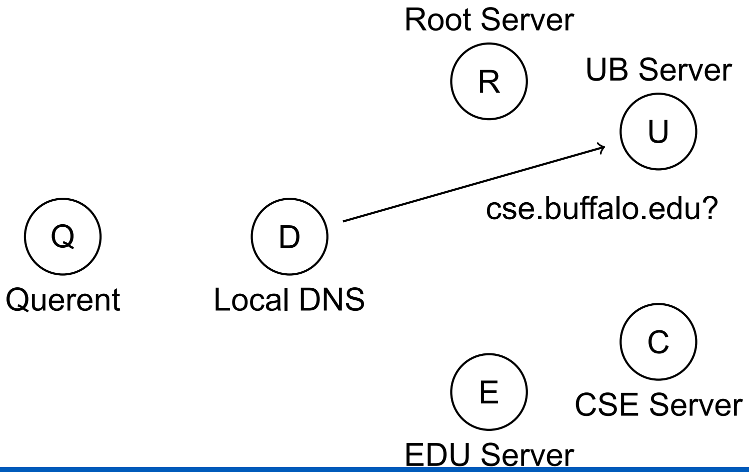
Local DNS iterates for buffalo.edu



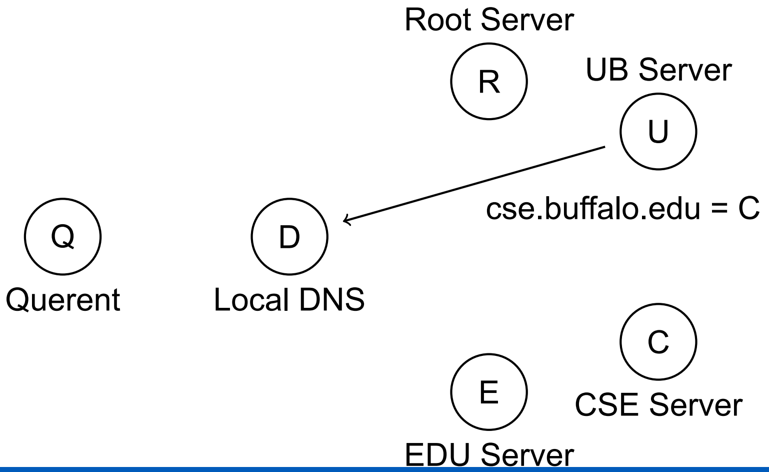
Full Query Example



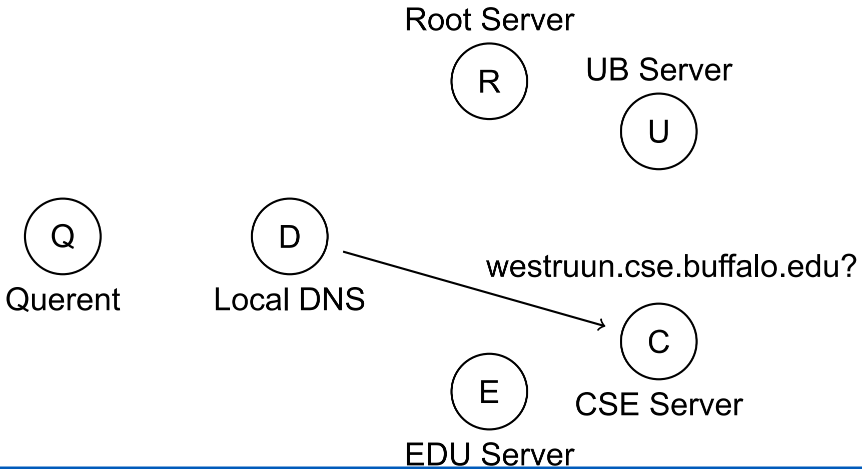
Full Query Example



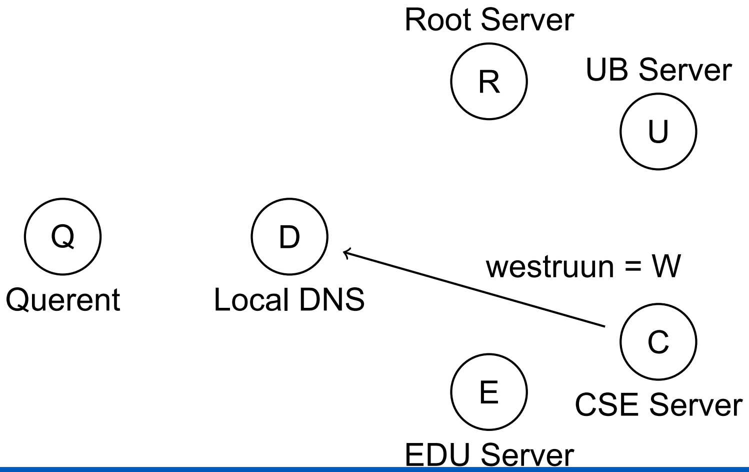
Full Query Example



Full Query Example

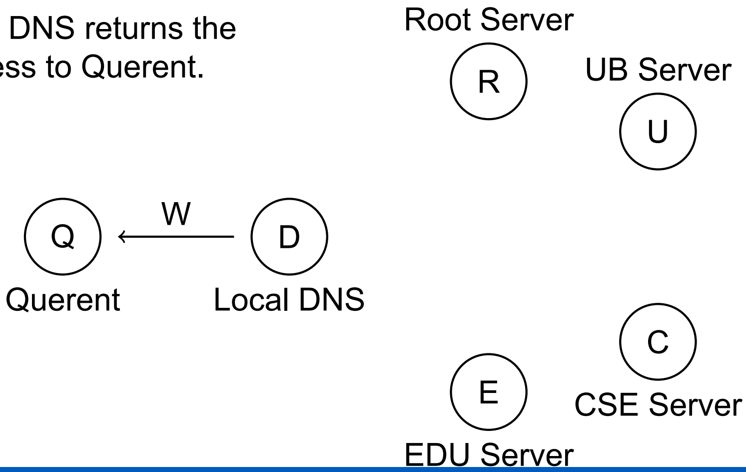


Full Query Example



Full Query Example

Local DNS returns the address to Querent.



Performance

This process can take a long time!

A complete iterative lookup can take hundreds of ms or more.

This is why caching is important!

If the local DNS server caches intermediate lookups, looking up (*e.g.*) www.cse.buffalo.edu next will be much faster!

However, caches have to time out so that names can change.

Balancing these things can be tricky.

Reliability

A single host lookup can require many servers.

That is many possible points of failure!

This is solved through redundancy:

```
$ host -t NS cse.buffalo.edu
cse.buffalo.edu name server dns02.buffalo.edu.
cse.buffalo.edu name server dns01.buffalo.edu.
cse.buffalo.edu name server dns04.buffalo.edu.
cse.buffalo.edu name server dns03.buffalo.edu.
```

Any one of those servers can serve cse.buffalo.edu.

The Chicken and the Egg

A DNS client needs to know the IP address of its server.

DNS client use servers to find IP addresses.

A DNS server needs to know the root servers.

The root servers are a.root-servers.org, *etc.*

???

The Chicken and the Egg

A DNS client needs to know the IP address of its server.

DNS client use servers to find IP addresses.

A DNS server needs to know the root servers.

The root servers are a.root-servers.org, *etc.*

???

Some configuration still has to use IP addresses!

Summary

- Naming is hard, and harder for distributed systems
- Naming can be:
  - Centralized at some authority
  - Delegated hierarchically
  - Distributed via global uniqueness
- DNS is a global distributed database that:
  - Delegates authority
  - Provides redundancy
  - Uses caching to improve performance

Next Time …

- Globally unique names
- Content-addressed naming
- Distributed Hash Tables

References I

Recommended Readings

[7] Paul Mockapetris. *Domain Names — Concepts and Facilities*. RFC 1034. Nov. 1987. URL: `https://www.rfc-editor.org/rfc/rfc1034.txt`.

Optional Readings

[1] ISO 3166 Maintenance Agency. *Codes for the Representation of Names of Countries and their Subdivisions — Part 1: Country Code*. ISO 3166-1:2020. Aug. 2020. URL: `https://www.iso.org/iso-3166-country-codes.html`.

References II

[2] Internet Assigned Numbers Authority. *Service Name and Transport Protocol Port Number Registry*. URL: https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml.

[3] Leon Bambrick. *Twitter Messsage*. Jan. 2015. URL: https://twitter.com/secretGeek/status/552779013890904064.

[4] Ralph Droms. *Dynamic Host Configuration Protocol*. RFC 2131. Mar. 1997. URL: https://www.rfc-editor.org/rfc/rfc2131.txt.

[5] Ken Harrenstien, Mary K. Stahl, and Elizabeth J. Feinler. *DOD Internet Host Table Specification*. RFC 952. Oct. 1985. URL: https://www.rfc-editor.org/rfc/rfc952.txt.

References III

[6] Paul J. Leach, Michael Mealling, and Rich Salz. *A Universally Unique IDentifier (UUID) URN Namespace*. RFC 4122. July 2005. URL: https://www.rfc-editor.org/rfc/rfc4122.txt.

[8] International Organization for Standardization. *Specification of Abstract Syntax Notation One (ASN.1)*. International Standard 8824. Dec. 1987.