Higher layer protocols

- DHCP
- DNS
- Real time applications
- RTP
- SIP

DHCP

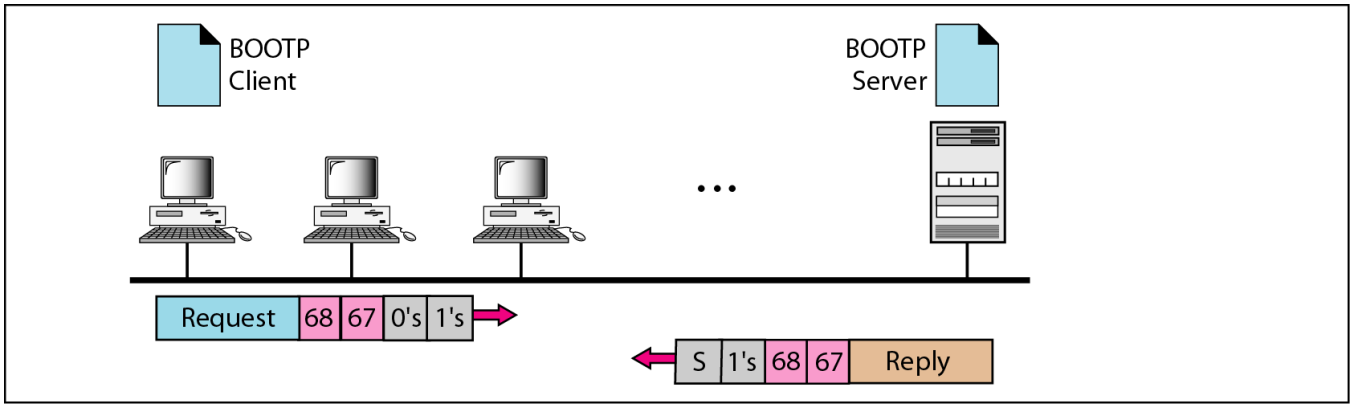
Configuring hosts so they can communicate

What to configure

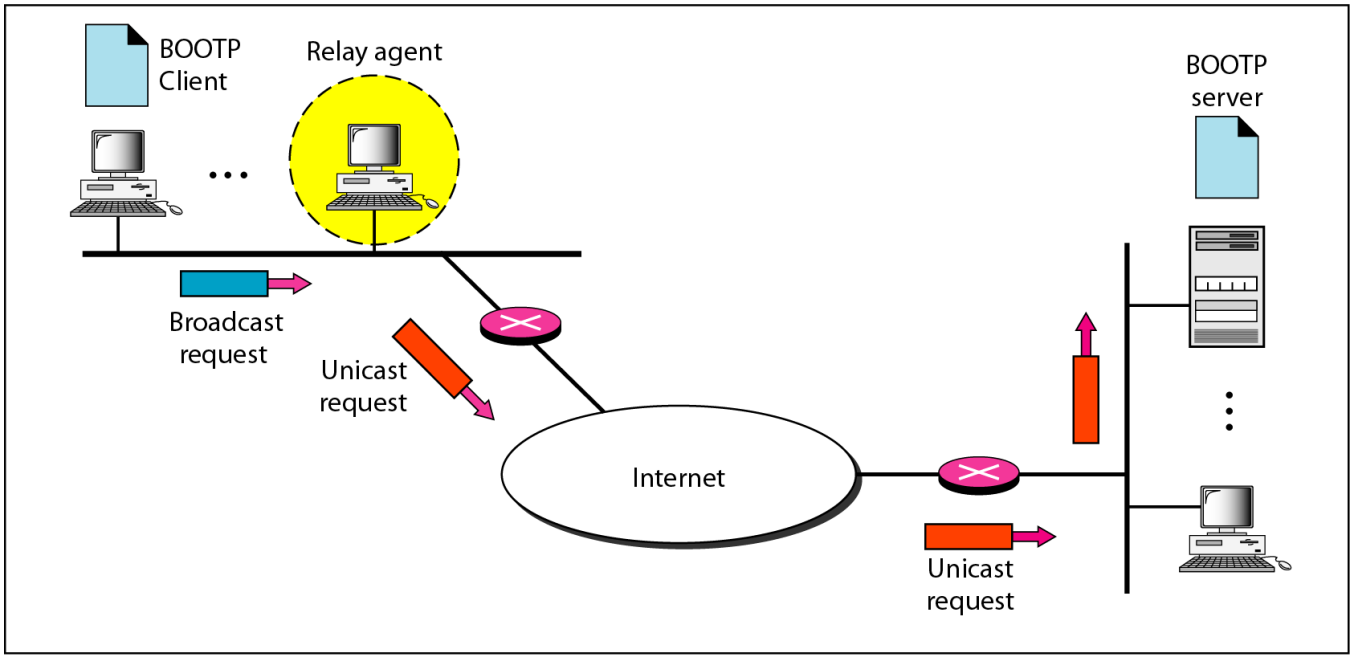
- IP address
- Net mask (specifies network id)
- Default Gateway (at least one)
- DNS server (at least one)
  - Server's IP address

- Other stuff
  - TFTP server
  - Configuration file
  - Executable image download

Obtaining an IP address (bootp – Bootstrap protocol)



a. Client and server on the same network

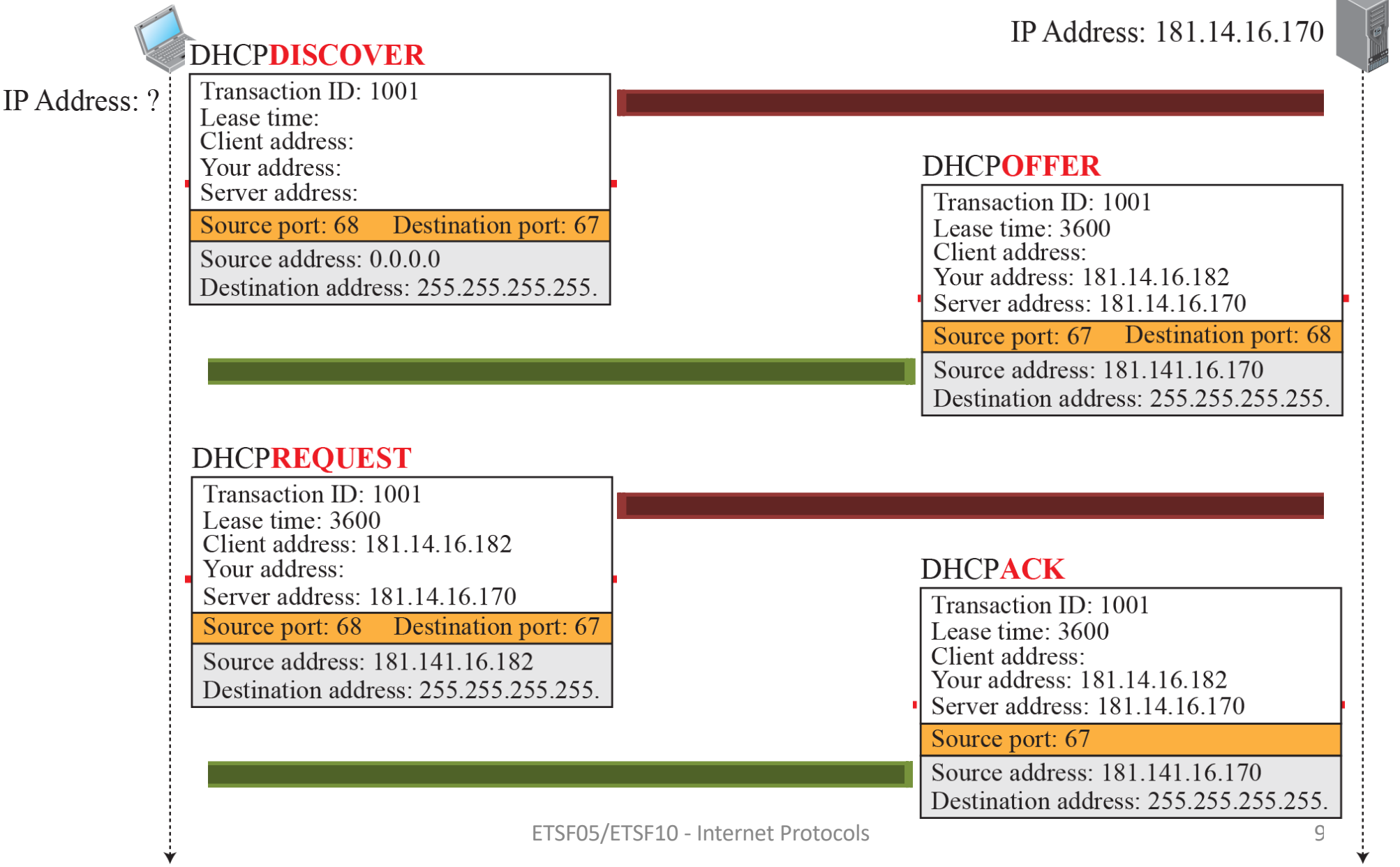


b. Client and server on different networks

Dynamic Host Configuration Protocol (DHCP)

- BOOTP
  - Not dynamic!
  - Server cannot reclaim IP address
- DHCP
  - IP address
    - Allocation from pool or static (mapping to e.g. MAC addr)
  - Network mask
  - Default gateway
  - DNS server(s)
  - Lease time

DHCP operation



How do I know where Google is?
DNS

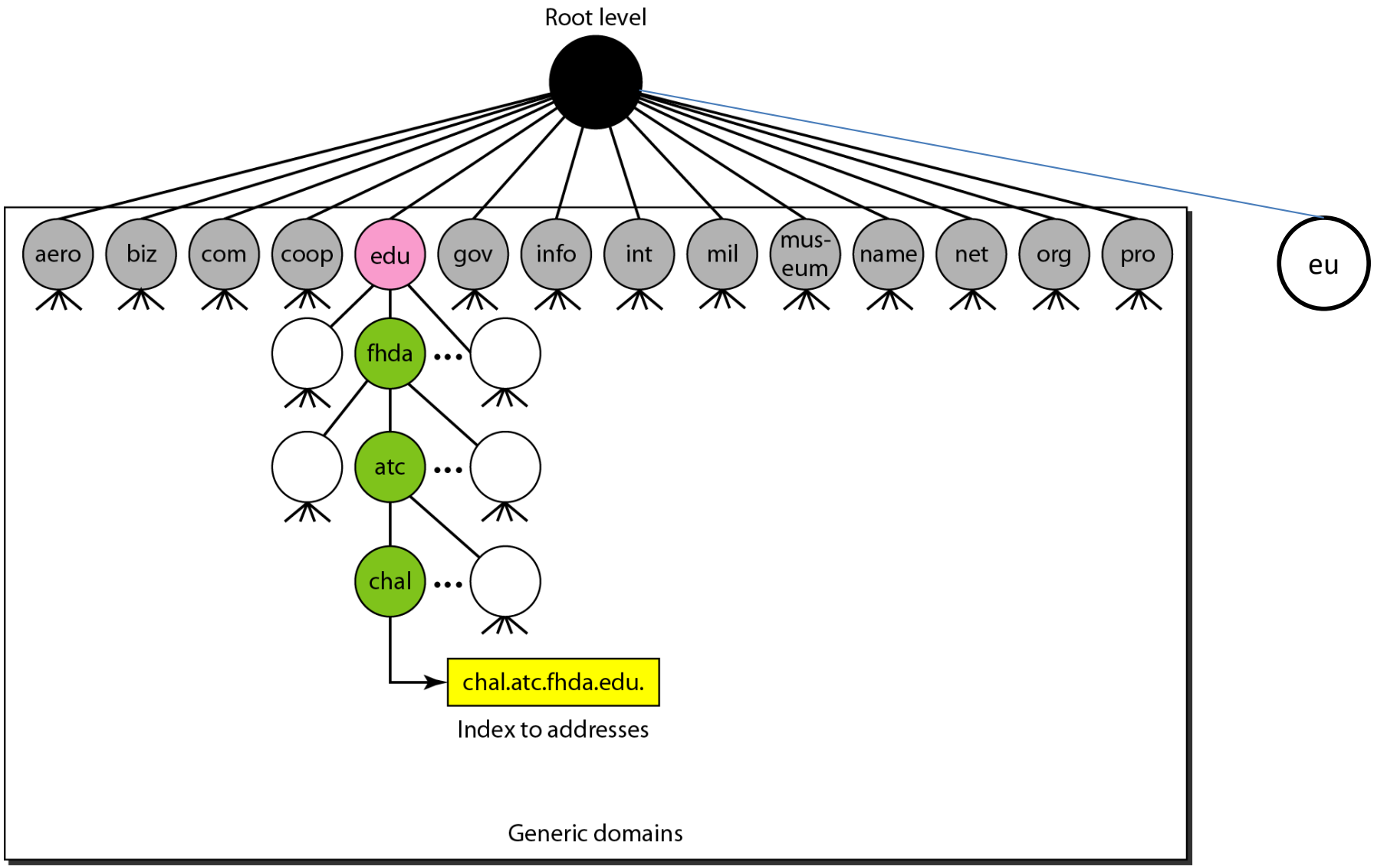
DNS

- People can’t remember 145.55.23.198
- People can remember bill.microsoft.com
- (Almost) Every networked device has identifier
- Fully Qualified Domain Name (FQDN)

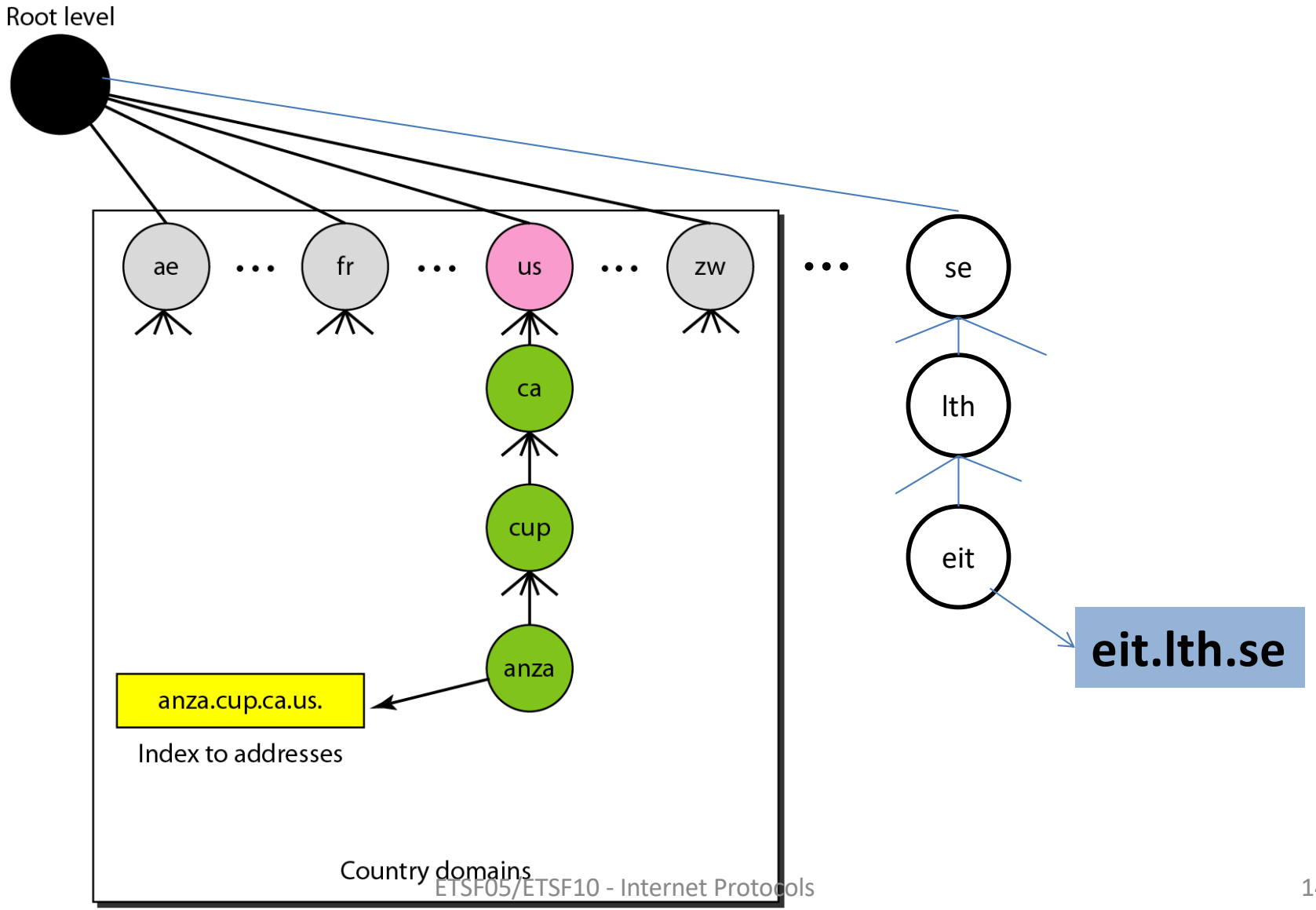
FQDN

- ICANN -Inet Corporation for Assigned Names and Numbers - www.icann.org
  - DNS administration
  - see the FAQ for recent info
- http://www.internic/net/regist.html
  - accredited list of registrars for .com/.net/.org

Generic domains



Country domains



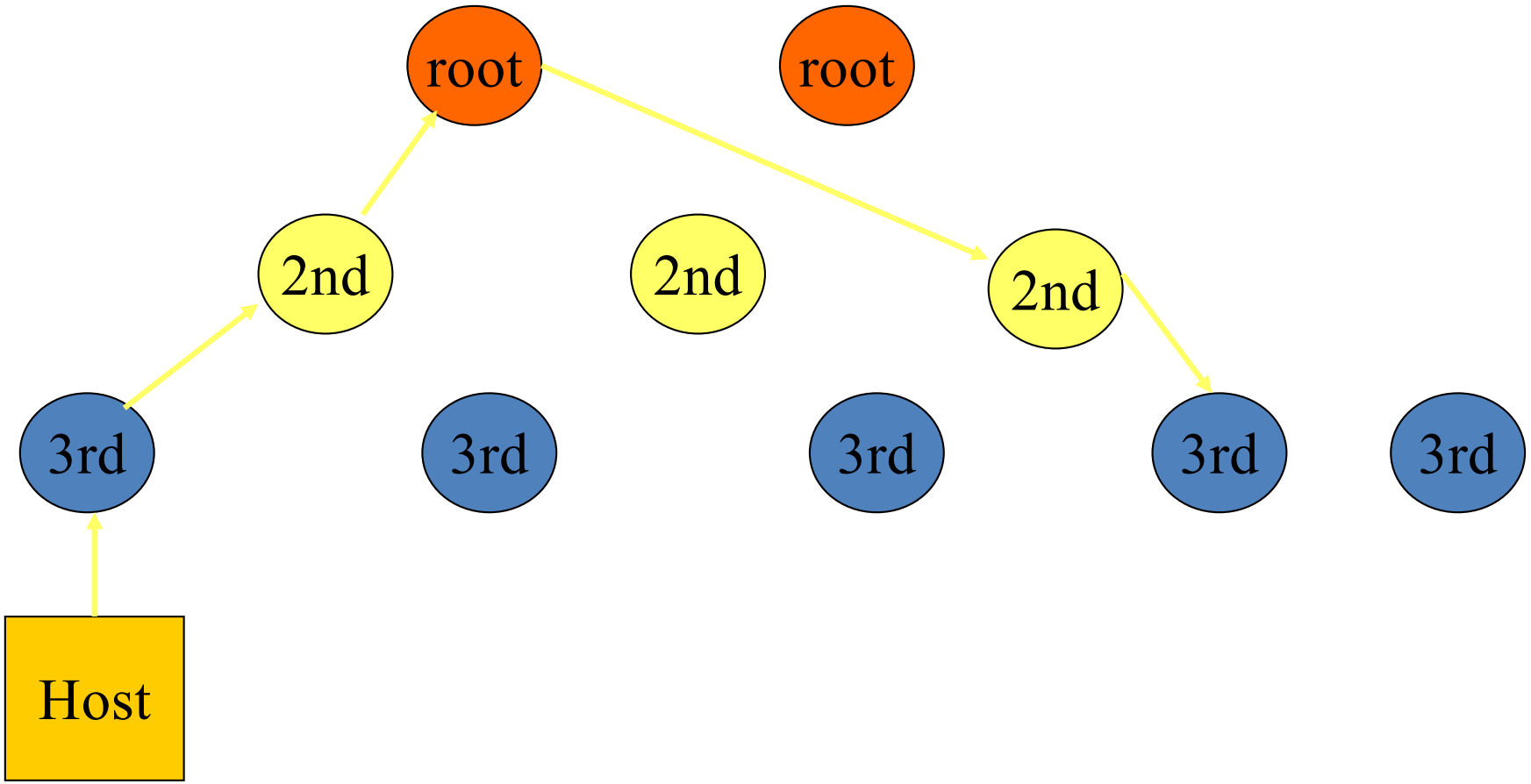
DNS

- Two components:
- Name server (named)
  - A database front end (can be text file)
- Resolver
  - Client side library implements gethostbyname(), gethostbyaddr()

DNS Hierarchy

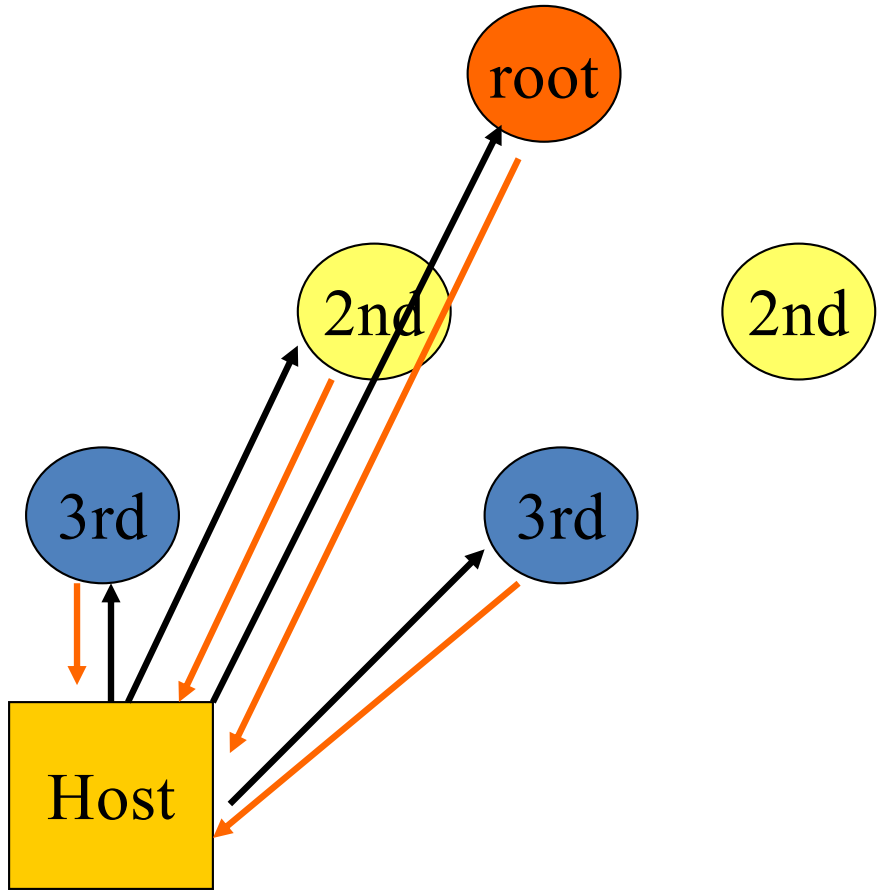
- DNS divided into levels with
  - root servers
  - 2nd level servers
  - Local servers (organisations)
- Queries in iterations
  - First local server
  - If not found 2nd level, then root
  - Root points to 2nd level server etc.

DNS Hierarchy



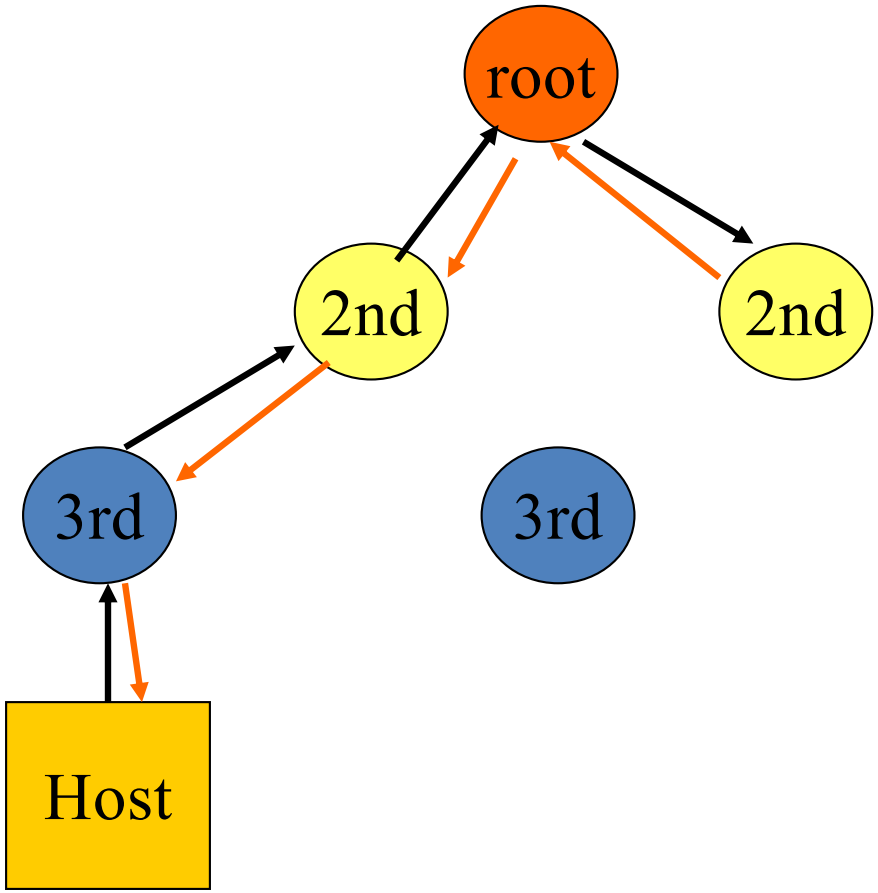
Queries

- Two versions:
  - Iterative



Queries

- Two versions:
  - Recursive



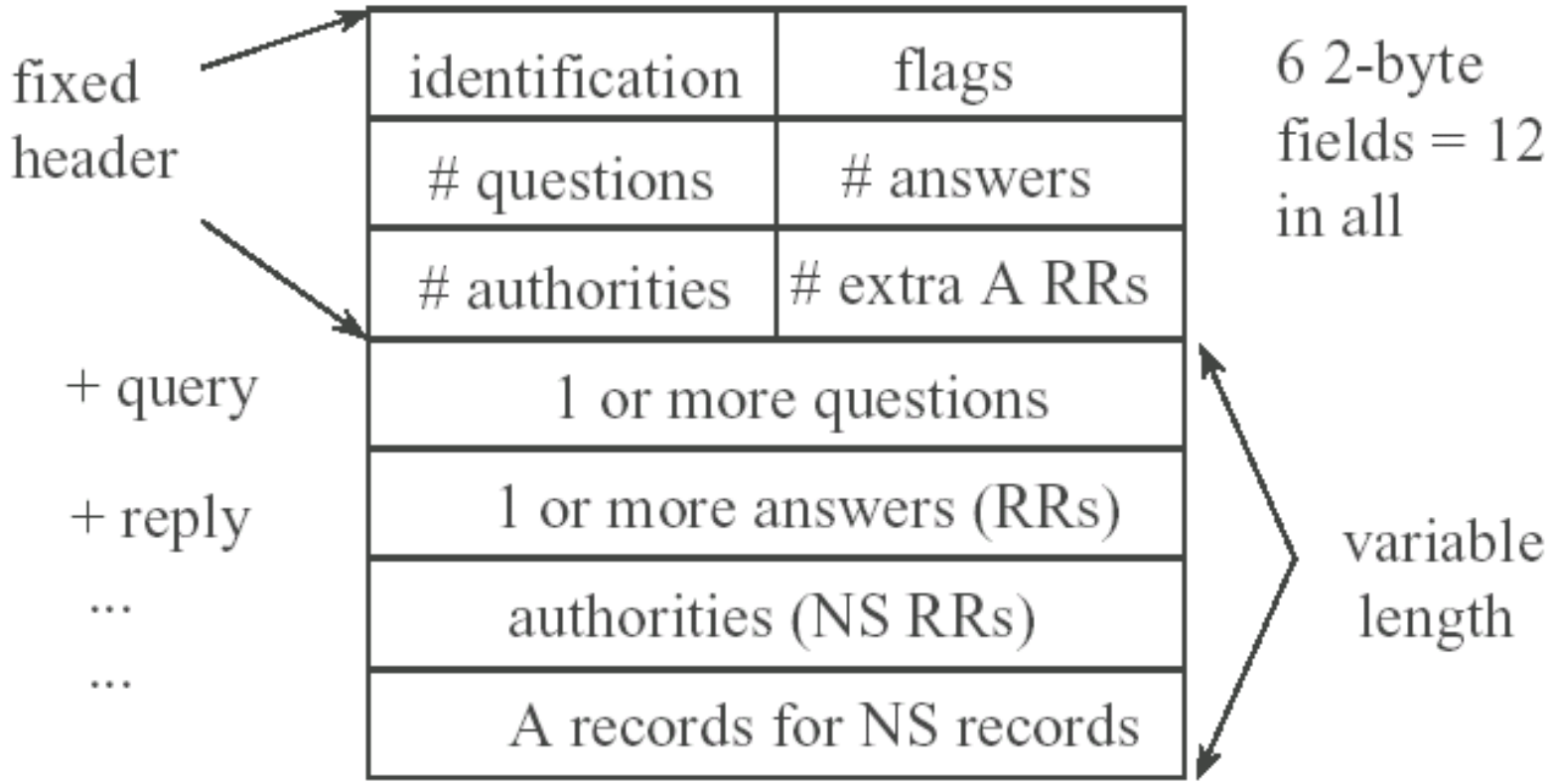
Header

- typically UDP request/reply format
- request = fixed header + question section
- reply = fixed header + query + answer sections
- question = (name, type, class = IP) with name in compressed format
- reply = set of **Resource Records** (RR)

Records

- Each entry in database returned as a specific record type
  - A for IPv4 address
  - AAAA for IPv6 address
- Other types available
  - SRV, TXT etc.

Protocol



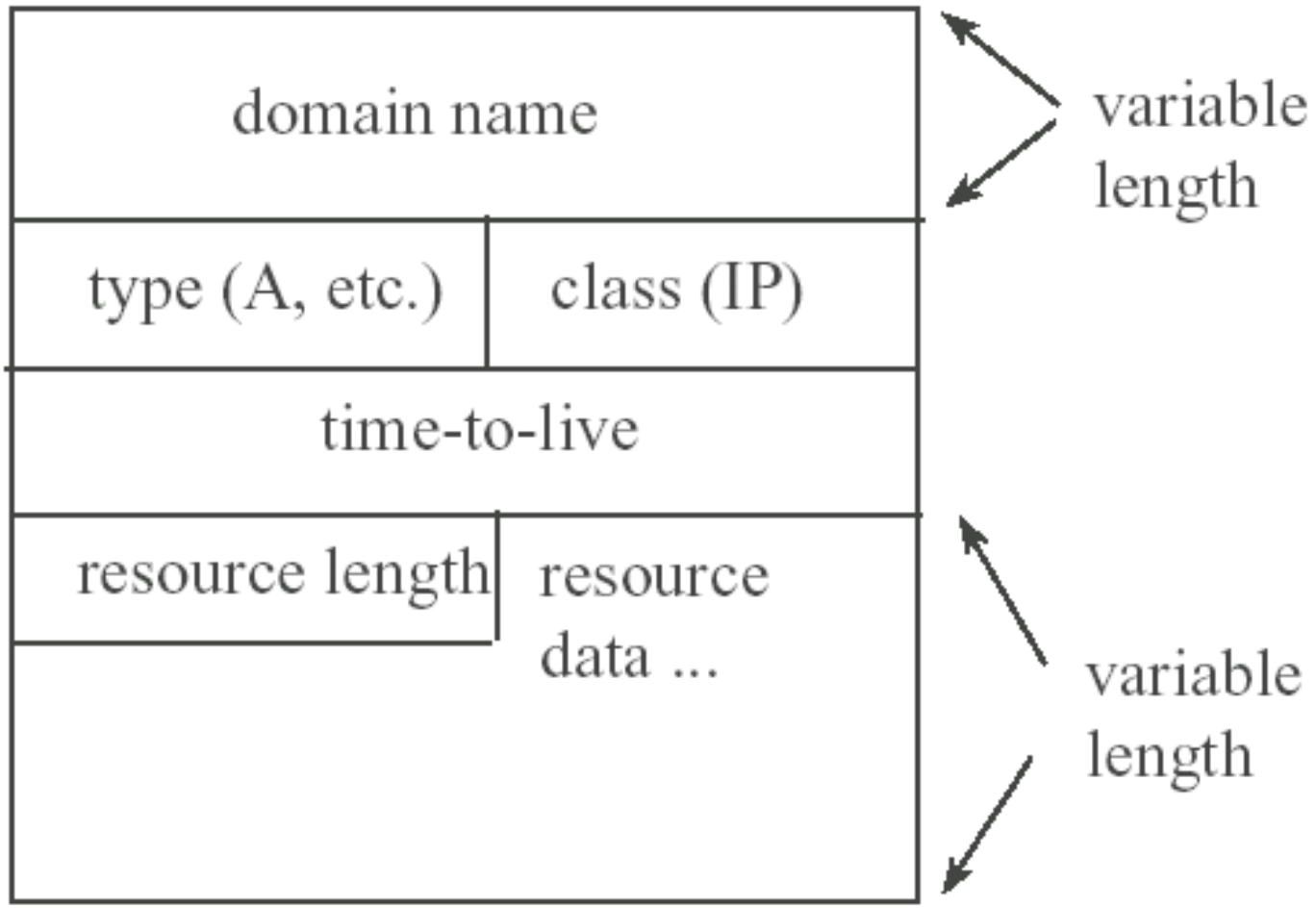
node: id field used by client to match responses

Flags

= 16 bits

QR - 0 if query, 1 if response
opcode - 0, standard query; 1, inverse; 2, server status request
AA - authoritative answer. Answer is from NS that maintains zone
TC - truncated. Using UDP, and answer was > 512 bytes
RD - recursion desired. If not set then iterative, NS may return list of other NS servers to try.
RA - recursion available. Set if server supports recursion
rcode - error codes, 0 means no error. Only set if name invalid.

Resource Record format



RR format

- domain name - same format as in query
- TTL - in seconds, value often 86400, 1 day
- resource data, the answer, for example if the query was **foo.com**, the answer might be **192.12.1.2**

Issues

- Caching:
  - Server may cache records and return non authoritative answer to resolver.
  - Good for scalability and performance (quick reply)
- However, load balancing may be broken.

Issues

- DNS can return different addresses to each query in round robin fashion to do server load balancing
  - Caching violates this scheme
- Security worked on in IETF

Application Types

App	Elastic	Non-elastic	Delay sensitive	BW critical	Errors
Telnet		X	X		
FTP	X			X	
Email	X			Hmm	
Streaming	X		Moderate	X	X
VOIP		X	X	X	X

Application types

- Which applications should use TCP? UDP?
- Point-to-multipoint (multicast)
  - Has to use UDP
- Streaming and real time
  - Low jitter and no retransmissions, use UDP
- No errors, no delay constraints
  - Use TCP
- No errors, delay constraints
  - Use UDP and FEC

EMAIL

- very simple protocol, 7-bit ASCII chars
- uses TCP, port 25
- typically uses DNS MX records
- MIME Multipurpose Internet Mail Extensions,allow “multimedia” mail

SMTP, Request/response protocol

- HELO client.dns.name - say hello to other side
  - 250 server-dns “Howdy”
- MAIL From: user@dns-site - id originator
- RCTP To: user@dns-site - id recipient
- DATA
  - text
  - . - DOT in column 1 for EOF
- QUIT - end of transmission

WWW

- WWW is one overlaid network architecture of many. Other examples are:
  - P2P apps
  - CDN
  - MBONE

HTTP

- Browsers use two basic components
  - HTML (SGML, XML etc.)
  - HTTP
- HTML describes content of a web page
- HTTP shuffles the components across the network

HTTP

- Simple ASCII based protocol
- commands called “methods”, but for the most part, just a variation on “get file”
- server status and errors similar to error strings found in ftp/email
  - 200 - successful
  - 300 - not done yet; e.g., 301 is moved permanently
  - 400 - client error; e.g., 403 forbidden (server refuses)
  - 500 - server error; 503 service unavailable at the moment

RTP

- Streaming and real time media
- UDP has two major drawbacks:
  - Lack support for lost or reordered packets
  - Lack support for jitter compensation
- RTP provides these functions

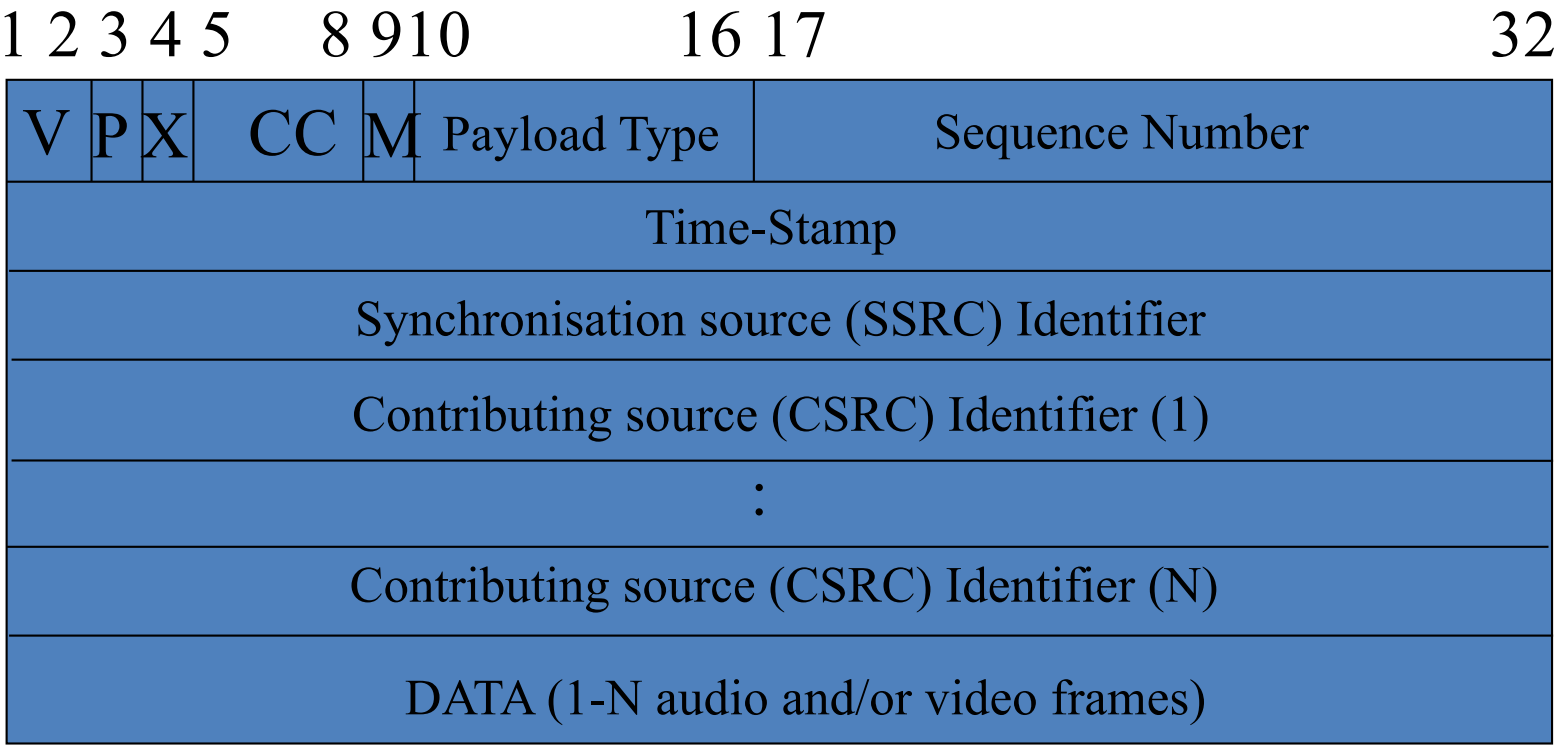
RTP

- Two network functions with RTP:
  - Mixer, mix different sources to one stream
    - New Synchronisation Source Identifier
    - Add old SSRC as Contributing Source Identifiers
  - Translator, manipulates the content, e.g. a transcoder

Why is a flow identifier needed?

I multiple flows come from the same process (audio and video), the receiver has to separate them (same IP and port)

RTP



V – version CC – CSRC count (N < 16)

P – pad M – marker bit

X – extension flag

RTCP

- Out of band control signalling (usually RTP port +1)
- Used to:
  - Integrate different streams
  - Inform about QoS (delay, jitter, packet loss)
  - Distribute information about participants
    - Who is participating in session
    - Who is speaking now

RTCP

- Reports generated:
  - Frequency inv. Proportional to no members
    - Excellent scalability (BW limited)
    - Initial “storm”

Report calculations

- Estimate total bandwidth of session
- Sender period T: $$T = \frac{\#Participants}{0.25 \times 0.05 \times SessionBW} \times avg.\_RTCP\_packet\_size$$
- Receiver period T: $$T = \frac{\#Participants}{0.75 \times 0.05 \times SessionBW} \times avg.\_RTCP\_packet\_size$$
- Next packet = last packet + Max(5s, T) + random (0.5-1.5)
- Random prevents “bunching”

SIP

- Bob wants to call Alice
  - In circuit switched networks easy, send voltage pulse to telephone, it rings.
  - In the Internet?
    - A protocol called SIP
- SIP is used to invite to
  - New sessions
  - Existing sessions

SIP Basics

- Agnostic about
  - Application
  - Media
  - Description
- A wrapper protocol for these functions
- Used to initiate, control and terminate sessions

SIP Basics

- Based on HTML, email, URL
  - Integrating communication

Calling card

Email: bob@abc.net

Phone: +61 2 9531 99865

Fax: +61 2 9531 99844

Web: www.abc.net/~bob

SIP, single identifier bob@abc.net

SIP Entities

- SIP server
  - SIP Redirect Server
  - SIP proxy server
- SIP Registrar
- SIP User Agents

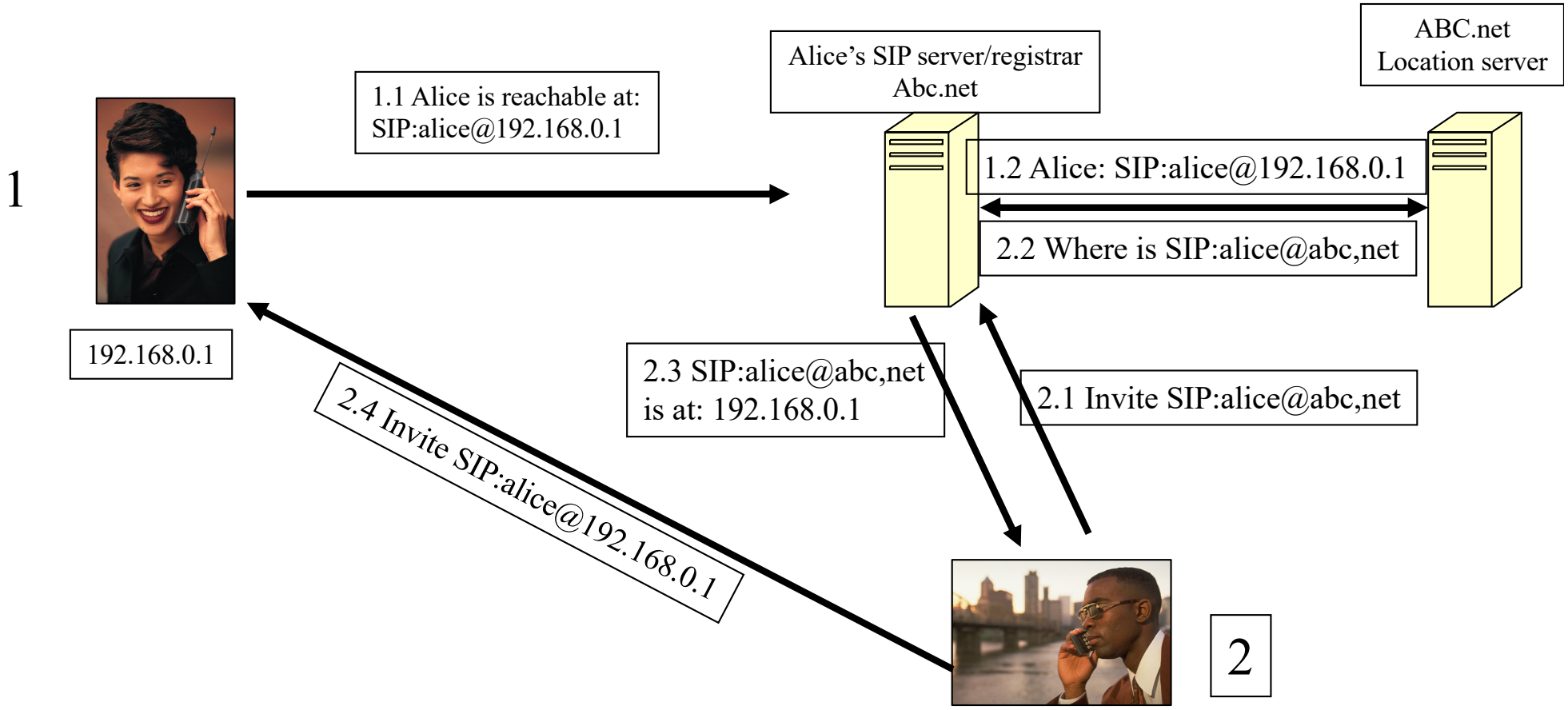
SIP User mobility

- User move around
- Users have different devices
- Devices get dynamic IP addresses
- Where to locate a user?
- SIP uses URL (mail addresses) to identify domain

SIP User Mobility

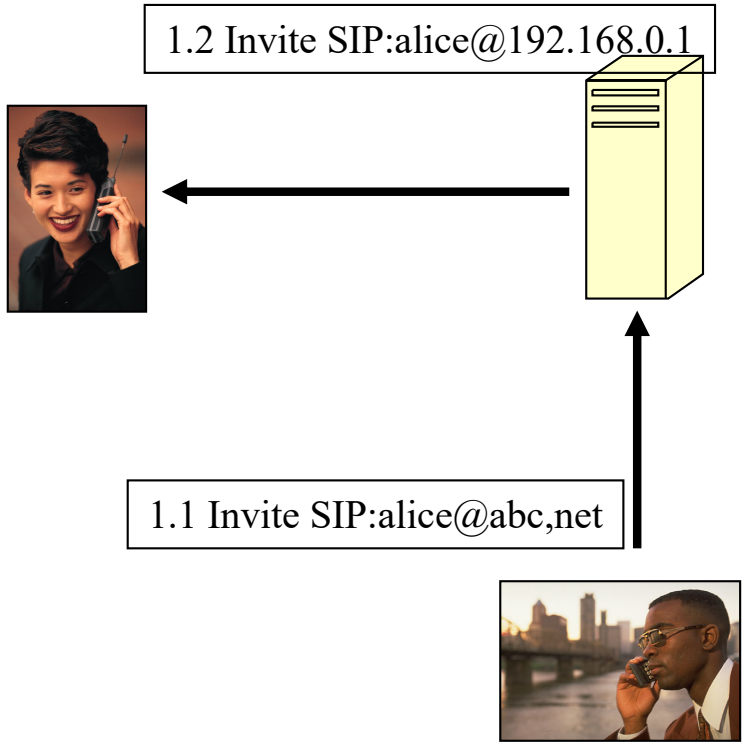
- Registrar accepts user location registrations (SIP)
- Registrar updates Location Server (other protocol)
- Peers query Location server (other protocol)

SIP User Mobility



SIP* Server

- The server can operate in two modes:
  - Proxy (this slide)
  - Redirect (previous slide)



Proxy location

- A SIP proxy can be located using a discovery protocol, e.g.
  - Dynamic Host Configuration Protocol, DHCP
  - Service Location Protocol, SLP
- Or statically entered in the application

SIP UA

- UA is an Application
- Can have User Interface
- SIP request processing

- SIP Servers route SIP messages – SIP UA processes SIP messages

SIP Operation

- Request – response (client – server)
  - User agent client, UAC
  - User agent server, UAS
- Responses:

Range	Response class
100-199	Informational
200-299	Success
300-399	Redirection
400-499	Client error
500-599	Server error
600-699	Global failure

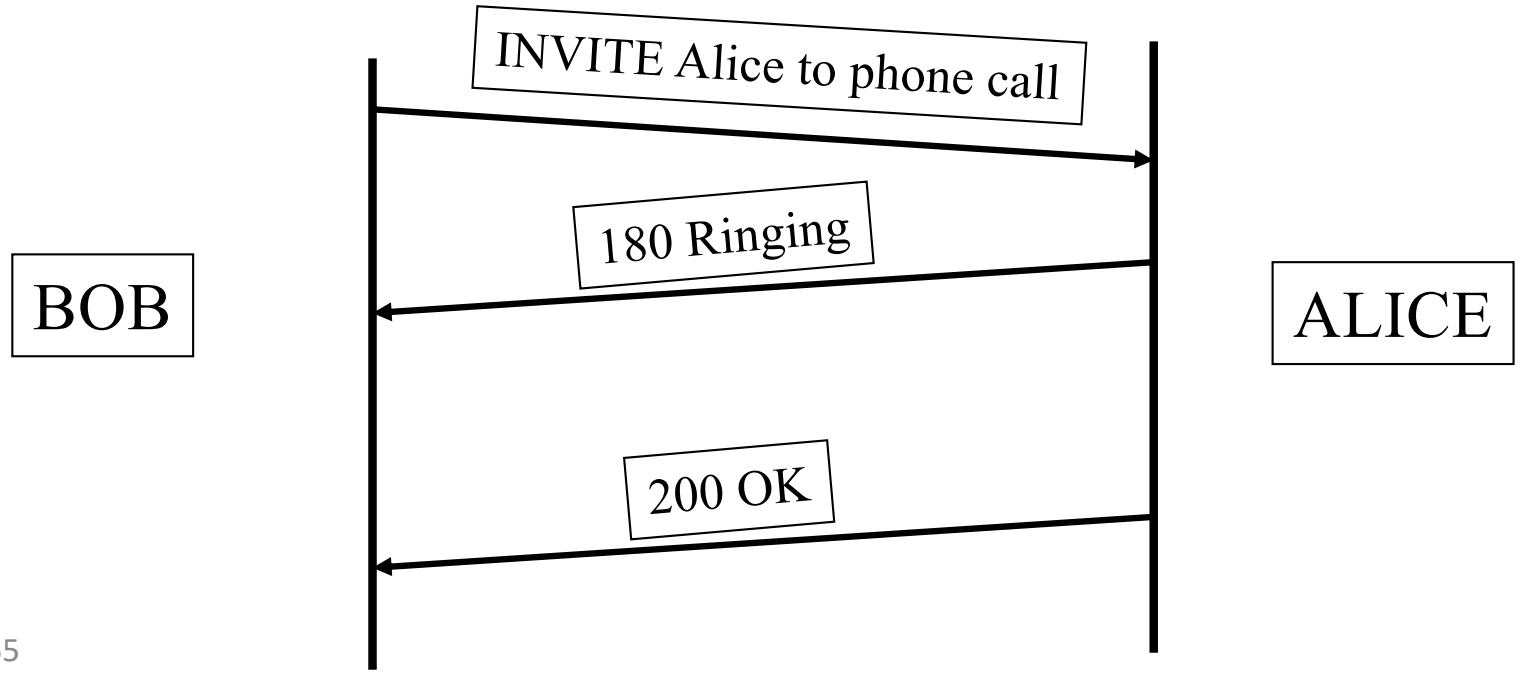
- Contain status code and phrase

Example Responses

- 100 Trying
- 180 Ringing
- 200 OK
- 202 Accepted
- 305 Use proxy
- 400 Bad request
- 403 Forbidden
- 404 Not found
- 408 Request timeout
- 415 Unsupported media type
- 500 Internal server error
- 503 Service unavailable
- 505 SIP version not supported
- 603 Decline

Requests

- INVITE
  - Carries session description (SDP)
  - Typical scenario:



Requests (Methods)

- ACK
  - Issued by client upon receiving final response
  - Three way handshake (tell server client still there after long delay)
- CANCEL
  - Cancels pending invite requests
  - Server returns 200 OK

Requests (Methods)

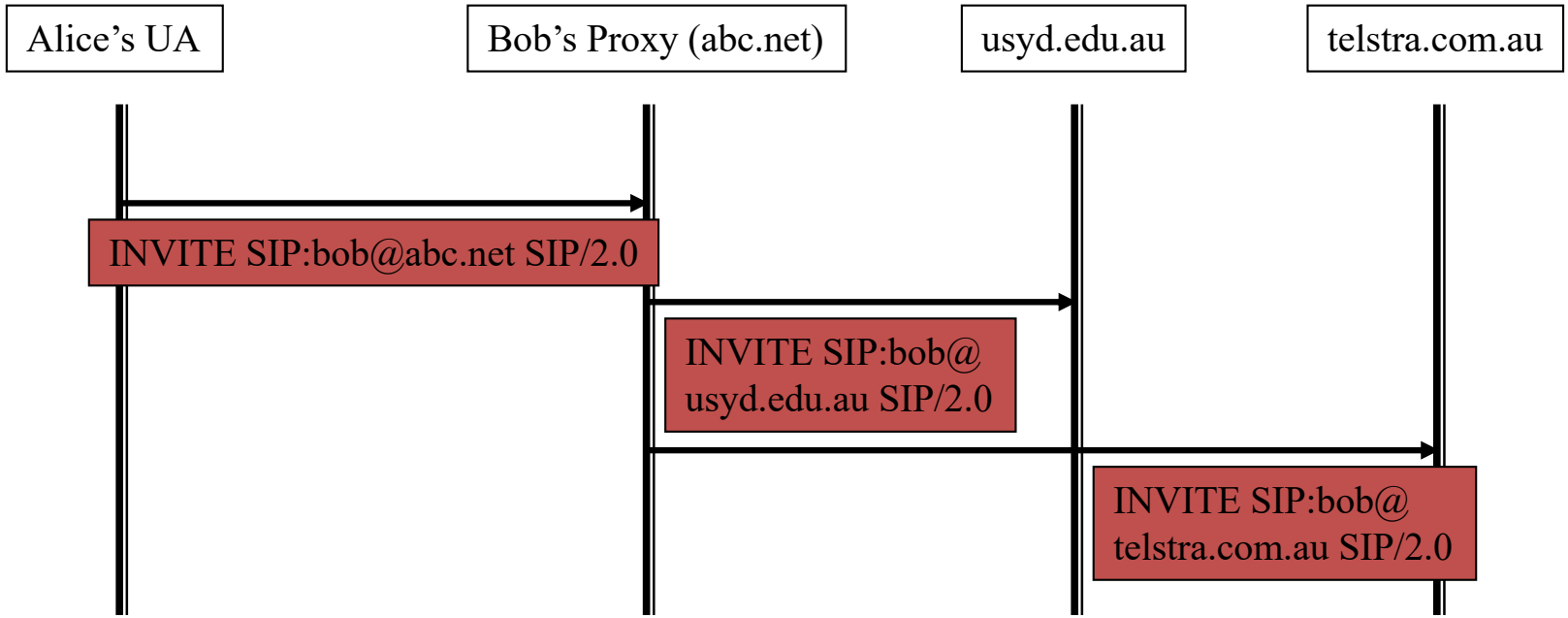
- BYE
  - Two party sessions, terminate session
  - Multi-party sessions, notify leaving – session continues
  - Reply 200 OK
- REGISTER
  - Tell registrar of user location
  - Reply 200 OK
- OPTIONS
  - List what methods a server supports + misc. info.
  - Reply 200 OK

SIP format

- Text based protocol
  - Flexible – inefficient?
- Request format
  - Request line
  - Several headers
  - Empty line
  - Message body
- Response format
  - Status line
  - Several headers
  - Empty line
  - Message body

SIP format

- Request line
  - Method, next hop URI, protocol version



Status line: version, status code, reason string (SIP/2.0 200 OK)

SIP declares that final responses have to be transmitted using a reliable transport protocol, but provisional responses not, why?

Reliable transmission

- Simply because the information whether or not a session has been established is vital. Session progress is optional information.

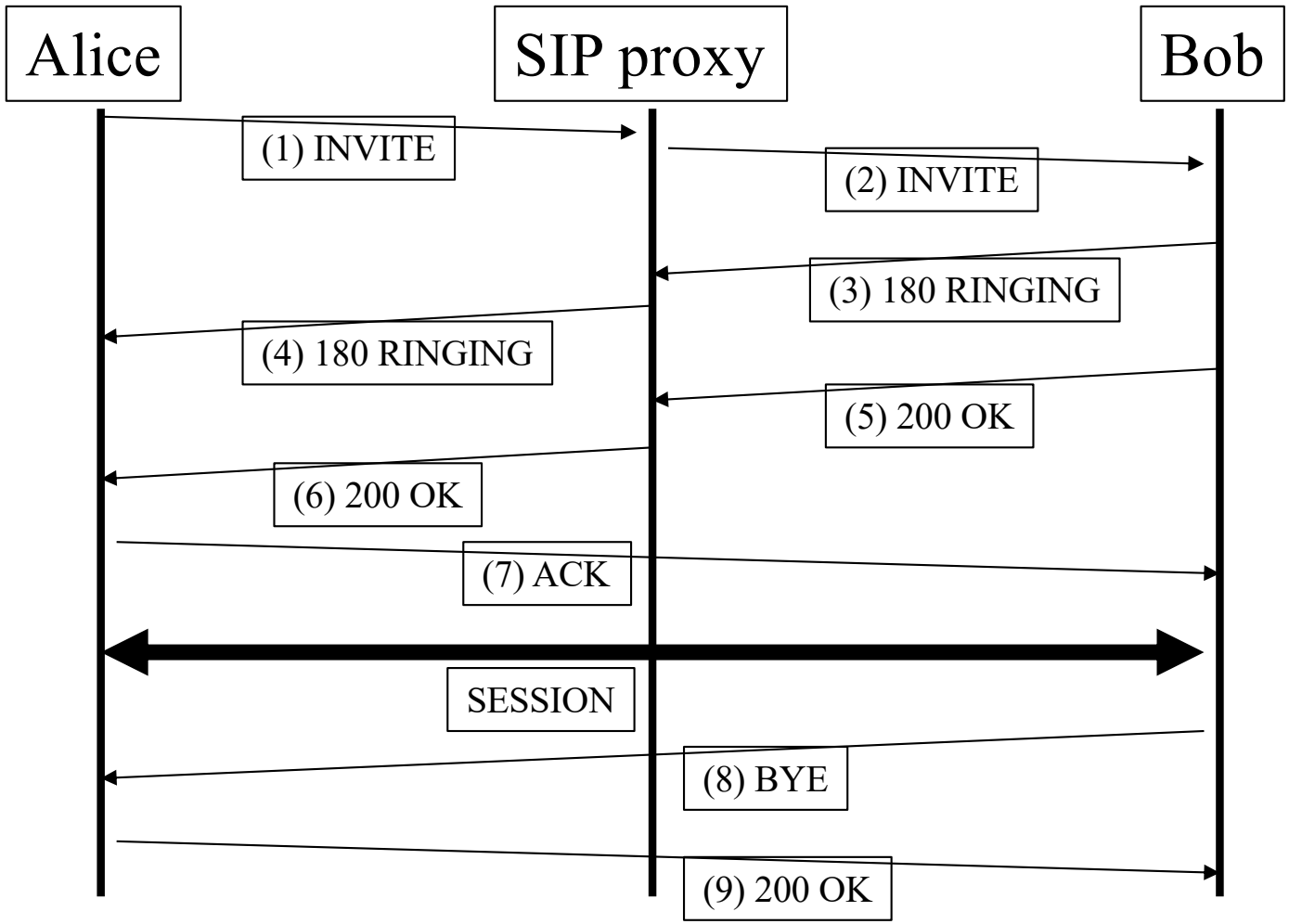
SIP Headers

- Example headers:
  - From: Bjorn Landfeldt bjornl@staff.usyd.edu.au
  - Call-ID: ijnd73nji3978g@192.168.0.1 (distinguish between two sessions with same participants)
  - Contact: Bjorn Landfeldt bjornl@192.168.0.1 (direct route)
  - Record-Route: <SIP:bjornl@staff.usyd.edu.au; maddr=192.168.0.125> (tell that proxy has to be in signalling path)

SIP Bodies

- Separated from headers by blank line
- Proxies ignore bodies
  - Can be encrypted end-to-end
- Typically contains session descriptions
- A SIP message can contain several bodies
  - E.g. SDP description and callers photo

SIP Example


Alice
SIP proxy
Bob
(1) INVITE
(2) INVITE
(3) 180 RINGING
(4) 180 RINGING
(5) 200 OK
(6) 200 OK
(7) ACK
SESSION
(8) BYE
(9) 200 OK