COMP2221 Networks

David Head

University of Leeds

Lecture 4

Reminder of the last lectures

In previous lectures we have seen how:

- Networks can be understand as a **layered architecture**.
- The Transport layer typically uses **UDP** or **TCP**.
- Headers are appended as data is sent down the **protocol stack**, and removed by the same layers as it is received.
- Each **host** (*i.e.* laptop, desktop, server, ...) can handle multiple incoming and outgoing messages using **ports**.

However, we have still not seen how the host *itself* is addressed.

Today's lecture

In today's lecture, we will see how readable internet addresses such as `www.leeds.ac.uk` are converted to IP addresses.

- Service provided by the Domain Name System, or **DNS**.
- How DNS servers are structured.
- How they can be queried.
- Other services they provide.

Next time we will look more closely at IP addresses, and see how to use them in Java.

DNS is now a critical support protocol for other network applications.

A host can be addressed in one of two compatible ways:

- A readable host name (*i.e.* `www.comp.leeds.ac.uk`); or
- An IP address:
  - IPv4 (*i.e.* 129.11.144.10)
  - IPv6 (*i.e.* 2001:630:62:59::53)

DNS provides a system that maps between these two addressing formats (so you don't have to remember long lists of numbers for your favourite sites):

`www.comp.leeds.ac.uk` $\longleftrightarrow$ 129.11.144.10

Key concepts

Indirection:

- Names replace IP addresses.
- We rarely use IP addresses directly.

Hierarchy:

- Apparent in IP addresses, their names, and the DNS server structure itself.

Distribution:

- No single DNS server contains all names or IP addresses.
- Scalable.

Caching:

- Local caching of DNS results for re-use.

Names *versus* addresses

Human recall of names is better than that for numbers . . .

. . . but names provide little information about location.

- The country level domains `.uk`, `.fr` *etc.* give *some* location information, but nothing beyond the country.
- Other domains (`.org`, `.net` *etc.*) give no location information.

Addresses are the 'raw data' for network communication.

- Part of the Network layer protocol.

Addresses are organised hierarchically, and more immediately relate to host location.

Uniqueness (1)

Every **public** device on the internet has a unique IP address.

Hosts in **private** networks, *i.e.* Local Area Networks or **LAN**s, can use their own internal addresses.

- Some ranges of IP addresses reserved for this purpose, *e.g.* 10.*.*.* in IPv4[1].

Outward-facing servers, *i.e.* the public hosts visible to the Wide Area Network (WAN), *do* have a unique IP address.

- They forward messages from LAN hosts to the WAN, and *vice versa*, using a Network Address Translation (NAT) table.
- Does this by (ab-)using port numbers.

[1] The asterisk '*' means **any** value for this byte.

Uniqueness (2)

One hostname can map to **multiple** IP addresses:

- Popular servers will typically have multiple IP addresses around the world.
- The DNS server may try to select the 'closest' to the sender.
- It may also rotate through the list of IP addresses, to reduce congestion.

For instance, popular web sites or CDNs (Content Distribution Networks), will typically have multiple addresses from which their content can be accessed.

- How these sites are *synchronised* is another matter.

Uniqueness (3)

Multiple names can map to a **single** IP address.

- This is known as **aliasing**.

Especially useful for mail addresses:

- For *e.g.* `f.bloggs@leeds.ac.uk`, `leeds.ac.uk` is a short address.
- It is *not* the name of the server.
- When sending a mail to this address, the email client can query the DNS to find the full, **canonical** address.

Can also be used to simplify host names for *e.g.* web sites in much the same way.

DNS Tools

`nslookup`

- Resolves hostnames to IP addresses.
- Deprecated, but still installed on most UNIX systems.

`host`

- Basic resolution of hostnames and IP addresses.

`dig`

- Resolves hostnames and IP addresses.
- Can also trace the DNS query.
- Allows for a more detailed response.

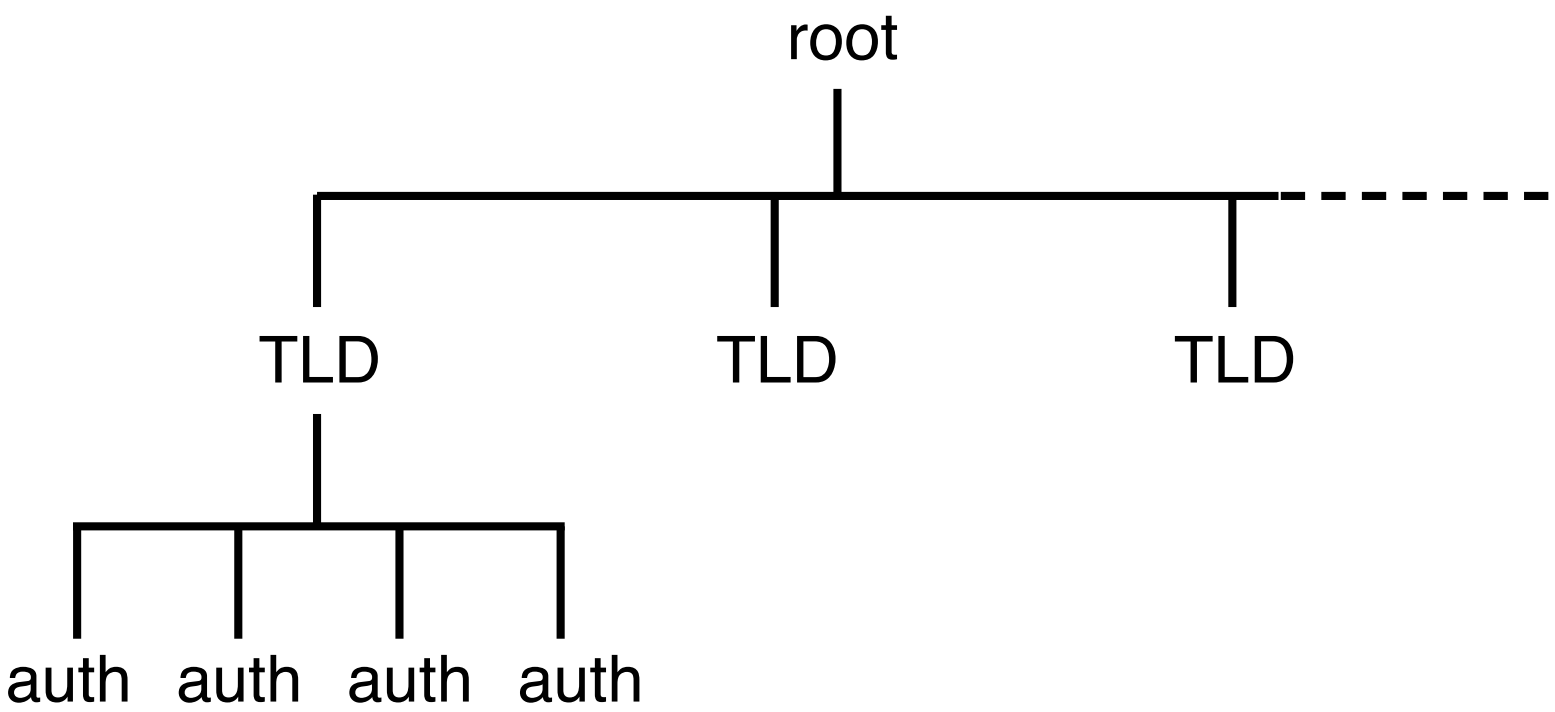
The Domain Name System

Hierarchical name space:

- Divided into zones.
- Distributed across DNS servers.
- There is no single, centralised list — so also no single point of failure.

Server structure matches the hierarchical name space:

- Root servers.
- Top-Level Domain (**TLD**) servers.
- Authoritative DNS servers.
- Local DNS servers.


root
TLD
TLD
TLD
auth
auth
auth
auth

Root Servers

There are 13 root servers covering the Internet

- Actually many more (>400), but only 13 IP addresses, *i.e.* we can only 'see' 13 from one host.
- Exception to the 'unique' IP address.

These are distributed geographically, although most are in the US.

Root server locations are 'hard-wired' into DNS servers.

TLD Servers

Responsible for top-level domains:

- `.com`, `.org`, `.net`, ...

Also country domains

- `.uk`, `.fr`, ...

One server per top-level domain, maintained by a company or organisation.

- *e.g.* Verisign Global Registry is responsible for the `.com` domain.

Authoritative DNS Servers

Organisations who want their sites to be viewed publicly **must** provide accessible DNS records for their hosts.

- Either their own authoritative DNS server, or that of a service provider.
- Universities, ISPs, ...
- Likely to also maintain a secondary (back-up) DNS server.

For example, the University of Leeds DNS servers are

- `dns.leeds.ac.uk` for IPv4.
- `dns6.leeds.ac.uk` for IPv6.

DNS Protocol

Uses UDP, although can use TCP in special circumstances (*i.e.* if transferring a large amount of data from one server to another).

Queries **fail** if not answered within a set time limit.

- May retry until successful.

Uses port 53.

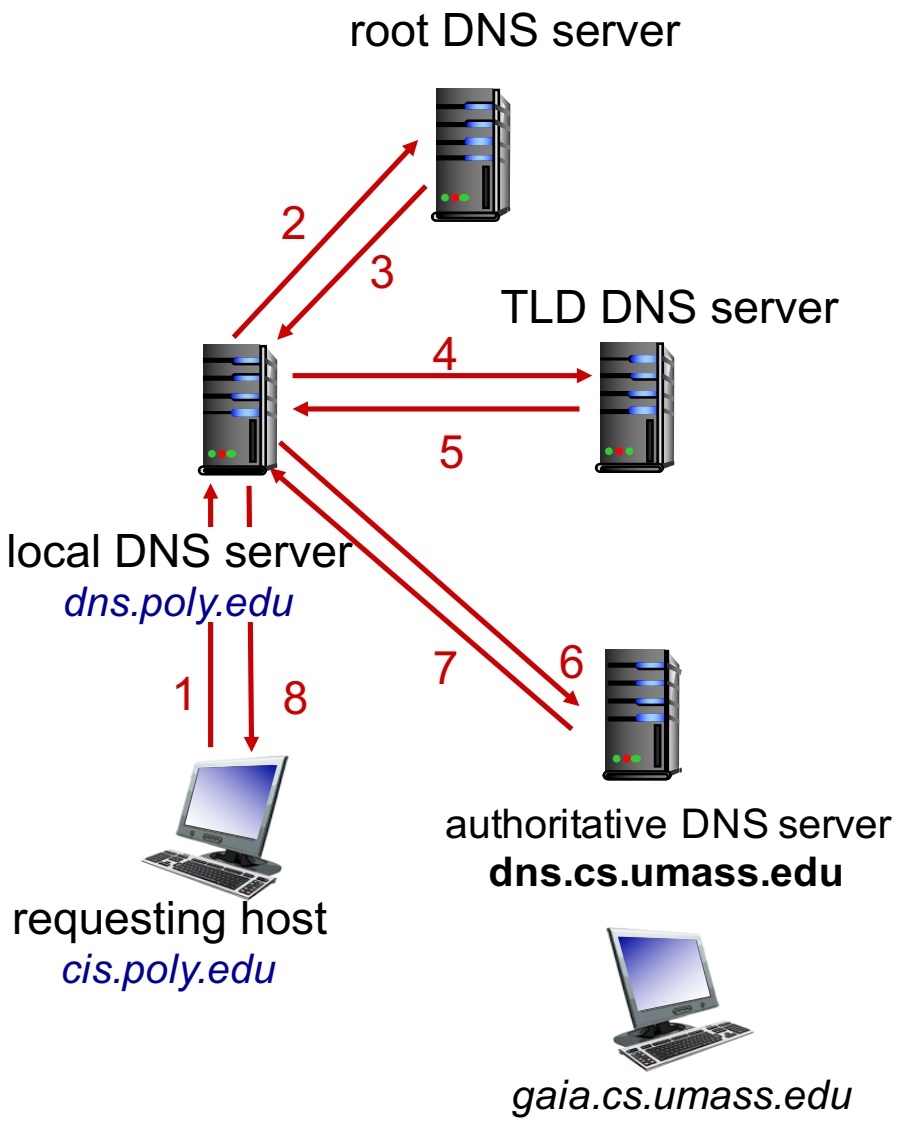
Messages are either 'query' or 'reply,' using the same format.

- If interested, Kurose and Ross §2.4 (7^{th} ed.) has more details.

Using DNS

A DNS query will pass through the hierarchy:

- Local host is configured to connect to a **local** DNS server . . .
- . . . which negotiates with **root** DNS server to find **TLD** server . . .
- . . . and then negotiates with TLD server to find **Authoritative** DNS server . . .
- . . . and then negotiates with Authoritative DNS server to resolve host name to IP address . . .
- . . . and then returns this IP address to the local host.



http://www-net.cs.umass.edu/kurose-ross-ppt-6e

Recursive and iterative queries

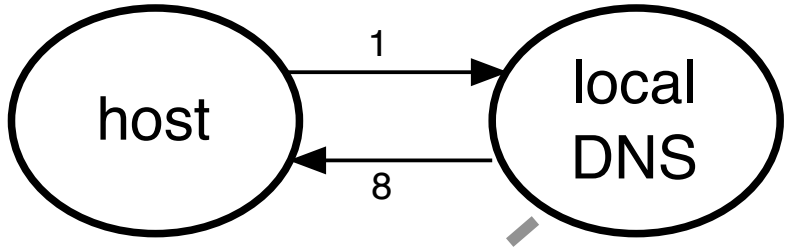
This example includes both **recursive** and **iterative queries:**

- The local host asking the local DNS server to resolve a host name is a **recursive** query.
- The 3 queries from the local DNS to the root, TLD and Authoritative servers in turn is an **iterative** query.

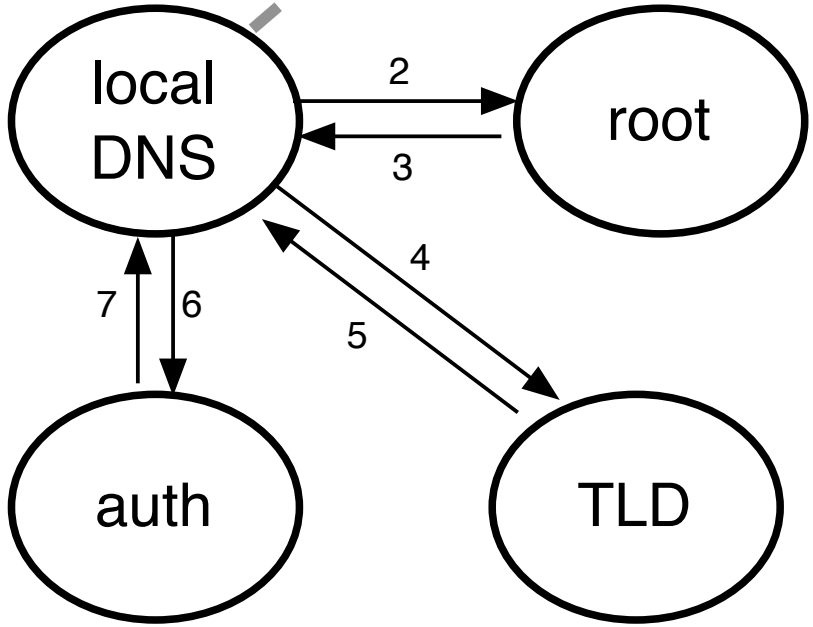
8 messages required in total, for each host name resolution.

A very lengthy process!

Recursive:



Iterative:



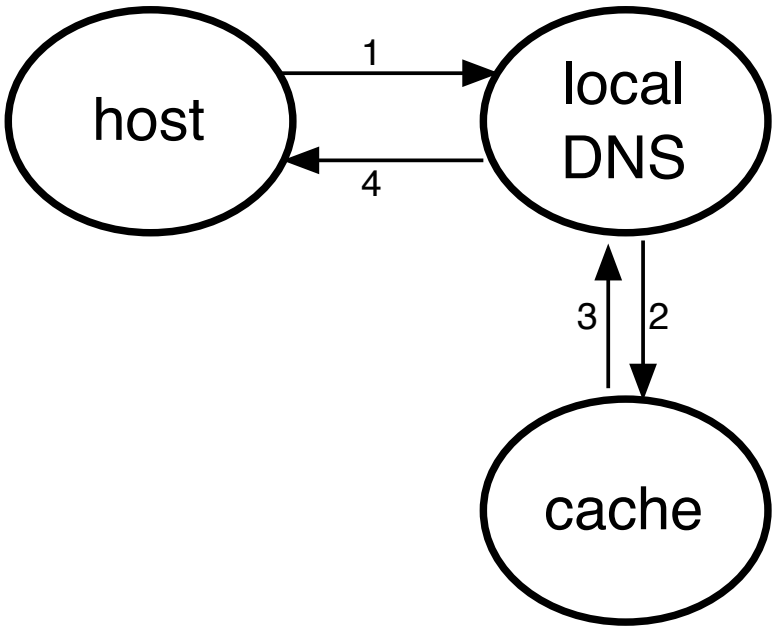
DNS Caching (1)

The query process has an overhead:

- Primarily a delay, especially if one of the queries is lost and needs to be re-sent.
- May also cause congestion, as the number of messages for each server can become very high.

To improve performance, query results are **cached** on the local DNS server:

- Cache is checked before the root DNS server is used.
- Can cache IP addresses for TLD servers, bypassing the need to access the root server.


host
local
DNS
cache
1
2
3
4

DNS Caching (2)

If a query is successful:

- The returned DNS message will include a TTL, or a **time-to-live**.
- Will cache until the TTL expires, typically days.

If a query is unsuccessful:

- Overhead for unknown domain names (*e.g.* mistyped domain) is greater than for known domains, as it requires an exhaustive search.
- Caching avoids a repeat of this overhead.
- In this instance, typically has a shorter TTL, or the order of hours.

Summary

Today we have looked at the Domain Name System:

- How it is required to support the use of human-readable host or server names.
- **Distributed**, **hierarchical** network of servers.
- **Caching** is critical for performance.

Next time we we start on Java (at last!) and the `java.net.InetAddress` class, as well as more details about IP addressing.