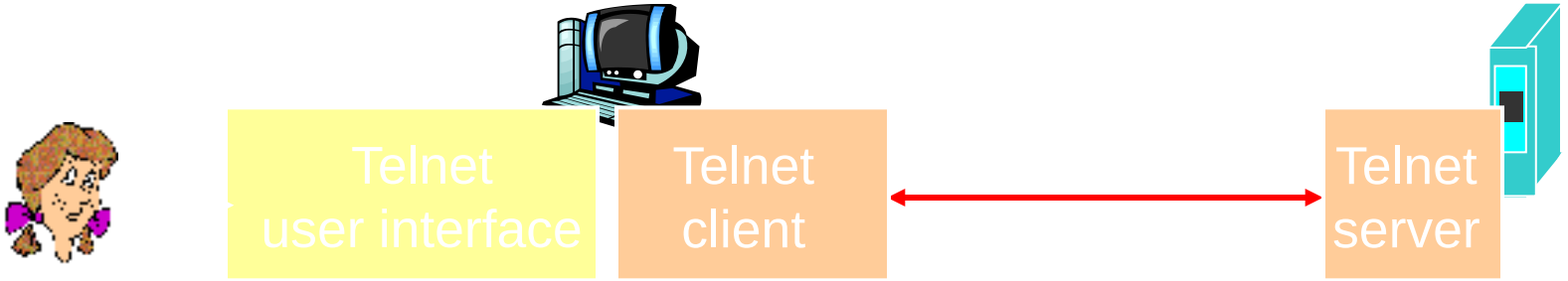
# Network Applications (2)

Andy Carpenter

(Andy.Carpenter@manchester.ac.uk)

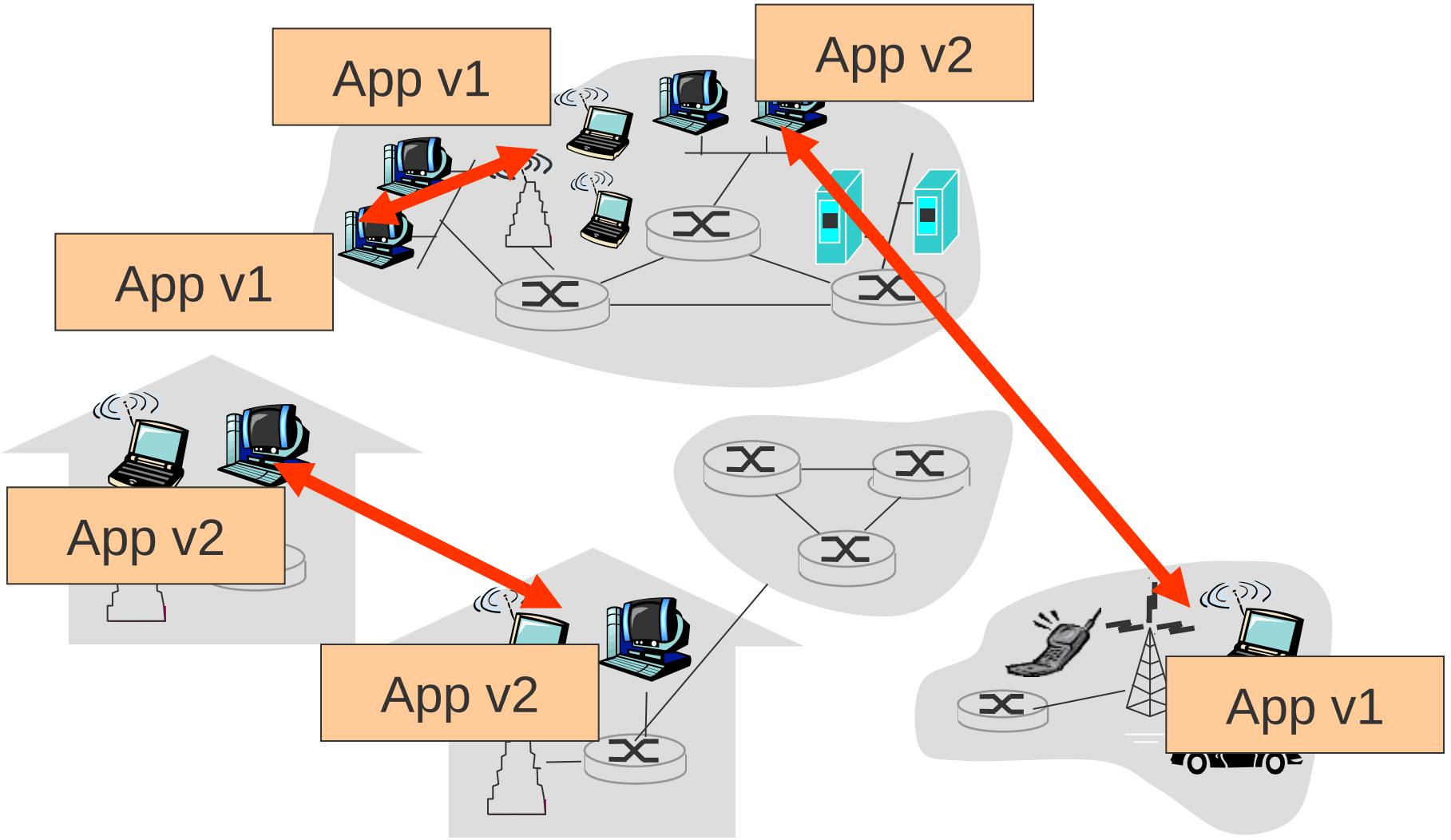
Elements these slides come from Kurose and Ross, authors of "Computer Networking: A Top-down Approach", and are copyright Kurose and Ross

# Telnet – A remote login application




- Uses canonical representation of terminal (NVT)
- Can change functionality of connection using options
- Issue: End-of-line representation
  - three possibilities: CR, LF and CR LF
  - network virtual terminal (NVT) always uses CR LF
- Issue: control vs. data
  - embeds control in data stream
- Issue: adding new options (extensibility)

# Application Extensibility



# Extensibility: Telnet Options

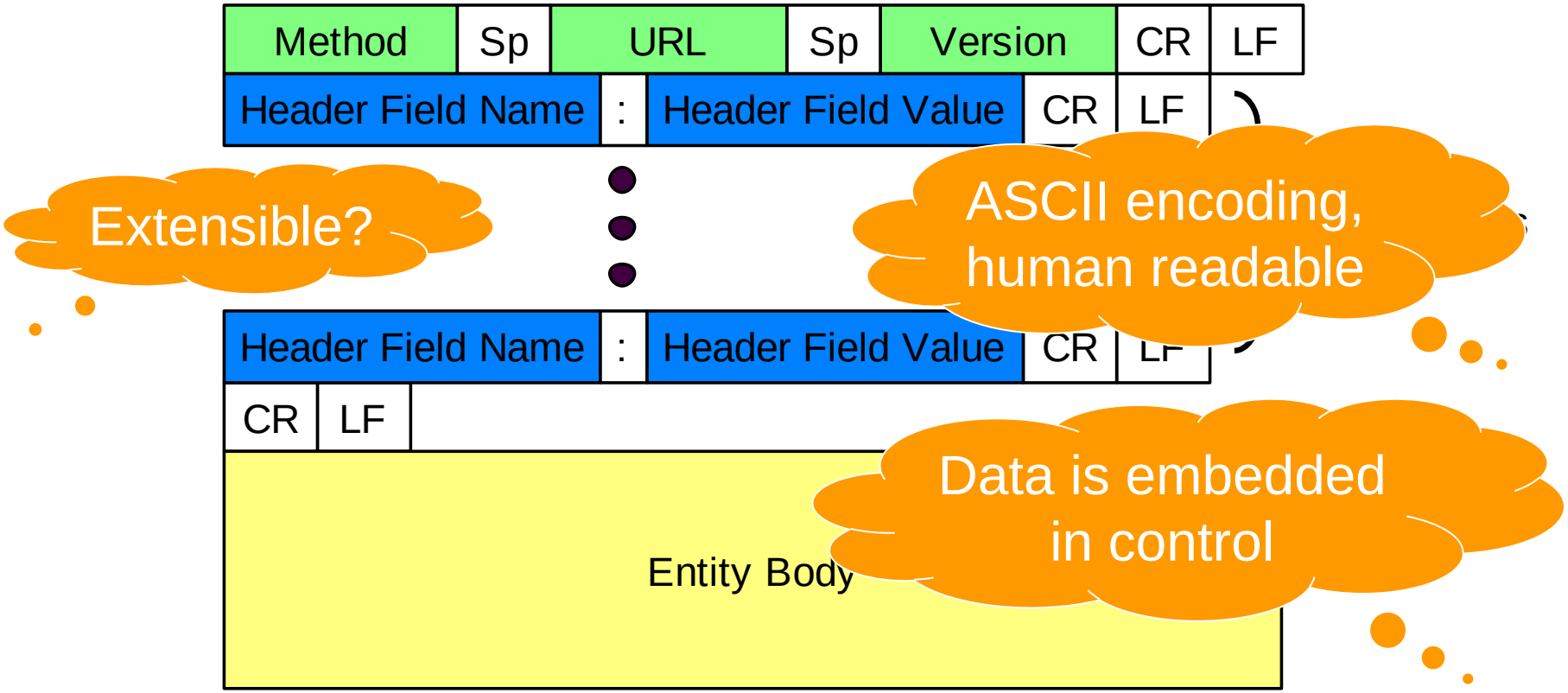
- Options change functionality of NVT
- Option negotiation mechanism:
  - request action, either accepted or rejected
- Based on DO/DON'T and WILL/WON'T commands

| Send | Positive Response | Negative Response |
|---|---|---|
| DO | WILL | WON'T |
| DON'T | WON'T | WILL |
| WILL | DO | DON'T |
| WON'T | DON'T | DO |

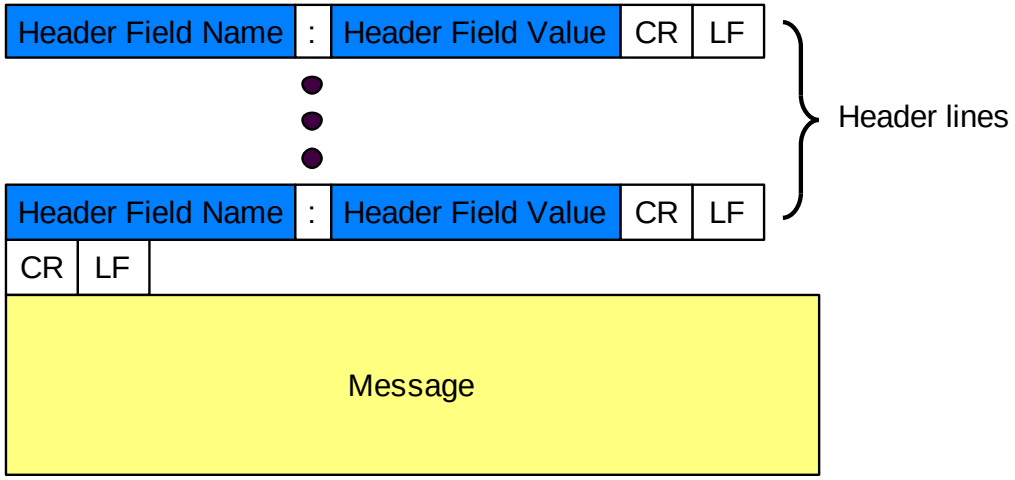
- Standard only defines how options are negotiated
  - adding new option does not require new standard
- Unknown requests are rejected (extensible)

# Extensibility: HTTP Messages

*request response*



# Extensibility: Email RFC822 Message

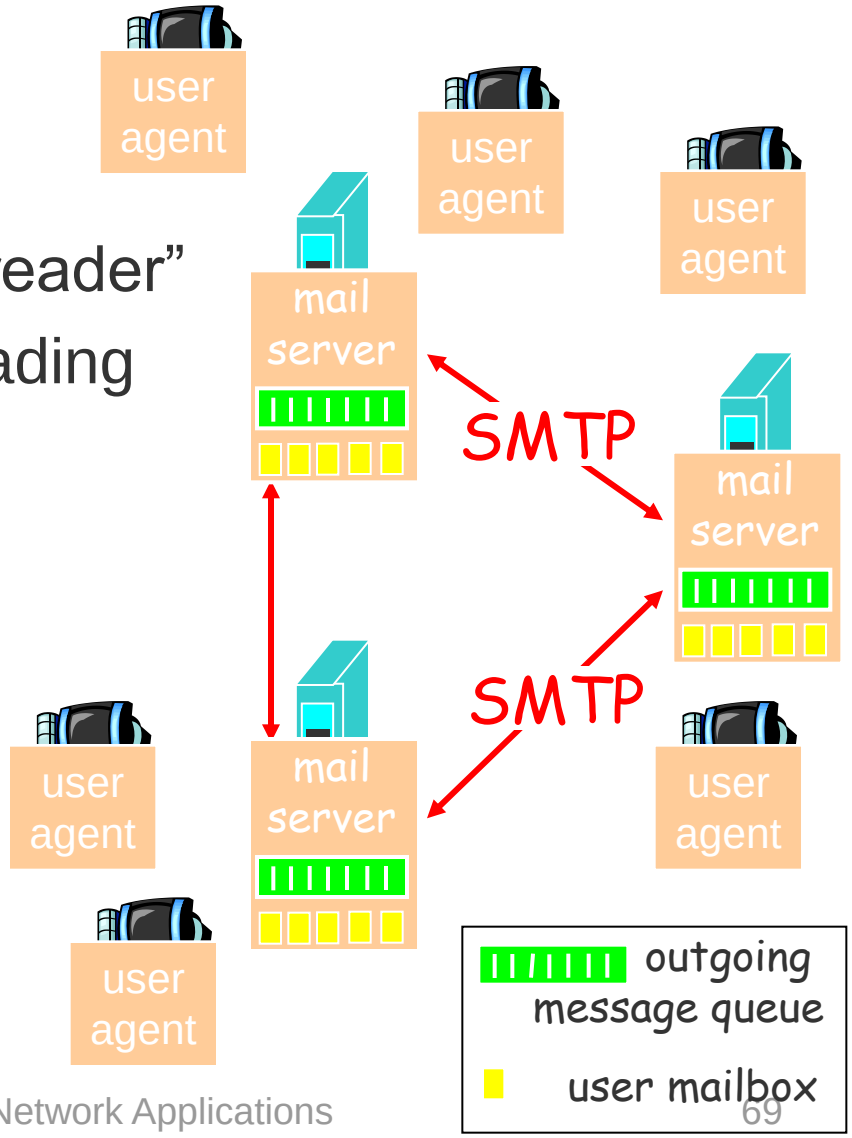


# Protocols: Electronic Mail

Major components:

- Mail servers
- User agents, a.k.a. “mail reader”
  - composing, editing, reading
- Sending protocol
  - push to server
- Access protocol
  - fetch from server
  - manage on server
- Message format

Issue: number of protoc



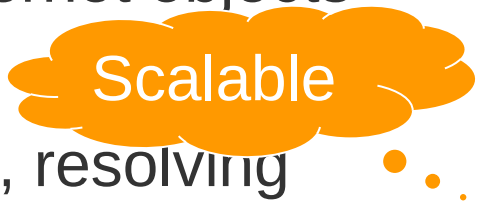
# Email: Transfers to Servers (SMTP)



- Simple Mail Transfer Protocol (RFC 5321)
- Uses persistent TCP connections to server port 25
- Three phases of transfer:
  - handshaking (greeting)
  - transfer of messages
  - closure
- Command/response interaction
  - commands: ASCII text
  - response: ASCII status code and phrase
- Messages must be in 7-bit ASCII
- SMTP is push; HTTP is pull


# Scalability: Domain Name System (DNS)

- Uses hierarchical name space for internet objects
- Provides way to decentralise:
  - naming, name and value mapping, resolving
- Not just names to address mapping; others:
  - host and mail server aliases (service names)
  - address to name
  - load balancing (multiple address for one name)
- Issues:
  - coordinated decentralisation, scalability
  - robustness, start point for searches

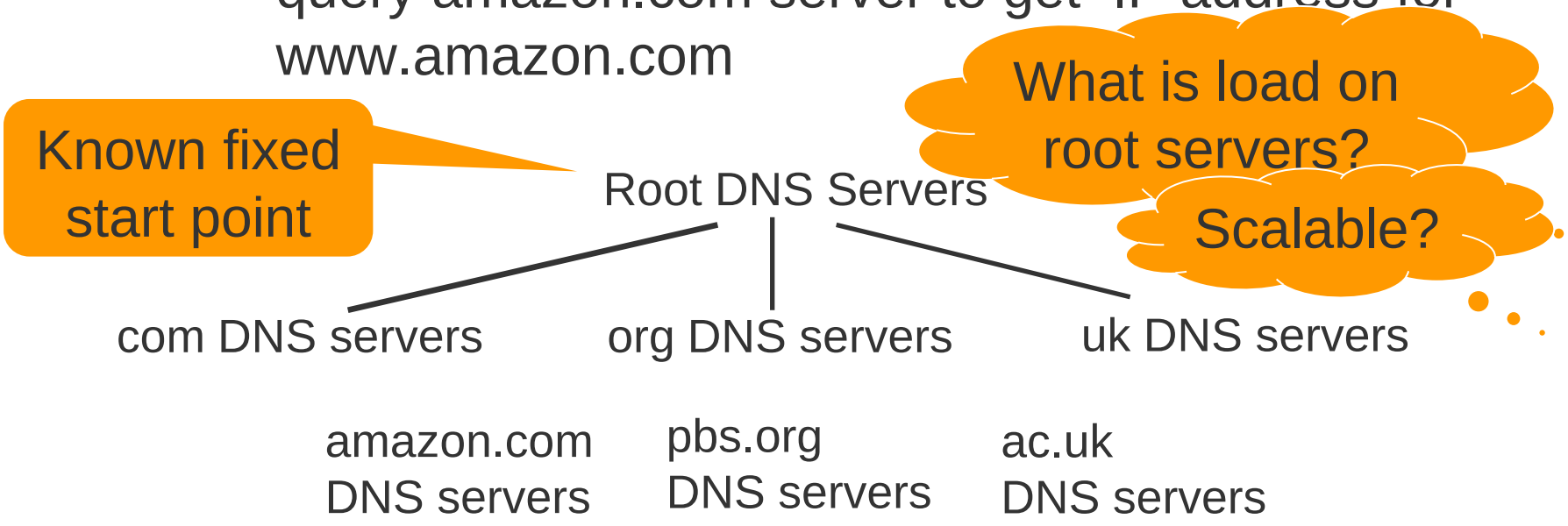


# DNS: Name Syntax and Naming

- Hierarchical names; levels separated by a dot
- At top there is a single ‘root’ domain; ‘.’
- A section of hierarchy is known as a domain or zone
- Names must be unique within a zone
- Standard assumes top level naming authority
- May delegate naming authority for a domain/zone
- Naming authority may be further delegated
- Domains are divided until contents are manageable

# Scalability: DNS 'Database'

- Every zone has, at least, one name server
- Client wants IP for www.amazon.com; 1st approx:
  - query a root server to find com DNS server
  - query com server to get amazon.com DNS server
  - query amazon.com server to get IP address for www.amazon.com



# DNS: Fixed Start Points for Queries

- 13 root name servers worldwide
  - named [a-m].root-servers.net
  - many have multiple locations (use anycasting)
- Addresses built into DNS implementation code

Auto configured

a Verisign, Dulles, VA
c Cogent, Herndon, VA (also LA)
d U Maryland College Park, MD
g US DoD Vienna, VA
h ARL Aberdeen, MD
j Verisign, ( 21 locations)

k RIPE London
(also 16 other locations)

i Stockholm
(plus 28 other locations)

m WIDE Tokyo
(also Seoul, Paris, SF)

e NASA Mt View, CA
f Internet Software C. Palo Alto, CA
(and 36 other locations)

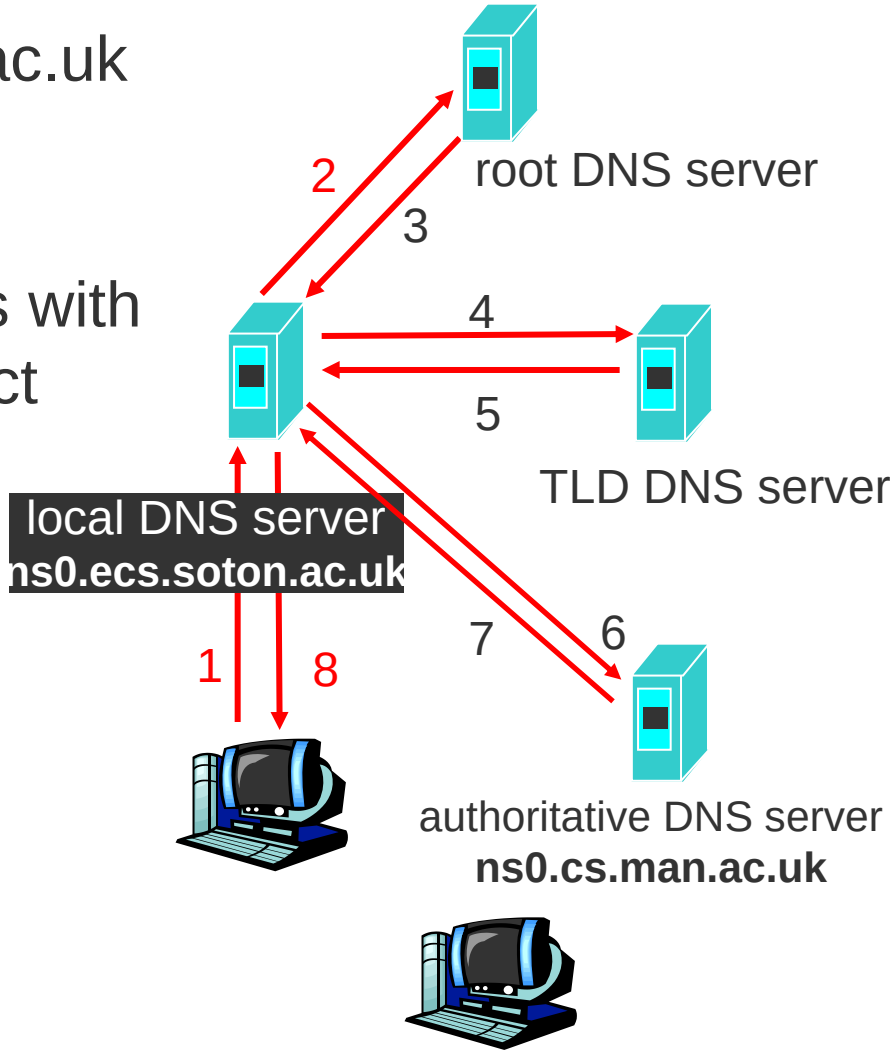
b USC-ISI Marina del Rey, CA
l ICANN Los Angeles, CA

What happens an address when changes?

K&R:2.5.2, P&D:9.1.3

# DNS: Iterative Resolution Example

- Host gander.ecs.soton.ac.uk wants IP address for ruby.cs.man.ac.uk
- Contacted server replies with name of server to contact



# Criteria for Good Design – Met?

- Service model
  - ?
- Global coordination; universal understanding
  - ?
- Minimise manual setup
  - ?
- Minimise volume of information at any point
  - ?
- Distribute information capture and management
  - ?

# Criteria for Good Design – Met?

- Extensibility
  - ?
- Integration/interoperation of heterogeneous systems
  - ?
- Error detection
  - ?
- Error recovery (reliability)
  - ?
- Scalability
  - ?

# Summary

- Good application design is good protocol design
- An application probably uses a collection of protocols
- Have content and data
  - request/response encapsulates data in control
  - control can be embedded in data (Telnet)
  - can separate control and data (FTP, RTP)
- Need to understand information transferred
- Extensible mechanisms
- Can reduce network traffic using caches
- Compared to lower levels, greater variety of requirements