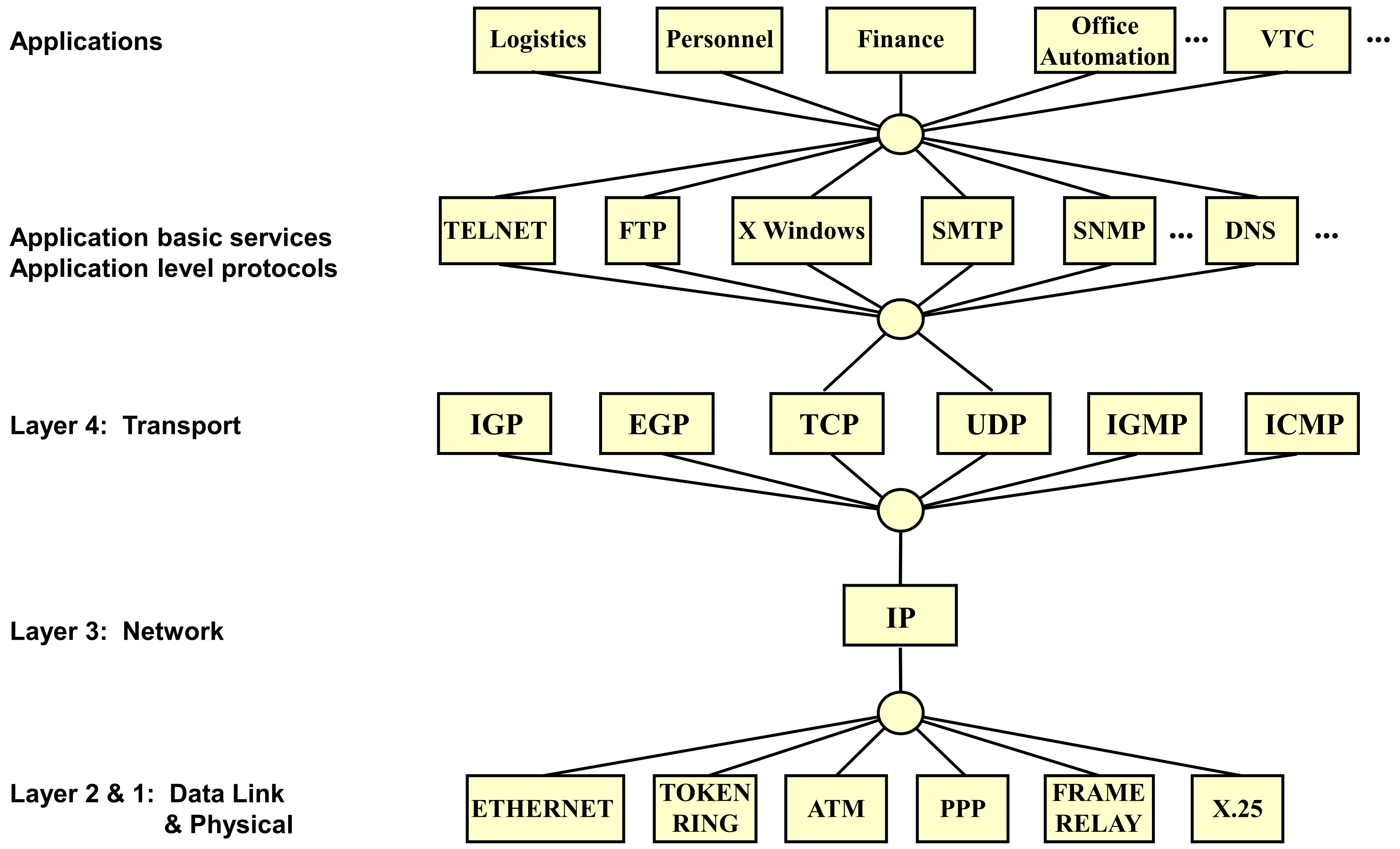
Application Level

Network core: routers, network of networks

Network edge: applications and hosts

- end systems (hosts):
  - run application programs
  - e.g., WWW, email
- client/server model:
  - client host requests, receives service from server
  - e.g., WWW client (browser)/ server; email client/server
- peer-peer model:
  - host interaction symmetric
  - e.g.: teleconferencing

Everything over IP & IP over Everything



Internet protocols provide:

- General-purpose facility for reliable data transfer
- Mechanism for contacting hosts

Application-level protocols provide high-level services:

- DNS, Electronic mail, Remote login,FTP, World Wide Web

All of these applications use *client-server* architecture

Application programs

- Use internet protocols to contact other applications
- Application must interact with protocol software *before* contact is made
- *Listening* application informs local protocol software that it is ready to accept incoming messages
- *Connecting* application uses internet protocol to contact listener
- Applications exchange messages through resulting connection
- Provide *user-level* services

Application Level

Enterprise Systems:

- Engineering/Manufacturing Systems
- Business/Office Systems

Application Systems:

- User Interfaces
- Processing Programs
- Databases and files

Application Support Services:

- Client/Server support
- Distributed OS

Application: Transport Service Requirements

Data loss

–some apps (e.g., audio) can tolerate some loss

–other apps (e.g., file transfer, telnet) require 100% reliable data transfer

Timing

–some apps (e.g., Internet telephony, interactive games) require low delay to be “effective”

Bandwidth

–some apps (e.g., multimedia) require minimum amount of bandwidth to be “effective”

–other apps (“elastic apps”) make use of whatever bandwidth they get

Transport Service Requirements of Common Apps

Application	Data loss	Bandwidth	Time Sensitive
file transfer	no loss	elastic	no
e-mail	no loss	elastic	no
Web documents	loss-tolerant	elastic	no
real-time audio/video	loss-tolerant	audio: 5Kb-1Mb video:10Kb-5Mb	yes, 100's msec
stored audio/video	loss-tolerant	same as above	yes, few secs
interactive games	loss-tolerant	few Kbps up	yes, 100's msec
financial apps	no loss	elastic	yes and no

Services Provided by Internet Transport Protocols

TCP service:

- *connection-oriented:* setup required between client, server
- Client establishes connection to server
- Client terminates connection
- *reliable transport* between sending and receiving process
- *flow control:* sender won't overwhelm receiver
- *congestion control:* throttle sender when network overloaded
- *does not providing:* timing, minimum bandwidth guarantees

Some services use both

•DNS

UDP service:

- unreliable data transfer between sending and receiving process
- does not provide: connection setup, reliability, flow control, congestion control, timing, or bandwidth guarantee
- Message must fit in one UDP datagram

Internet Applications: their protocols and transport protocols

Application	Application layer protocol	Underlying transport protocol
e-mail	smtp [RFC 821]	TCP
remote terminal access	telnet [RFC 854]	TCP
Web	http [RFC 2068]	TCP
file transfer	ftp [RFC 959]	TCP
streaming multimedia	proprietary (e.g. RealNetworks)	TCP or UDP
remote file server	NFS	TCP or UDP
Internet telephony	proprietary (e.g., Vocaltec)	typically UDP

Traditional Distributed Applications

Those based on Internet: distributed applications

Have the following features:

- Application logic
- Application protocol support code
  - Example: The X protocol for transmitting graphical images
- Transport interface code
  - Makes the appropriate network calls to send and receive the messages that make up the application protocol over a specific network transport
  - Usually divided into transport-independent and transport-dependent parts

Middleware provides transparency of the transport interface code

Software between application programs and OS/network.

Provides a set of higher-level distributed computing capabilities and a set of standards-based interfaces.

Interfaces allow applications to be distributed and to take advantage of other services provided over the network.

Middleware can be viewed as a set of (transport) services that are accessible to application programmers through an API (Application Programming Interface)

Example: Sockets, RPC.

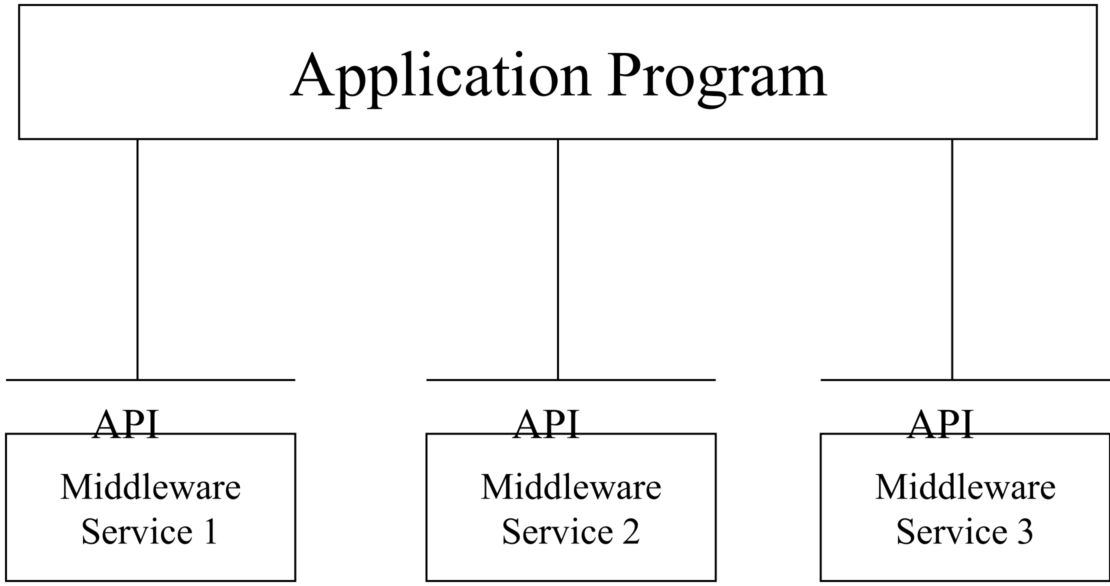
Middleware & API

API: Application Programming Interface

Defines interface between application and transport layer

Socket API: specific Internet API

Two processes communicate by sending data into socket, reading data out of socket



•Defined by programming/operating system

•Includes collection of procedures for application program

•Protocols do not typically specify API

•API defined by programming system

•Allows greatest flexibility - compatibility with different programming systems

•Originated with Berkeley BSD UNIX, also available on Windows and other operating systems

Client-server model

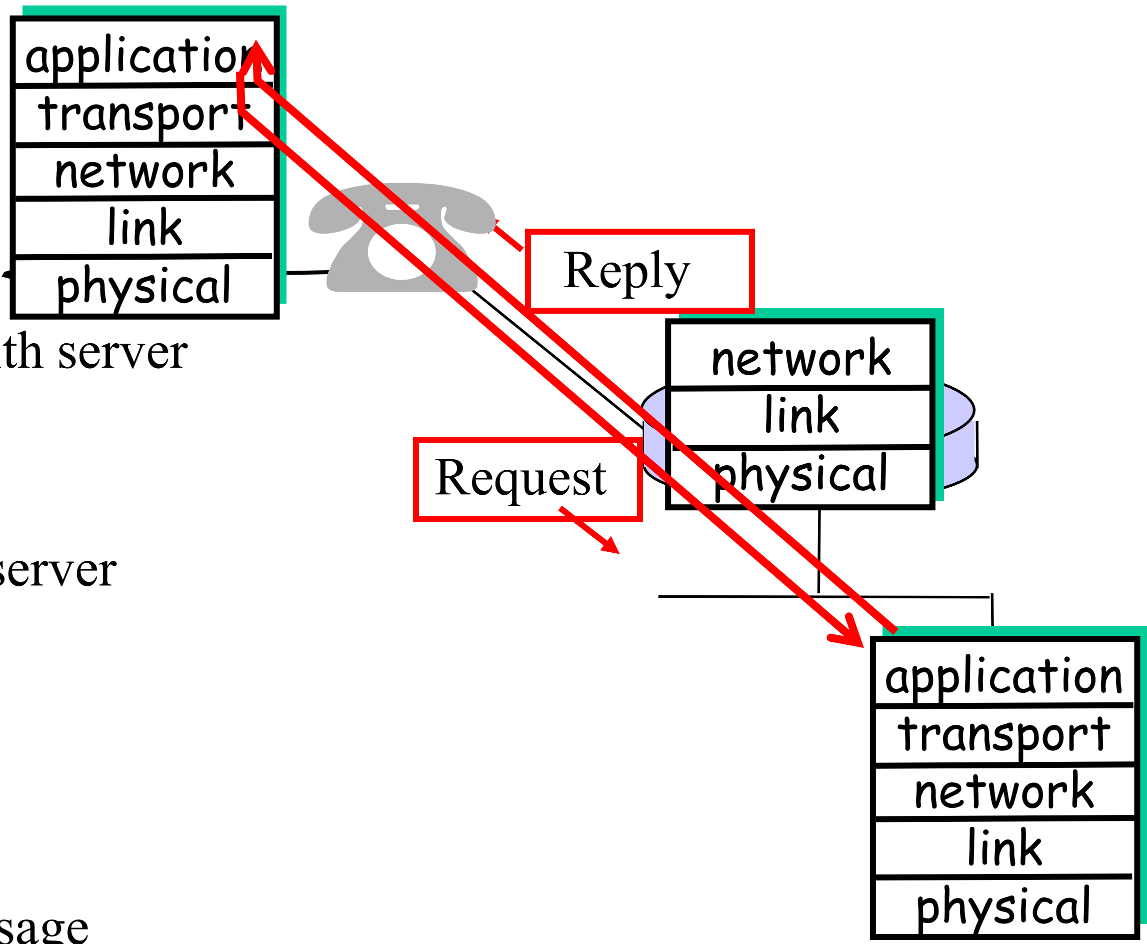
Typical network application has two pieces: *client* and *server*

Client application

- initiates contact (connection) with server ("speaks first")
- Sends message to server
- typically requests service from server
- e.g.: request WWW page
- Waits for return message

Server application is ``listener"

- Waits for incoming message
- Performs service
- Returns results
- e.g., sends requested WWW page.



Domain Name System

The **Domain Name System** (DNS) provides translation between symbolic names for IP hosts and their IP addresses

–People: use many identifiers: name, Passport #, …

–Internet hosts:

- IP address (32 bit) - used for addressing in IP datagrams
- "name", e.g., www.iitb.eirnet.ie - used by humans

Provides logical hierarchical view of the Internet

– DNS: globally *distributed database* implemented in hierarchy of many *name servers*

Functioning:

–*application-layer protocol* to communicate to *resolve* names (address/name translation): application calls *resolver*

–A client/server interaction

clients: query servers to resolve names (*nslookup* function)

servers: run name server daemons, reply to queries *(bind, named*)

–gethostbyname: UNIX based resolver library call that can be invoked from an application program

DNS: Example

Host named: xyz.iitb.ronet.ro wants IP address of web server: www.ibm.com

Resolver Steps:

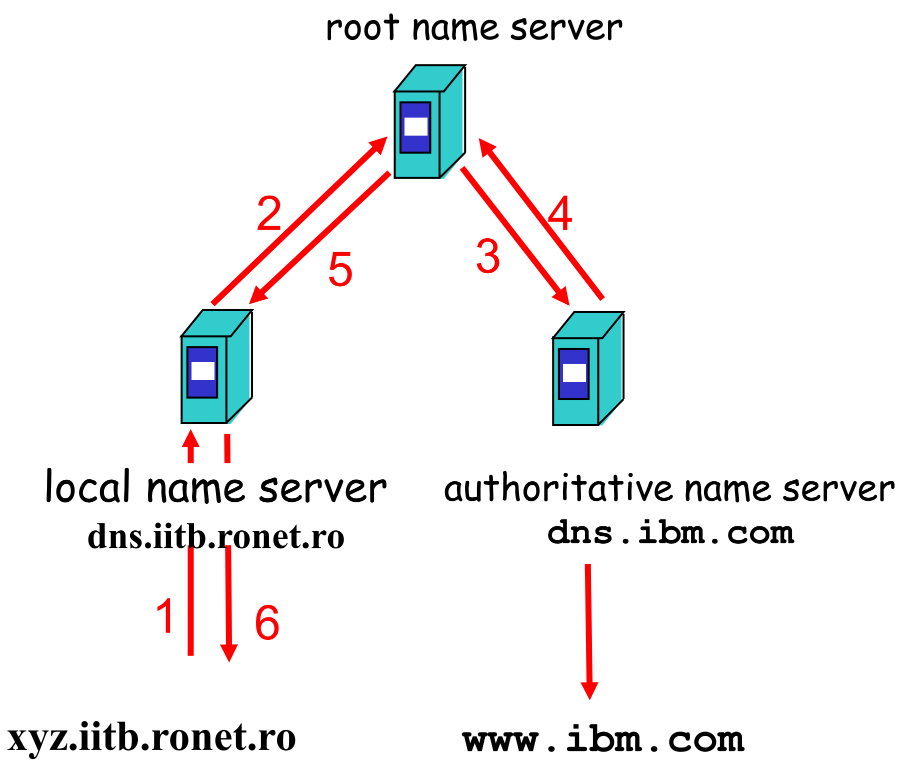
1. After checking locally, contacts its **local DNS server**:

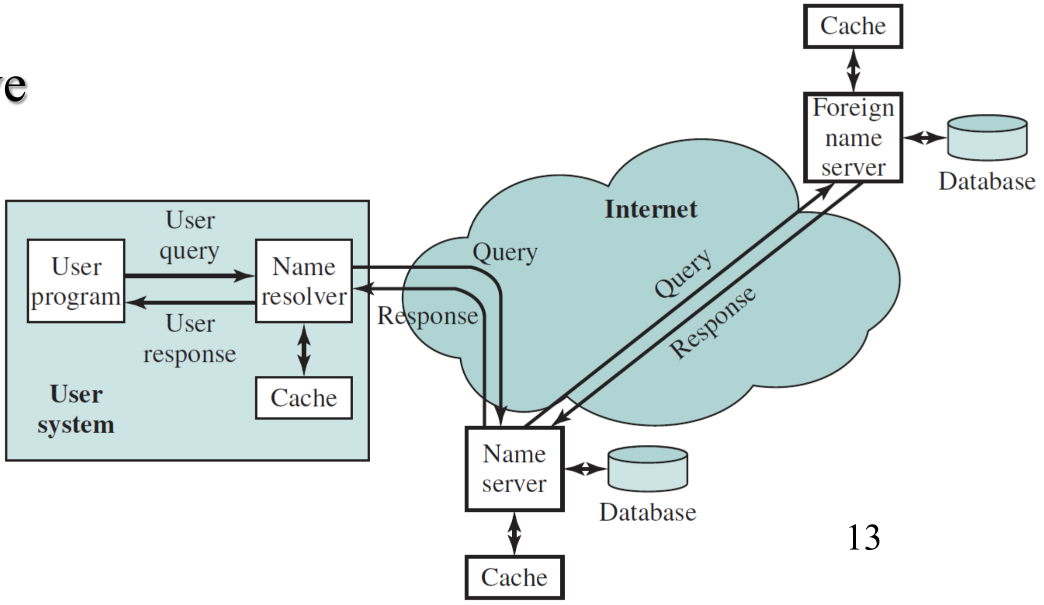
   dns.iitb.ronet.ro

2. dns.iitb.ronet.ro contacts **root name server**, if necessary

3. root name server contacts **authoritative name server** (responsible server): dns.ibm.com, if necessary

4,5,6 Return of info





DNS Name Servers

Centralized DNS?

- single point of failure
- traffic volume
- distant centralized database
- maintenance
- doesn't *scale!*

Distributed DNS:

- no server has all name-to-IP address mappings

Local Name Servers:

- each organization/ISP has *local (default) name server*
- host DNS query first goes to local name server

Authoritative Name Server:

- for a host: stores that host's IP address & name
- can perform name/address translation for that host's name

Root Name Server:

contacts authoritative name server if name mapping not known

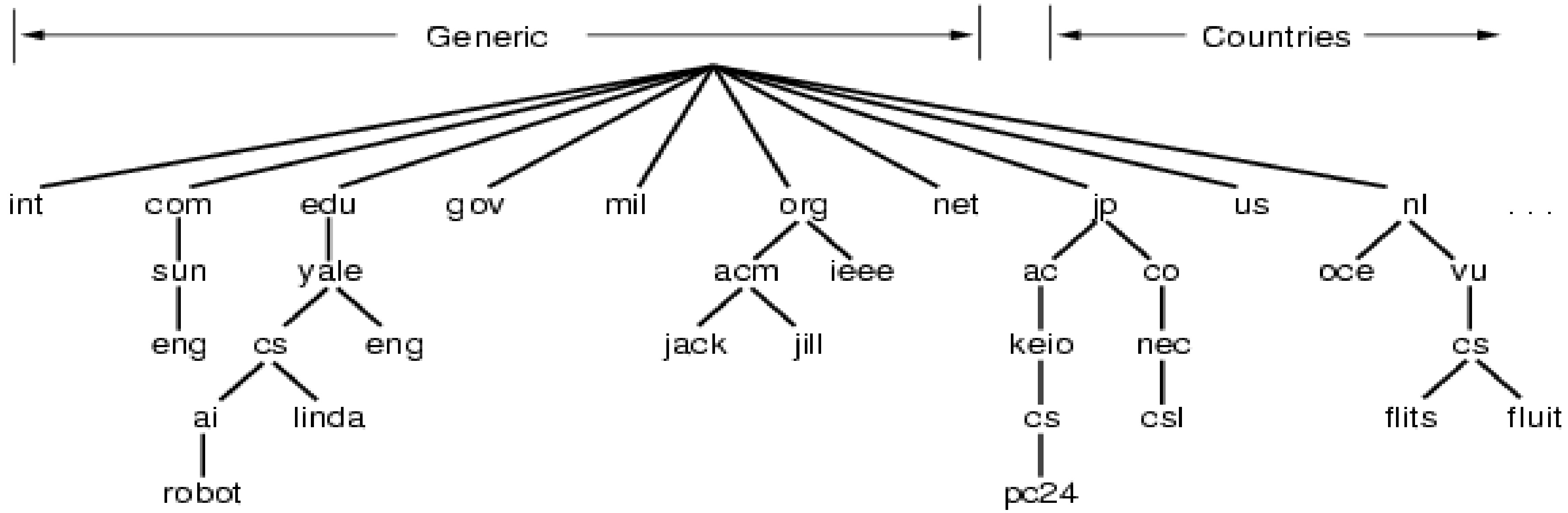
gets mapping

returns mapping to local name server

Several root name servers worldwide

Structure of DNS names

- Each name consists of a sequence of alphanumeric components separated by periods; domain names are case insensitive; each component name up to 63 characters long, entire path up to 255 characters
- Examples: www.eg.bucknell.edu www.netbook.cs.purdue.edu charcoal.eg.bucknell.edu
- Names are hierarchical, with most-significant component on the right
- Left-most component is the computer name



A portion of the Internet domain name space.

DNS naming structure

Top level domains (right-most components; also known as *TLD*s) defined by global authority (only Internet authority, what will be for the future?)

com	Commercial organization
edu	Educational institution
gov	Government organization
mil	Military organization

Organizations apply for names in a top-level domain:
bucknell.edu   macdonalds.com
Organizations determine own internal structure
csis.ul.ie   cs.purdue.edu

Geographic structure

Top-level domains are US-centric (e.g., cs.yale.edu could be accessed as cs.yale.ct.us), each organization in US is under a generic domain

Geographic TLDs are used for organizations in other countries:

TLD	Country
.uk	United Kingdom
.fr	France
.ch	Switzerland
.ie	Ireland

Countries define their own internal hierarchy: ac.uk and .edu.au are used for academic organizations in the United Kingdom and Australia; ac.uk or co.uk are mirrors for .edu or .com

Romania makes not (yet) that distinction, all organizations are under .ro

Choosing DNS server architecture

- Small organizations can use a single DNS server
  - Easy to administer
  - Inexpensive
- Large organizations often use multiple servers
  - Reliability through redundancy
  - Improved response time through load-sharing
  - Delegation of naming authority
- Locality of reference applies - users will most often look up names of computers within same organization

Domain names within an organization

Organizations can create any internal DNS hierarchy

- Uniqueness of TLD and organization name guarantee uniqueness of any internal name (much like file names in your directories)
- All but the left-most component of a domain name, is called the *domain* for that name:

Name	**Domain**
www.netbook.cs.purdue.edu	netbook.cs.purdue.edu
regulus.eg.bucknell.edu	eg.bucknell.edu
coral.bucknell.edu	bucknell.edu

Authority for creating new *subdomains* is delegated to each domain

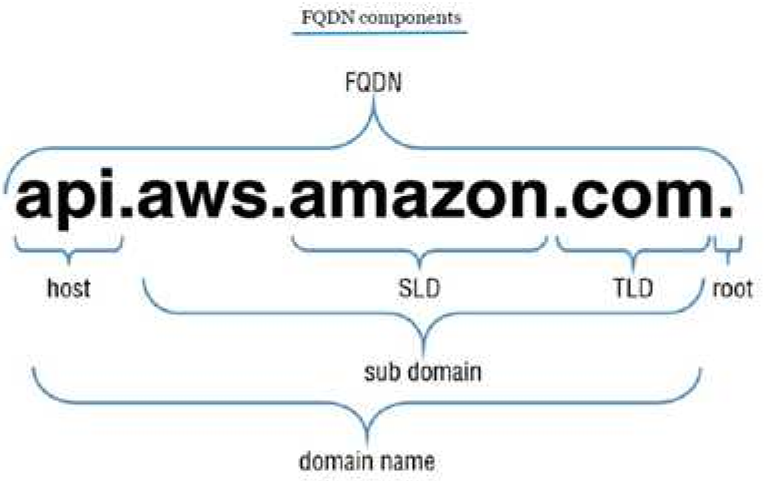
- Administrator of bucknell.edu has authority to create eg.bucknell.edu and need not contact any central naming authority

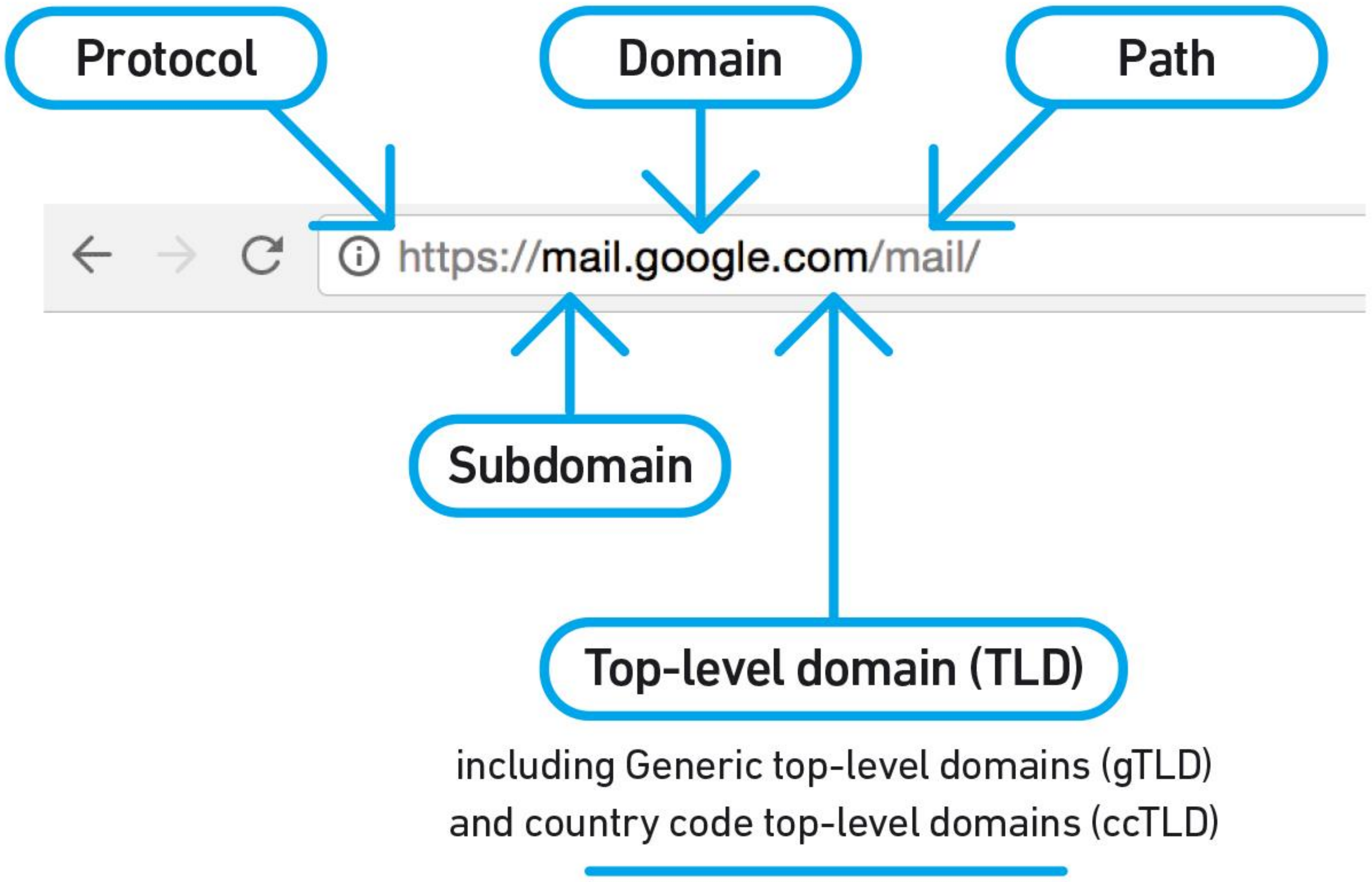
DNS names and physical location

- DNS domains are logical concepts and need not correspond to physical location of organizations
- DNS domain for an organization can span multiple networks
  - utcluj.ro covers all networks at UTCN
  - cs.utcluj.ro all from Computer Science department
  - laptop.cs.utcluj.ro could be connected to a network in … California

Abbreviations

- May be convenient to use abbreviations for local computers; e.g. coral for coral.bucknell.edu
- Abbreviations are handled in the *resolver*; DNS servers only know *full-qualified domain names* (FQDNs)
- Local resolver is configured with list of suffixes to append
- Suffixes are tried sequentially until match found




Protocol
Domain
Path
https://mail.google.com/mail/
Subdomain
Top-level domain (TLD)
including Generic top-level domains (gTLD)
and country code top-level domains (ccTLD)

DNS: Name Resolution

Root name server:

–may not know authoritative name server

–may know *intermediate name server* to contact to find authoritative name server

–Two ways of action:

Recursive queries:

–puts burden of name resolution on contacted name server (each server, having not requested information, goes & finds it, reporting back)

–not scalable under heavy load

Iterated queries:

–contacted server replies only with the name of next server to contact

DNS: Name Resolution (continued)

- Resolver software typically available as library procedures
    - Implement DNS application protocol
    - Configured for local servers
    - Example - UNIX *gethostbyname*
- Calling program is *client*
    - Constructs DNS protocol message - a *DNS request*
    - Sends message to local DNS server
- DNS *server* resolves name
    - Constructs DNS protocol message - a *DNS reply*
    - Sends message to client program and waits for next request

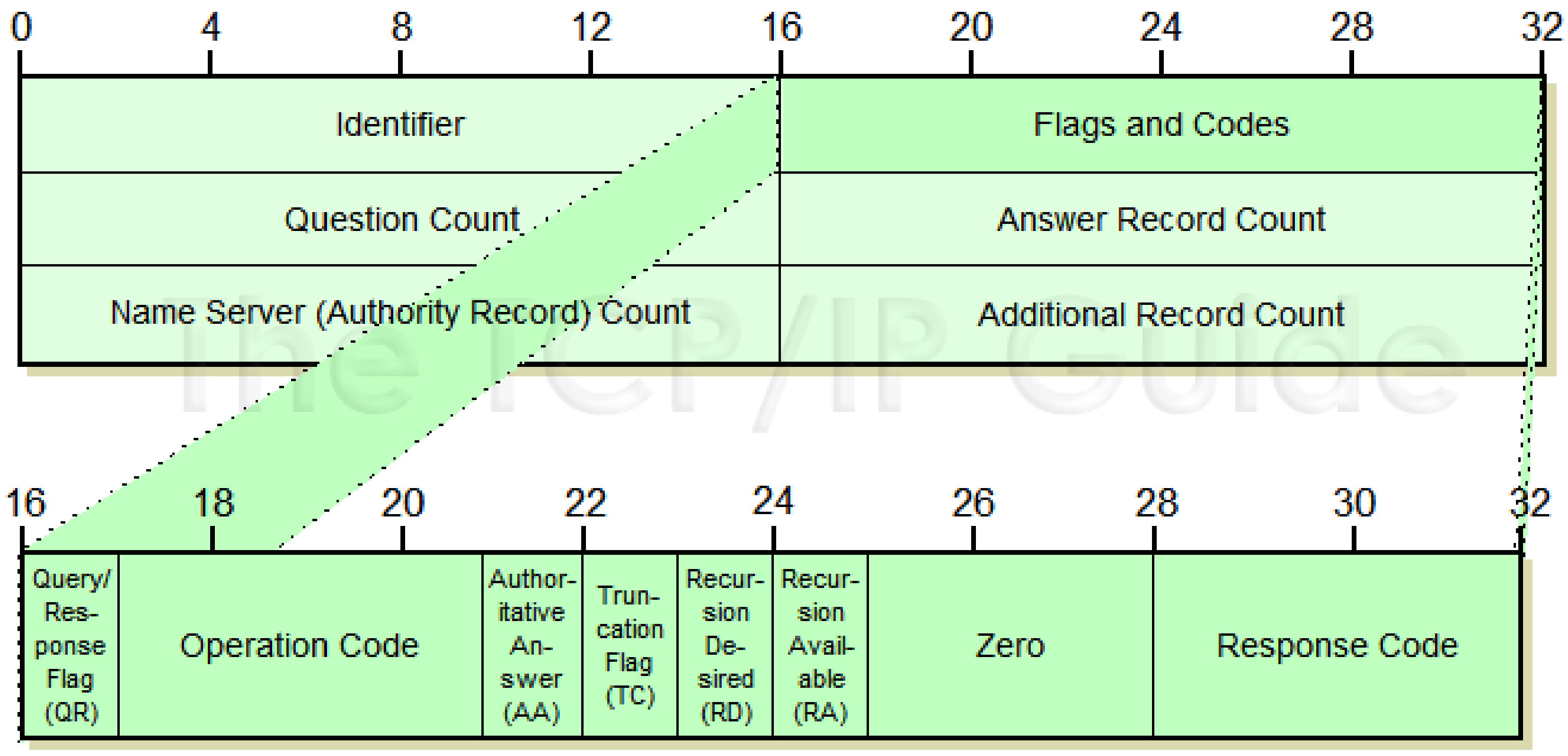
DNS: Database

- Contains:
  - ***resource records (RRs)*** that include the name, IP address, and other information about hosts
- Key features:
  - *Variable-depth hierarchy for names*: DNS allows essentially unlimited levels and uses the period (.) as the level delimiter in printed names
  - *Distribution controlled by the database*: DNS database is divided into thousands of separately managed zones, which are managed by separate administrators. The database software controls distribution and update of records.

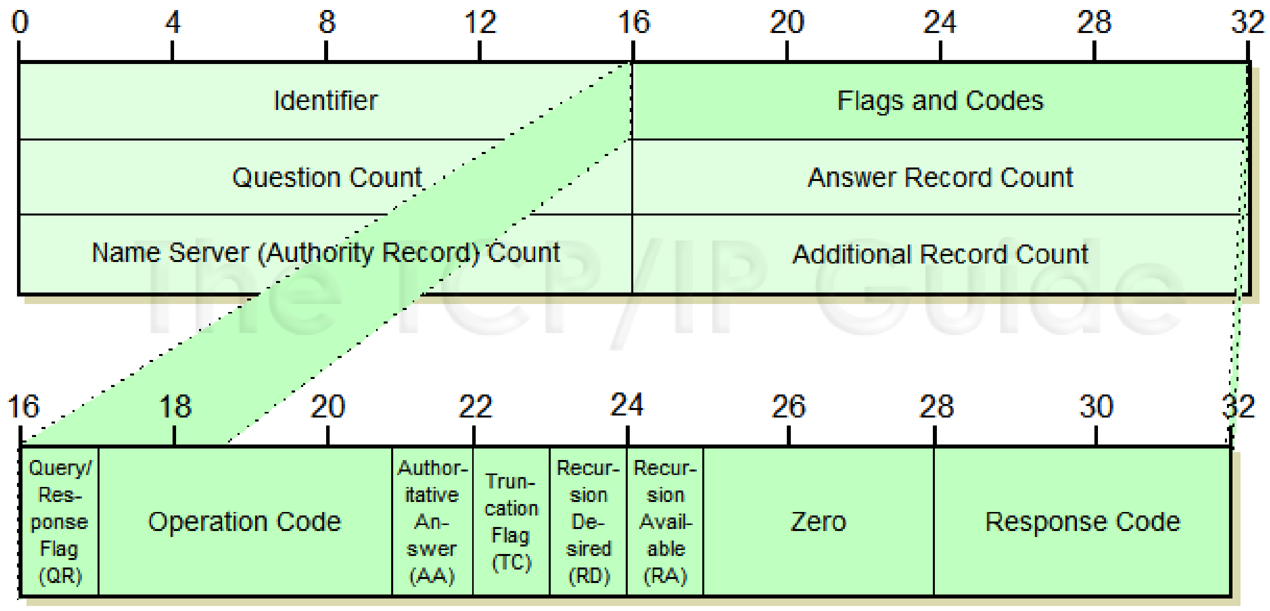
Type	Description
A	A host address. This RR type maps the name of a system to its IPv4 address. Some systems (e.g., routers) have multiple addresses, and there is a separate RR for each.
AAAA	Similar to A type, but for IPv6 addresses.
CNAME	Canonical name. Specifies an alias name for a host and maps this to the canonical (true) name.
HINFO	Host information. Designates the processor and operating system used by the host.
MINFO	Mailbox or mail list information. Maps a mailbox or mail list name to a host name.
MX	Mail exchange. Identifies the system(s) via which mail to the queried domain name should be relayed.
NS	Authoritative name server for this domain.
PTR	Domain name pointer. Points to another part of the domain name space.
SOA	Start of a zone of authority (which part of naming hierarchy is implemented). Includes parameters related to this zone.
SRV	For a given service, provides name of server or servers in domain that provide that service.
TXT	Arbitrary text. Provides a way to add text comments to the database.
WKS	Well-known services. May list the application services available at this host.

DNS messages (main idea)

DNS request contains name to be resolved
DNS reply contains IP address for the name in the request



Identifier: A 16-bit identification field generated by the device that creates the DNS query. It is copied by the server into the response, so it can be used by that device to match that query to the corresponding reply received from a DNS server.
Question Count: Specifies the number of questions in the *Question* section of the message.
Answer Record Count: Specifies the number of resource records in the *Answer* section of the message.
Authority Record Count: Specifies the number of resource records in the *Authority* section of the message
Additional Record Count: Specifies the number of resource records in the *Additional* section of the message.



www.tcpipguide.com/

Email – Electronic Mail

Electronic mail paradigm

Heavily used application on any network

Electronic version of paper-based office memo

Quick, low-overhead written communication

Dates back to time-sharing systems, in 1960s

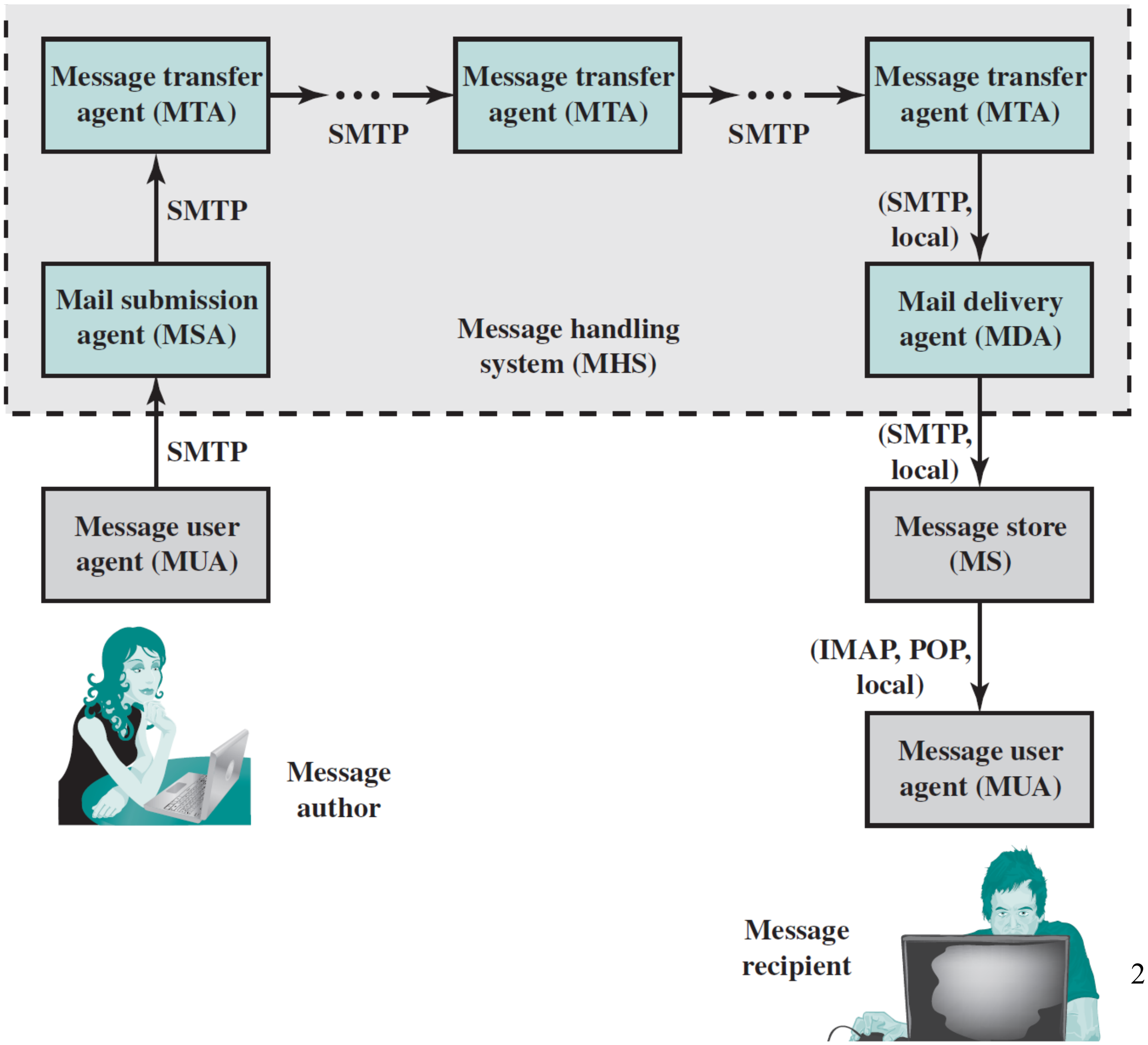
Because e-mail is encoded in an electronic medium, new forms of interaction are possible

Automatic processing - sorting, reply

Can carry other content: if basic **Simple Mail Transfer Protocol** (SMTP), based on TCP/IP, was delivering only simple text messages, by use of **Multi-purpose Internet Mail Extension** (MIME) now have delivery of other types of data (voice, images, video clips,…)

History: first standard: CCITT X.400 - too complex; base for OSI’s MOTIS application; replaced by TCP/IP based standards RFC 821(transmission protocol), and RFC 822 (message format)

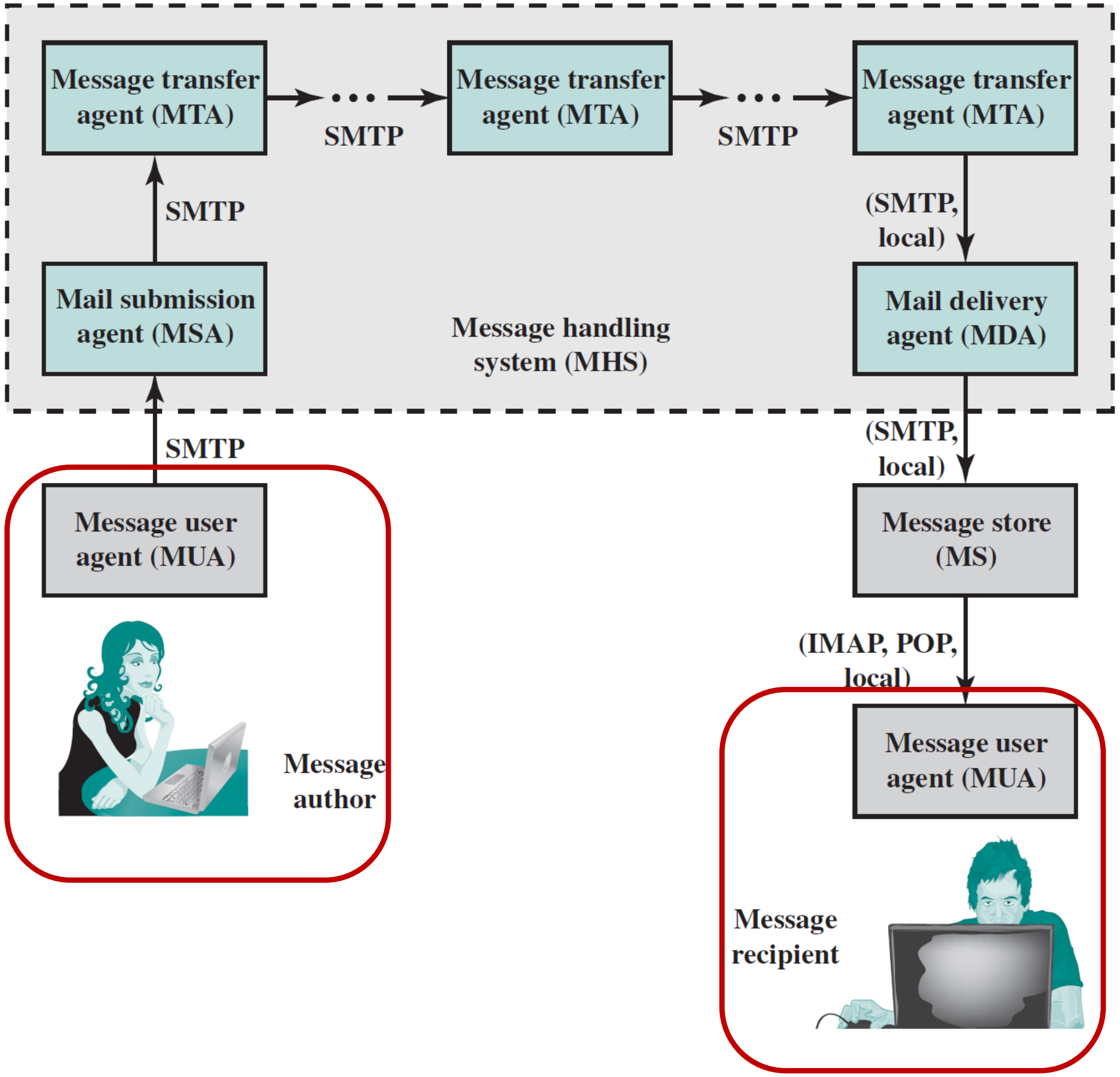
Email Architecture RFC 5598 (*Internet Mail Architecture*)


Message transfer agent (MTA)
SMTP
Message transfer agent (MTA)
SMTP
Message transfer agent (MTA)
SMTP
(SMTP, local)
Mail submission agent (MSA)
Message handling system (MHS)
Mail delivery agent (MDA)
SMTP
(SMTP, local)
Message user agent (MUA)
Message store (MS)
(IMAP, POP, local)
Message author
Message user agent (MUA)
Message recipient

Email Architecture - components

Message User Agent (MUA)

- –on behalf of user actors and user applications
- –client e-mail program or a local network e-mail server
- –sender MUA: formats a message and performs initial submission into the MHS via an MSA
- –receiver MUA: processes receive mail for storage and/or display
- –Thunderbird, Outlook, Opera Mail, Mailbird, etc


Message transfer agent (MTA)
SMTP
Message transfer agent (MTA)
SMTP
Message transfer agent (MTA)
SMTP
(SMTP, local)
Mail submission agent (MSA)
Message handling system (MHS)
Mail delivery agent (MDA)
SMTP
(SMTP, local)
Message user agent (MUA)
Message store (MS)
(IMAP, POP, local)
Message author
Message user agent (MUA)
Message recipient

Email Architecture - components

Message Handling Service (MHS)

<u>Mail Submission Agent (MSA)</u> – component of MTA

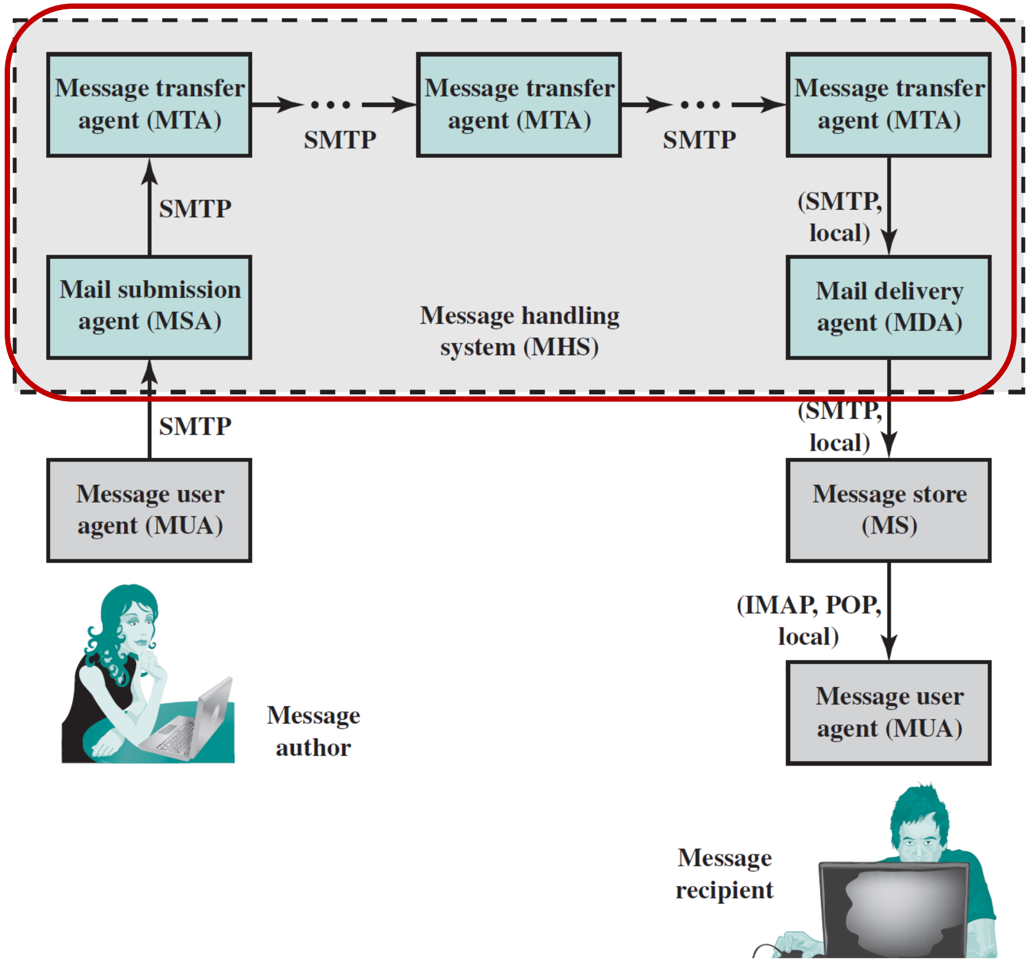
- –accepts messages from MUA
- –enforces the policies of the hosting domain and the requirements of Internet standards
- –Simple Mail Transfer Protocol (SMTP) is used between the MUA and the MSA.

<u>Message Transfer Agent (MTA)</u>

- –Relays mail for one application-level hop
- –SMTP is used between MTAs
- –client: the sending mail server, server: receiving mail server

<u>Mail Delivery Agent (MDA)</u> – component of MTA

- –transfers messages from the MHS to the MS

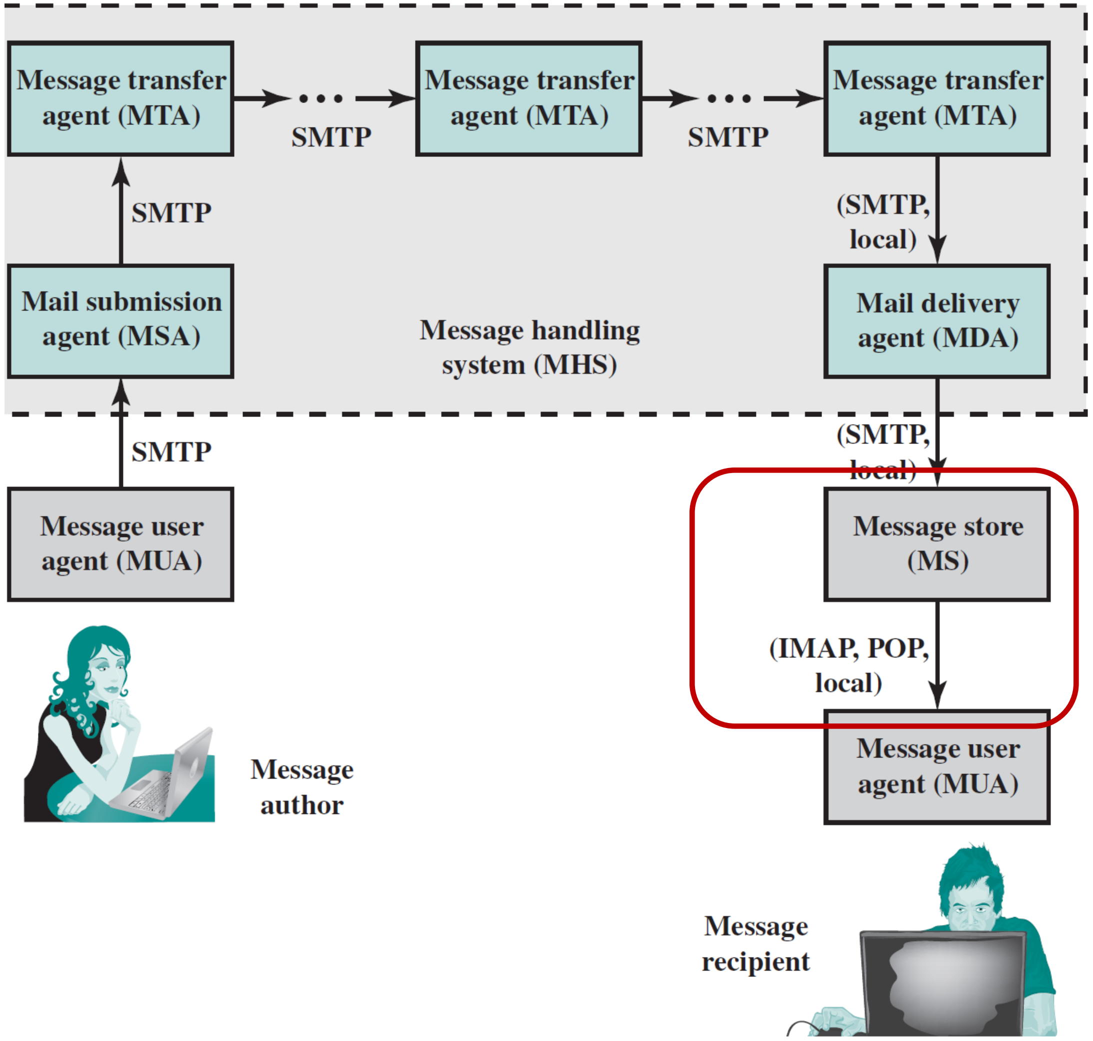


Message transfer agent (MTA)
SMTP
Message transfer agent (MTA)
SMTP
Message transfer agent (MTA)
SMTP
(SMTP, local)
Mail submission agent (MSA)
Message handling system (MHS)
Mail delivery agent (MDA)
SMTP
(SMTP, local)
Message user agent (MUA)
Message store (MS)
(IMAP, POP, local)
Message author
Message user agent (MUA)
Message recipient

Email Architecture - components

Message Store (MS)

- –located on a remote server or on the same machine as the MUA
- –retrieve message from remote server using:
- –POP (Post Office Protocol) or IMAP (Internet Message Access Protocol).


Message transfer agent (MTA)
SMTP
Message transfer agent (MTA)
SMTP
Message transfer agent (MTA)
SMTP
(SMTP, local)
Mail submission agent (MSA)
Message handling system (MHS)
Mail delivery agent (MDA)
SMTP
(SMTP, local)
Message user agent (MUA)
Message store (MS)
(IMAP, POP, local)
Message user agent (MUA)
Message author
Message recipient

Post Office Protocol (POP) and Internet Message Access Protocol (IMAP)

Role: message user agent (MUA) retrives mail from message store (MS)

POP version 3 - RFC 1939; connection to server using TCP on port 110

- *Authentication state*: user ID/password or more sophisticated methods
- *Transaction state*: the client can access the mailbox to retrieve and delete messages
- *Update state*: the server passes all of the changes requested by the client's commands and then closes the connection

IMAP version 4 - RFC 3501; connection to server using TCP on port 143 or 993 over SSL

provides more functionality than POP3

Clients can:

- have multiple remote mailboxes
- specify criteria for downloading messages
- make changes both when connected and when disconnected (periodic re-synchronization)
- IMAP ***always*** keeps messages on the server and replicates copies to the clients

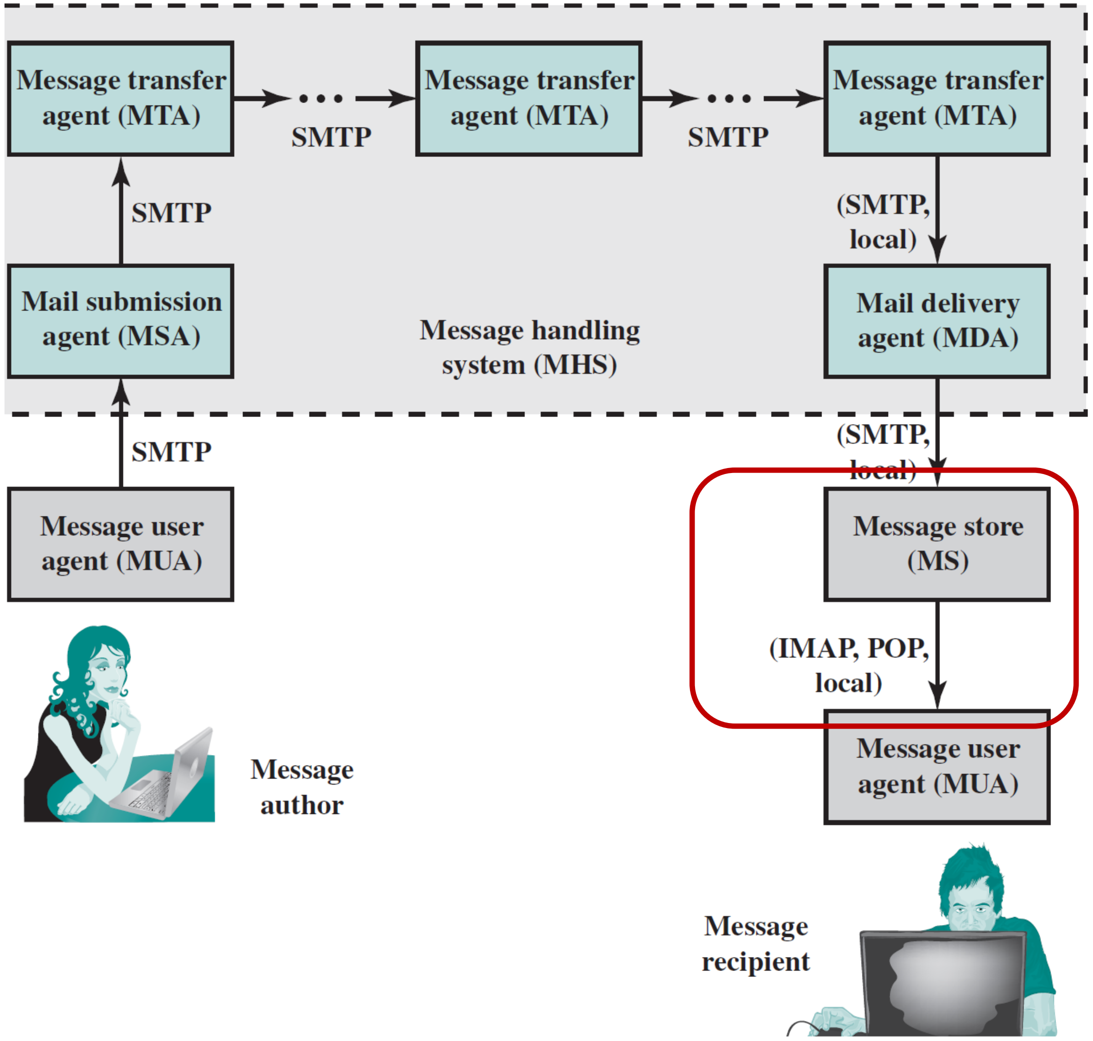
Email Architecture - components

Other components

Internet e-mail provider named also Administrative management domain (ADMD)

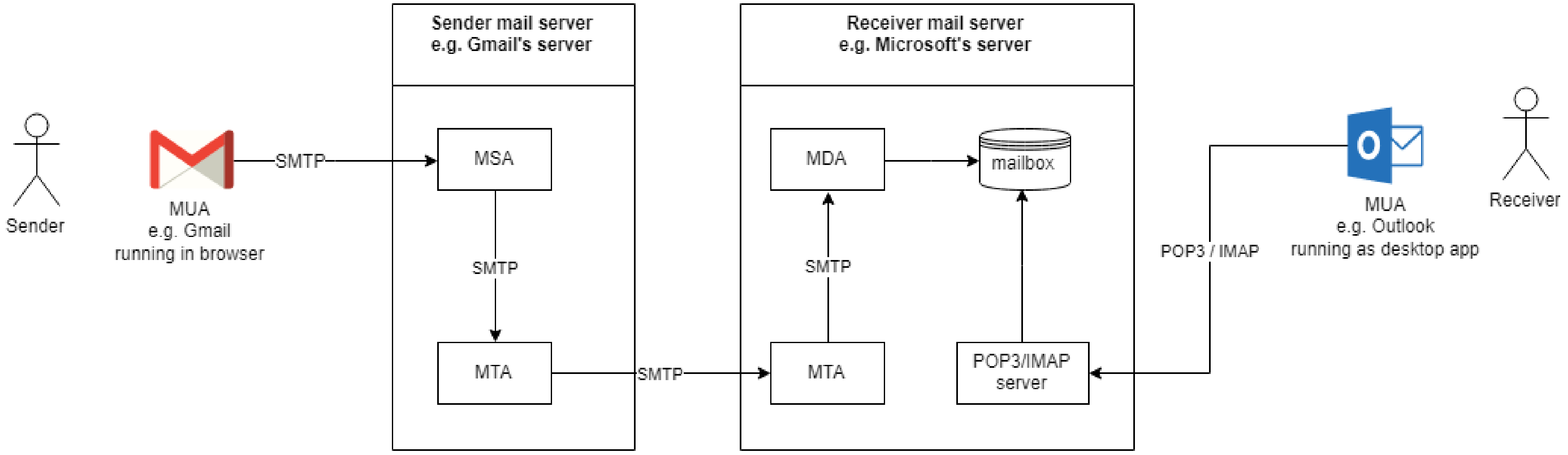
- –examples:
  - –IT department that operates a mail relay (local or enterprise)
  - –an ISP that operates a public shared e-mail service

Domain name system (DNS)


Message transfer agent (MTA)
SMTP
Message transfer agent (MTA)
SMTP
Message transfer agent (MTA)
SMTP
(SMTP, local)
Mail submission agent (MSA)
Message handling system (MHS)
Mail delivery agent (MDA)
SMTP
(SMTP, local)
Message user agent (MUA)
Message store (MS)
(IMAP, POP, local)
Message author
Message user agent (MUA)
Message recipient

Example

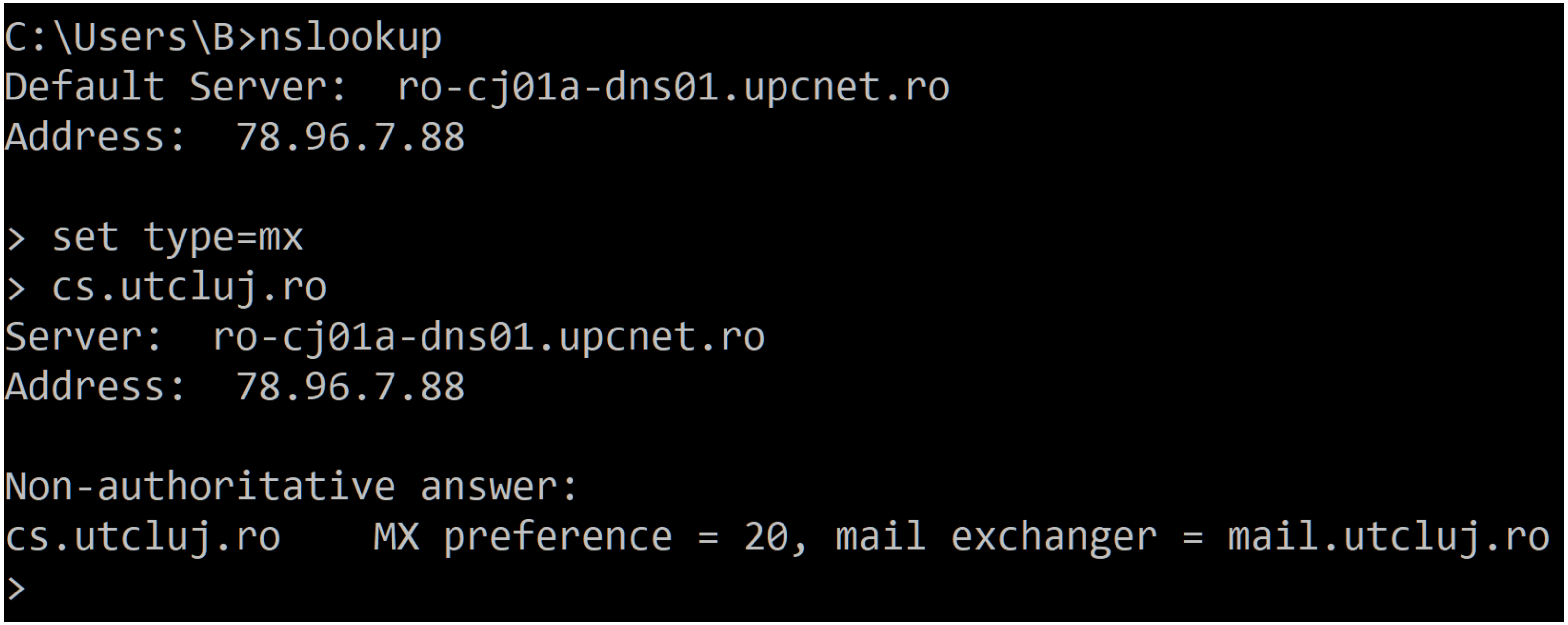
- From Gmail user to Microsoft user
  - *also, security (Public key infrastructure)*



https://afreshcloud.com/sysadmin/mail-terminology-mta-mua-msa-mda-smtp-dkim-spf-dmarc

Example – steps to send an email to *Firstname.Lastname@cs.utcluj.ro*

- Sender Message User Agent (MUA): web-based email interface (gmail)
- Mail Submission Agent (MSA): get message from MUA and sends it to MTA via SMTP
- Sender Message Transfer Agent (MTA): query MX record for *cs.utcluj.ro*
  - DNS 'mail exchange' (MX) record -> directs email to a mail server (see example below for a *nslookup* of *MX type)*



```
C:\Users\B>nslookup
Default Server:  ro-cj01a-dns01.upcnet.ro
Address:  78.96.7.88

> set type=mx
> cs.utcluj.ro
Server:  ro-cj01a-dns01.upcnet.ro
Address:  78.96.7.88

Non-authoritative answer:
cs.utcluj.ro    MX preference = 20, mail exchanger = mail.utcluj.ro
>
```

- Receiver Message Transfer Agent (MTA): responsible for mail.utcluj.ro
- Mail Delivery Agent (MDA): put's the email in the receiver's inbox (***Firstname.Lastname@cs.utcluj.ro*** *vs* ***localname@mail.utcluj.ro***)
- Receiver Message User Agent (MUA): receiver's outlook client (configured for utcluj email / web interface - intranet.utcluj.ro)

Mailbox: Electronic mailbox

E-mail users have an *electronic mailbox* into which incoming mail is deposited; identified by an *e-mail address*

User then accesses mail with a mail reader program

Usually associated with computer account (user's account ID); one user may have different electronic mailboxes

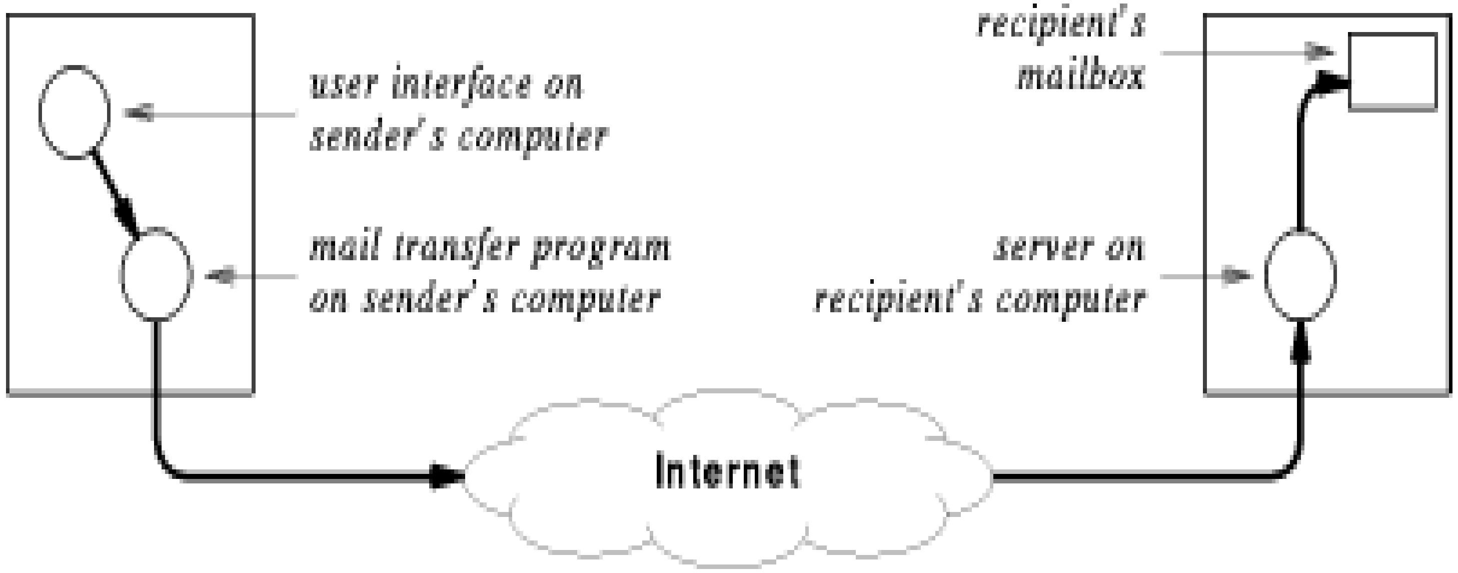
On networked multi-user computer, need to identify computer as well as *mailbox* :

e-mail address is composed of computer name and mailbox name

Mail transfer

- E-mail communication is really a two-part process:
    - User composes mail with an *e-mail interface* program
    - *Mail transfer* program delivers mail to destination
        - Waits for mail to be placed in outgoing message queues
        - Picks up message and determines recipient(s)
        - Becomes *client* and contacts *server* on recipient's computer
        - Passes message to server for delivery

Mail Transfer Illustration


user interface on sender's computer
mail transfer program on sender's computer
Internet
recipient's mailbox
server on recipient's computer

Simple Mail Transfer Protocol (SMTP) is the standard application protocol for delivery of mail from source to destination (RFC 821)

•Provides reliable delivery of messages

•Uses TCP well-known port 25 for message exchange between client and server

•Command/Response interaction:

commands: ASCII text (message character set as 7-bit ASCII)

response: status code and phrase

Other functions:

E-mail address lookup & address verification

General characteristics

Three phases of transfer: handshaking, mail transfer, closure

Attempts to provide reliable service

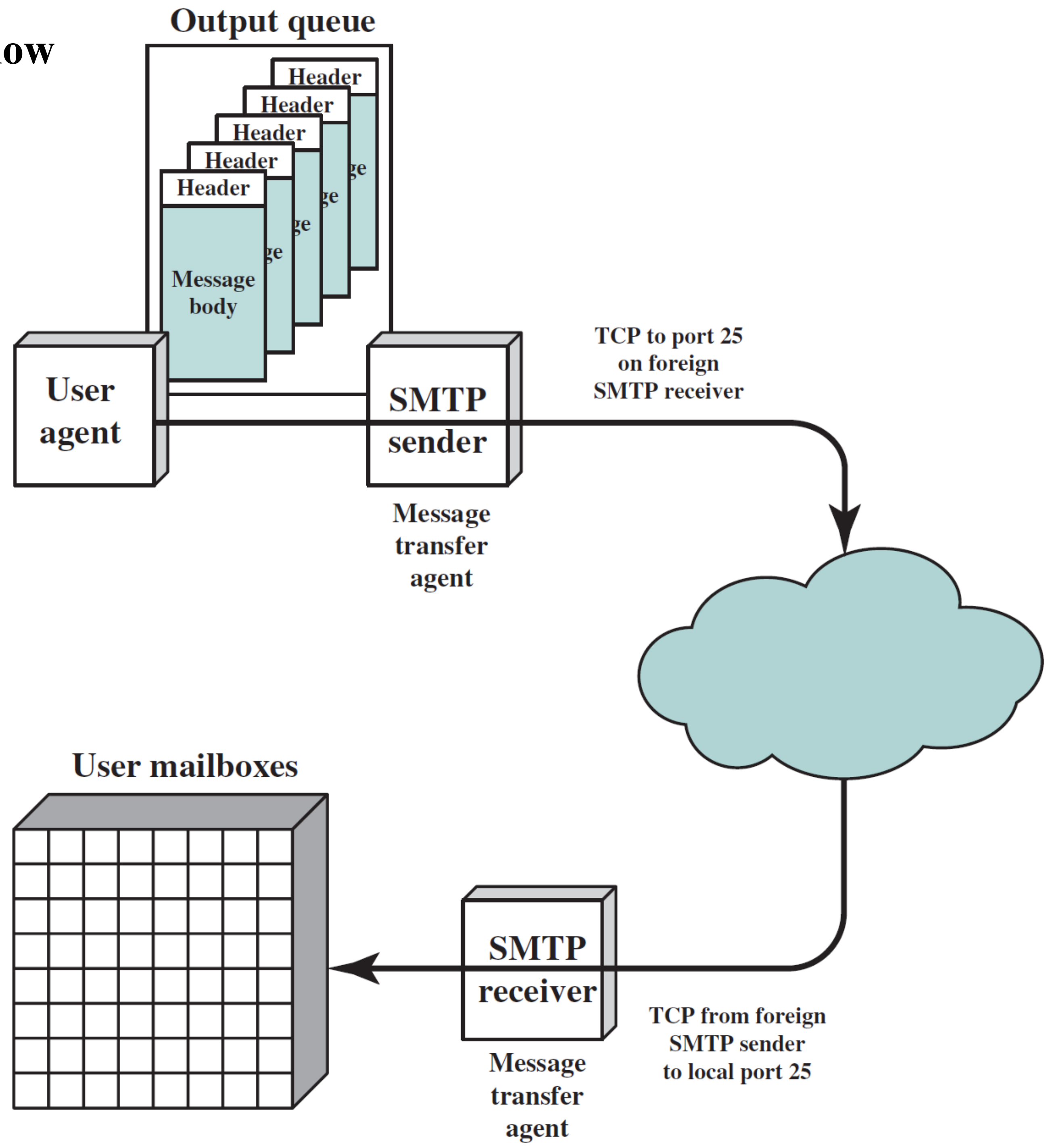
No guarantee to recover lost messages

No end to end acknowledgement to originator

Error indication delivery not guaranteed

Generally considered reliable!

SMTP Mail Flow



Mail Message Contents

Each queued message has in composition:

- Message text
  - RFC 822 header with: message envelope and list of recipients
  - Message body, composed by user
- A list of mail destinations
  - Derived by user agent from header
  - May be listed in header
  - May require expansion of mailing lists
  - May need replacement of mnemonic names with mailbox names

If Blind Carbon Copies (BCC) indicated, user agent needs to prepare correct message format

SMTP Sender

- Takes message from queue
- Transmits to proper destination host
    - Via SMTP transaction
    - Over one or more TCP connections to port 25
- Host may have multiple senders active
- Host should be able to create receivers on demand
- When delivery complete, sender deletes destination from list for that message
- When all destinations processed, message is deleted

Optimization

- If message destined for multiple users on a given host, it is sent only once
    - Delivery to users handled at destination host
- If multiple messages ready for given host, a single TCP connection can be used
    - Saves overhead of setting up and dropping connection

Possible Errors:

Host unreachable

Host out of operation

TCP connection fail during transfer

Sender can re-queue mail

- Give up after a period

Faulty destination address

- User error
- Target user changed address
- Redirect if possible
- Inform user if not delivered

SMTP Protocol - Reliability

Used to transfer messages from sender to receiver over TCP connection

Attempts to provide reliable service

No guarantee to recover lost messages

No end to end acknowledgement to originator

Error indication delivery not guaranteed

Generally considered reliable!

SMTP Receiver

- Accepts arriving message
- Places in user mailbox or copies to outgoing queue for forwarding
- Receiver must:
  - Verify local mail destinations
  - Deal with errors
    - Transmission
    - Lack of disk space
- Sender responsible for message until receiver confirm complete transfer
  - Indicates mail has arrived at host, not user

SMTP Forwarding

- Mostly direct transfer from sender host to receiver host
- May go through intermediate machine via forwarding capability
  - Sender can specify route
  - Target user may have moved

Format for Text Messages
RFC 882

Message viewed as having <u>envelope</u> and <u>contents</u>

Envelope contains information required to transmit and deliver message

Message is sequence of lines

of text

- Uses general memo framework
- Header usually keywor followed by colon followed by arguments

Header	Meaning
To:	Email address(es) of primary recipient(s)
Cc:	Email address(es) of secondary recipient(s)
Bcc:	Email address(es) for blind carbon copies
From:	Person or people who created the message
Sender:	Email address of the actual sender
Received:	Line added by each transfer agent along the route
Return-Path:	Can be used to identify a path back to the sender

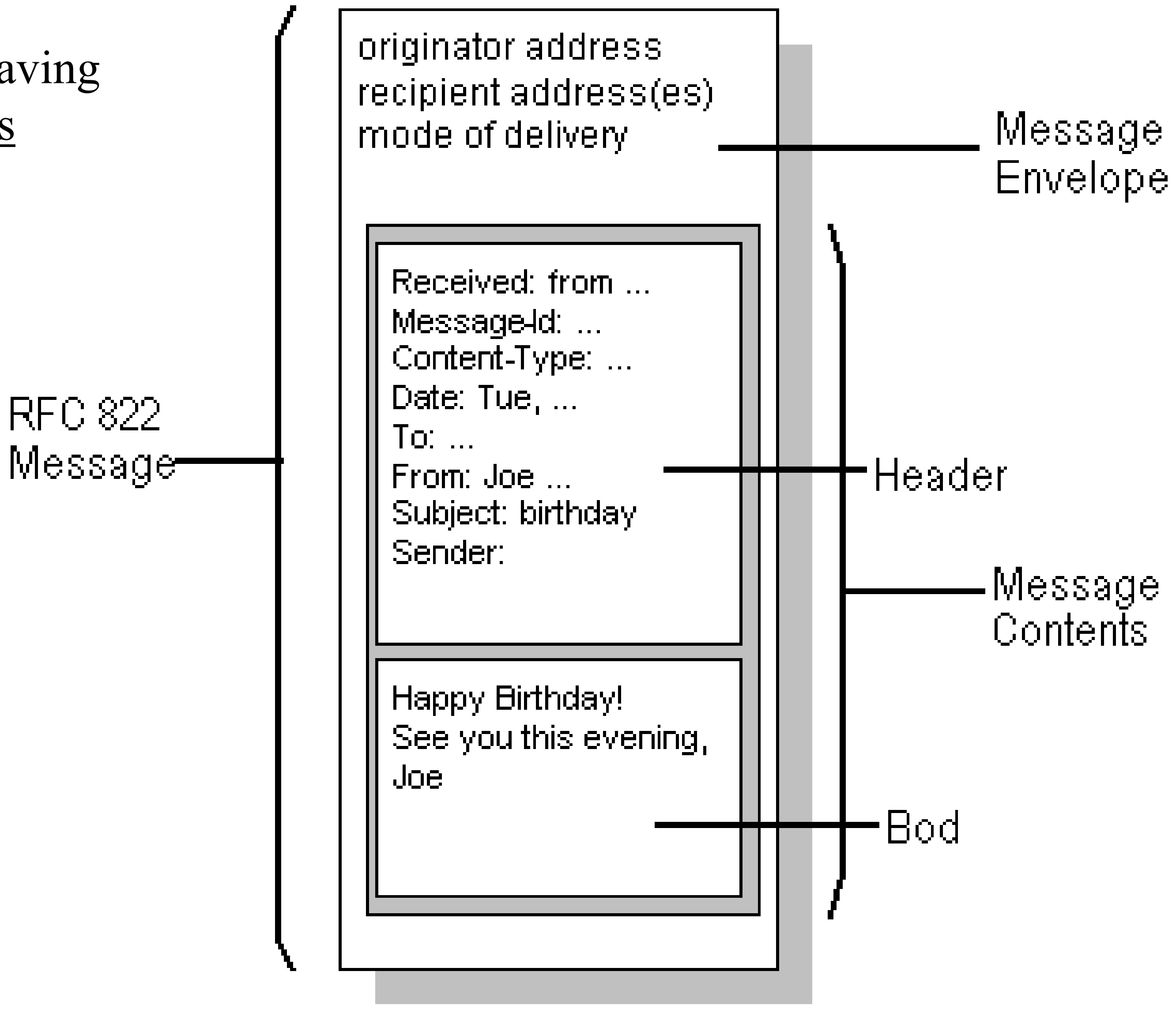
RFC 822 header fields related to message transport.

Header	Meaning
Date:	The date and time the message was sent
Reply-To:	Email address to which replies should be sent
Message-Id:	Unique number for referencing this message later
In-Reply-To:	Message-Id of the message to which this is a reply
References:	Other relevant Message-Ids
Keywords:	User chosen keywords
Subject:	Short summary of the message for the one-line display

Some fields used in the RFC 822 message header.

Format for Text Messages
RFC 882

Message viewed as having envelope and contents



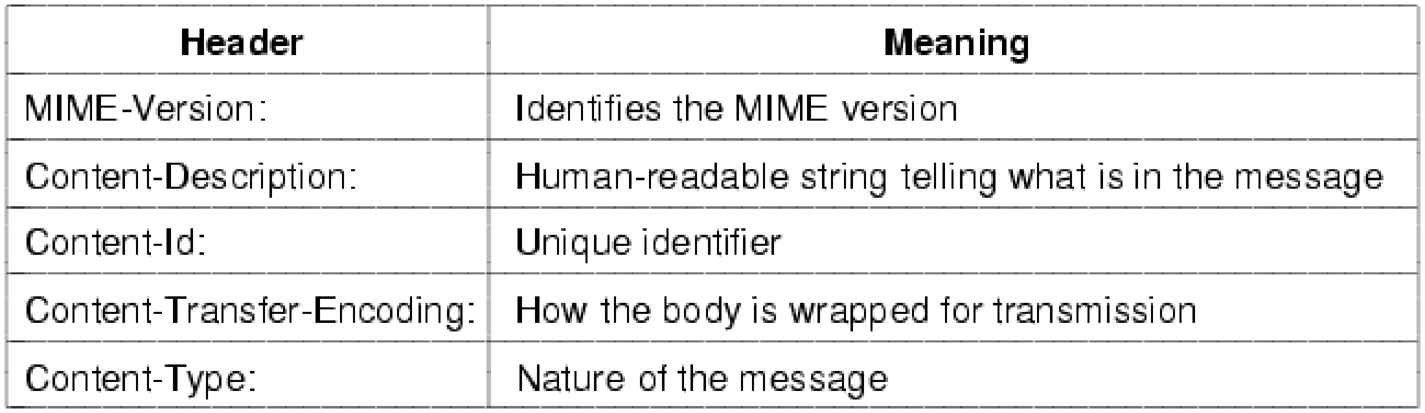
Multipurpose Internet Mail Extension (MIME)

Extension to RFC 822 for message format; given in RFC 2045, 2056

Additional lines in message header declare MIME content type

Five new message header fields

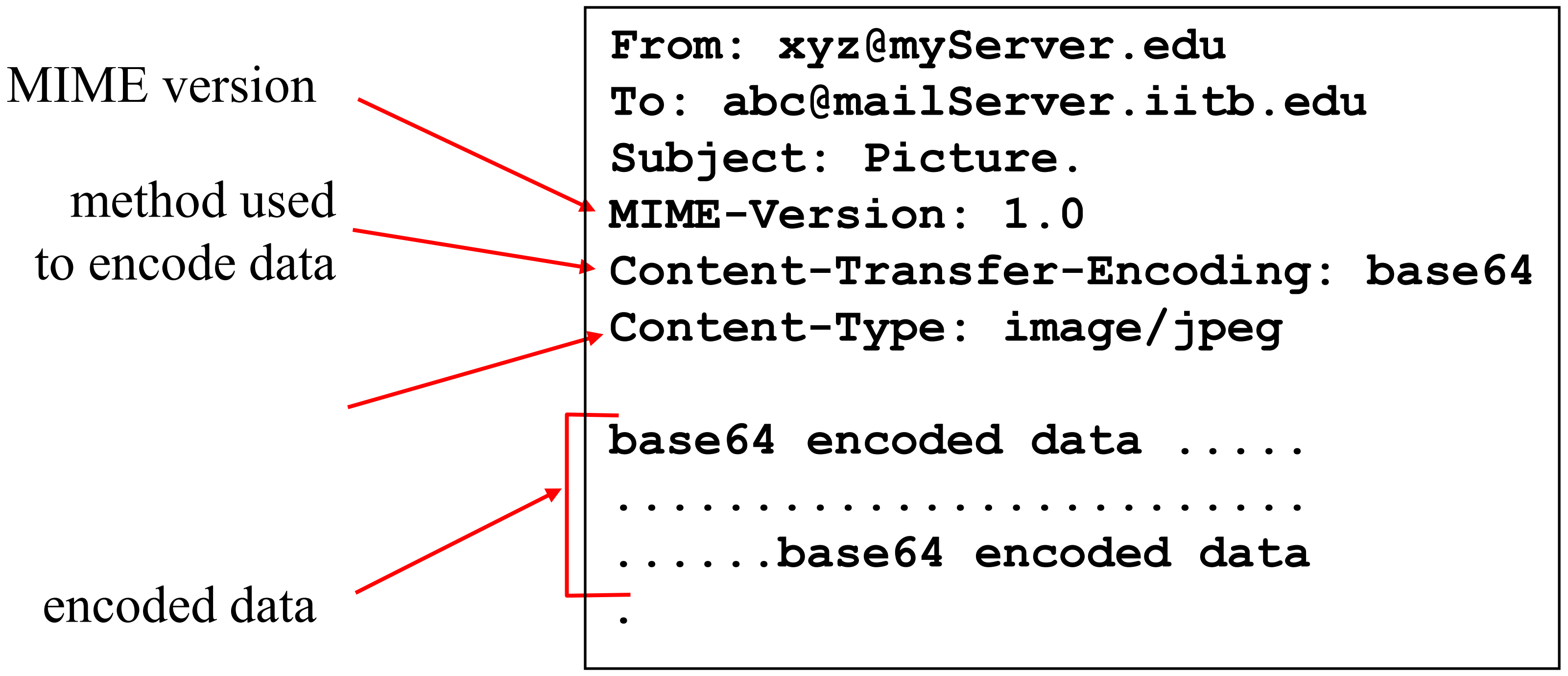
- *MIME version*
- *Content type*
- *Content transfer encoding*
- *Content Description*
- *Content Id*



Header	Meaning
MIME-Version:	Identifies the MIME version
Content-Description:	Human-readable string telling what is in the message
Content-Id:	Unique identifier
Content-Transfer-Encoding:	How the body is wrapped for transmission
Content-Type:	Nature of the message

RFC 822 headers added by MIME.

Message Format: Multimedia Extensions



```
From: xyz@myServer.edu
To: abc@mailServer.iitb.edu
Subject: Picture.
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

base64 encoded data .....
.........................
......base64 encoded data
.
```

•Extends and automates encoding mechanisms - *Multipart Internet Mail Extensions*

•Allows inclusion of separate components - programs, pictures, audio clips - in a single mail message (see next table)

•Sending program identifies the components, so receiving program can automatically extract and inform mail recipient

•Separator line gives information about specific encoding

•MIME is extensible - sender and receiver agree on encoding scheme

•MIME is compatible with existing mail systems

•Everything encoded as ASCII

•Headers and separators ignored by non-MIME mail systems

•MIME *encapsulates* binary data in ASCII mail envelope

MIME Content Types

Type	Subtype	Description
Text	Plain	Unformatted text; may be ASCII or ISO 8859.
Multipart	Mixed	The different parts are independent but are to be transmitted together. They should be presented to the receiver in the order that they appear in the mail message.
	Parallel	Differs from Mixed only in that no order is defined for delivering the parts to the receiver.
	Alternative	The different parts are alternative versions of the same information. They are ordered in increasing faithfulness to the original and the recipient's mail system should display the "best" version to the user.
	Digest	Similar to Mixed, but the default type/subtype of each part is message/rfc822
Message	rfc822	The body is itself an encapsulated message that conforms to RFC 822.
	Partial	Used to allow fragmentation of large mail items, in a way that is transparent to the recipient.
	External-body	Contains a pointer to an object that exists elsewhere.
Image	jpeg	The image is in JPEG format, JFIF encoding.
	gif	The image is in GIF format.
Video	mpeg	MPEG format.
Audio	Basic	Single-channel 8-bit ISDN mu-law encoding at a sample rate of 8 kHz.
Application	PostScript	Adobe Postscript.
	octet-stream	General binary data consisting of 8-bit bytes.

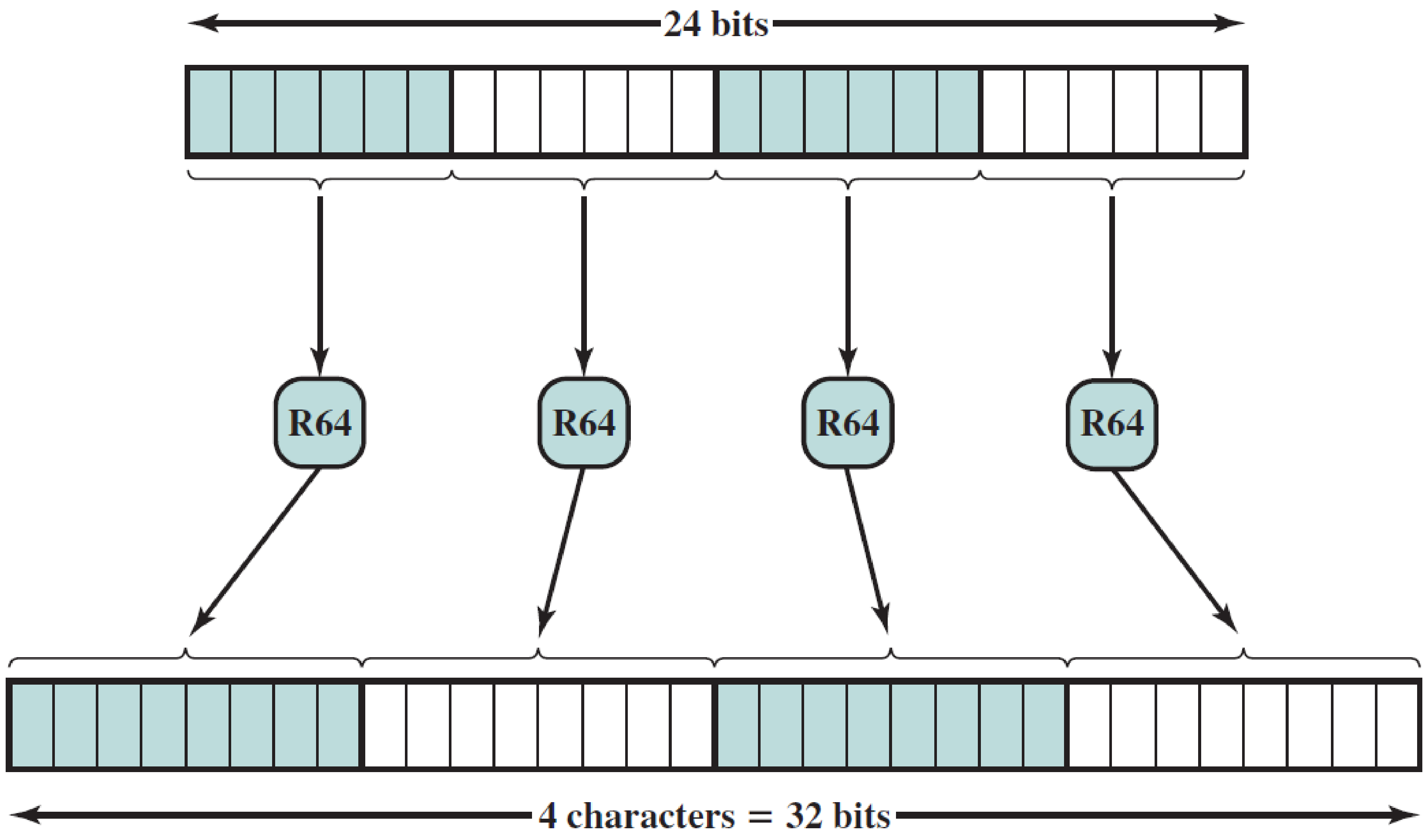
MIME Transfer Encodings

Reliable delivery across large range of environments

Content transfer encoding field takes on six values (see next table):

- Three of them (7bit, 8bit, binary): no encoding done
  - Provide some info about nature of data
  - SMTP transfer uses 7bit form
- Quoted-printable
  - Data contains largely printable ASCII characters
  - Non-printing characters represented by hex code
- Base64 (Radix-64 Encoding)
  - Maps arbitrary binary input (6 bits) onto printable output 8 bit characters
- X-token
  - Named nonstandard encoding

MIME Transfer Encodings



Binary data to Base64 encoding scheme