Computer Networks - *Xarxes de Computadors*

Outline

- Course Syllabus
- Unit 1: Introduction
- Unit 2. IP Networks
- Unit 3. TCP
- Unit 4. LANs
- **Unit 5. Network applications**

Based on: https://studies.ac.upc.edu/FIB/grau/XC/#slides

Unit 5. Network applications

Outline

- **DNS**
- Email
- Web
- Charsets
- HTML

Unit 2: IP Networks

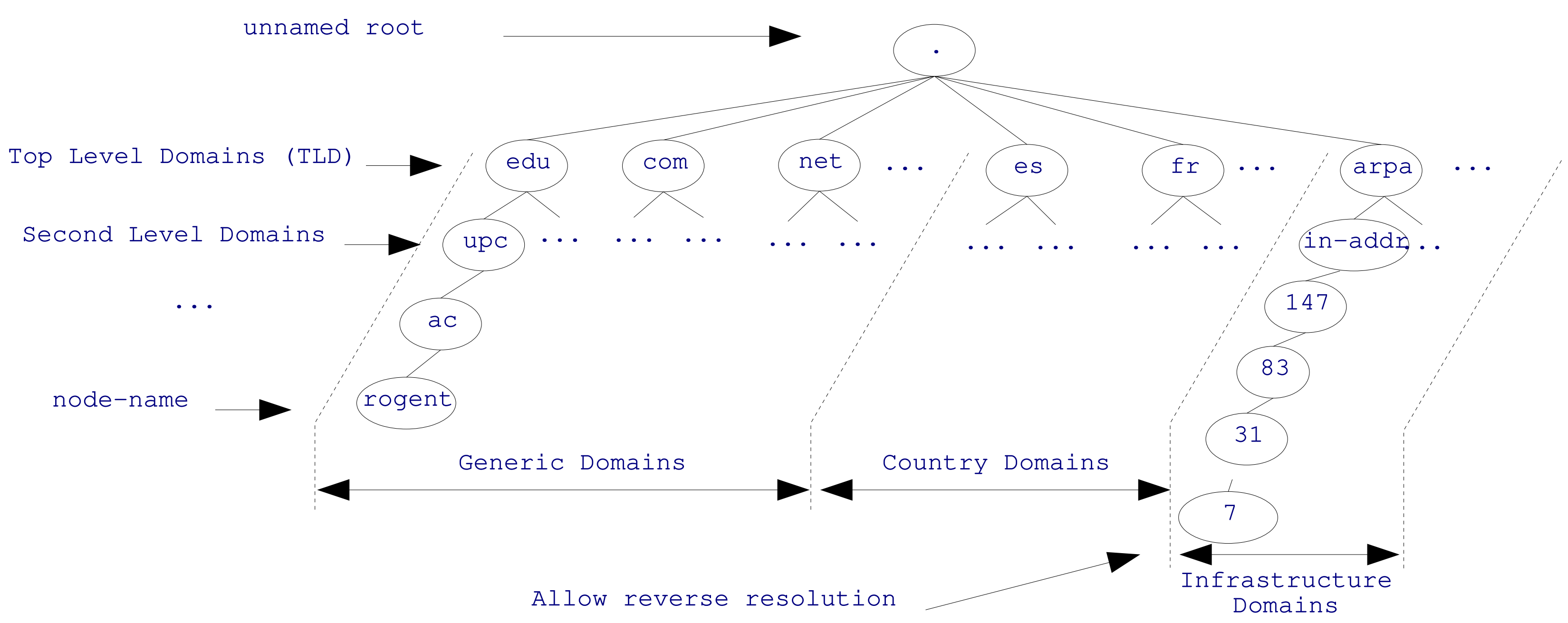
Domain Name System DNS (RFC 1034, 1035)

- Allows users to use names instead of IP addresses: e.g. rogent.ac.upc.edu instead of 147.83.31.7, www.upc.edu instead of 147.83.194.21, etc.
- Names consists of a node-name and a domain-name
  - e.g. rogent.ac.upc.edu, www.upc.edu
- DNS consists of a worldwide distributed data base.
- DNS data base entries are referred to as Resource Records (RR).
- The information associated with a name is composed of 1 or more RRs.
- Names are case insensitive (e.g. www.upc.edu and WWW.UPC.EDU are equivalent).

Unit 2: IP Networks

DNS – Domain Hierarchy

- DNS data base is organized in a tree:

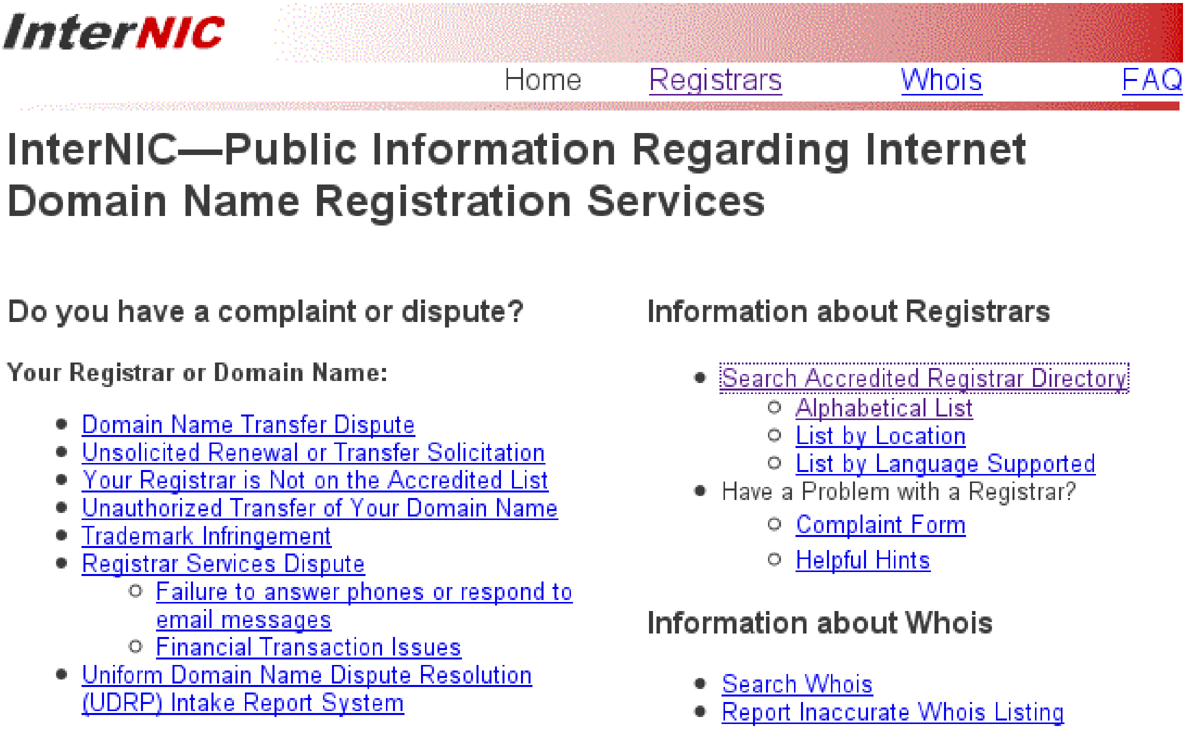


List of TLDs https://data.iana.org/TLD/tlds-alpha-by-domain.txt

Unit 2: IP Networks

DNS – Domain Hierarchy

- The *Internet Corporation for Assigned Names and Numbers* (ICANN) is responsible for managing and coordinating the DNS.
- ICANN delegates Top Level Domains (TLD) administration to registrars (list of accredited registrars: https://www.icann.org/en/accredited-registrars)
- Domains delegate the administration of subdomains.



InterNIC—Public Information Regarding Internet Domain Name Registration Services

Do you have a complaint or dispute?

Your Registrar or Domain Name:

- Domain Name Transfer Dispute
- Unsolicited Renewal or Transfer Solicitation
- Your Registrar is Not on the Accredited List
- Unauthorized Transfer of Your Domain Name
- Trademark Infringement
- Registrar Services Dispute
  - Failure to answer phones or respond to email messages
  - Financial Transaction Issues
- Uniform Domain Name Dispute Resolution (UDRP) Intake Report System

Information about Registrars

- Search Accredited Registrar Directory
  - Alphabetical List
  - List by Location
  - List by Language Supported
- Have a Problem with a Registrar?
  - Complaint Form
  - Helpful Hints

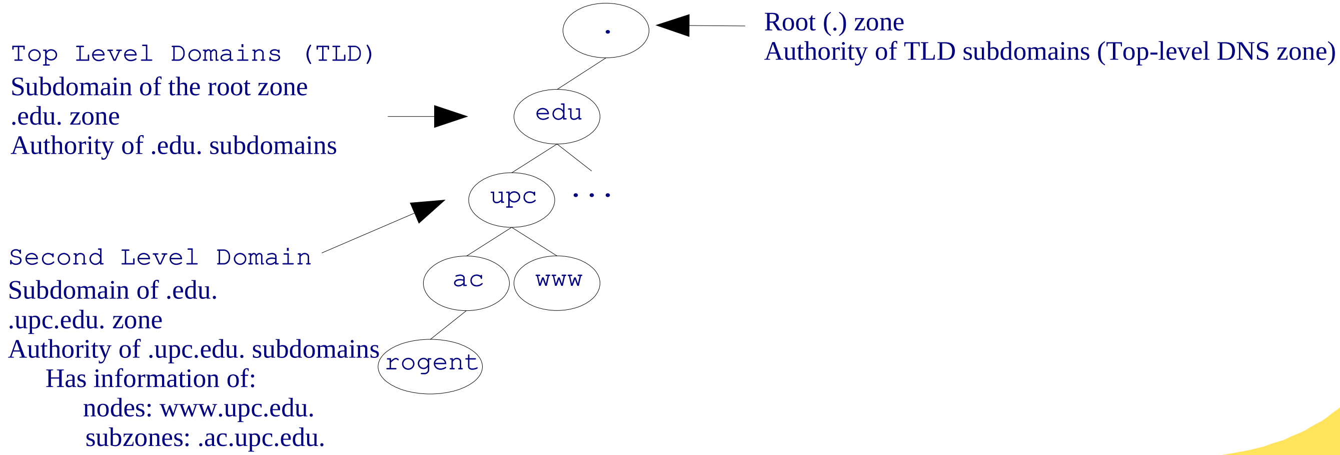
Information about Whois

- Search Whois
- Report Inaccurate Whois Listing

Unit 2: IP Networks

DNS – Data Base Organization

- Access to DNS data base is done using *Name Servers* (NS).
- NSs may hold permanent and cached RRs. Cached RRs are removed after a timeout.
- Each subdomain has an *authority* which consists of a primary and backup NSs.
- In this context, subdomains are referred to as *zones*, and delegated subdomains *subzones*.
- An authority has the complete information of a zone:
  - Names and addresses of all nodes within the zone.
  - Names and addresses of all subzone authorities.



Root (.) zone
Authority of TLD subdomains (Top-level DNS zone)

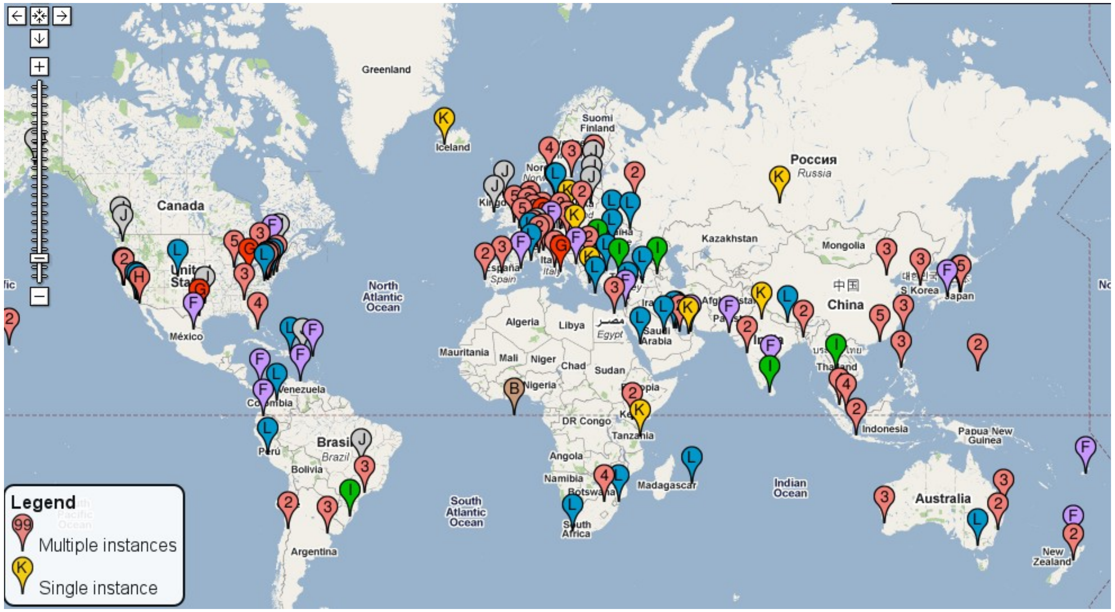
Top Level Domains (TLD)
Subdomain of the root zone
.edu. zone
Authority of .edu. subdomains

Second Level Domain
Subdomain of .edu.
.upc.edu. zone
Authority of .upc.edu. subdomains
Has information of:
nodes: www.upc.edu.
subzones: .ac.upc.edu.

Unit 2: IP Networks

DNS – Data Base Organization

- Root Servers are the entry point to the domain hierarchy.
- Root Servers are distributed around the world and have the TLD addresses:
  http://www.root-servers.org
- Root server addresses are needed in a NS configuration.



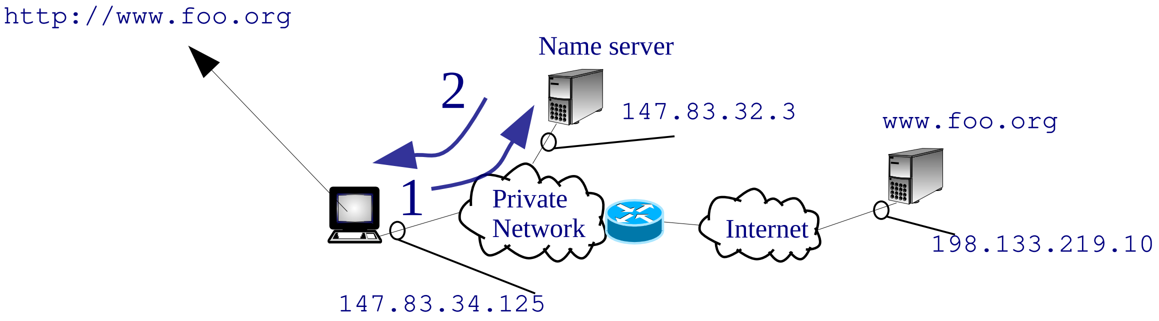
Source: http://www.root-servers.org

The root zone 13 root name servers clusters operated by 12 independent organizations.
Official name of the clusters: a.root-servers.net to m.root-servers.net.
As of May 2023 there are ~1700 instances in total.

Unit 2: IP Networks

DNS - Protocol

- Client-server paradigm
- UDP/TCP. Short messages uses UDP.
- well-known port: 53



```
1  18:36:00.322370 IP (proto: UDP) 147.83.34.125.1333 >
          147.83.32.3.53:  53040+ A? www.foo.org. (31)
2  18:36:00.323080 IP (proto: UDP) 147.83.32.3.53 > 147.83.34.125.1333:
          53040 1/2/2 www.foo.org. A 198.133.219.10 (115)
```

Server-client paradigm:
Client side

DNS - Unix example: The resolver

- The applications use the calls (*resolver* library):

```
struct hostent *gethostbyname(const char *name) ;
struct hostent *gethostbyaddr(const void *addr, int len, int type);
```

- The resolver first looks the /etc/hosts file:

```
# hosts         This file describes a number of hostname-to-address
#               mappings for the TCP/IP subsystem.  It is mostly
#               used at boot time, when no name servers are running.
#               On small systems, this file can be used instead of a
#               "named" name server.
# Syntax:
# IP-Address  Full-Qualified-Hostname  Short-Hostname
127.0.0.1       localhost
10.0.1.1        massanella.ac.upc.edu massanella
```

- Otherwise a *name server* is contacted using /etc/resolv.conf file:

```
search ac.upc.edu
nameserver 147.83.32.3
nameserver 147.83.33.4
```

`search` Domain attached by the OS if not specified by the user (e.g. ‘ping rogent’ → ‘‘ping rogent.ac.upc.edu’’

`nameserver` Name servers to be used by preference (subsequent NS is only used if the precedent has timed out

Unit 2: IP Networks

Server-client paradigm:
Server side

DNS – Unix example: Basic NS configuration

- Unix NS implementation is BIND (Berkeley Internet Name Domain), http://www.isc.org.
- named is the BIND NS daemon.
- BIND basic configuration files:
  - `/etc/named.conf` global configuration
  - `/var/lib/named/root.hint` root servers addresses
  - `/var/lib/named/*.db` zone files

Source of truth:
https://www.internic.net/domain/named.root

Source of truth of the root-zone:
https://www.internic.net/domain/root.zone

Unit 2: IP Networks

Server-client paradigm:
Server side

DNS – Unix example: root servers addresses

```
linux # cat /var/lib/named/root.hint

;       This file holds the information on root name servers needed to
;       initialize cache of Internet domain name servers
;       (e.g. reference this file in the "cache  .  <file>"
;       configuration file of BIND domain name servers).
;
;       This file is made available by InterNIC
;       under anonymous FTP as
;           file                /domain/named.root
;           on server           FTP.INTERNIC.NET
;       -OR-                    RS.INTERNIC.NET
.                         3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.       3600000  IN  A     198.41.0.4
.                         3600000  IN  NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.       3600000  IN  A     192.228.79.201
.                         3600000  IN  NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.       3600000  IN  A     192.33.4.12

...
.                         3600000  IN  NS    M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.       3600000  IN  A     202.12.27.33
```

comments

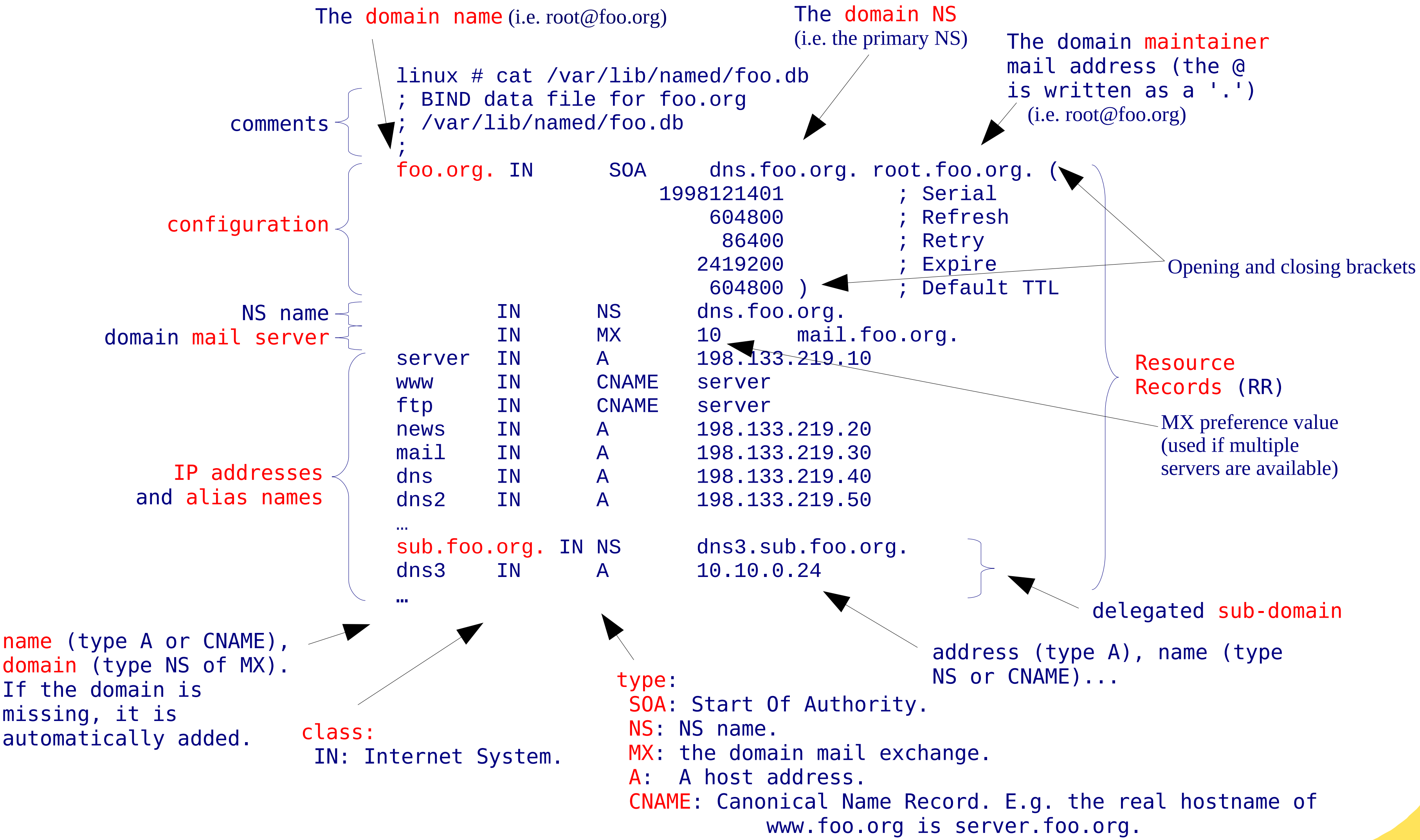
Resource Records (RR)
pointing to root-servers

address of a name
NS name

Unit 2: IP Networks

DNS – Unix example: zone file

Server-client paradigm:
Server side



The domain name (i.e. root@foo.org)

The domain NS
(i.e. the primary NS)

The domain maintainer
mail address (the @
is written as a '.')
(i.e. root@foo.org)

```
linux # cat /var/lib/named/foo.db
; BIND data file for foo.org
; /var/lib/named/foo.db
;
foo.org. IN      SOA     dns.foo.org. root.foo.org. (
                     1998121401          ; Serial
                         604800          ; Refresh
                          86400          ; Retry
                        2419200          ; Expire
                         604800 )        ; Default TTL
           IN      NS      dns.foo.org.
           IN      MX      10      mail.foo.org.
server     IN      A       198.133.219.10
www        IN      CNAME   server
ftp        IN      CNAME   server
news       IN      A       198.133.219.20
mail       IN      A       198.133.219.30
dns        IN      A       198.133.219.40
dns2       IN      A       198.133.219.50
…
sub.foo.org. IN NS         dns3.sub.foo.org.
dns3       IN      A       10.10.0.24
…
```

comments

configuration

NS name

domain mail server

IP addresses
and alias names

Opening and closing brackets

Resource
Records (RR)

MX preference value
(used if multiple
servers are available)

delegated sub-domain

name (type A or CNAME),
domain (type NS of MX).
If the domain is
missing, it is
automatically added.

class:
IN: Internet System.

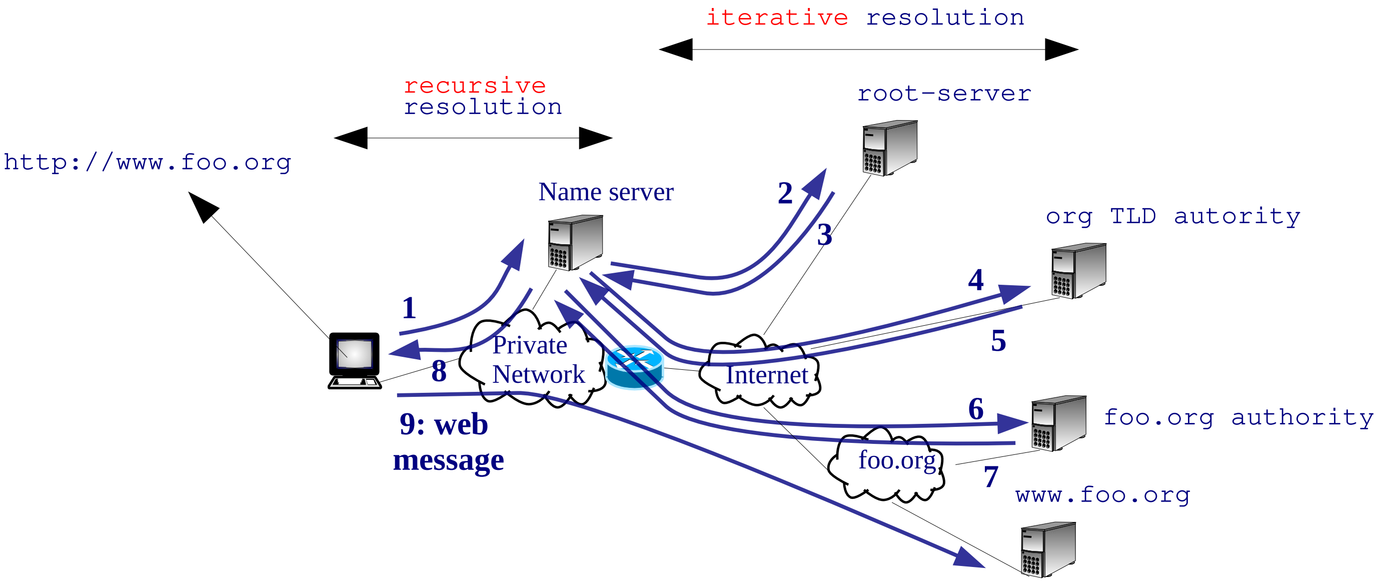
type:
SOA: Start Of Authority.
NS: NS name.
MX: the domain mail exchange.
A: A host address.
CNAME: Canonical Name Record. E.g. the real hostname of
www.foo.org is server.foo.org.

address (type A), name (type
NS or CNAME)...

Unit 2: IP Networks

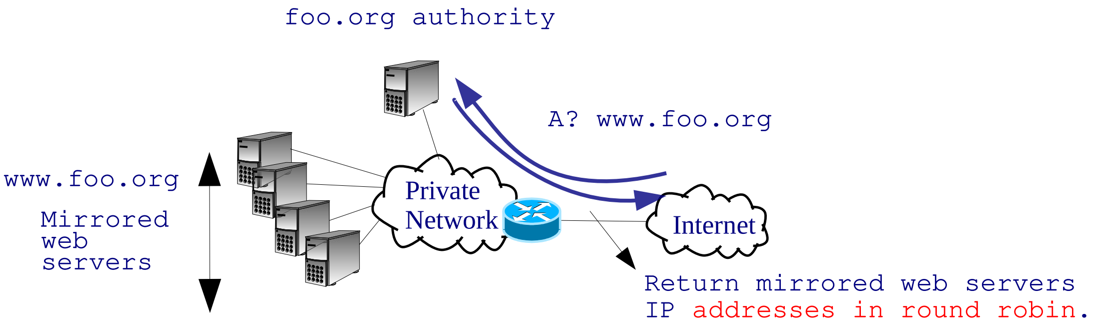
DNS – Resolution

- NSs cache name resolutions.
- A cached RR is returned without looking for in the NS authority.
- The same name may be associated with several IP addresses (e.g. load balancing).
- The addresses of a common domain may not belong to the same IP network (e.g. Content Distribution Networks).



Unit 2: IP Networks

DNS – Load balancing, example



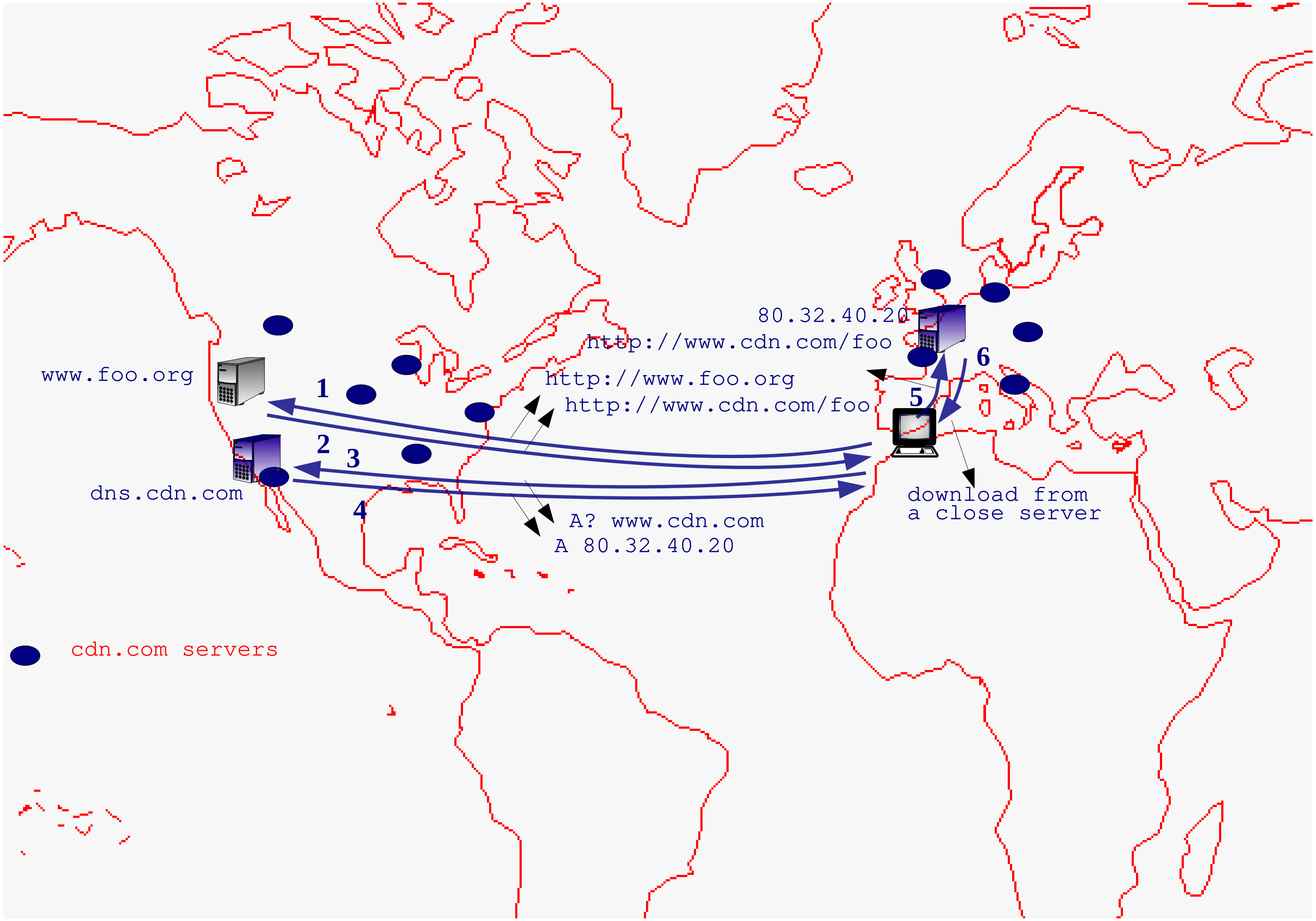
- Example using dig:

```
linux ~> dig www.microsoft.com

; <<>> DiG 9.3.2 <<>> www.microsoft.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 31808
;; flags: qr rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;www.microsoft.com.             IN      A

;; ANSWER SECTION:
www.microsoft.com.      3135    IN      CNAME   toggle.www.ms.akadns.net.
toggle.www.ms.akadns.net. 181   IN      CNAME   g.www.ms.akadns.net.
g.www.ms.akadns.net.    181     IN      CNAME   lb1.www.ms.akadns.net.
lb1.www.ms.akadns.net.  181     IN      A       207.46.19.60
lb1.www.ms.akadns.net.  181     IN      A       207.46.18.30
lb1.www.ms.akadns.net.  181     IN      A       207.46.20.60
lb1.www.ms.akadns.net.  181     IN      A       207.46.19.30
lb1.www.ms.akadns.net.  181     IN      A       207.46.198.30
lb1.www.ms.akadns.net.  181     IN      A       207.46.225.60

;; Query time: 42 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Sun Mar 11 10:48:11 2007
;; MSG SIZE  rcvd: 203
```

```
linux ~> dig www.microsoft.com

; <<>> DiG 9.3.2 <<>> www.microsoft.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17923
;; flags: qr rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;www.microsoft.com.             IN      A

;; ANSWER SECTION:
www.microsoft.com.      3469    IN      CNAME   toggle.www.ms.akadns.net.
toggle.www.ms.akadns.net. 215   IN      CNAME   g.www.ms.akadns.net.
g.www.ms.akadns.net.    215     IN      CNAME   lb1.www.ms.akadns.net.
lb1.www.ms.akadns.net.  215     IN      A       207.46.198.30
lb1.www.ms.akadns.net.  215     IN      A       207.46.199.30
lb1.www.ms.akadns.net.  215     IN      A       207.46.18.30
lb1.www.ms.akadns.net.  215     IN      A       207.46.19.60
lb1.www.ms.akadns.net.  215     IN      A       207.46.198.60
lb1.www.ms.akadns.net.  215     IN      A       207.46.20.60

;; Query time: 43 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Sun Mar 11 10:42:38 2007
;; MSG SIZE  rcvd: 203
```

Unit 2: IP Networks

DNS - Content Distribution Networks, example



Unit 2: IP Networks

DNS – Messages: Message Format

- All DNS messages have the same format:
  - Header: type of message.
  - Question: What is to be resolved.
  - Answer: Answer to question.
  - Authority: Domain authority names.
  - Additional: Typically, the authority name's addresses.

```
 -------------------------------------------------
|             Header (12 bytes)                   |
 -------------------------------------------------
/             Question (variable)                 /
 -------------------------------------------------
/             Answer (variable)                   /
 -------------------------------------------------
/             Authority (variable)                /
 -------------------------------------------------
/             Additional (variable)               /
 -------------------------------------------------
```

Unit 2: IP Networks

DNS – Messages: Header

- Identification: 16 random bits used to match query/response
- Flags. Some of them:
  - Query-Response, QR: 0 for query, 1 for response.
  - Authoritative Answer, AA: When set, indicates an authoritative answer.
  - Recursion Desired, RD: When set, indicates that recursion is desired.
- The other fields indicate the number of Questions, Answer, Authority and Additional fields of the message.

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 bits
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Identification       |      Flags                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         #Questions            |      #Answers                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         #Authorities          |      #Additional              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Unit 2: IP Networks

DNS – Messages: Question

- QName: Indicates the name to be resolved.
- QType: Indicates the question type:
  - Address, A.
  - Name Server, NS.
  - Pointer, PTR: For an inverse resolution.
  - Mail Exchange, MX: Domain Mail Server address.
- Qclass: For Internet addresses is 1.

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 bits
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                       QName (variable)                        /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         QType                 |           QClass              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 bytes
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|6|r|o|g|e|n|t|2|a|c|3|u|p|c|3|e|d|u|0|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

QName codification example of `rogent.ac.upc.edu`

Unit 2: IP Networks

DNS – Messages: Resource Records (RRs)

- The fields Answer, Authority and Additional are composed of RRs:
  - Name, Type, Class: The same as in the Question field.
  - TTL (Time To Live): Number of seconds the RR can be cached.
  - RDLenth: RR size in bytes.
  - Rdata: E.g. An IP address if the Type is 'A', or a name if the Type is 'NS', 'MX' or 'CNAME'.

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 bits
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                        Name (variable)                        /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Type               |             Class             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              TTL                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           RDLenth             |        RData (variable)       /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Unit 2: IP Networks

DNS – Messages: Example

```
# tcpdump -s1500 -vvpni eth0 port 53
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 200 bytes
11:17:30.769328 IP (UDP, length: 55) 147.83.30.137.1042 > 147.83.30.70.53: 36388+ A? ns.uu.net. (27)
11:17:30.771324 IP (UDP, length: 145) 147.83.30.70.53 > 147.83.30.137.1042: 36388
                  q: A? ns.uu.net. 1/2/2 ns.uu.net. A 137.39.1.3
                  ns: ns.uu.net. NS auth00.ns.uu.net., ns.uu.net. NS auth60.ns.uu.net.
                  ar: auth00.ns.uu.net. A 198.6.1.65, auth60.ns.uu.net. A 198.6.1.181 (117)
```

Query message:

- 36388: Identifier.
- +: Recursion-Desired is set.
- A?: Qtype = A.
- ns.uu.net.: Name to resolve.

Response message:

- 36388: Identifier.
- q: A? ns.uu.net.: Repeat the Question field.
- 1/2/2: 1 Answers, 2 Authorities, 2 Additional follows.
- ns.uu.net. A 137.39.1.3: The answer (RR of type A, address: 137.39.1.3).
- ns: ns.uu.net. NS auth00.ns.uu.net., ns.uu.net. NS auth60.ns.uu.net.: 2 Authorities (RRs of type NS: the domain ns.uu.net. authorities are auth00.ns.uu.net. and auth60.ns.uu.net).
- ar: auth00.ns.uu.net. A 198.6.1.65, auth60.ns.uu.net. A 198.6.1.181: 2 Additional (RRs of type A: authorities IP addresses).

Unit 5. Network applications

Outline

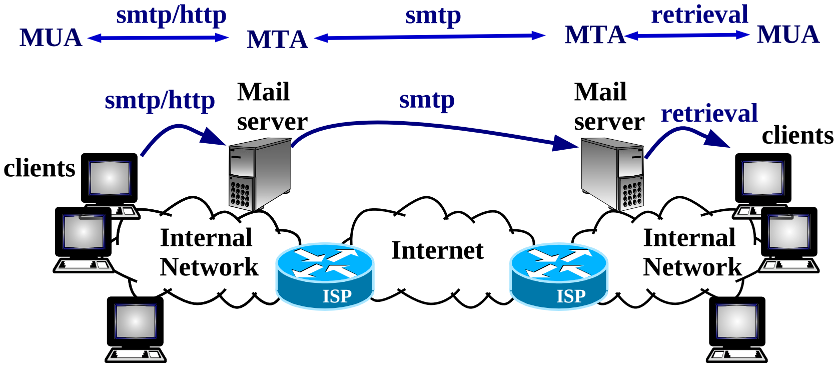
- DNS
- **Email**
- Web
- Charsets
- HTML

Unit 5. Network applications

Email

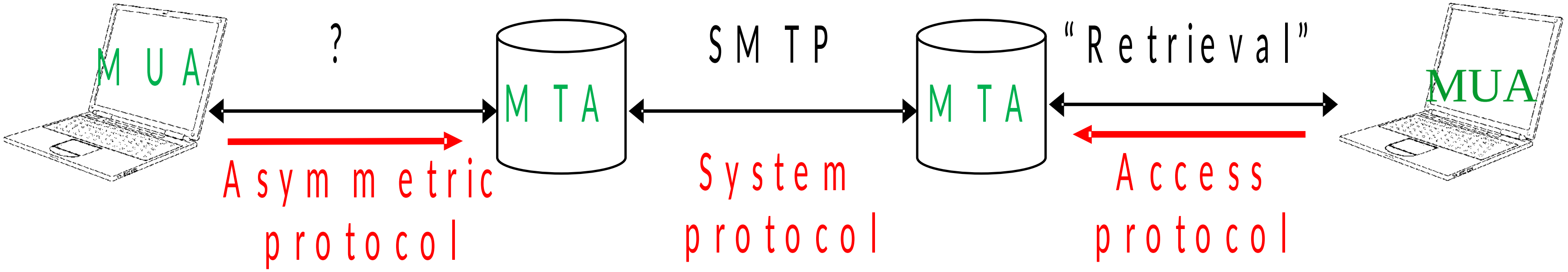
- Electronic mail (email): One of the first applications used in the Internet to electronic messaging.
- Components:
  - Transport layer: TCP, well-known port: 25.
  - Application layer protocol: Simple Mail Transfer Protocol (SMTP). First defined by RFC-821 and last updated by RFC-5321.
  - Retrieval protocols (IMAP, POP, HTTP).

MUA: Mail User Agent
MTA: Mail Transfer Agent



Unit 5. Network applications

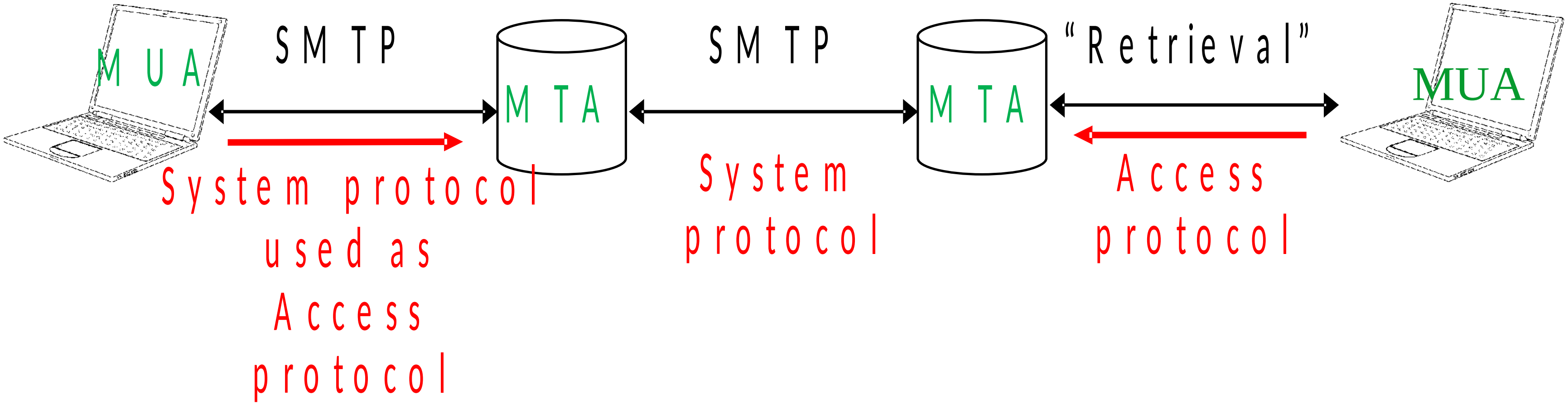
Email – Architecture



- **MUA: Mail User Agent**
- **MTA: Mail Transfer Agent**

Unit 5. Network applications

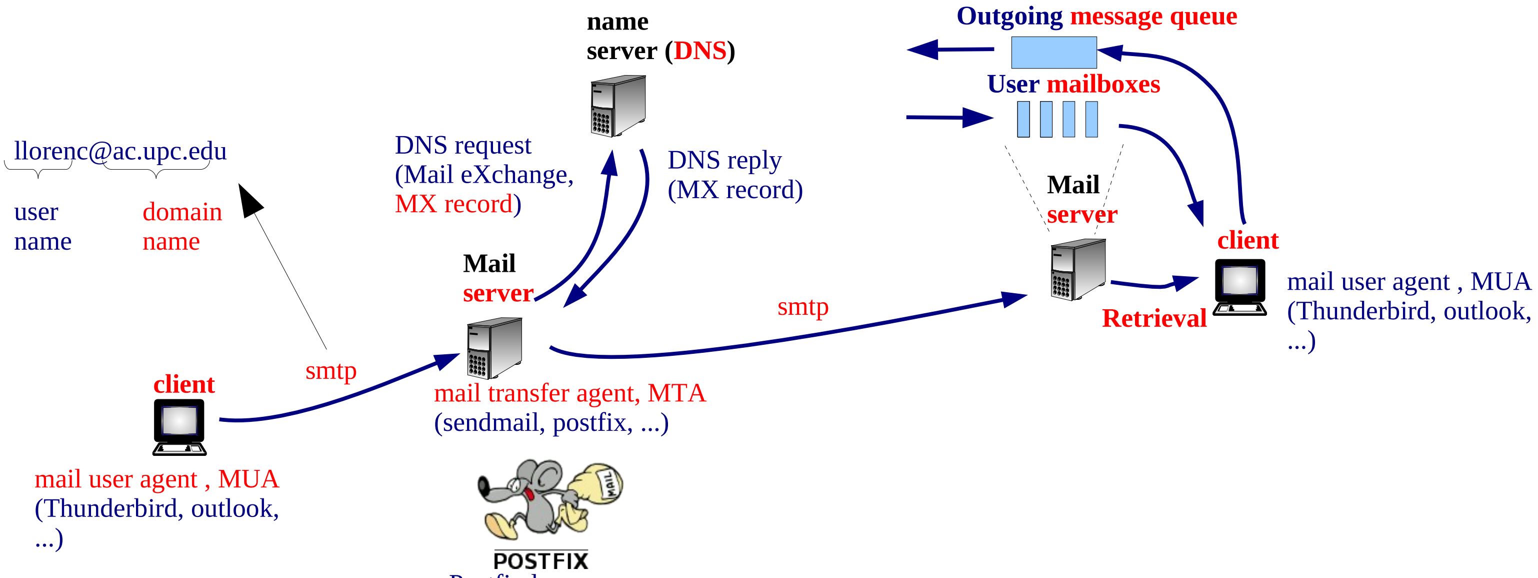
Email – Protocols



- **"Retrieval" protocols (mailbox access):**
  - Post Office Protol (POP3)
  - Internet Message Access Protocol (IMAP)
- **Simple Mail Transfer Protocol (SMPT)**

Unit 5. Network applications

Email - SMTP processing model



llorenc@ac.upc.edu

user name

domain name

name server (DNS)

DNS request (Mail eXchange, MX record)

DNS reply (MX record)

Outgoing message queue

User mailboxes

Mail server

client

mail user agent , MUA (Thunderbird, outlook, ...)

smtp

Mail server

mail transfer agent, MTA (sendmail, postfix, ...)

smtp

Retrieval

client

mail user agent , MUA (Thunderbird, outlook, ...)

Postfix logo
http://www.postfix.org
(UNIX, free and open-source)

Unit 5. Network applications

Email - SMTP protocol (RFC-821, last update RFC-5321)

- Designed as a simple (few commands) and text-based protocol (ASCII).
  - Client basic commands: HELO (identify SMTP client), MAIL FROM: (identify sender mailbox), RCPT TO: (identify recipient mailbox), DATA (mail message), QUIT (close transaction).
  - Server replies: Three digit number (identify what state the client to enter next), and a human understandable message.
- Example: Manually send an email using telnet to port 25.

Other possible ports: 587 (TLS - STARTTLS), 465 (SSL: outdated)

CLIENT

SERVER

COMMANDS

SMTP transaction

```
linux ~> telnet relay.upc.edu 25
Trying 147.83.2.12...
Connected to relay.upc.edu.
Escape character is '^]'.
220 dash.upc.es ESMTP Sendmail 8.14.1/8.13.1; Fri, 4 Feb 2011 14:57:15 +0100
HELO linux.ac.upc.edu
250 dash.upc.es Hello linux.ac.upc.edu [147.83.34.125], pleased to meet you
MAIL FROM: <llorenc@ac.upc.edu>
250 2.1.0 <llorenc@ac.upc.edu>... Sender ok
RCPT TO: <albert@ac.upc.edu>
250 2.1.5 <albert@ac.upc.edu>... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself

Hello world
.
250 2.0.0 p14DvFOQ008320 Message accepted for delivery
QUIT
221 2.0.0 dash.upc.es closing connection
Connection closed by foreign host.
linux ~>
```

Try `telnet mail.guifi.net 587`

With STARTTLS:

```
openssl s_client -debug -starttls smtp \
-crlf -connect mail.guifi.net:587
```

Multipurpose Internet Mail Extensions: MIME

- Used in mail, web, etc
- Specification for “Transport” of composite multimedia objects
  - Transport type information (receiver can automatically present)
  - Encoding to enable/facilitate the transfer
- The internal format becomes invisible to users
- Include one or more objects, text in diverse alphabets, large objects (fragments, refs), alternatives, etc.

MIME: examples

```
From: Nathaniel Borenstein <nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: Plain old email

This is a plain old email message.
It contains ASCII text, nothing more.
```

```
From: Nathaniel Borenstein <nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: Plain text mail
Content-type: text/plain; charset=us-ascii

This is plain text mail.
```

```
...Subject: French mail
Content-type: text/plain; charset=iso-8859-1
Content-transfer-encoding: quoted-printable

Le courrier =E9lectronique =E0 la fran=E7aise ...
```

```
...Content-type: image/gif
Content-Transfer-Encoding: base64

R0lGODdhSgGgAfUAAENDQ01NTTw8PEVF...
```

MIME: example multipart

```
  From: Nathaniel Borenstein <nsb@bellcore.com>
  To: Ned Freed <ned@innosoft.com>
  Subject: A multipart example
  Content-Type: multipart/mixed; boundary=CUT_HERE

--CUT_HERE
  Content-type: text/plain

  Hey, Ned, look at this neat picture:
--CUT_HERE
  Content-type: image/gif
  Content-Transfer-Encoding: base64

  5WVlZ6enqqqr....
--CUT_HERE
  Content-type: text/plain

  Wasn’t that neat?
--CUT_HERE--
```

Note the ending ‘--’ of the last boundary

MIME: content type

- Text:
  - Attribute: charset=iso-8859-1
  - text/plain (simple text), text/html ...
- Image: image/gif, image/jpeg, image/png ...
- Audio: sound, voice, music …
- Application: application specific content
  - Application/octet-stream: data without any associated application
  - Application/organization-product
- Multipart: a set of objects
  - Mixed: a combination of several objects
  - Alternative: an object in several formats to select one (text/html/rtf)
  - Parallel: several objs for simultaneous presentation (e.g. audio+video)
  - Digest: collection of messages
  - Related: set of objects part of a single object (web page)
- Message:
  - RFC822: a complete message (eg. resent message)
  - Partial: a fragment …
  - External-Body: a reference to an external object

Registration scheme
Type/subtype:
mantained by IANA

MIME: transfer encoding

Ways to encode content: (to “get through” a 7 bit transport)

- Quoted-Printable:
  - The majority of text is 7 bits, transform some characters €→ =E4
  - The result “almost" legible without decoding. Depends on table (charset)
- Base64:
  - 3 bytes (24 bits) <=> 4 ASCII (32 bits)
  - A-Za-z0-9+/=
  - '=' as padding, other are ignored (\r, \n, …)
- Binary: No encoding: any character and lines of any length
- 7Bit: No character encoding (all 7 bits) and lines of appropriate length
- 8Bit: No character encoding (8 bits) and lines of appropriate length
- In the heading:

```
MIME-Version: 1.0
Subject: =?iso-8859-1?Q?acentuaci=F3n=20t=EDpica?=
```

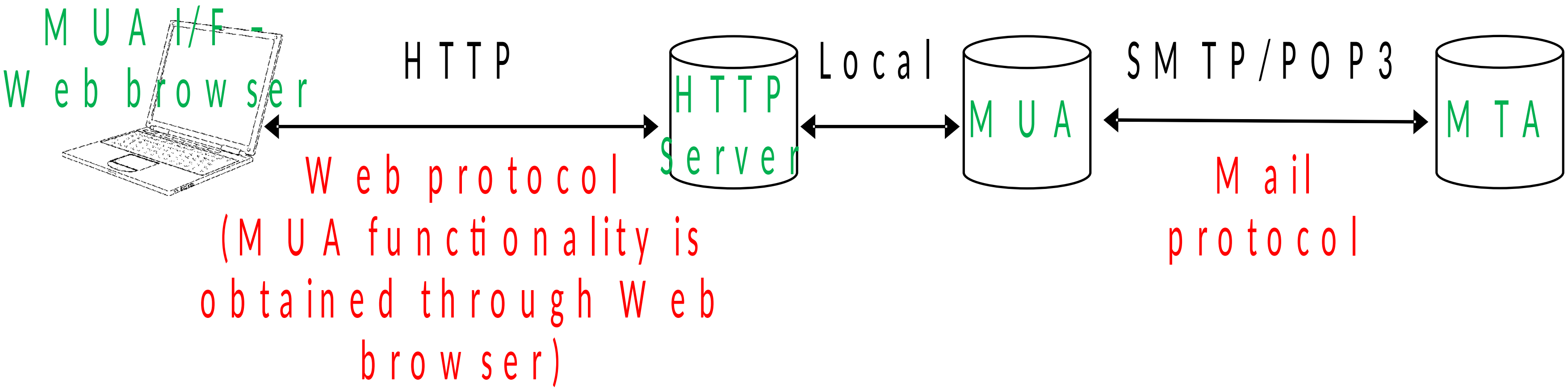
Unit 5. Network applications

Email - retrieval protocols

- Post Office Protocol (POP), RFC-1939:
  - POP server listens on well-known port 110
  - User normally deletes messages upon retrieval.
- Internet Message Access Protocol (IMAP) RFC-3501:
  - IMAP server listens on well-known port 143
  - Messages remain on the server until the user explicitly deletes them.
  - Provide commands to create folders, move messages, download only parts of the messages (e.g. only the headers)
- Web based Email (HTTP)
  - A web server handles users mailboxes. User agent is a web browser, thus, using HTTP to send and retrieve email messages.

Unit 5. Network applications

Email - Webmail



- Web front-end for mail services. The MUA is a web browser.
- Real protocol to access the services: HTTP (web).
- The HTTP server machine uses SMTP or POP3, as required.

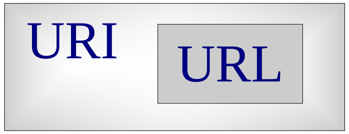
Unit 5. Network applications

Outline

- DNS
- Email
- **Web**
- Charsets
- HTML

Unit 5. Network applications

Web – links



URI

URL

- Uniform Resource Identifier (URI) RFC3986
  - Generic syntax to identify a resource.
- Uniform Resource Locator (URL) RFC1738
  - Subset of URIs identifying the locating a resource in the Internet.
- The URL general syntax is

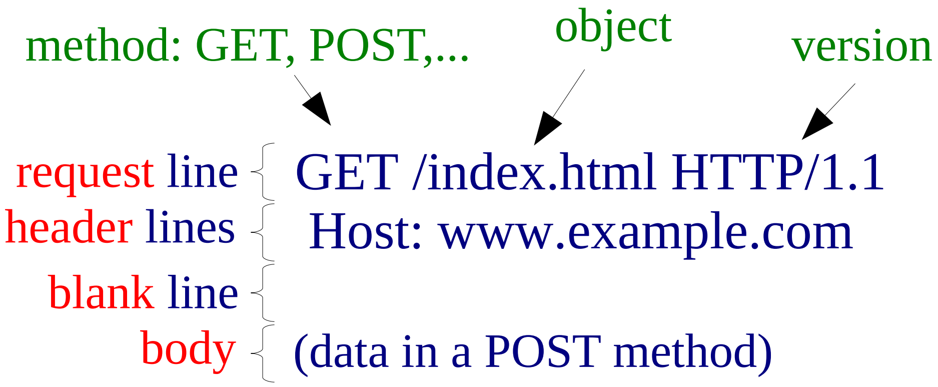
  scheme://username:password@domain:port/path?query_string#fragment_id

  - scheme: Purpose, and the syntax of the remaining part. http, gopher, file, ftp...
  - domain name or IP address gives the destination location. The port is optional.
  - query_string: contains data to be passed to the server.
  - fragment_id: specifies a position in the html page.
  - Examples:
    - http://tools.ietf.org/html/rfc1738
    - http://147.83.2.135
    - http://studies.ac.upc.edu/FIB/grau/XC/#Practs
    - file:///home/llorenc/gestio/2010/cd/autors.html
    - http://www.amazon.com/product/03879/refs9?pf_ra=ATVPD&pf_rd=07HR2

Unit 5. Network applications

Web – HTTP Messages, RFC2616

- Client (HTTP request):



method: GET, POST,...

object

version

request line

header lines

blank line

body

```
GET /index.html HTTP/1.1
 Host: www.example.com

(data in a POST method)
```

- Header: Allows the client to give additional information about the request and the client itself.
    - Host:
        - host of the resource being requested
        - mandatory in HTTP/1.1

Unit 5. Network applications

Web – HTTP Messages, RFC2616

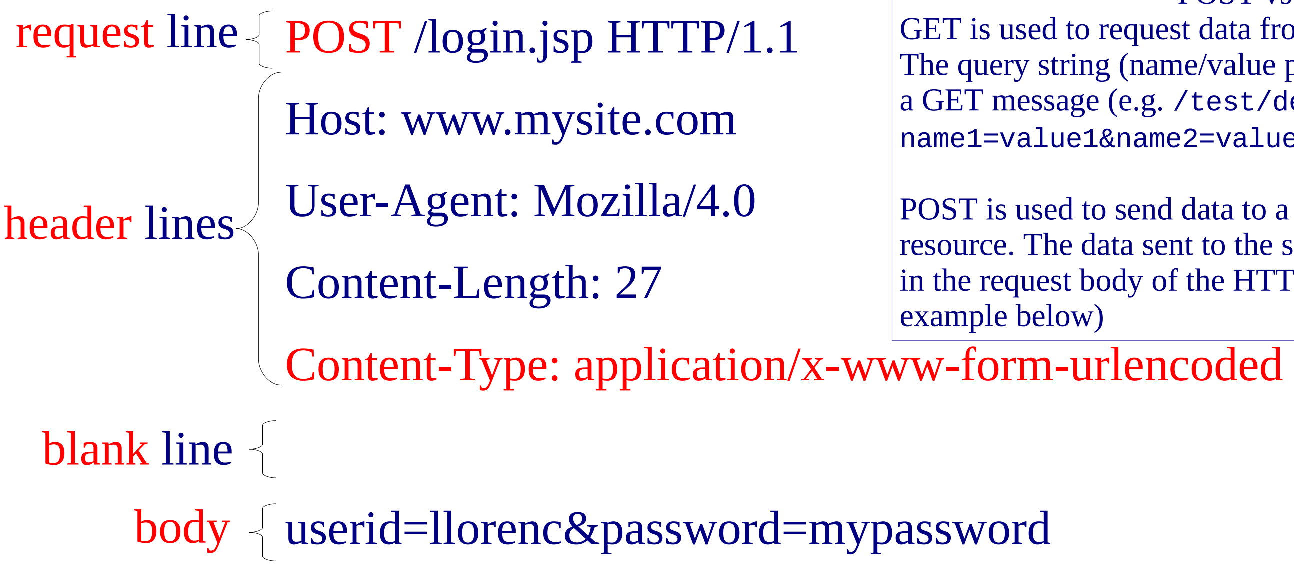
- **Methods:**
  - GET Typical command. Requests an object.
  - POST Request an object qualified by the data in the body. This data is the contents of the HTML form fields, provided by the client.
  - HEAD the server returns only the header
  - OPTIONS request communication options
  - PUT store entity
  - PATCH modify an existing resource
  - DELETE delete entity
  - TRACE final recipient echoes the received message back
  - CONNECT used with a proxy
- NOTES
  - Most used: GET, POST
  - Safe and mandatory: GET, HEAD

Any server must implement these methods at least

Unit 5. Network applications

Web – HTTP Messages, RFC2616

- POST uses MIME types: application/octet-stream, to send raw binary data, and application/x-www-form-urlencoded, to send name-value pairs. Example:



request line
POST /login.jsp HTTP/1.1

header lines
Host: www.mysite.com
User-Agent: Mozilla/4.0
Content-Length: 27
Content-Type: application/x-www-form-urlencoded

blank line

body
userid=llorenc&password=mypassword

POST vs. GET

GET is used to request data from a specified resource. The query string (name/value pairs) is sent in the URL of a GET message (e.g. `/test/demo_form.php?name1=value1&name2=value2`)

POST is used to send data to a server to create/update a resource. The data sent to the server with POST is stored in the request body of the HTTP request (as show in the example below)

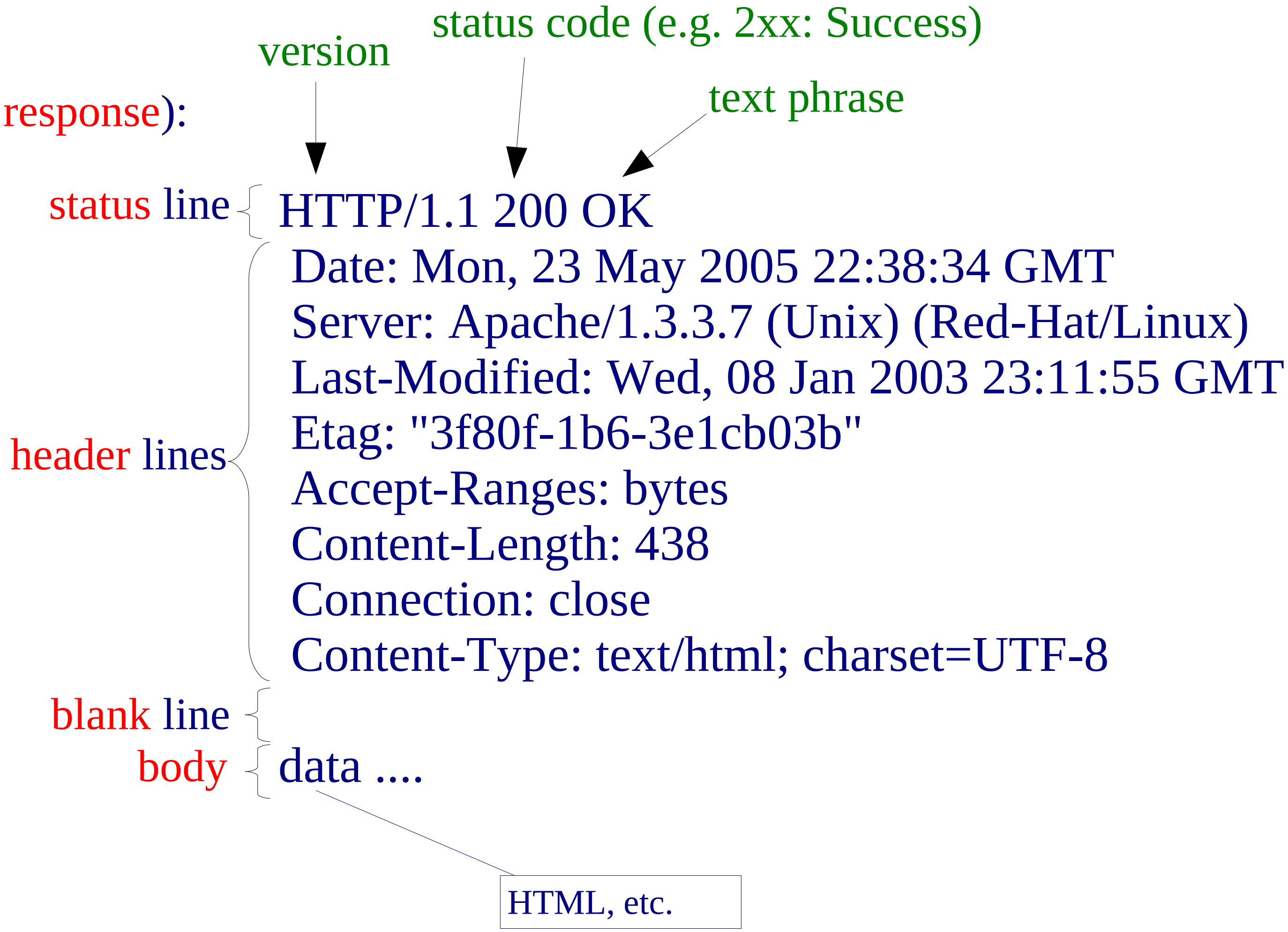
Example ('| jq' is ancillary):

```
curl https://gw1.vocdoni.net/dvote -X POST -H Content-Type:application/json –data \
'{"id": "req00'$RANDOM'", "request": {"method": "getStats", "timestamp":'$(date + \
%s)'}}' | jq
```

Unit 5. Network applications

Web – HTTP Messages, RFC2616

- Server (HTTP response):



version

status code (e.g. 2xx: Success)

text phrase

status line

header lines

blank line

body

```
HTTP/1.1 200 OK
 Date: Mon, 23 May 2005 22:38:34 GMT
 Server: Apache/1.3.3.7 (Unix) (Red-Hat/Linux)
 Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT
 Etag: "3f80f-1b6-3e1cb03b"
 Accept-Ranges: bytes
 Content-Length: 438
 Connection: close
 Content-Type: text/html; charset=UTF-8

data ....
```

HTML, etc.

Unit 5. Network applications

Web – HTTP Messages, RFC2616

- Status codes:
  - 1xx informational response – the request was received, continuing process
  - 2xx successful – the request was successfully received, understood, and accepted
  - 3xx redirection – further action needs to be taken in order to complete the request
  - 4xx client error – the request contains bad syntax or cannot be fulfilled
  - 5xx server error – the server failed to fulfill an apparently valid request

- Some well-know status codes:
  - 200 OK
  - 304 Not Modified
  - 400 Bad Request
  - 403 Forbidden
  - 404 Not Found
  - 500 Internal Server Error
  - 502 Bad Gateway

Used by proxies

Unit 5. Network applications

Web – HTTP Messages, RFC2616

- Header
  - Last-Modified: date, used in conditional retrieval.
  - Etag: id, used in conditional retrieval.
  - Connection: keep-alive/close, controls whether or not the network connection stays open after the current transaction.
  - Accept: <MIME_type>/<MIME_subtype>, acceptable mime types.
  - ...

Example HTTP 1.0 interactive sessions (double intro needed):

```
$ telnet www.google.com 80
Trying 2a00:1450:4003:80d::2004...
Connected to www.google.com.
Escape character is '^]'.
GET / HTTP/1.0

HTTP/1.0 200 OK
Date: Thu, 26 May 2022 09:57:36 GMT
Expires: -1
...
```

Example HTTP 1.1 interactive session (Host is mandatory in HTTP 1.1):

```
$ telnet www.google.com 80
Trying 2a00:1450:4003:80d::2004...
Connected to www.google.com.
Escape character is '^]'.
GET / HTTP/1.1
Host: www.google.com

HTTP/1.1 200 OK
Date: Thu, 26 May 2022 09:59:59 GMT
Expires: -1
...
```

Example HTTP 1.1 TLS interactive session:

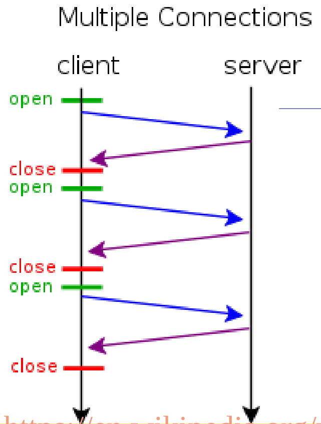
```
$ openssl s_client -connect \
www.upc.edu:443
CONNECTED(00000003)
depth=2 C = US, O = DigiCert Inc, OU =
www.digicert.com, CN = DigiCert
Assured ID Root CA
verify return:1
...
    Extended master secret: yes
---
GET /ca HTTP/1.1
Host: www.upc.edu

HTTP/1.1 200 OK
Date: Wed, 25 May 2022 18:31:52 GMT
...
```

GET commands must be manually typed

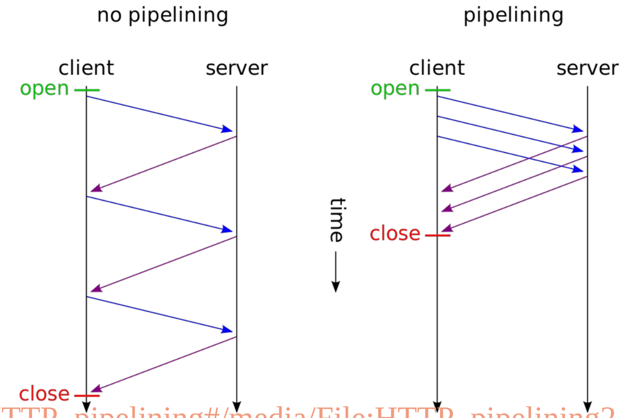
Unit 5. Network applications

Web – Persistent/non Persistent connections

- Non persistent (default in HTTP/1.0): The server close the TCP connection after every object. E.g, for an html page with 10 jpeg images, 11 TCP connections are sequentially opened.
- Persistent (default in HTTP/1.1) : The server maintains the TCP connection opened until an inactivity time. In the example: All 11 objects would be sent over the same TCP connection.
- Persistent connections with pipelining (supported only in HTTP/1.1): The client issues new requests as soon as it encounter new references, even if the objects have been not completely downloaded. In the example: All 11 objects would be sent over the same TCP connection.



Parallel TCP connections are possible. In fact, many browsers do it.



Source: https://en.wikipedia.org/wiki/HTTP_pipelining#/media/File:HTTP_pipelining2.svg

Source: https://en.wikipedia.org/wiki/File:HTTP_persistent_connection.svg

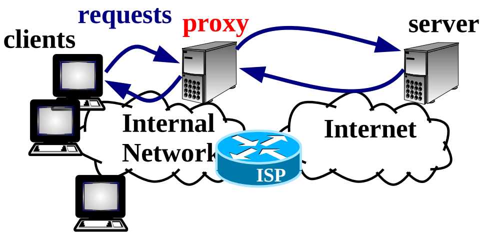
Unit 5. Network applications

Web – Caching and Proxies

- Caching: The client stores downloaded pages in a local cache. Conditional GET requests are used to download pages if necessary. It can use the Date and/or Etag:

```
GET /index.html HTTP/1.1
Host: www.example.com
If-Modified-Since: October 21, 2002 4:57 PM
If-None-Match: "686897696a7c876b7e"
```

- Proxy server: Acts as an intermediary for requests from clients.
  - Advantages:
    - Security (the proxy may reject the access to unauthorized servers)
    - Logs
    - Caching
    - Save public IP addresses (only the proxy may have access to the Internet)
    - ...



Drawback: cannot work in HTTPS due to end-to-end encryption

Unit 5. Network applications

Web – web based applications

- Components:
  - Presentation: A web browser (client side).
  - Engine generating “on the fly” HTML pages (server side).
  - Storage: a database (e.g. mysql).
- Benefits:
  - Fast to deploy and upgrade (only server side).
  - Only a compatible browser is required at the client side.
  - Provide cross-platform compatibility (i.e., Windows, Mac, Linux, etc.)

Unit 5. Network applications

Outline

- DNS
- Email
- Web
- **Charsets**
- HTML

Languages, cultures, alphabets

7400 million people (2016)

22% speak Chinese, 11% English, 7% Spanish, 0,1% Catalan

Apart from languages, there are cultures and alphabets

- Language with several cultures: es_ES, es_CO ("locale")
- Alphabet shared by several languages (e.g. català & français)

Culture:

- Messages, character sets, transliteration, ordering, search in strings, hours and dates, numbers and currency, pronunciation, …

Interaction between agents in different languages and cultures: alphabets and character sets

Languages, cultures, alphabets

Internacionalization (i18n), Localization (l10n)

Alphabets

- "base": ascii
- National: e.g.: latin-1 (includes ascii), kanji
- International: e.g.: unicode (includes latin-1 and “all” languages)

Expression or language negotiation (in HTTP):

Accept-Language: es, ca, en-gb, en
Accept-Charset: iso-8859-15, unicode-9-0
. . .

Content-Language: ca
Content-Type: text/html; charset=utf-8
. . .

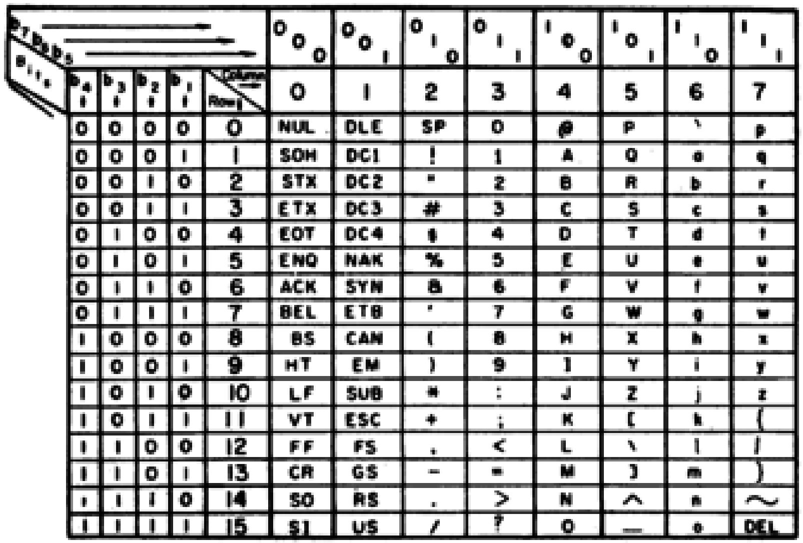
English is the default …

Character sets

Characters are encoded following several conventions:

- **repertoire**: a set of characters (name and representation (glyph))
- **code**: correspondence between repertoire and natural numbers.
- **encoding**: method (algorithm) to convert code numbers into a sequence of octets (> 256 characters)
- US-ASCII: 95 characters + control=128: 7 bits (1 octet sent)

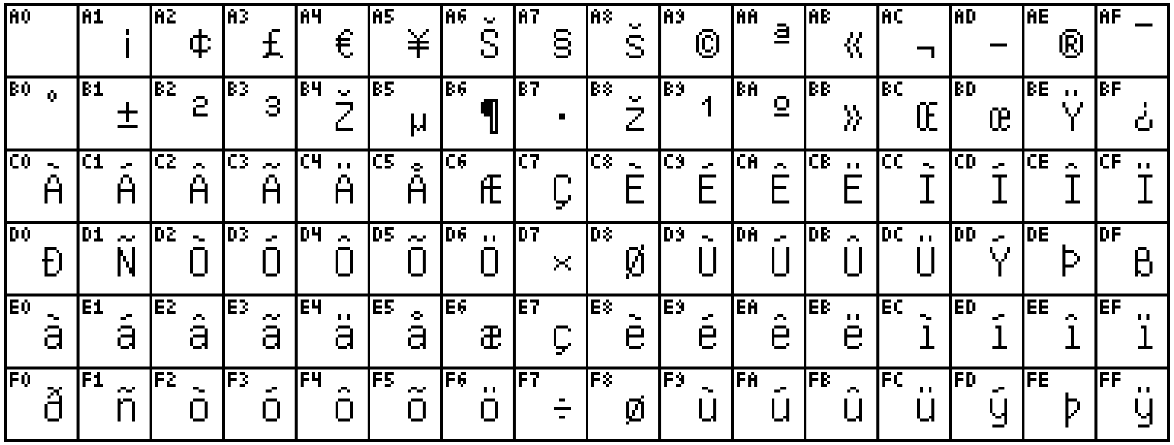
USASCII code chart



				b7 b6 b5	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b4	b3	b2	b1	Column / Row	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	|
1	1	0	1	13	CR	GS	-	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

ISO 8859

- ISO 8859-1 (ISO Latin 1): 190 + control = 256: 1 octet Western European, default for HTTP



A0	A1 ¡	A2 ¢	A3 £	A4 €	A5 ¥	A6 Š	A7 §	A8 š	A9 ©	AA ª	AB «	AC ¬	AD -	AE ®	AF ¯
B0 °	B1 ±	B2 ²	B3 ³	B4 Ž	B5 µ	B6 ¶	B7 ·	B8 ž	B9 ¹	BA º	BB »	BC Œ	BD œ	BE Ÿ	BF ¿
C0 À	C1 Á	C2 Â	C3 Ã	C4 Ä	C5 Å	C6 Æ	C7 Ç	C8 È	C9 É	CA Ê	CB Ë	CC Ì	CD Í	CE Î	CF Ï
D0 Ð	D1 Ñ	D2 Ò	D3 Ó	D4 Ô	D5 Õ	D6 Ö	D7 ×	D8 Ø	D9 Ù	DA Ú	DB Û	DC Ü	DD Ý	DE Þ	DF ß
E0 à	E1 á	E2 â	E3 ã	E4 ä	E5 å	E6 æ	E7 ç	E8 è	E9 é	EA ê	EB ë	EC ì	ED í	EE î	EF ï
F0 ð	F1 ñ	F2 ò	F3 ó	F4 ô	F5 õ	F6 ö	F7 ÷	F8 ø	F9 ù	FA ú	FB û	FC ü	FD ý	FE þ	FF ÿ

- More variants

ISO 8859-15 extends -1 + Ÿ, €

ISO 8859-2 (Central European)

ISO 8859-4 (North European)

ISO 8859-5 (Cyrillic)

ISO 8859-6 (Arabic) — Most common Arabic glyphs

ISO 8859-7 (Greek)

ISO 8859-8 (Hebrew) — modern Hebrew.

ISO 8859-9 (Turkish, Kurdish)

ISO 8859-11 (Thai) — Contains most glyphs needed

Universal Coded Character Set
Unicode

All characters from all written languages + math + emoticons + +=Universal Character set (ucs)

Encoding: UCS-4 bytes (fixed length)

Proportional spacing, language independent

Unicode consortium: synchronized with🤣O,🥘

- Unicode 9.0.0 (7/2016): 140,186 symbols (2023/05/29)
- U+hex code: U+0020 = ' '

Character Encoding: Universal Transformation Format (UTF)

- Difficulty or impossibility to transport 8 o 16 bits data in Internet protocols:
- **UTF-8**, UTF-16, UTF-32 (variable length)

http://www.unicode.org

Variable length encodings

- UTF-8 (8 bits) (rfc2044)
  - One to four 8-bit code units
  - Most common in the Internet
  - Preserves ASCII codes

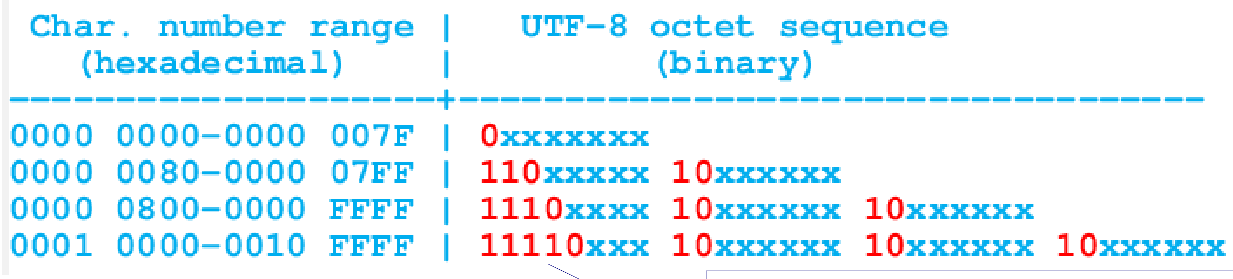
```
Content-Type: text/plain; charset=UTF-8
Content-Transfer-Encoding: 8bit
CatalÃ  , FranÃ§ais, TÃ¤mÃ¤ on testi.
```

- UTF-16 (16 bits)
  - One or two 16-bit code units
- UTF-32 (32 bits)
  - Fixed-length 32-bit code units

Universal Coded Character Set Unicode

UTF-8 Encoding

- Determine high-order bits from the number of octets
- Fill in the bits marked x



```
 Char. number range  |        UTF-8 octet sequence
    (hexadecimal)    |              (binary)
--------------------+---------------------------------------------
0000 0000-0000 007F | 0xxxxxxx
0000 0080-0000 07FF | 110xxxxx 10xxxxxx
0000 0800-0000 FFFF | 1110xxxx 10xxxxxx 10xxxxxx
0001 0000-0010 FFFF | 11110xxx 10xxxxxx 10xxxxxx 10xxxxxx
```

• Self-synchronization => it is possible to identify any character at any time (no need to restart reading from the beginning of the communication -it is not the case in ASCII):
The number of bytes of the character is determined by the combination of the initial bits

- Example
  - character: €
  - code point: U+20AC
  - code point in binary (12 bits): 10 0000 1010 1100
  - 3 code units required:
  - UTF-8: 11100010 10000010 10101100
  - UTF-8 in hex: E282AC

Unit 5. Network applications

Outline

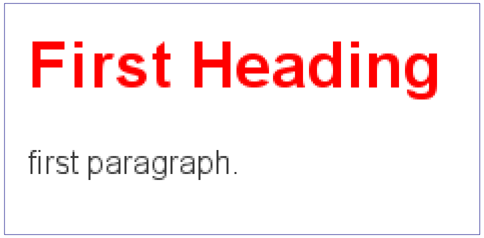
- DNS
- Email
- Web
- Charsets
- **HTML**

Unit 5. Network applications

HTML – Hyper-Text Markup Language, HTML

- Tim Berners-Lee defined HTML in 1989. HTML design mail goal was displaying formated text documents with hyperlinks (including links to other documents) in web browsers.
- Based on tags e.g. <head> data </head>
- Example:

```
<html>
<head>
 <title>Basic html document</title>
</head>
<body>
  <h1><font color="red">First Heading</font></h1>
  <p>first paragraph.</p>
</body>
</html>
```



First Heading

first paragraph.

Terminology:

- element
- attribute
- text

Unit 5. Network applications

HTML – Hyper-Text Markup Language, HTML

- HTML features (1):
  - Hyperlinks: Click on a link and jump to another document
  - Forms: The document accept user inputs that are sent to the server
  - Scripting: Allow adding programs. The program executes on the client's machine when the document loads, or at some other time such as when a link is activated.

HTTP GET

HTTP POST

JavaScript

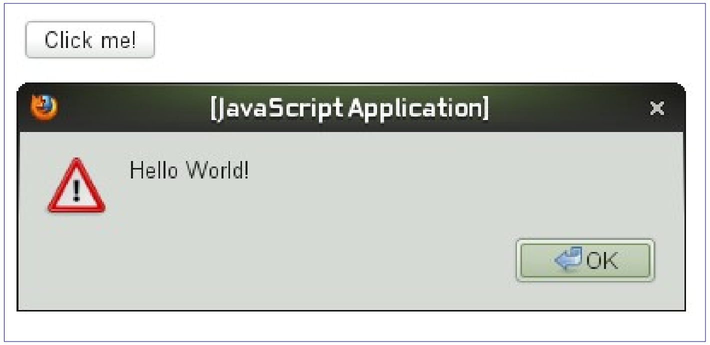
- Hyperlinks
  - <a> tag defines an hyperlink
  - Syntax:
    - » <a href="url">link text</a>
  - Example:
    - » <a href="https://studies.ac.upc.edu/FIB/grau/XC/">XC-GRAU</a>

Unit 5. Network applications

HTML – Hyper-Text Markup Language, HTML

- HTML features (2):
  - javascript example:

```
<html>
<head>
<script type="text/javascript">
 function displaymessage() {
   alert("Hello World!");
 }
</script>
</head>
<body>
 <form>
  <input type="button"
   value="Click me!" onclick="displaymessage()" />
 </form>
</body>
</html>
```



Hello World!

Unit 5. Network applications

WEB & HTML – Example: “Hello, world!”

1) Create (copy & paste) index.html and index2.html files (source code from previous slides)

2) In terminal 1 run Wireshark and observe the captured traces while working in terminal 3 and the web browser:

```
wireshark -n -i any -k -f "host 127.0.1.1"
```

3) In terminal 2 run the Python http module (server):

```
python3 -m http.server -d <path> -b 127.0.1.1 8080
```

Path to directory of index.html and index2.html

4) In terminal 3 run a telnet client (GET … commands must be explicitly typed and intro pushed twice):

```
telnet 127.0.1.1 8080
...
GET / HTTP/1.0
...
```

5) In terminal 3 run a telnet client:

```
telnet 127.0.1.1 8080
...
GET /index2.html HTTP/1.1
...
```

6) Open a web browser and visit http://127.0.1.1:8080 and http://127.0.1.1:8080/index2

index.html

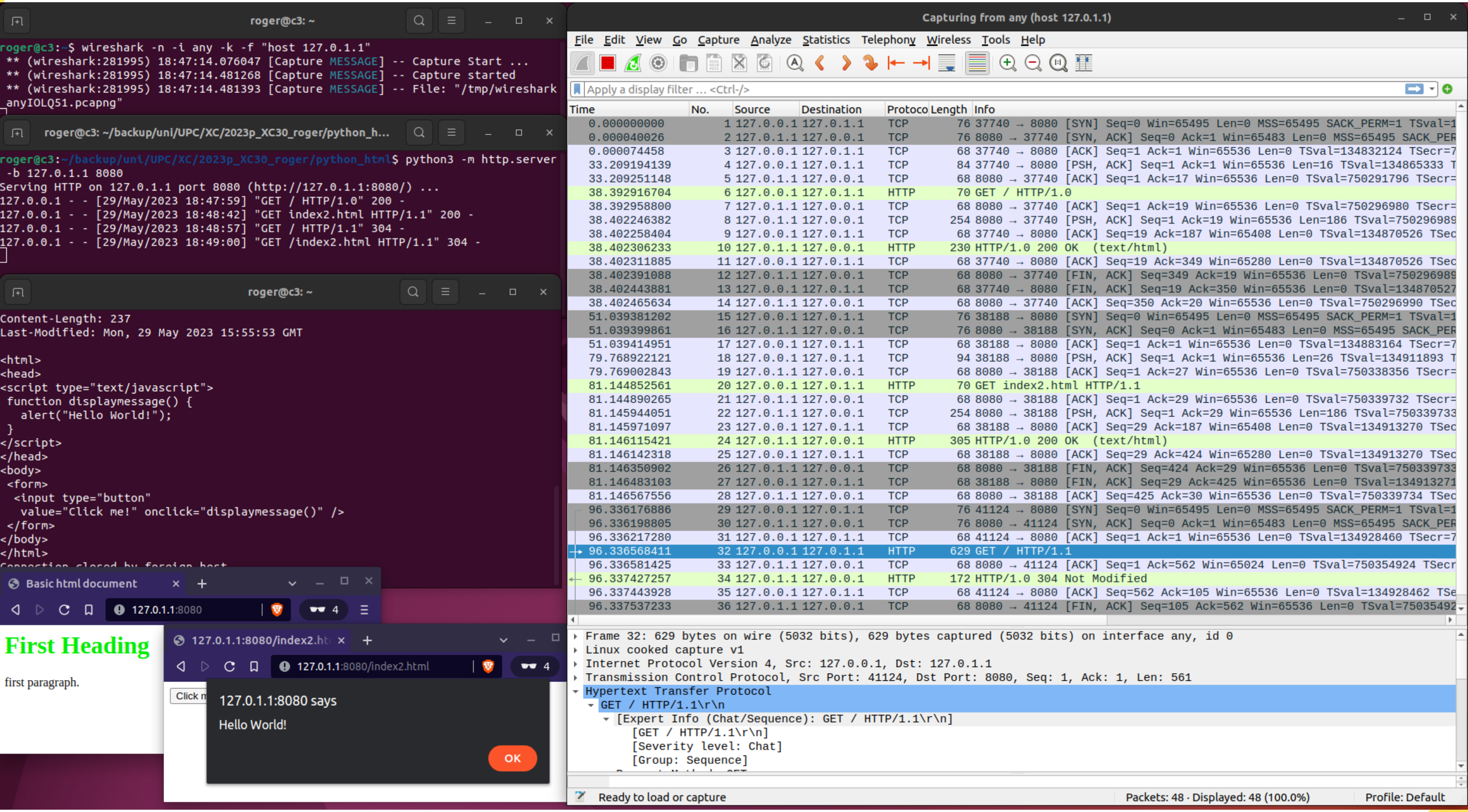
```
<html>
<head>
 <title>Basic html document</title>
</head>
<body>
  <h1><font color=”red”>First Heading</font></h1>
  <p>first paragraph.</p>
</body>
</html>
```

index2.html

```
<html>
<head>
<script type="text/javascript">
 function displaymessage() {
   alert("Hello World!");
 }
</script>
</head>
<body>
 <form>
  <input type="button"
   value="Click me!" onclick="displaymessage()" />
 </form>
</body>
</html>
```

Unit 5. Network applications

WEB & HTML – Example: Hello world



```
roger@c3:~$ wireshark -n -i any -k -f "host 127.0.1.1"
 ** (wireshark:281995) 18:47:14.076047 [Capture MESSAGE] -- Capture Start ...
 ** (wireshark:281995) 18:47:14.481268 [Capture MESSAGE] -- Capture started
 ** (wireshark:281995) 18:47:14.481393 [Capture MESSAGE] -- File: "/tmp/wireshark
_anyIOLQ51.pcapng"
```

```
roger@c3:~/backup/uni/UPC/XC/2023p_XC30_roger/python_html$ python3 -m http.server
 -b 127.0.1.1 8080
Serving HTTP on 127.0.1.1 port 8080 (http://127.0.1.1:8080/) ...
127.0.0.1 - - [29/May/2023 18:47:59] "GET / HTTP/1.0" 200 -
127.0.0.1 - - [29/May/2023 18:48:42] "GET index2.html HTTP/1.1" 200 -
127.0.0.1 - - [29/May/2023 18:48:57] "GET / HTTP/1.1" 304 -
127.0.0.1 - - [29/May/2023 18:49:00] "GET /index2.html HTTP/1.1" 304 -
```

```
Content-Length: 237
Last-Modified: Mon, 29 May 2023 15:55:53 GMT

<html>
<head>
<script type="text/javascript">
 function displaymessage() {
   alert("Hello World!");
 }
</script>
</head>
<body>
 <form>
  <input type="button"
   value="Click me!" onclick="displaymessage()" />
 </form>
</body>
</html>
Connection closed by foreign host
```

First Heading

first paragraph.

127.0.1.1:8080 says

Hello World!

Capturing from any (host 127.0.1.1)
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
Time          No. Source    Destination Protoco Length Info
  0.000000000   1 127.0.0.1 127.0.1.1   TCP      76 37740 → 8080 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=1
  0.000040026   2 127.0.1.1 127.0.0.1   TCP      76 8080 → 37740 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PER
  0.000074458   3 127.0.0.1 127.0.1.1   TCP      68 37740 → 8080 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=134832124 TSecr=7
 33.209194139   4 127.0.0.1 127.0.1.1   TCP      84 37740 → 8080 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=16 TSval=134865333 T
 33.209251148   5 127.0.1.1 127.0.0.1   TCP      68 8080 → 37740 [ACK] Seq=1 Ack=17 Win=65536 Len=0 TSval=750291796 TSecr=
 38.392916704   6 127.0.0.1 127.0.1.1   HTTP     70 GET / HTTP/1.0
 38.392958800   7 127.0.1.1 127.0.0.1   TCP      68 8080 → 37740 [ACK] Seq=1 Ack=19 Win=65536 Len=0 TSval=750296980 TSecr=
 38.402246382   8 127.0.1.1 127.0.0.1   TCP     254 8080 → 37740 [PSH, ACK] Seq=1 Ack=19 Win=65536 Len=186 TSval=750296989
 38.402258404   9 127.0.0.1 127.0.1.1   TCP      68 37740 → 8080 [ACK] Seq=19 Ack=187 Win=65408 Len=0 TSval=134870526 TSec
 38.402306233  10 127.0.1.1 127.0.0.1   HTTP    230 HTTP/1.0 200 OK  (text/html)
 38.402311885  11 127.0.0.1 127.0.1.1   TCP      68 37740 → 8080 [ACK] Seq=19 Ack=349 Win=65280 Len=0 TSval=134870526 TSec
 38.402391088  12 127.0.1.1 127.0.0.1   TCP      68 8080 → 37740 [FIN, ACK] Seq=349 Ack=19 Win=65536 Len=0 TSval=750296989
 38.402443881  13 127.0.0.1 127.0.1.1   TCP      68 37740 → 8080 [FIN, ACK] Seq=19 Ack=350 Win=65536 Len=0 TSval=134870527
 38.402465634  14 127.0.1.1 127.0.0.1   TCP      68 8080 → 37740 [ACK] Seq=350 Ack=20 Win=65536 Len=0 TSval=750296990 TSec
 51.039381202  15 127.0.0.1 127.0.1.1   TCP      76 38188 → 8080 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=1
 51.039399861  16 127.0.1.1 127.0.0.1   TCP      76 8080 → 38188 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PER
 51.039414951  17 127.0.0.1 127.0.1.1   TCP      68 38188 → 8080 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=134883164 TSecr=7
 79.768922121  18 127.0.0.1 127.0.1.1   TCP      94 38188 → 8080 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=26 TSval=134911893 T
 79.769002843  19 127.0.1.1 127.0.0.1   TCP      68 8080 → 38188 [ACK] Seq=1 Ack=27 Win=65536 Len=0 TSval=750338356 TSecr=
 81.144852561  20 127.0.0.1 127.0.1.1   HTTP     70 GET index2.html HTTP/1.1
 81.144890265  21 127.0.1.1 127.0.0.1   TCP      68 8080 → 38188 [ACK] Seq=1 Ack=29 Win=65536 Len=0 TSval=750339732 TSecr=
 81.145944051  22 127.0.1.1 127.0.0.1   TCP     254 8080 → 38188 [PSH, ACK] Seq=1 Ack=29 Win=65536 Len=186 TSval=750339733
 81.145971097  23 127.0.0.1 127.0.1.1   TCP      68 38188 → 8080 [ACK] Seq=29 Ack=187 Win=65408 Len=0 TSval=134913270 TSec
 81.146115421  24 127.0.1.1 127.0.0.1   HTTP    305 HTTP/1.0 200 OK  (text/html)
 81.146142318  25 127.0.0.1 127.0.1.1   TCP      68 38188 → 8080 [ACK] Seq=29 Ack=424 Win=65280 Len=0 TSval=134913270 TSec
 81.146350902  26 127.0.1.1 127.0.0.1   TCP      68 8080 → 38188 [FIN, ACK] Seq=424 Ack=29 Win=65536 Len=0 TSval=750339733
 81.146483103  27 127.0.0.1 127.0.1.1   TCP      68 38188 → 8080 [FIN, ACK] Seq=29 Ack=425 Win=65536 Len=0 TSval=134913271
 81.146567556  28 127.0.1.1 127.0.0.1   TCP      68 8080 → 38188 [ACK] Seq=425 Ack=30 Win=65536 Len=0 TSval=750339734 TSec
 96.336176886  29 127.0.0.1 127.0.1.1   TCP      76 41124 → 8080 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=1
 96.336198805  30 127.0.1.1 127.0.0.1   TCP      76 8080 → 41124 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PER
 96.336217280  31 127.0.0.1 127.0.1.1   TCP      68 41124 → 8080 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=134928460 TSecr=7
 96.336568411  32 127.0.0.1 127.0.1.1   HTTP    629 GET / HTTP/1.1
 96.336581425  33 127.0.1.1 127.0.0.1   TCP      68 8080 → 41124 [ACK] Seq=1 Ack=562 Win=65024 Len=0 TSval=750354924 TSecr
 96.337427257  34 127.0.1.1 127.0.0.1   HTTP    172 HTTP/1.0 304 Not Modified
 96.337443928  35 127.0.0.1 127.0.1.1   TCP      68 41124 → 8080 [ACK] Seq=562 Ack=105 Win=65536 Len=0 TSval=134928462 TSe
 96.337537233  36 127.0.1.1 127.0.0.1   TCP      68 8080 → 41124 [FIN, ACK] Seq=105 Ack=562 Win=65536 Len=0 TSval=75035492
```

```
▸ Frame 32: 629 bytes on wire (5032 bits), 629 bytes captured (5032 bits) on interface any, id 0
▸ Linux cooked capture v1
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.1.1
▸ Transmission Control Protocol, Src Port: 41124, Dst Port: 8080, Seq: 1, Ack: 1, Len: 561
▾ Hypertext Transfer Protocol
  ▾ GET / HTTP/1.1\r\n
    ▾ [Expert Info (Chat/Sequence): GET / HTTP/1.1\r\n]
        [GET / HTTP/1.1\r\n]
        [Severity level: Chat]
        [Group: Sequence]
```

Ready to load or capture | Packets: 48 · Displayed: 48 (100.0%) | Profile: Default