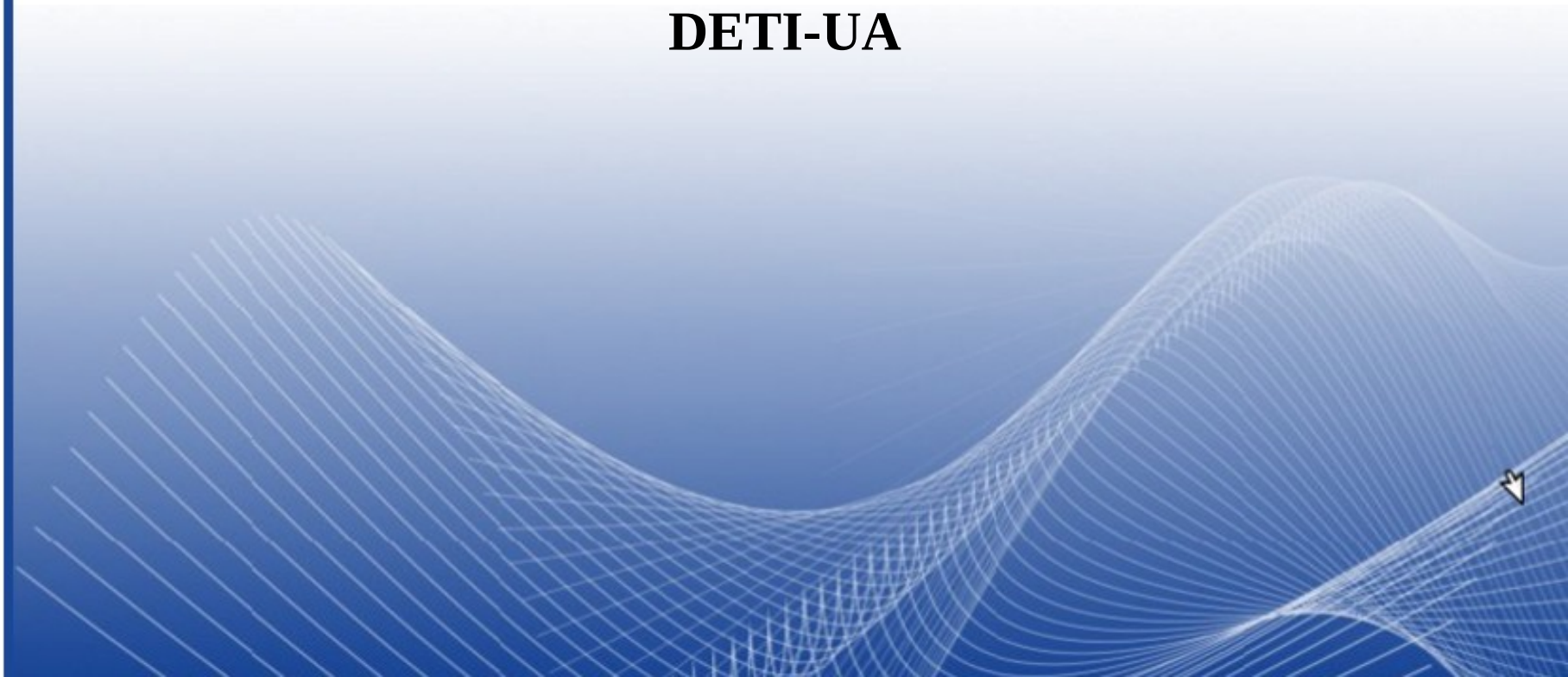
Services and Applications
(Client-Server Model)

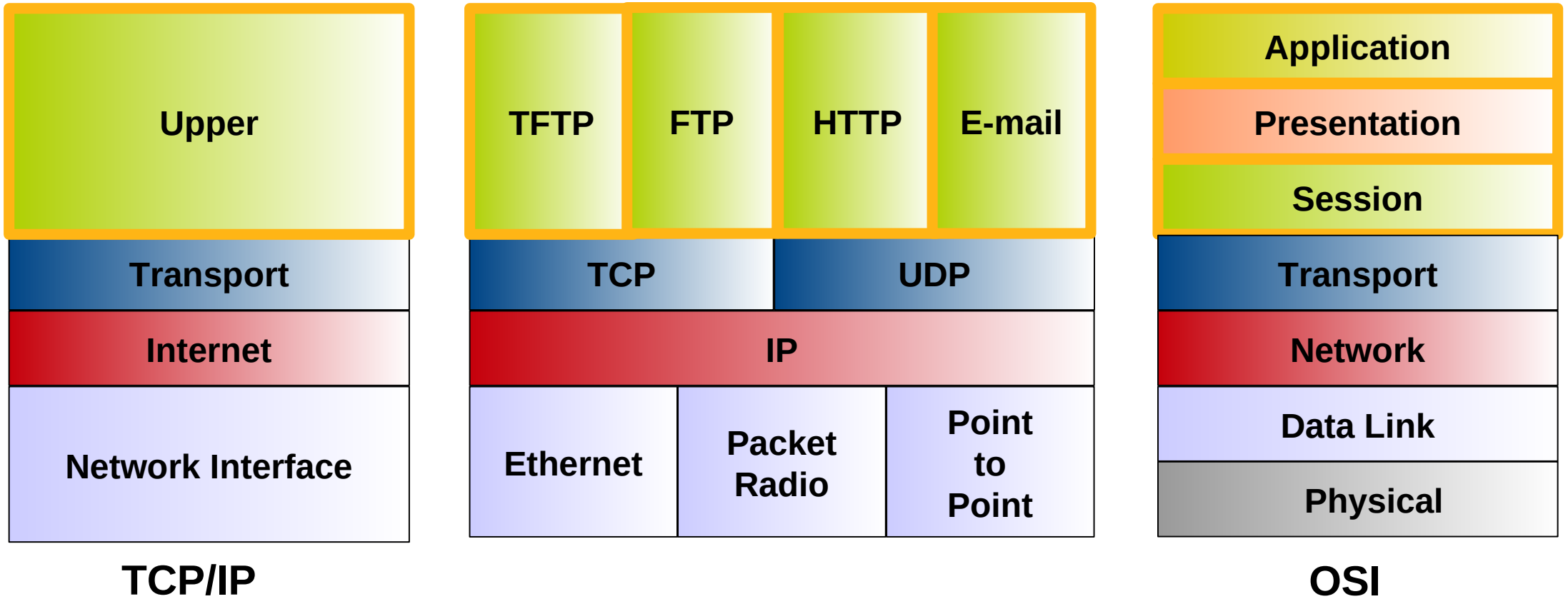
Redes de Comunicações I

Licenciatura em Engenharia de Computadores e Informática
DETI-UA

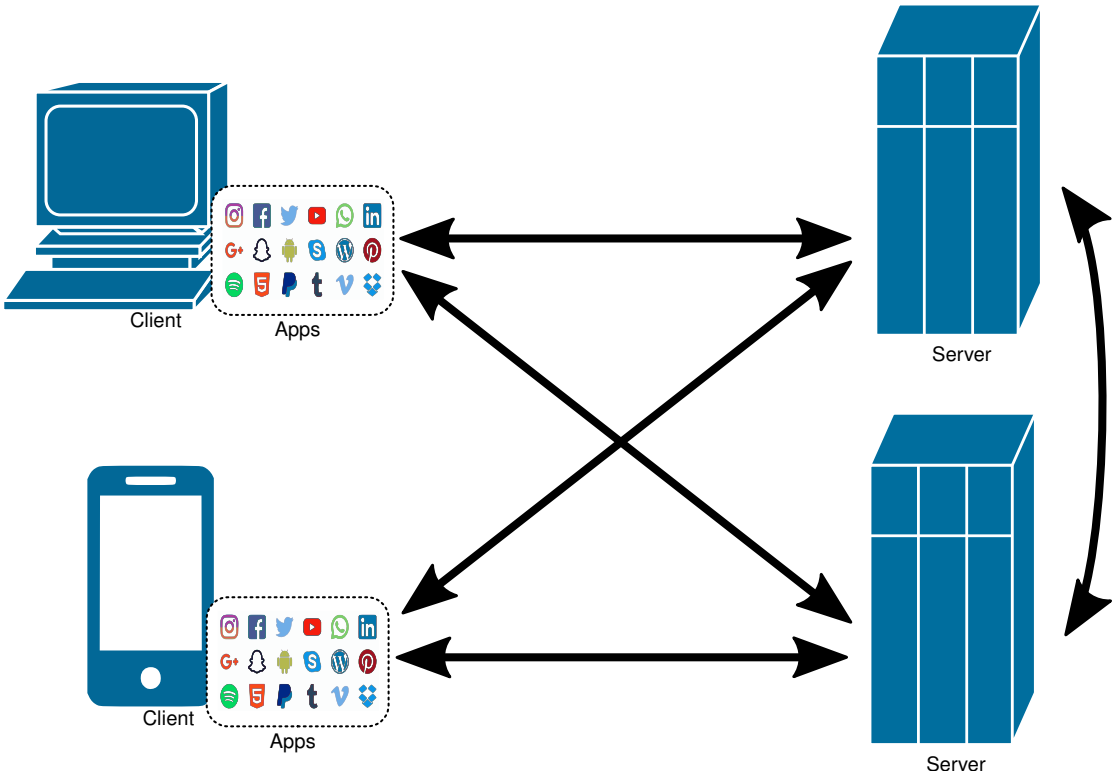


universidade de aveiro
deti.ua.pt

TCP/IP Reference Model



Client-Server Model



Servers:

- Always ON.
- IP address is always the same or exists a static association between a name and a dynamic IP address.
- May communicate between them.
  - May act as client.

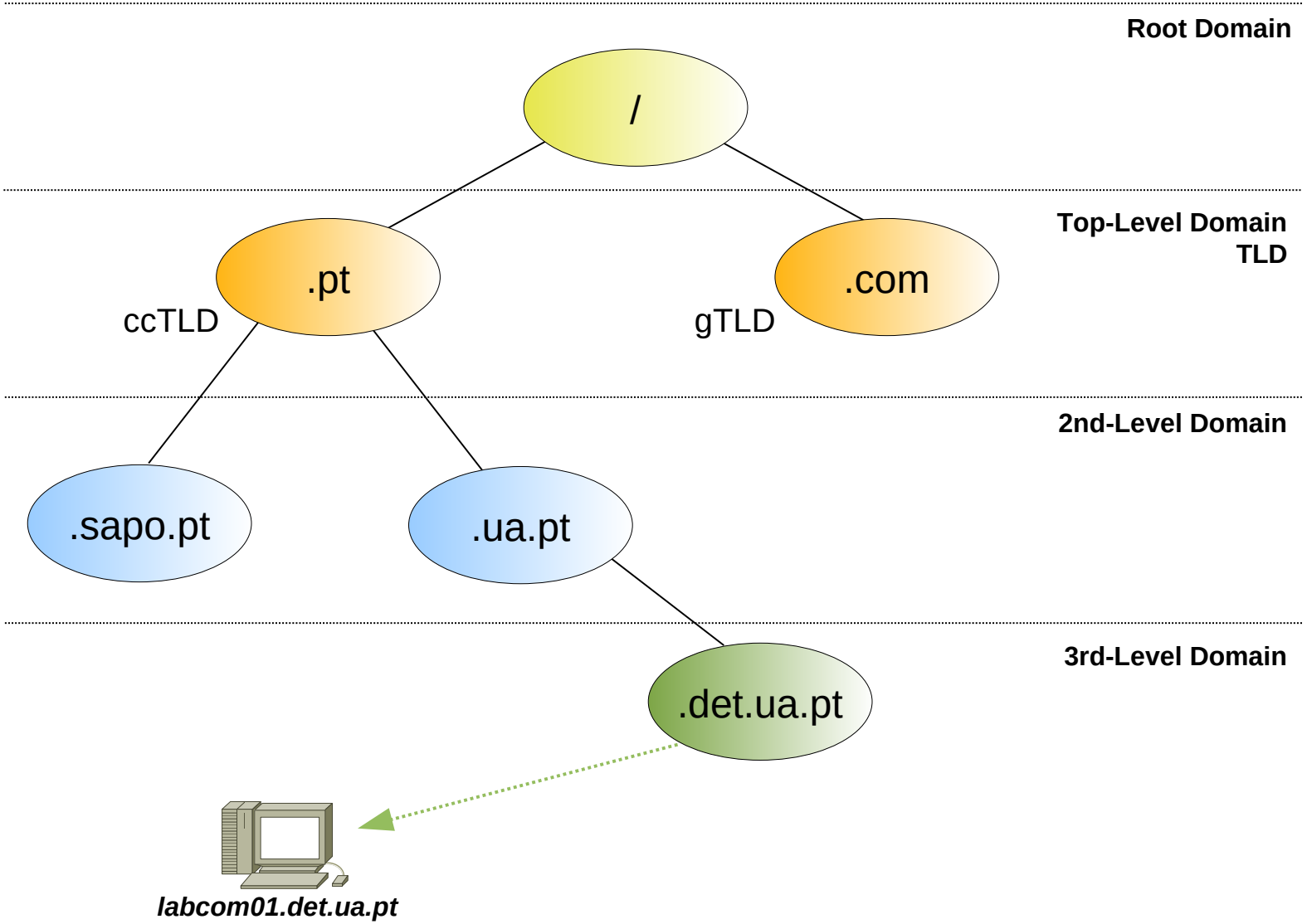
Clients:

- Communicate with servers.
- Can be ON only when in operation.
- May have dynamic addresses.
- Within this model, they do not communicate between themselves.
  - P2P is another communication model.

Domain Name System (DNS)

- Distributed database system that facilitates a translation service (resolution) between host names and IP addresses.
- Allows also the translation/resolution between IP addresses and host names
  - The name ”DD.CC.BB.AA.**in-addr.arpa**” allows the resolution of the IPv4 address AA.BB.CC.DD
  - The name 0.0.0.0.8.b.d.0.1.0.0.2.**ip6.arpa** allows the resolution of the IPv6 address 2001:0db8:0000::/48
  - Resolution name-ip and ip-name is not symmetrical.
- Organizes the names in domains according to an hierarchical structure.
- Each DNS system defines one or more zones over which has the resolution authority.

Hierarchical Structure of Domain Names



Root Servers & Root Zone File

- Root servers

- Root Zone File (sample)

```
.....
COM. NS A.GTLD-SERVERS.NET.
COM. NS G.GTLD-SERVERS.NET.
COM. NS H.GTLD-SERVERS.NET.
COM. NS C.GTLD-SERVERS.NET.
.....
PT. NS NS.DNS.BR.
PT. NS NS2.NIC.FR.
PT. NS NS.DNS.PT.
PT. NS SUNIC.SUNET.SE.
PT. NS NS2.DNS.PT.
PT. NS NS-EXT.ISC.ORG.
.....
NET. NS A.GTLD-SERVERS.NET.
NET. NS G.GTLD-SERVERS.NET.
NET. NS H.GTLD-SERVERS.NET.
NET. NS C.GTLD-SERVERS.NET.
.....
INFO. NS B0.INFO.AFILIAS-NST.ORG.
INFO. NS C0.INFO.AFILIAS-NST.INFO.
INFO. NS D0.INFO.AFILIAS-NST.ORG.
.....
```

Top-Level Domains (TLD)

- gTLDs (generic TLDs)
  - .com, .edu, .gov, .mil, .net, .org, .int, .aero, .biz, .coop, .info, .museum, .name, .pro, .cat, .jobs, .mobi, .travel, .tel, .asia
- ccTLDs (country code TLDs)
  - 2 letter domains that identify a specific country (ISO 3166)
  - Management is delegated (by ICANN) to a governmental institution from each country.
    - Those can (re)-delegate in private companies.
  - Ex: .pt, .es, .us, .fr, etc...
- New gTLDs
  - Over 1300 new gTLDs could become available in the next few years.
  - Trademarks: **.goog**, **.goggle**, **.apple**, **.yahoo**, **.honda**, **.barcelona**, ...
  - .xyz, .top, .wang, .win, .link, .site, .club, .app, .live, .cloud, .bank, .online, .bet, .book, .cars, .hotel, ...

TLD Zone Files (sample)

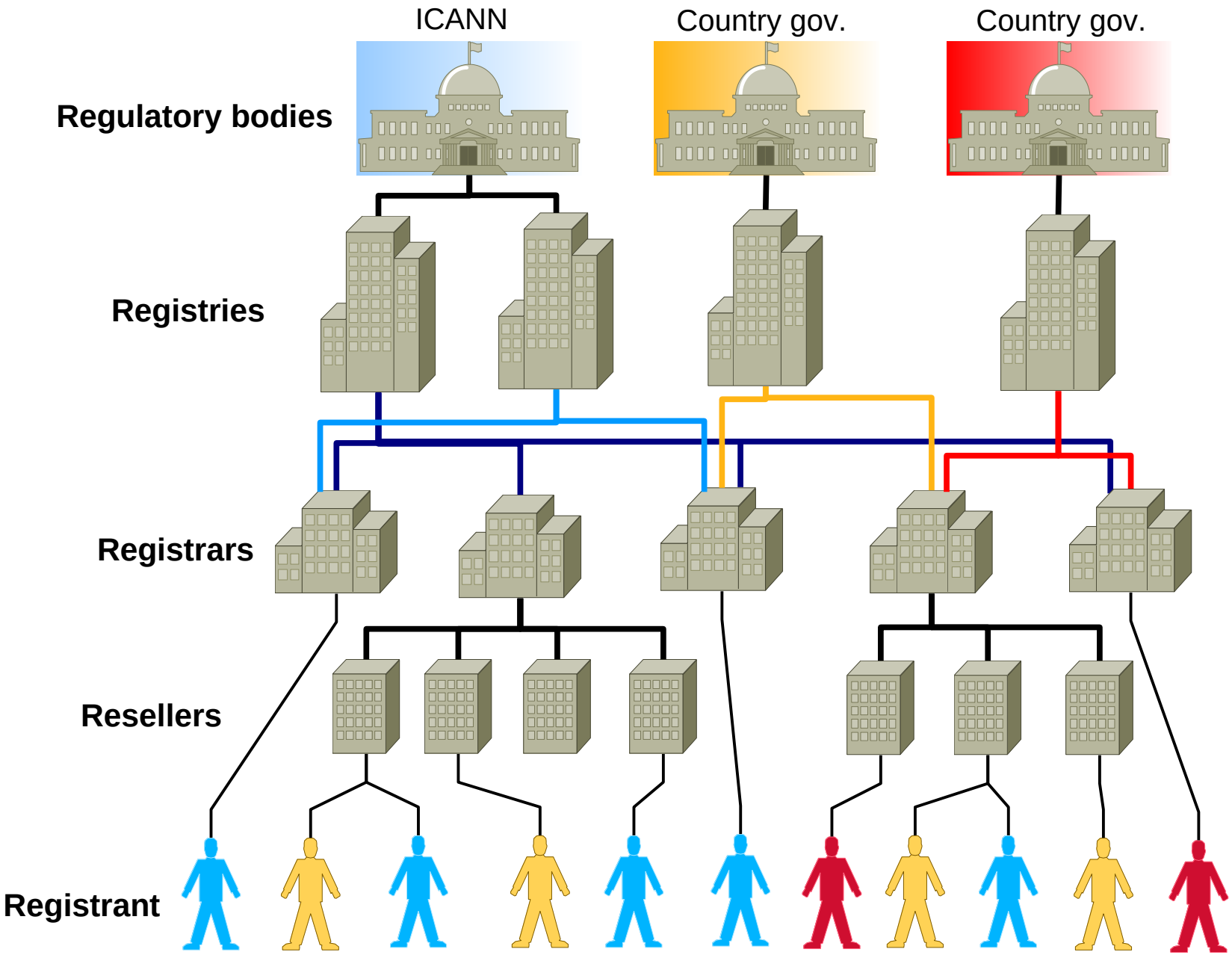
- .ORG (Public Interest Registry)

```
.....
AASELFSTORAGE.ORG. NS DNS02.GPN.REGISTER.COM.
AASELFSTORAGE.ORG. NS DNS03.GPN.REGISTER.COM.
AASELFSTORAGE.ORG. NS DNS04.GPN.REGISTER.COM.
AASELFSTORAGE.ORG. NS DNS05.GPN.REGISTER.COM.
AASEMI.ORG. NS DPNS1.DNSNAMESERVER.ORG.
AASEMI.ORG. NS DPNS2.DNSNAMESERVER.ORG.
AASEMI.ORG. NS DPNS3.DNSNAMESERVER.ORG.
AASEMI.ORG. NS DPNS4.DNSNAMESERVER.ORG.
AASEN.ORG. NS NS1.MAILBANK.COM.
AASEN.ORG. NS NS2.MAILBANK.COM.
AASENIORMORTGAGE.ORG. NS NS13.DOMAINCONTROL.COM.
AASENIORMORTGAGE.ORG. NS NS14.DOMAINCONTROL.COM.
AASENT.ORG. NS NS51.1AND1.COM.
AASENT.ORG. NS NS52.1AND1.COM.
AASENTMORTGAGE.ORG. NS NS51.1AND1.COM.
AASENTMORTGAGE.ORG. NS NS52.1AND1.COM.
AASENY.ORG. NS NS27.1AND1.COM.
AASENY.ORG. NS NS28.1AND1.COM.
AASEP.ORG. NS NS1.CASTIRONCODING.COM.
AASEP.ORG. NS NS2.CASTIRONCODING.COM.
AASERV.ORG. NS NS1.RENEWYOURNAME.NET.
.....
```

- .COM (Verisign)

```
.....
AMERICANHUNTING NS NS1.HITFARM
AMERICANHUNTING NS NS2.HITFARM
ATSCAF NS CBRU.BR.NS.ELS-GMS.ATT.NET.
ATSCAF NS CMTU.MT.NS.ELS-GMS.ATT.NET.
ACTIONNETS NS NS.TULSAWEB
ACTIONNETS NS NS.TIBP
ACI-APPLICAD NS NS2.WEBNJ.NET.
ACI-APPLICAD NS NS1.WEBNJ.NET.
ANZAPACK NS DNS3.TERRA.ES.
ANZAPACK NS DNS4.TERRA.ES.
ALPHASOFTDE NS DNS1.EPAG.NET.
ALPHASOFTDE NS DNS2.EPAG.NET.
ALPHASOFTDE NS DNS01.KUTTIG.NET.
AAI-TENN NS AUTH00.DNS.BELLSOUTH.NET.
AAI-TENN NS AUTH01.DNS.BELLSOUTH.NET.
AAI-TENN NS AUTH02.DNS.BELLSOUTH.NET.
ALLIEDMAXCUT NS NS3.DHCNET.NET.
ALLIEDMAXCUT NS NS0.DHCNET.NET.
ATLANTAEXOTICS NS NS1.APHOST
ATLANTAEXOTICS NS NS2.APHOST
ATLANTA-EXOTICS NS NS3.LNHI.NET.
ATLANTA-EXOTICS NS NS2.LNHI.NET.
ATLANTA-EXOTICS NS NS1.LNHI.NET.
.....
```

Domain Management Model (1)

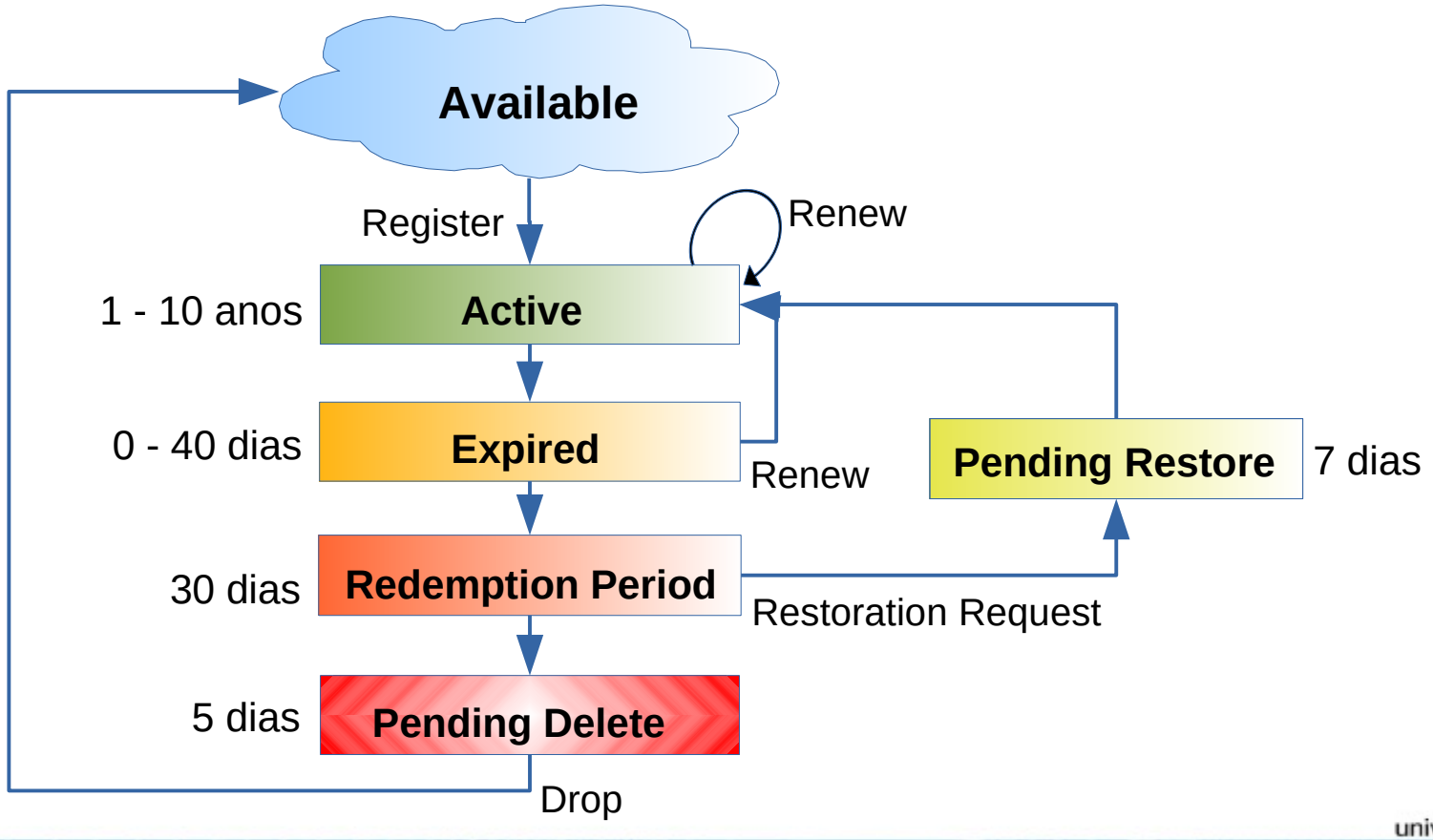


Domain Management Model (2)

- Delegation and Authority lie at the core of the domain name system hierarchy.
- The Authority for the root domain lies with Internet Corporation for Assigned Numbers and Names (ICANN).
  - gTLDs are authoritatively administered by ICANN and delegated to a series of accredited entities.
  - ccTLDs are delegated to the individual countries for administration purposes.
- The entity responsible by a specif domain is called **Registry**.
  - In charge of maintaining the Zone File of the TLD.
- **Registries** (usually) delegate in **Registrar** the operational management and marketing of a domain.
  - One **Registry** can delegate to multiple **Registrars**
  - The **Registrar** stores and manages the information and status of a domain.
- One **Registrar** may still accept **Resellers**
  - A **Reseller** sells domains from a **Registrar** (for a commission)
  - The management of the domains is not responsibility of a **Reseller.**
- A **Registrant** is any entity that want to register a domain name.

Domain Name Life Cycle

- A domain can be registered for a period of 1 to 10 years.
  - After that period the domain must be renewed.
- In case of no renewal, it's initiated the process of deletion of the domain name from the DNS database.
  - Nowadays, the Registrars do not release the domain immediately after the redemption period, they initiate a reselling mechanism (usually some kind of auction) of the domain on the secondary market.



WHOIS Service and Information

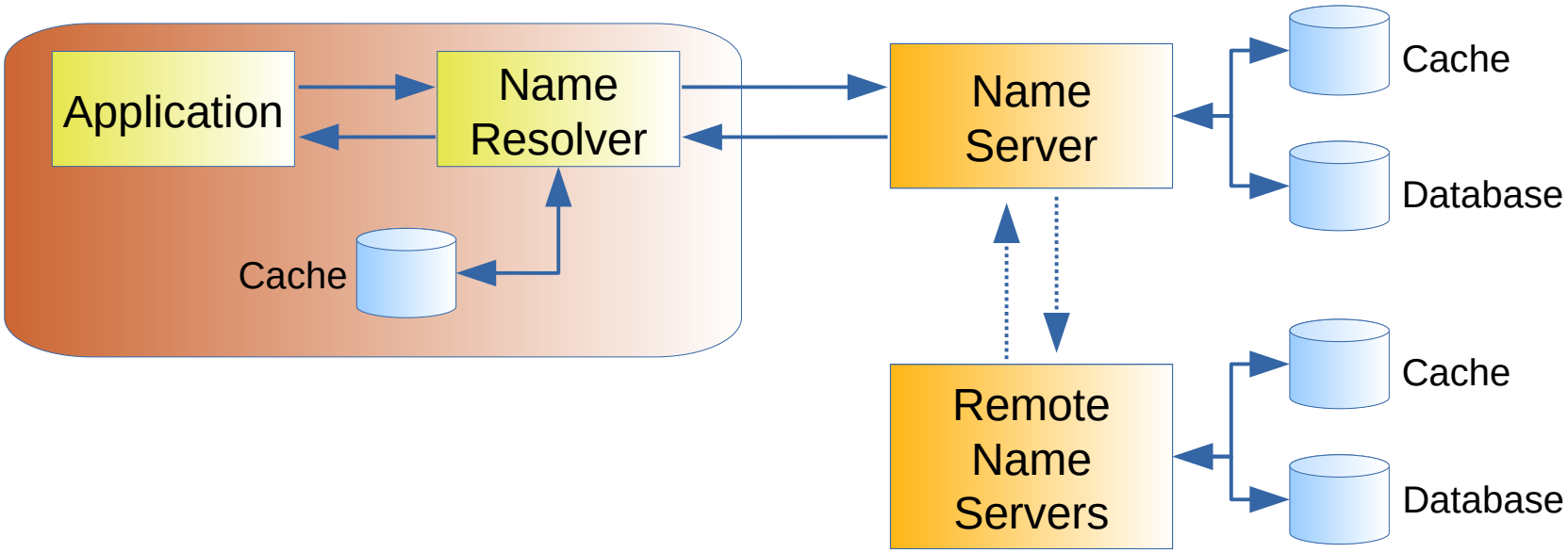
- Contains information about the registrant of a domain
  - Name servers
  - Status of the domain
    - Registry-Registrar Protocol (RPP)
    - Extensible Provisioning Protocol(EPP)
  - Creation, expiration and last update dates.
  - Registrant contacts
    - General
    - Administrative
    - Technical
    - Billing
- This information can be retrieved using the WHOIS service
  - Executes recursive queries of Registry and Registrant databases.

```
Domain Name: NAME.COM
Registrar: NAME.COM LLC
Whois Server: whois.name.com
Referral URL: http://www.name.com
Name Server: NS1.NAME.COM
Name Server: NS2.NAME.COM
Name Server: NS3.NAME.COM
Name Server: NS4.NAME.COM
Status: ok
Updated Date: 30-jan-2009
Creation Date: 03-jan-1995
Expiration Date: 04-nov-2015
.....
REGISTRANT CONTACT INFO
Name.com LLC
DNS Admin, 125 Rampart Way, Suite 300, Denver, CO 80230, US
Phone:       +1.7202492374
Email Address: dns@name.com

ADMINISTRATIVE CONTACT INFO
Name.com LLC
DNS Admin, 125 Rampart Way, Suite 300, Denver, CO 80230, US
Phone:       +1.7202492374
Email Address: dns@name.com

TECHNICAL CONTACT INFO
Name.com LLC
DNS Admin, 125 Rampart Way, Suite 300, Denver, CO 80230, US
Phone:       +1.7202492374
Email Address: dns@name.com

BILLING CONTACT INFO
Name.com LLC
DNS Admin, 125 Rampart Way, Suite 300, Denver, CO 80230, US
Phone:       +1.7202492374
Email Address: dns@name.com
```

Name Servers Registration

- In order to set up a DNS server outside of your registrar, you need to:
  - Explicitly register your name server names and IPs.
    - i.e. Associate name with IP (ex: ns1.domain.com – 10.1.1.1).
  - Define server names (minimum 2) to your domain registration at your registrar.

Name Resolution



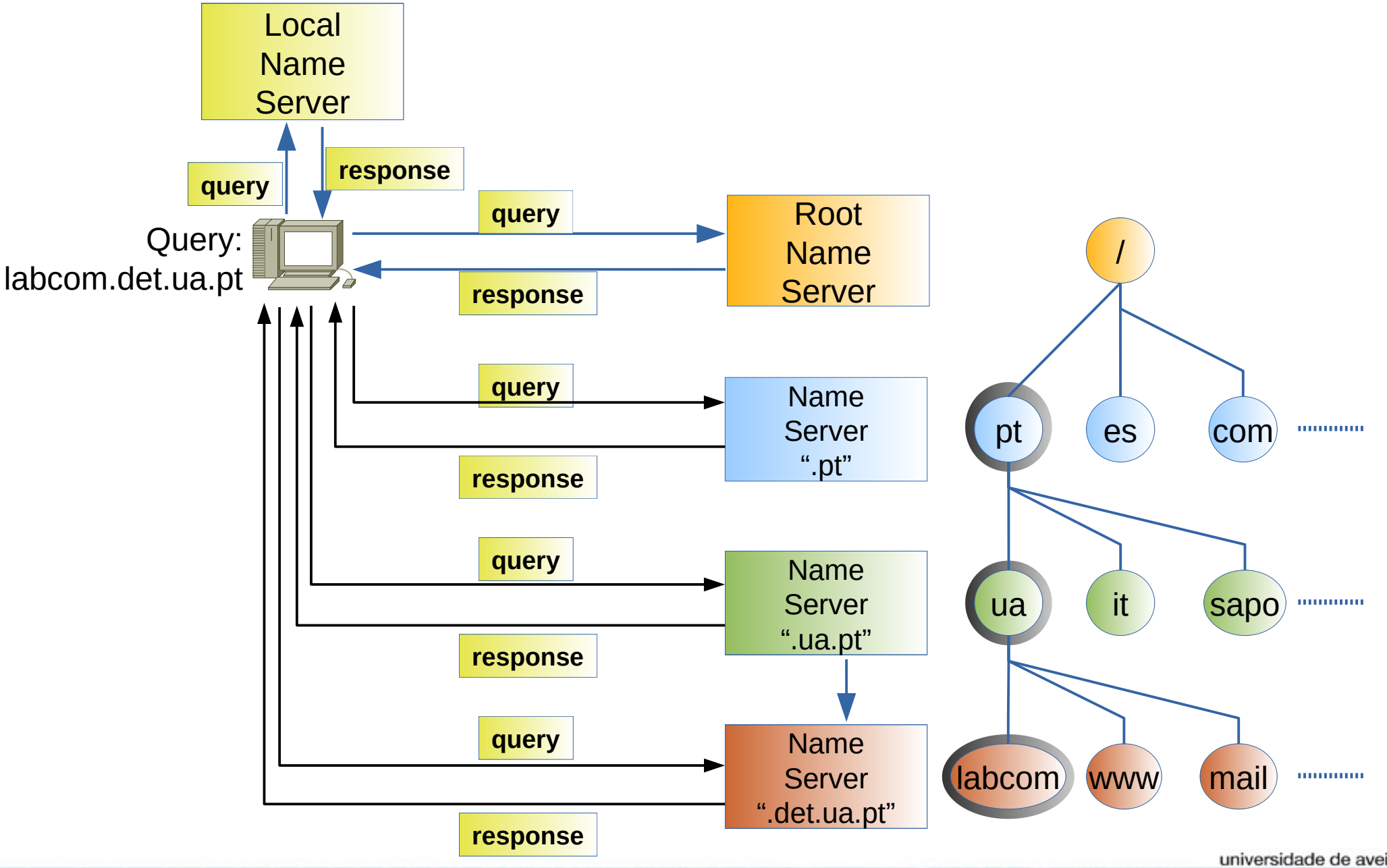
- Received answers are (may be) temporarily stored in cache (have an associated TTL)
  - Can be reused in future queries to speed up answers.
- Cache use improves the systems efficiency by eliminating unnecessary external queries.

DNS Query & DNS Response

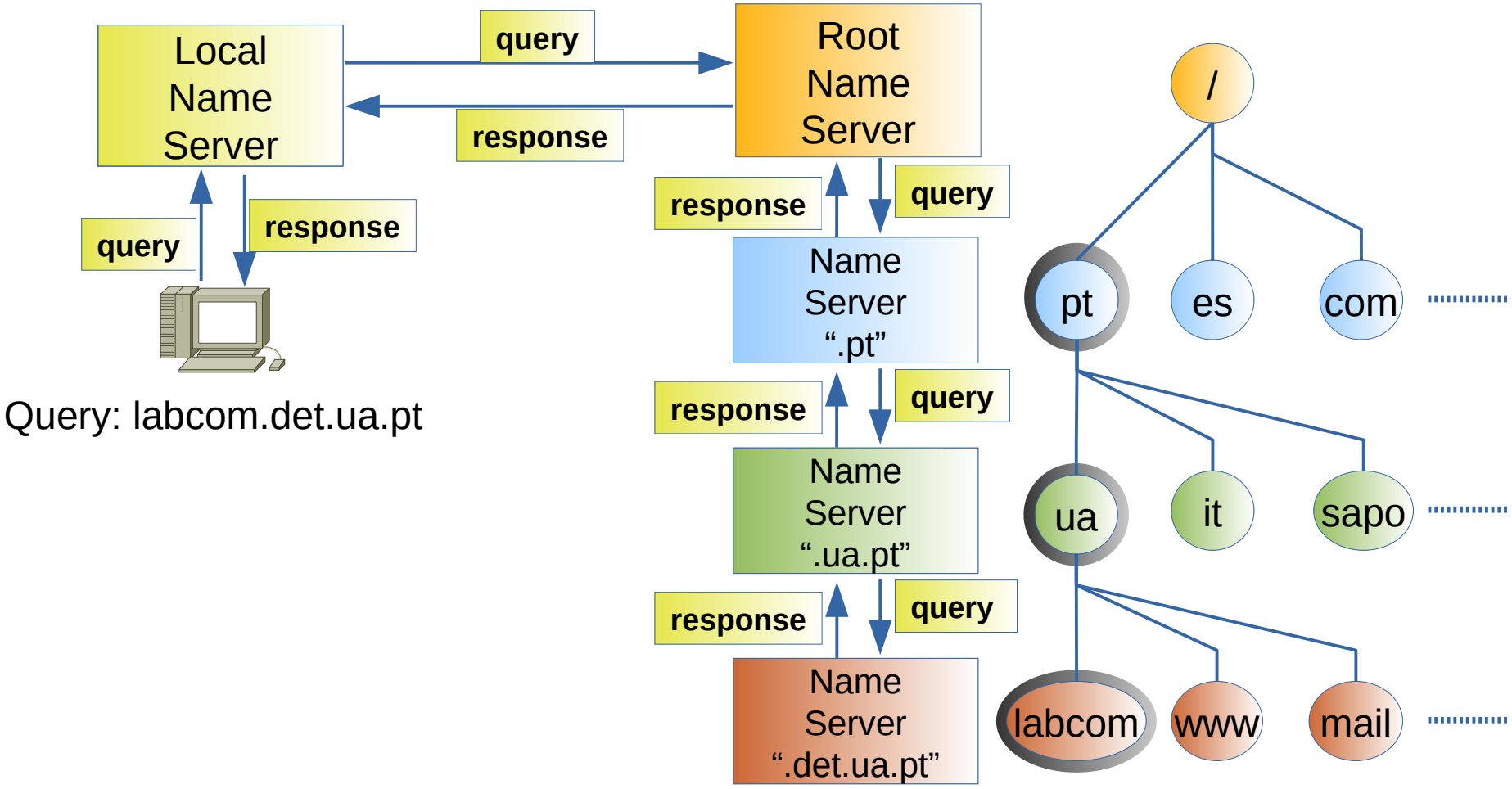
```
Frame 1928 (69 bytes on wire, 69 bytes captured)
Ethernet II, Src: 00:15:f2:9f:38:9d, Dst: 00:60:08:1f:b8:26
Internet Protocol, Src: 193.136.92.160, Dst: 193.136.92.65
User Datagram Protocol, Src Port: 54277, Dst Port: 53
    Source port: 54277 (54277)
    Destination port: 53 (53)
    Length: 35
    Checksum: 0x3c27 [incorrect, should be 0xabba (maybe
    caused by "UDP checksum offload"?)]
Domain Name System (query)
    [Response In: 1929]
    Transaction ID: 0xf1e4
    Flags: 0x0100 (Standard query)
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
    Queries
        www.ua.pt: type A, class I
```

```
Frame 1929 (152 bytes on wire, 152 bytes captured)
Ethernet II, Src: 00:60:08:1f:b8:26, Dst: 00:15:f2:9f:38:9d
Internet Protocol, Src: 193.136.92.65, Dst: 193.136.92.160
User Datagram Protocol, Src Port: 53, Dst Port: 54277
    Source port: 53 (53)
    Destination port: 54277 (54277)
    Length: 118
    Checksum: 0x1167 [correct]
Domain Name System (response)
    [Request In: 1928]
    [Time: 0.005100000 seconds]
    Transaction ID: 0xf1e4
    Flags: 0x8180 (Standard query response, No error)
    Questions: 1
    Answer RRs: 1
    Authority RRs: 2
    Additional RRs: 2
    Queries
        www.ua.pt: type A, class IN
    Answers
        www.ua.pt: type A, class IN, addr 193.136.173.25
    Authoritative nameservers
        ua.pt: type NS, class IN, ns ns2.ua.pt
        ua.pt: type NS, class IN, ns ns.ua.pt
    Additional records
        ns.ua.pt: type A, class IN, addr 193.136.172.18
        ns2.ua.pt: type A, class IN, addr 213.228.152.1
```

Iterative (Non-Recursive) Resolution



Recursive Resolution



Iterative vs. Recursive Resolution

- Iterative resolution:
  - Less efficient: increases the average time between a DNS query and its response.
  - Server loads are lower: each server responds immediately to a query,
    - Do not have to store any temporary information,
    - Do nor perform any interaction with other DNS servers.
- Recursive resolution:
  - More efficient: minimizes the average time between a DNS query and its response.
  - Higher server loads: each server must simultaneously manage the state of multiple DNS queries.
    - More memory, more CPU.
    - Not a problem with current servers.

Zone Configuration

- A zone is defined by
  - A zone declaration, which holds the type of the zone, a pointer to the zone file and type specif configuration statements (optional).
  - A zone file, which holds the DNS resource records for all of the domain names associated with the zone.
- Zone files store all of the data served by a DNS server.
- The basic format of the zone file is a time to live (TTL) field followed by the Start Of Authority (SOA) records.
  - The overall TTL instructs non-authoritative DNS servers how long to cache records retrieved from the zone file.
    - With large values it will take more time to propagate changes.
    - With smaller value, the DNS server load will increase (non-authoritative servers will have to send the same requests more frequently).
    - Typical values: 1 hour to a 1 day.
  - The SOA record defines the zone name, an e-mail contact and various time and refresh values applicable to the zone.

Zone Files

- Zone files contain Resource Records that describe a domain or sub-domain.
  - Format of zone files is an IETF standard defined by RFC 1035.
- Contents
  - Data that indicates the top of the zone and some of its general properties,
    - A SOA Record.
  - Authoritative data for all nodes or hosts within the zone,
    - A (IPv4) or AAAA (IPv6) Records.
  - Data that describes global information for the zone
    - Mail MX Records and Name Server NS Records.
  - In the case of sub-domain delegation the name servers responsible for this sub-domain
    - One or more NS Records.
    - One or more A or AAAA Records

Name Server Records

- SOA (RFC 1035): Start of Authority. Defines the zone name, an e-mail contact and various time and refresh values applicable to the zone.
- A (RFC 1035): IPv4 Address record. An IPv4 address for a host.
- AAAA (RFC 3596): IPv6 Address record. An IPv6 address for a host.
- NS (RFC 1035): Name Server. Defines the authoritative name server(s) for the domain (defined by the SOA record).
- MX (RFC 1035) Mail Exchanger. A preference value and the host name for a mail server/exchanger.
- CNAME (RFC 1035): Canonical Name. An alias name for a host.
- PTR (RFC 1035): IP address (IPv4 or IPv6) to host. Used in reverse maps.
- TXT (RFC 1035): Text information associated with a name.

SOA Record (1)

- @ - represents the base domain
- IN - class of the zome (INternet)
- SOA - record identifier
- The master DNS server for the zone
  - The host where the file was created (nameserver.domain.com)
- Contact e-mail - The e-mail address of the person responsible for administering the domain's zone file.
  - "." is used instead of an "@" in the e-mail name
  - adm.domain.com <=> adm@domain.com email

```
@   IN   SOA      nameserver.domain.com.   adm.domain.com. (
                                  1              ; serial number
                                  3600           ; refresh    [1h]
                                  600            ; retry      [10m]
                                  86400          ; expire     [1d]
                                  3600 )         ; min TTL    [1h]
```

SOA Record (2)

- Serial number - The revision number of this zone file.
  - Increment this number each time the zone file is changed.
  - It is important to increment this value each time a change is made, so that the changes will be distributed to any secondary DNS servers.
- Refresh Time - The time, in seconds, a secondary DNS server waits before querying the primary DNS server's SOA record to check for changes.
  - When the refresh time expires, the secondary DNS server requests a copy of the current SOA record from the primary.
  - The secondary DNS server compares the serial number of the primary DNS server's current SOA record and the serial number in it's own SOA record. If they are different, the secondary DNS server will request a zone transfer from the primary DNS server.
  - The default value is 3,600.
- Retry time - The time, in seconds, a secondary server waits before retrying a failed zone transfer.
  - Usually, the retry time is less than the refresh time. The default value is 600.
- Expire time - The time, in seconds, that a secondary server will keep trying to complete a zone transfer.
  - If this time expires prior to a successful zone transfer, the secondary server will expire its zone file (stops answering queries).
  - The default value is 86,400.
- Negative caching TTL – the time, in seconds, a negative answers (such as when a requested record does not exist) can be cached on non-authoritative servers.
  - This field acts like the overall TTL but specifically for negative answers.
  - Small values are appropriate (15m to 2h).

```
@   IN   SOA     nameserver.domain.com.   adm.domain.com. (
                                  1             ; serial number
                                  3600          ; refresh    [1h]
                                  600           ; retry      [10m]
                                  86400         ; expire     [1d]
                                  3600 )        ; min TTL    [1h]
```

Other Records (1)

- IPv4 Address Record (A)
  - Syntax: "*name ttl class rr ipv4*"

```
; zone fragment for example.com
$TTL 2d ; zone default = 2 days or 172800 seconds
joe         IN      A       192.168.0.3  ; joe & www = same ip
www         IN      A       192.168.0.3
www.example.com.    A       192.168.0.3
fred  3600 IN       A       192.168.0.4  ; TTL overrides $TTL default
ftp         IN      A       192.168.0.24 ; round robin with next
            IN      A       192.168.0.7
mail        IN      A       192.168.0.15  ; mail = round robin
mail        IN      A       192.168.0.32
mail        IN      A       192.168.0.3
```

- IPv6 Address Record (AAAA)
  - Syntax: "*name ttl class rr ipv6*"

```
; zone fragment for example.com
$TTL 2d ; zone default = 2 days or 172800 seconds
$ORIGIN example.com.
joe         IN      AAAA      2001:db8::3  ; joe & www = same ip
www         IN      AAAA      2001:db8::3
; functionally the same as the record above
www.example.com.    AAAA      2001:db8::3
fred  3600 IN       AAAA      2001:db8::4  ; TTL overrides $TTL default
ftp         IN      AAAA      2001:db8::5 ; round robin with next
            IN      AAAA      2001:db8::6
squat       IN      AAAA      2001:db8:0:0:1::13  ; address in another subnet
```

Other Records (2)

- Name Server Record (NS)
  - Syntax: "*name ttl class rr name*"

```
                IN      NS      ns1   ; unqualified name
; the line above is functionally the same as the line below
; example.com. IN       NS      ns1.example.com.
; at least two name servers must be defined
                IN      NS      ns2
; the in-zone name server(s) have an A record
ns1             IN      A       192.168.0.3
ns2             IN      A       192.168.0.3
```

- Mail Exchange Record (MX)
  - Syntax*: "name ttl class rr pref name*"
  - The pref (Preference) field is relative to any other MX record for the zone (value 0 to 65535). Low values are more preferred.

```
                IN      MX      10   mail   ; short form
; the line above is functionally the same as the line below
; example.com. IN       MX      10   mail.example.com.
; any number of mail servers may be defined
                IN      MX      20   mail2.example.com.
; use an external back-up
                IN      MX      30   mail.example.net.
; the local mail server(s) need an A record
mail            IN      A       192.168.0.3
mail2           IN      A       192.168.0.3
```

Other Records (3)

- Canonical Name Record (CNAME)
  - Syntax: "*name ttl class rr canonical_name*"

```
; zone fragment for example.com
$TTL 2d ; zone default = 2 days or 172800 seconds
$ORIGIN example.com.
....
server1     IN      A       192.168.0.3
www         IN      CNAME   server1
ftp         IN      CNAME   server1
```

  - Do not use CNAME records with NS and MX records,
    - Usually it works, but is theoretically not permitted!

Wrong!

```
            IN      MX  10  mail.example.com.
mail        IN      CNAME   server1
server1     IN      A       192.168.0.3
```

Correct!

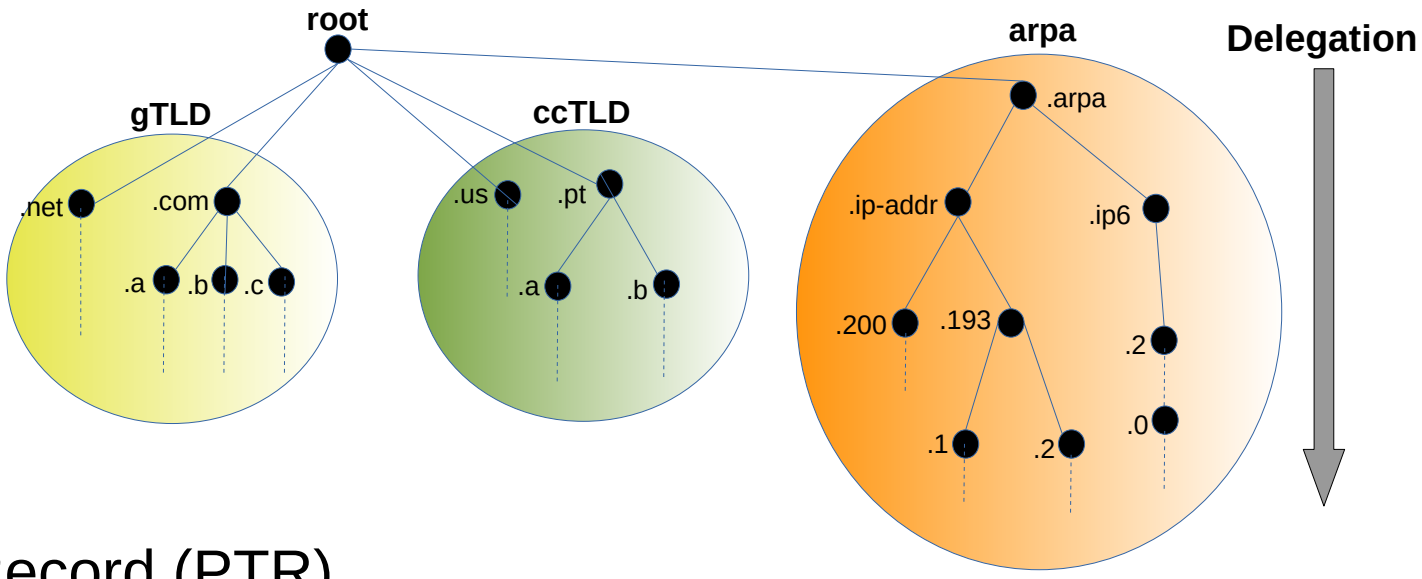
```
            IN      MX  10  mail.example.com.
server1     IN      CNAME   mail
mail        IN      A       192.168.0.3
```

Example

```
$ORIGIN teste.com.
@       IN      SOA     teste.com. adm.teste.com. (
                        199609206       ; serial, todays date + todays serial #
                        8H              ; refresh, seconds
                        2H              ; retry, seconds
                        4W              ; expire, seconds
                        1D )            ; minimum, seconds
                NS      ns1.teste.com.
                NS      ns2.teste.com.
                MX      10 teste.com.  ; Primary Mail Exchanger
                TXT     "TESTE Corp"

localhost       A       127.0.0.1
router          A       206.6.177.1
teste.com.      A       206.6.177.2
ns1             A       206.6.177.3
ns2             A       206.6.177.4
www             A       207.159.141.192

ftp             CNAME   teste.com.
mail            CNAME   teste.com.
news            CNAME   teste.com.

funn            A       206.6.177.2

;       Workstations
ws-177200       A       206.6.177.200
ws-177201       A       206.6.177.201
```

Reverse DNS

- In order to perform Reverse Resolution using normal recursive and Iterative queries the DNS designers defined a special (reserved) Domain Name called:
  - IN-ADDR.ARPA for IPv4 addresses,
    - Resolves <reversed_(partial)_IPv4_Address>.in-addr.arpa
  - IP6.ARPA for IPv6 addresses.
    - Resolves <reversed_(partial)_IPv6_Address>.ip6.arpa



- Uses the Pointer Record (PTR)
  - Pointer records are the opposite of A and AAAA.
    - Syntax: “name *ttl class rr name*”

IPv4 Reverse DNS - Example

```
zone "200.136.193.in-addr.arpa" {
        type master;
        file "zones/193.136.200";
};
```

```
$TTL 3D
@               IN      SOA     land-5.com. root.land-5.com. (
                                199609206       ; Serial
                                28800   ; Refresh
                                7200    ; Retry
                                604800  ; Expire
                                86400)  ; Minimum TTL
                        NS      land-5.com.
                        NS      ns2.psi.net.
;       Servers
1       PTR     router.land-5.com.
2       PTR     land-5.com.
2       PTR     funn.land-5.com.

;       Workstations

200     PTR     ws-177200.land-5.com.
201     PTR     ws-177201.land-5.com.
202     PTR     ws-177202.land-5.com.
203     PTR     ws-177203.land-5.com.
```

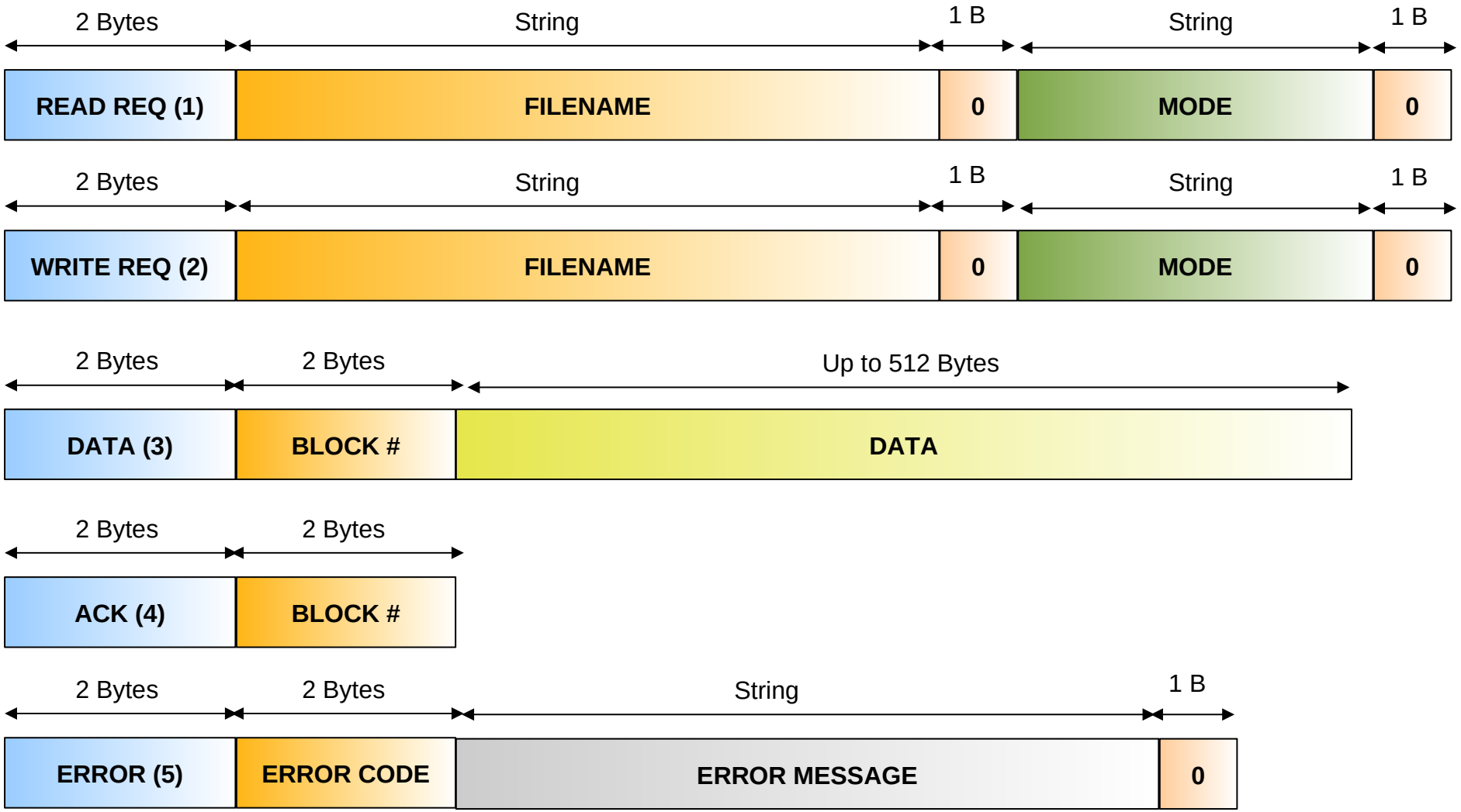
IPv6 Reverse DNS – Example

```
$TTL 2d    ; default TTL for zone 172800 secs
$ORIGIN 0.0.0.0.8.b.d.0.1.0.0.2.IP6.ARPA.
@          IN      SOA   ns1.example.com. hostmaster.example.com. (
                         2003080800 ; sn = serial number
                         12h         ; refresh = refresh
                         15m        ; retry = update retry
                         3w         ; expiry = expiry
                         2h         ; min = minimum
                         )
; name servers Resource Recordsfor the domain
           IN      NS      ns1.example.com.
; the second name servers is
; external to this zone (domain).
           IN      NS      ns2.example.net.
; PTR RR maps a IPv6 address to a host name
; hosts in subnet ID 1
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.0.0.0          IN      PTR     ns1.example.com.
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.0.0.0          IN      PTR     mail.example.com.
; hosts in subnet ID 2
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.0.0          IN      PTR     joe.example.com.
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.0.0          IN      PTR     www.example.com.
```

Trivial File Transfer Protocol (TFTP)

- Simple file transference service (IETF RFC 1350)
  - Does not allow directory/folder listing.
  - Does not incorporate any user authentication mechanism.
- Uses UDP.
  - The first packet from the client is sent to the server's port (default is port UDP 69).
  - The server chooses another (ephemeral) port number, and responds from that port.
  - Next client packets are sent to the second (chosen) server port.
- Implements "Stop and Wait" as flow control mechanism.
- Based on five primitives:
  - Read Request (RRQ)
  - Write Request (WRQ)
  - Data
  - Acknowledgement (ACK)
  - Error (ERR)

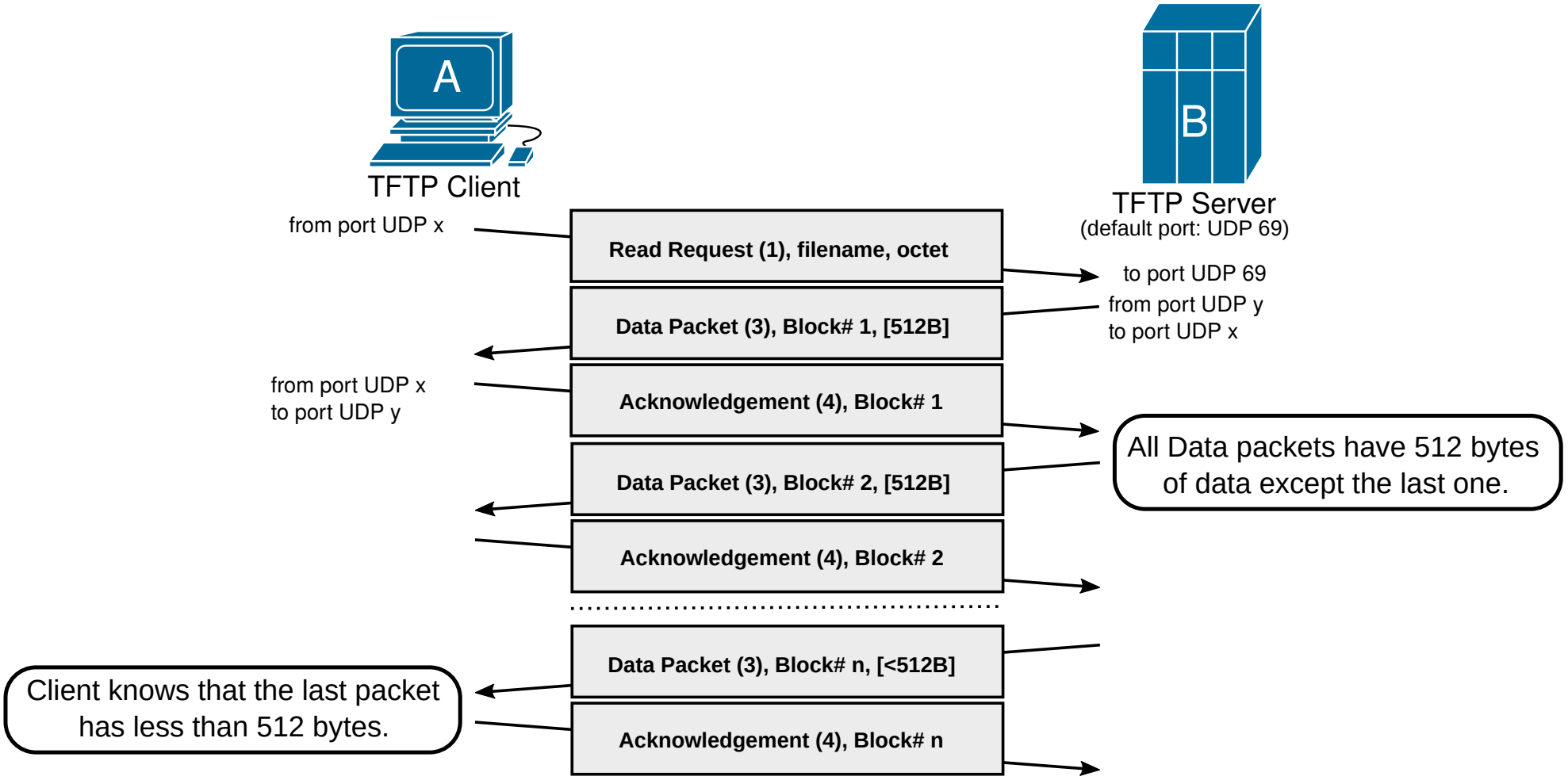
TFTP Messages Format (1)



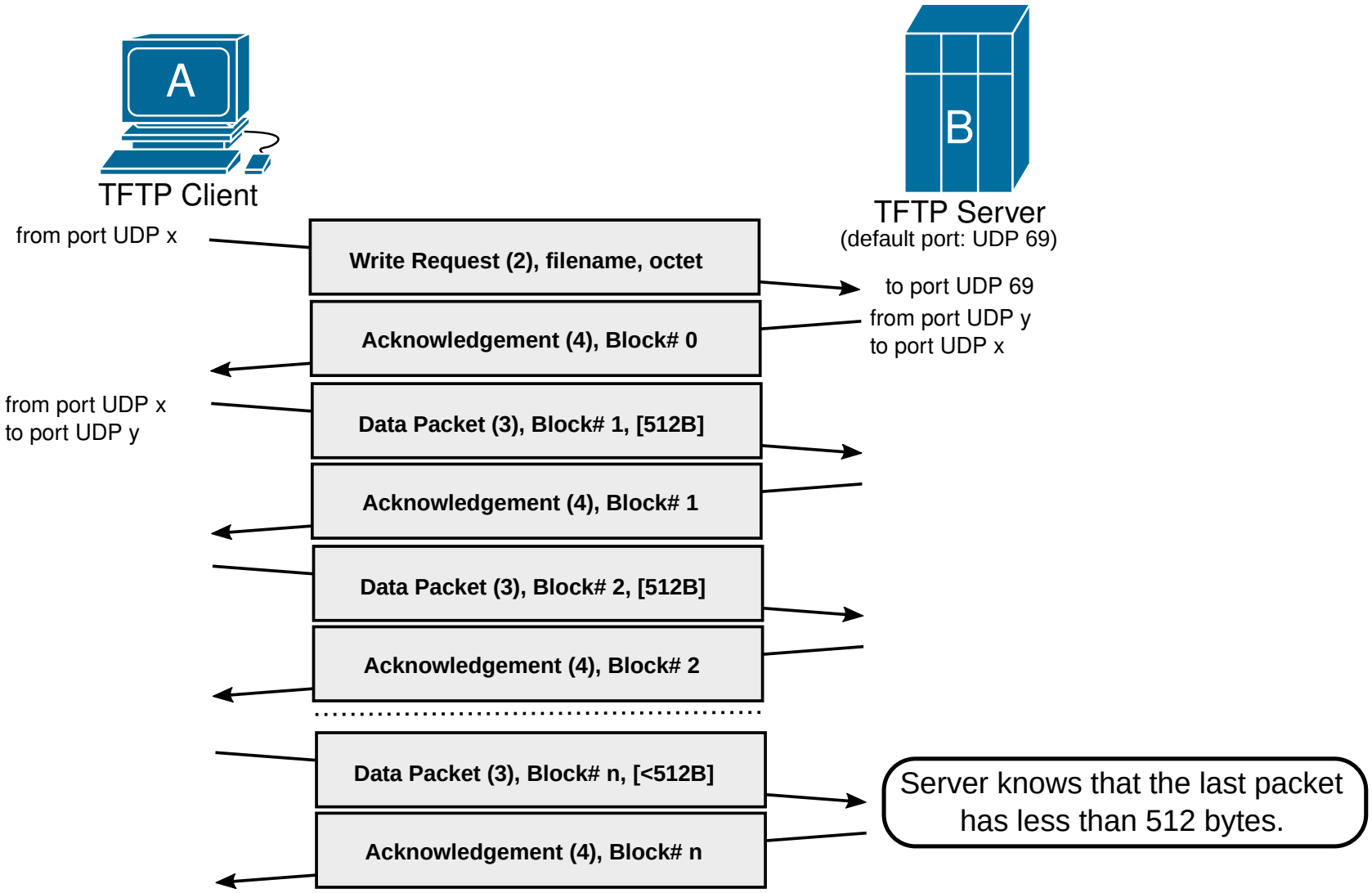
TFTP Messages Format (2)

- Read and Write Request messages: Used to request a file download or upload, respectively.
  - FILENAME – ASCII character string that species the name of the file to write or read.
  - MODE – ASCII character string that species how the file will be exchanged (netascii or octet).
- Data message:
  - BLOCK # - Identifies the order/index of the data block.
- Acknowledge message:
  - BLOCK # - It is used to define which data block is being acknowledged.
- ERROR message: Acts as a NACK (not acknowledge). It may trigger a retransmission or end of connection.
  - ERROR MESSAGE – ASCII character string that species an error.
  - ERROR CODE: 00 – Not defined; 01 – File not found; 02 – Access violation; 03 – Disk full; 04 – Invalid operation code; 05 – Unknown port number; 06 – File already exists; 07 – No such user.

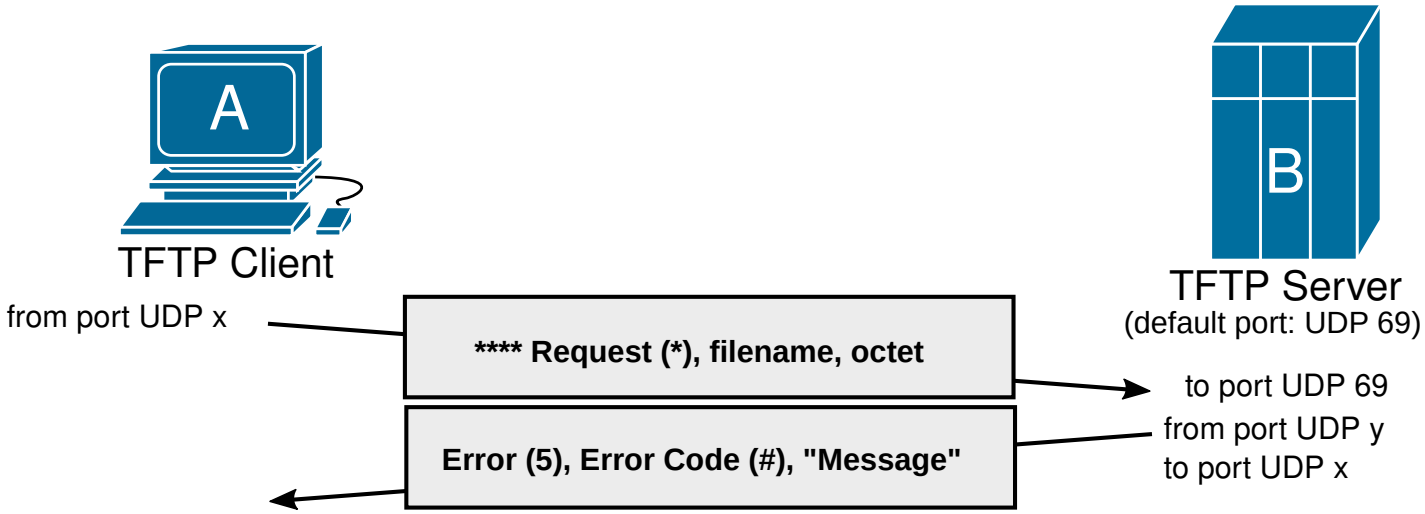
TFTP Read Request



TFTP Write Request



TFTP Errors

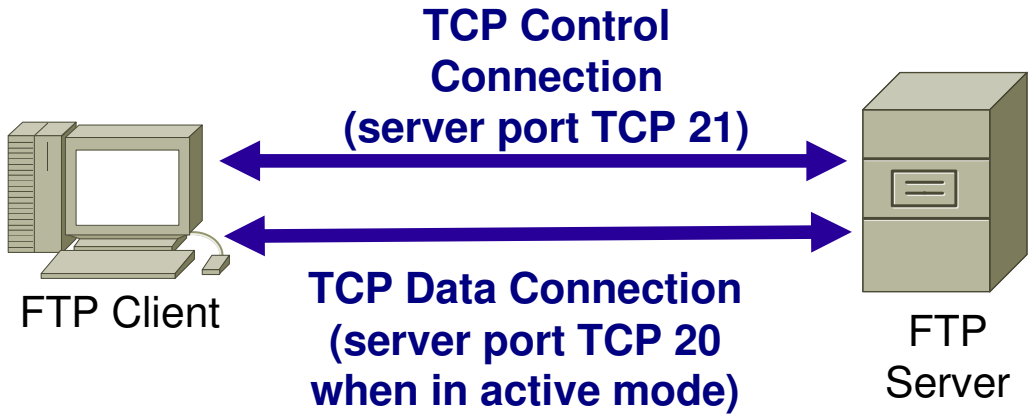


Error Codes

- **1** File not found.
- **2** Access violation.
- **3** Disk full or allocation exceeded.
- **4** Illegal TFTP operation.
- **5** Unknown transfer ID.
- **6** File already exists.

File Transfer Protocol (FTP)

- File Transference Service.
- Uses TCP.
- Server uses two TCP ports:
  - Control: TCP port 21.
  - Data: TCP port 20.
- Supports user authentication
  - Username + Password or Username Anonymous.
  - Credentials are transmitted in open text.



- The client establishes a TCP control connection with the server, by which the commands and responses are exchanged.
  - Commands and Responses are sent as ASCII text.
  - The TCP control connection will be maintained active until the end of the FTP session.
- Every time data must be exchanged, it is established a TCP data connection between the server and the client.
  - After each data set is exchanged, the TCP data connection is closed.
- Supports two data formats: ASCII and Binary (modes).
- Supports two data transfer modes:
  - Active: The server opens the TCP data connection to a client address and (ephemeral) port, announced by the client using a PORT command.
  - Passive: The client opens the TCP data connection to a server address and (ephemeral) port, announced by the server in a “227” response to a PASV command sent by the client.

FTP Control Requests and Responses

Sample Client Request commands:

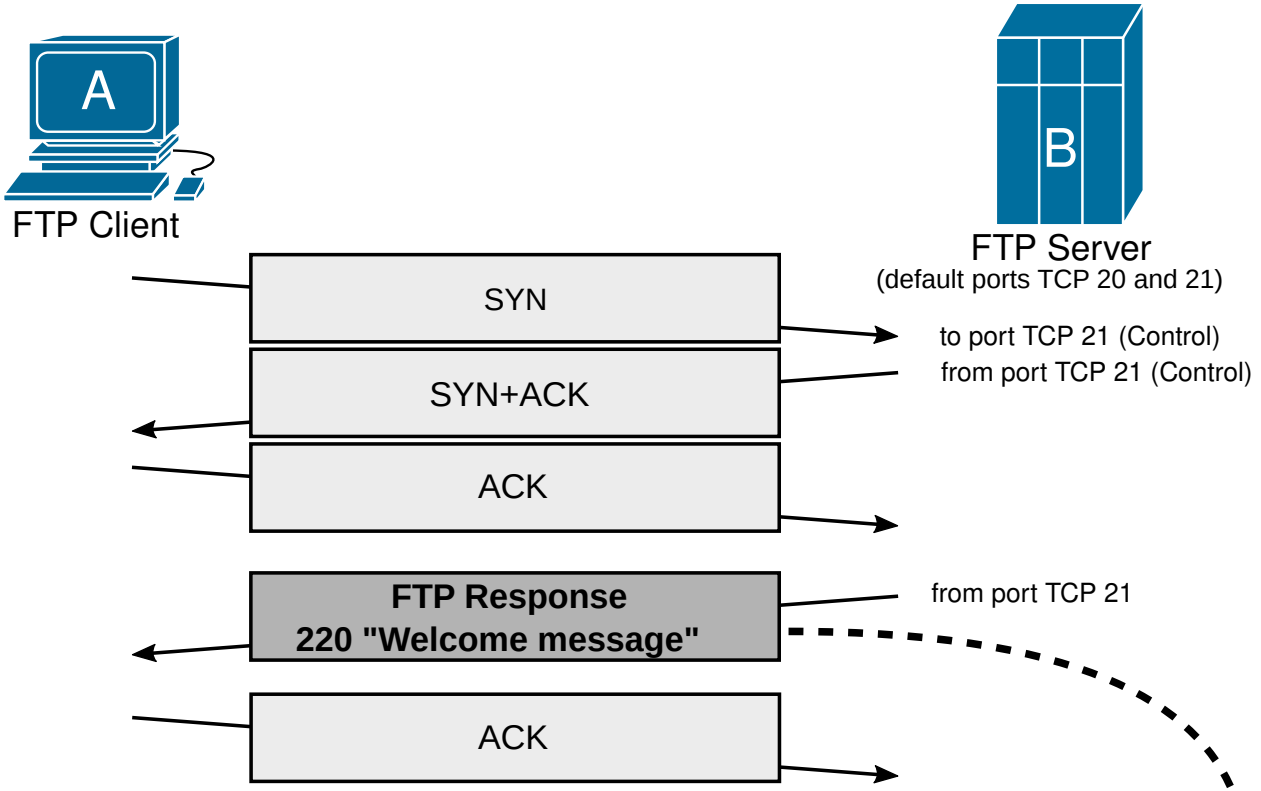
USER Authentication username.

- **PASS** Authentication password.
- **TYPE** Sets the transfer mode (ASCII/Binary).
- **CDUP** Change to Parent Directory.
- **CWD** Change working directory.
- **PWD** Print working directory. Returns the current directory of the host.
- **LIST** Returns information of a file or directory if specified, else information of the current working directory is returned.
- **PASV** Enter passive mode.
- **PORT** Specifies an address and port to which the server should connect.
- **RETR** Retrieve a copy of the file
- **STOR** Accept the data and to store the data as a file at the server site
- **DELE** Delete file.
- **QUIT** Disconnect.

Sample Server Response codes

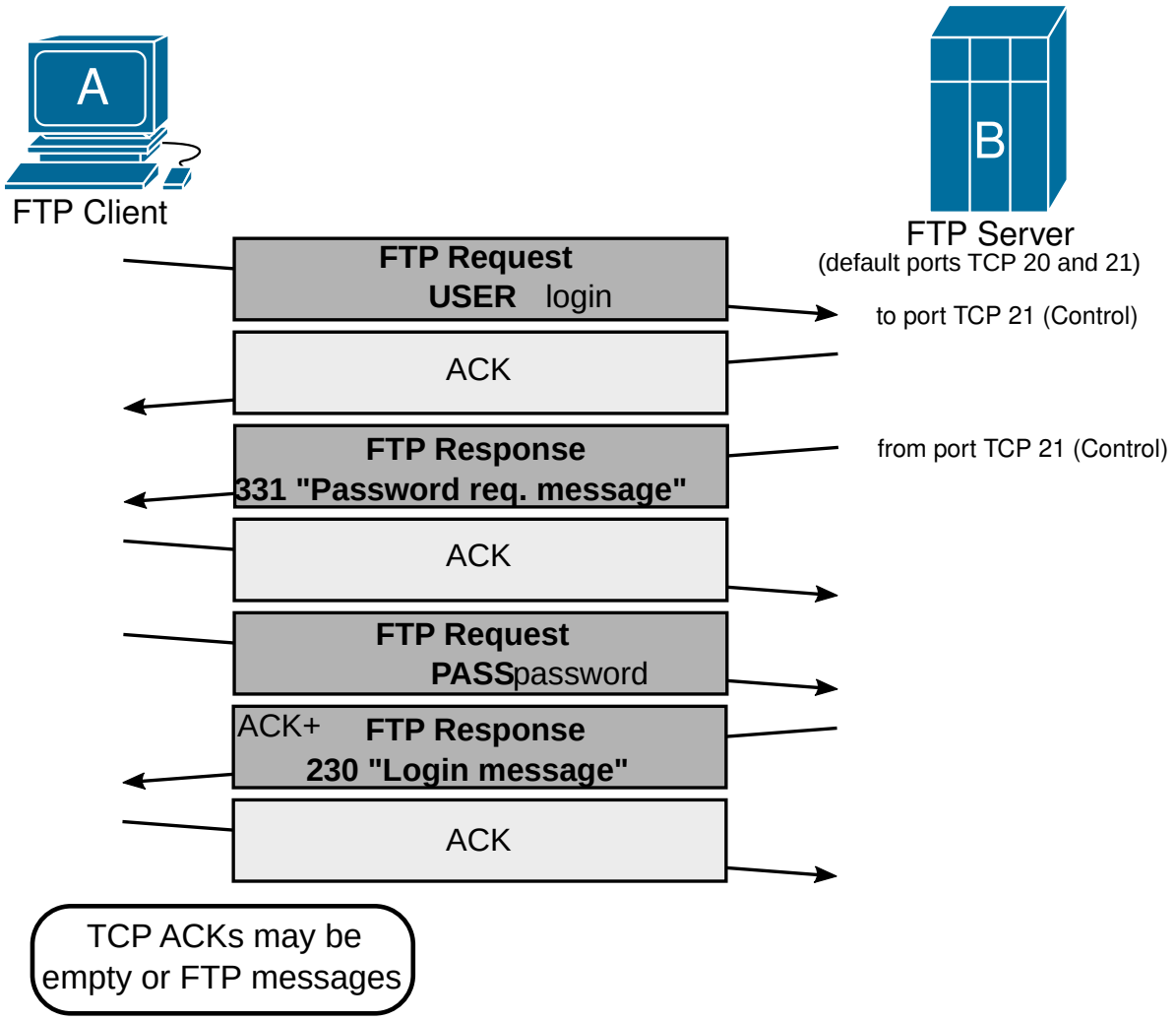
- **200** The requested action has been successfully completed.
- **220** Service ready for new user.
- **221** Service closing control connection.
- **226** Closing data connection. Requested file action successful.
- **227** Entering Passive Mode
- **230** User logged in, proceed.
- **331** Username OK, password required.
- **125** data connection already open transfer starting.
- **150** File status okay; about to open data connection.
- **425** Can’t open data connection.
- **452** Error writing file.

Initial FTP Connection




```
▸ Transmission Control Protocol, Src Port: 21, Dst Port: 59062
▾ File Transfer Protocol (FTP)
  ▾ 220 (vsFTPd 3.0.2)\r\n
      Response code: Service ready for new user (220)
      Response arg: (vsFTPd 3.0.2)
```

FTP Authentication



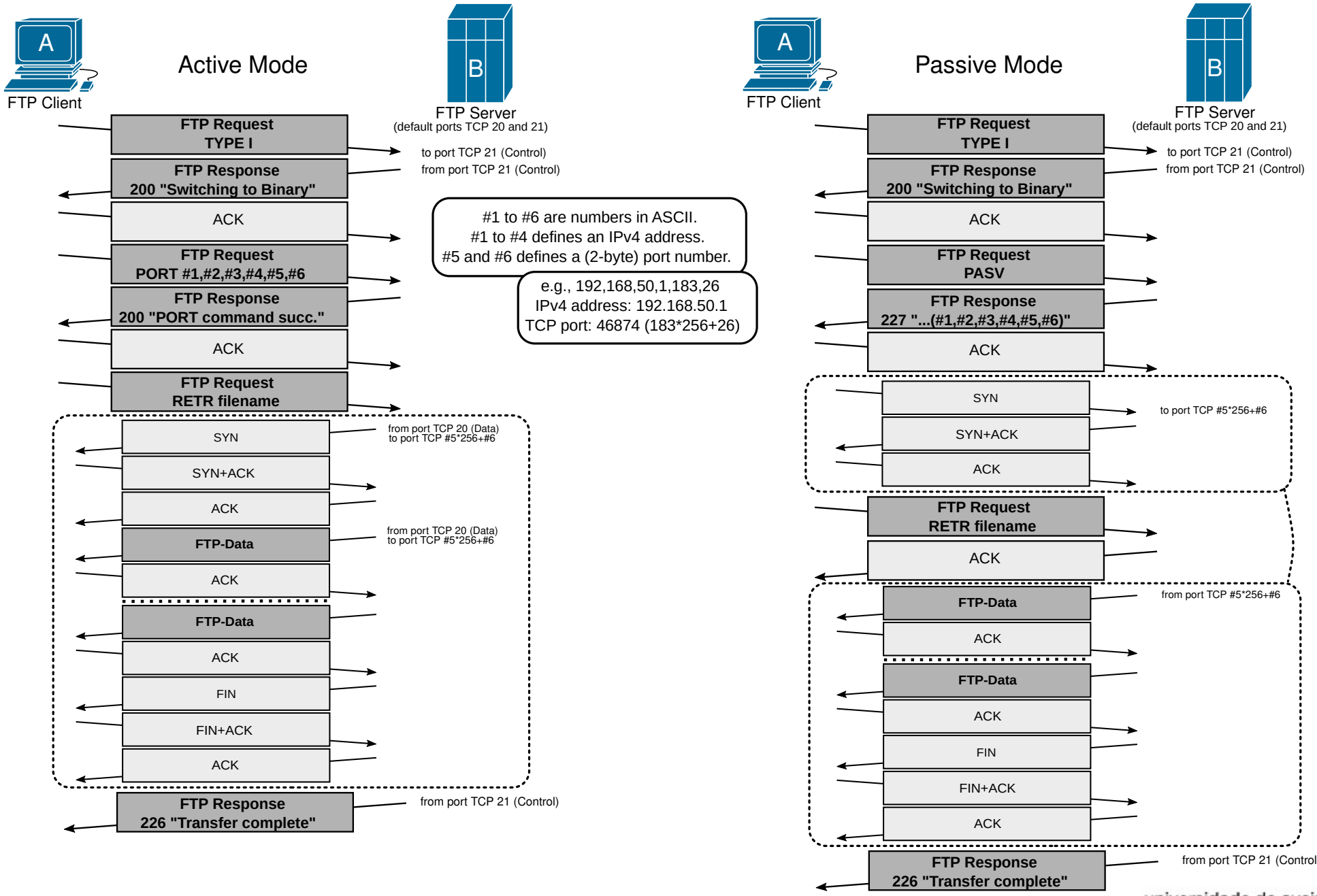

```
▸ Transmission Control Protocol, Src Port: 59062, Dst Port: 21
▾ File Transfer Protocol (FTP)
  ▾ USER labcom\r\n
      Request command: USER
      Request arg: labcom
```

```
▸ Transmission Control Protocol, Src Port: 21, Dst Port: 59062
▾ File Transfer Protocol (FTP)
  ▾ 331 Please specify the password.\r\n
      Response code: User name okay, need password (331)
      Response arg: Please specify the password.
```

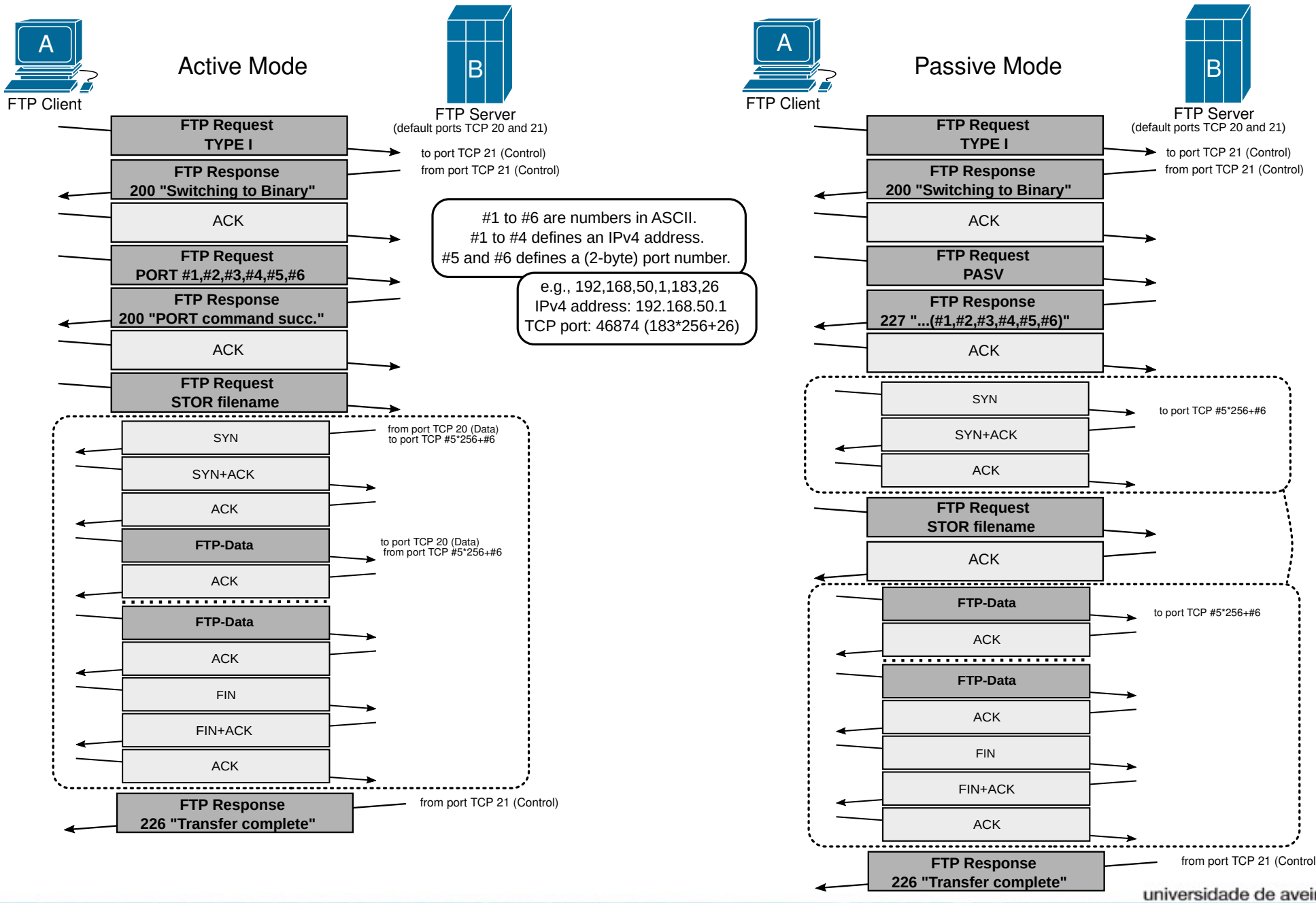
```
▸ Transmission Control Protocol, Src Port: 59062, Dst Port: 21
▾ File Transfer Protocol (FTP)
  ▾ PASS labcom\r\n
      Request command: PASS
      Request arg: labcom
```

```
▸ Transmission Control Protocol, Src Port: 21, Dst Port: 59062
▾ File Transfer Protocol (FTP)
  ▾ 230 Login successful.\r\n
      Response code: User logged in, proceed (230)
      Response arg: Login successful.
```

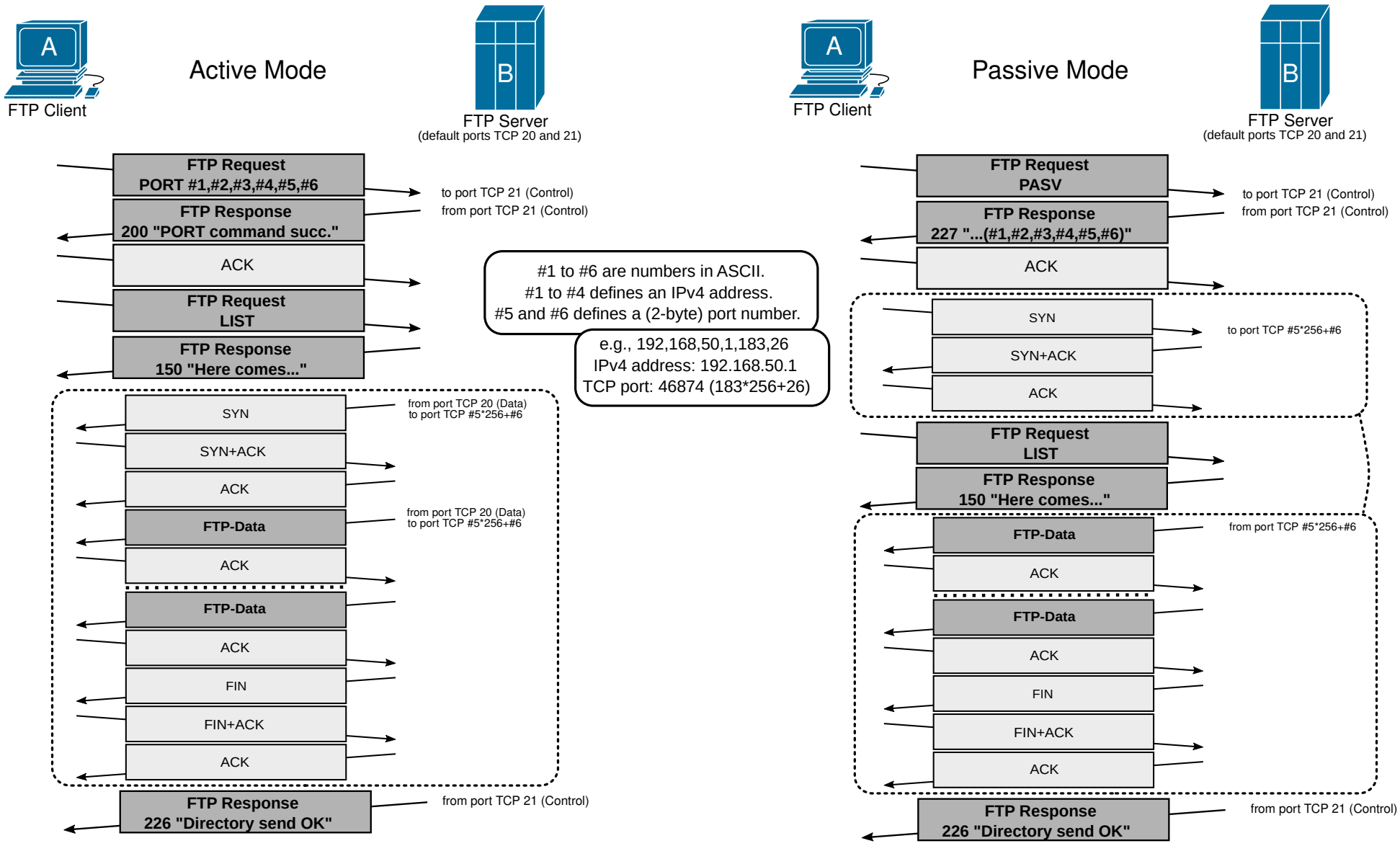
FTP Operations (Download)



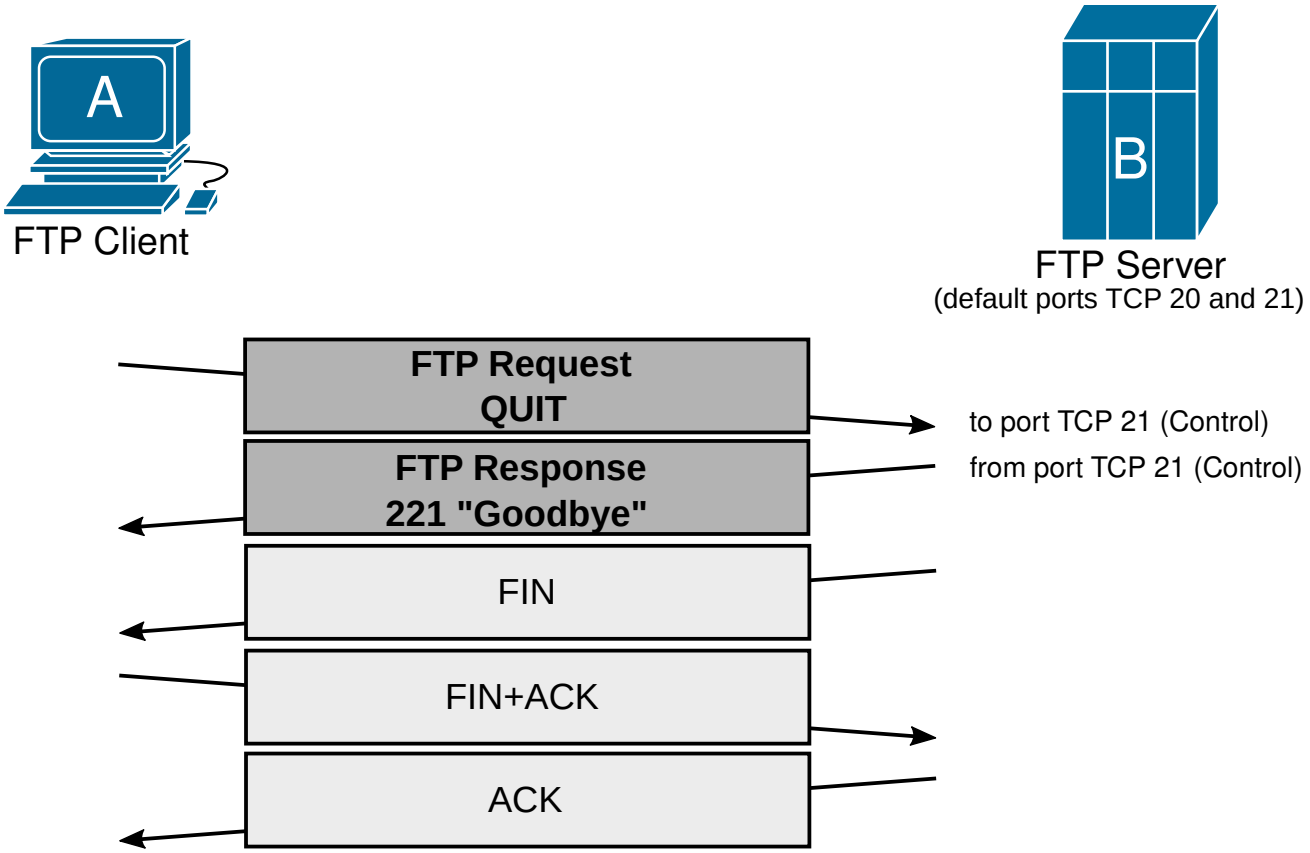
FTP Operations (Upload)



FTP Operations (Directory Listing)

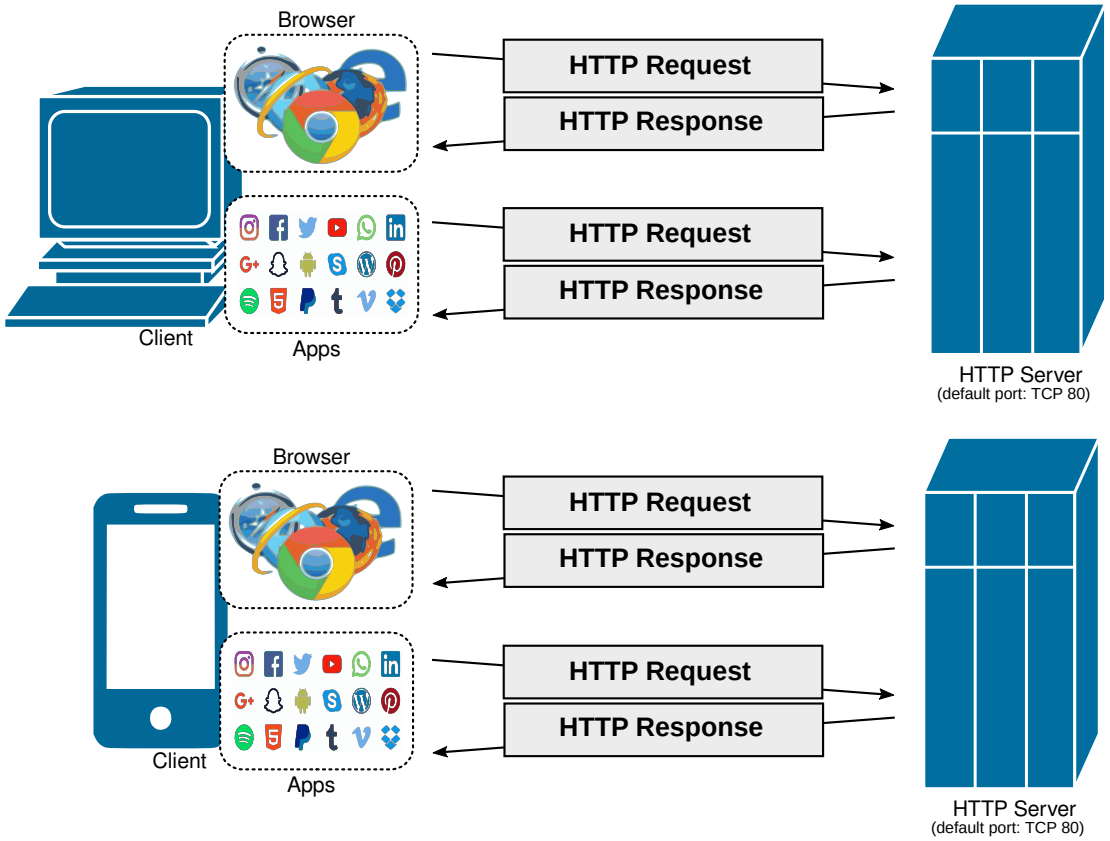


End of FTP Connection



Hyper-Text Transport Protocol (HTTP)

- Web's application layer protocol.
- Follows a client/server model.
  - Client:
    - Browser that requests, receives, and "displays" Web objects.
    - Application that sends and receives data.
  - Server: Web server sends data objects in response to requests. Also, receives data from clients.
- Client side sends HTTP Requests to server.
- Server responds with HTTP Responses.

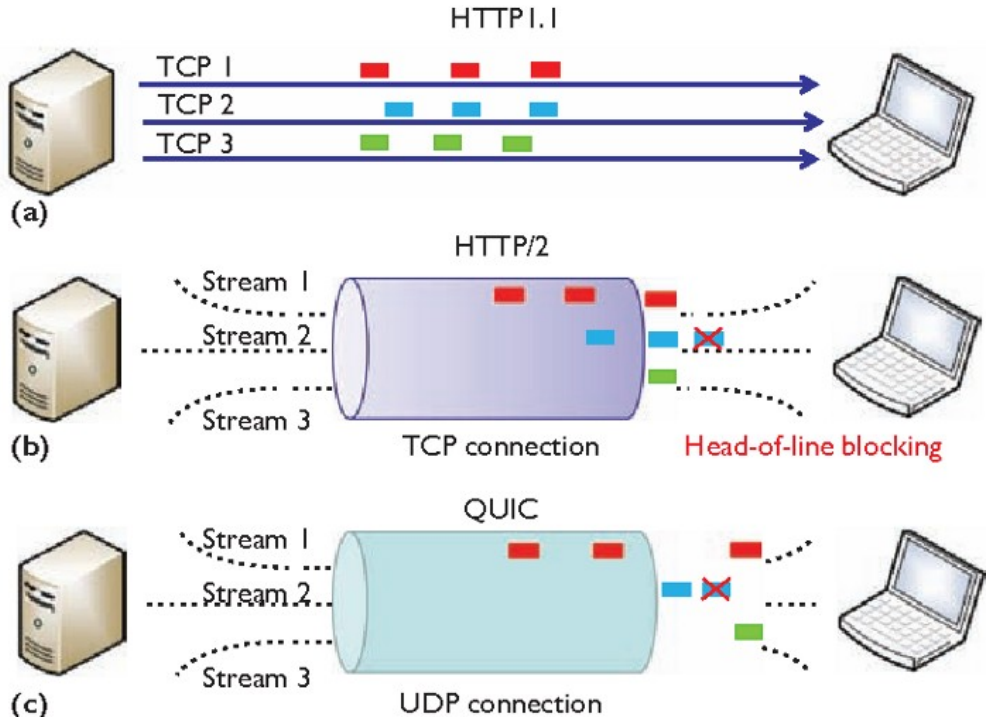


HTTP Connections

- HTTP uses TCP.
  - With non-persistent connections.
    - The client establishes a TCP connection to send the Request and the server terminates the TCP connection after sending the Response.
    - Performance is penalized by the establishment time of each TCP connection and its slow start behavior.
    - Possibility of using parallel TCP connections (configurable number in browsers) to decrease server response time.
  - With persistent connections.
    - The client establishes a TCP connection to send the Request, the Request includes a directive indicating to the server not terminate the TCP connection after sending the Response.
    - The client can use established connections with pipelining or without pipelining.
      - Pipelining=Sending multiple requests without waiting for the responses of previous requests.
    - The server waits for a timeout (typically configurable) time to terminate connections that are not used and were not closed.

HTTP Versions

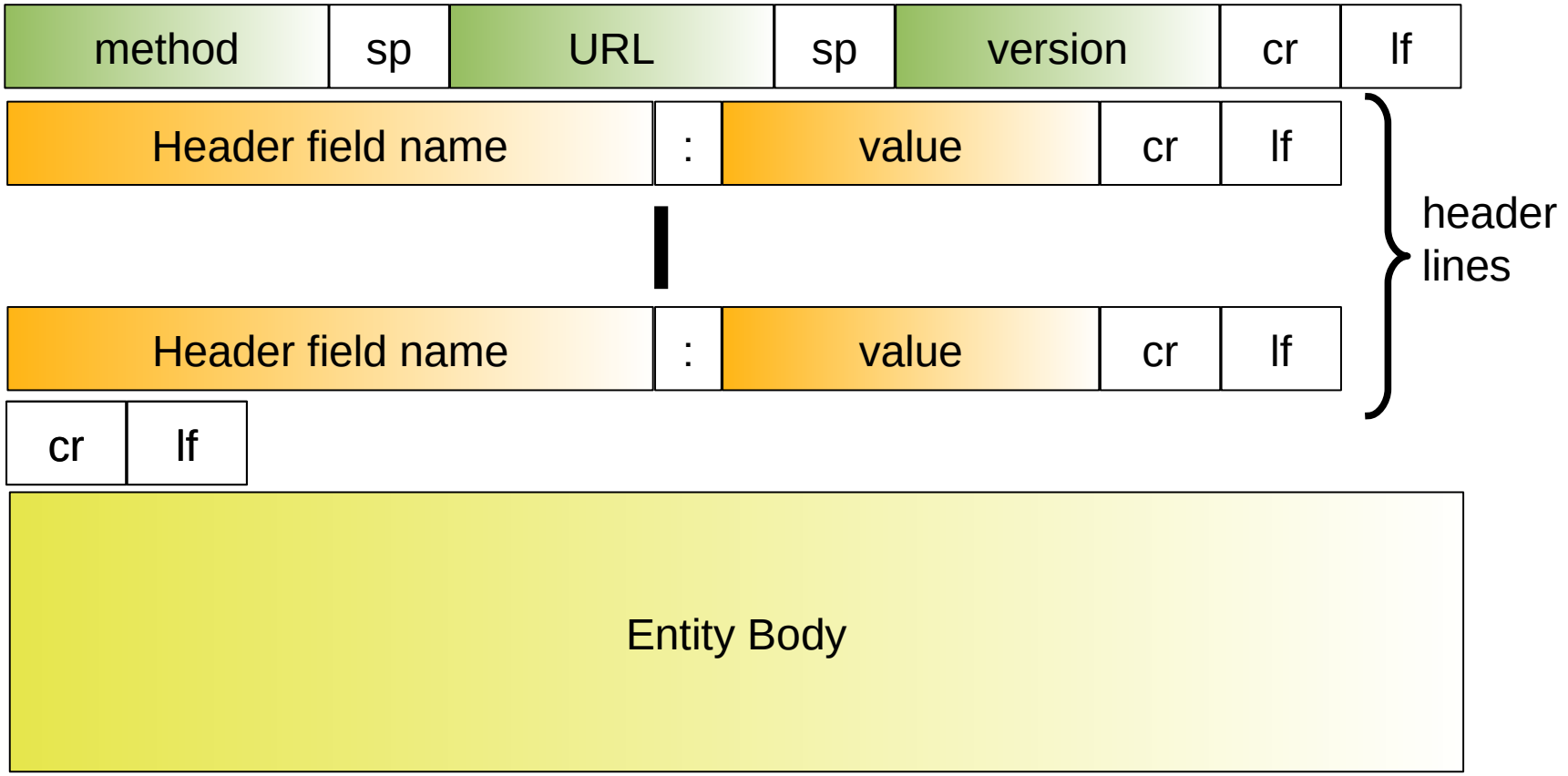
- HTTP 1.0 was defined in RFC 1945
  - Supports only non-persistent connections.
  - Motivation to use it nowadays:
    - Low complexity web/data objects.
    - Limited server capabilities.
- HTTP 1.1 was defined in RFC 2616
  - By default, works with persistent TCP connections and pipelining.
  - Compatible with version 1.0.
- HTTP 2.0 was defined in RFC 7540 (2015)
  - Derived from Google's SPDY protocol.
  - Data compression of HTTP headers.
  - Requests prioritization.
  - Server Push
    - Server sends data before client request.
  - Highly compatible with version 1.1.
- HTTP 3.0 (in draft)
  - Based on Google's QUIC.
  - Multiple data streams over an
  - UDP connection.
  - Faster establishment.
  - Does not block data streams after one packet loss.
  - Usable in (very) low packet losses networks.



Uniform Resource Locator (URL)

- http://www.someschool.edu:1024/somedir/page.html
  - http://
    - Protocol used to communicate with server.
    - Protocols supported by browsers: http, https, file, ftp, mailto.
  - www.someschool.edu
    - DNS name associated with the server's IP address.
    - Can be an IP address (193.136.173.5 ou www.ua.pt)
  - :1024/
    - Indicates the server's port (optional)
    - If omitted, the default port is used.
  - somedir/
    - Path for the data object (optional)
    - If omitted, it is assumed that the data object is in root folder.
  - page.html
    - Name of the data object.

HTTP Request Format



HTTP Methods

- GET
  - Requests a representation of the specified resource.
  - Requests using GET should only retrieve data.
- HEAD
  - Asks for a response identical to that of a GET request, but without the response body.
- POST
  - Used to submit an entity to the specified resource, often causing a change in state or side effects on the server
- PUT
  - Replaces all current representations of the target resource with the request payload.
- DELETE
  - Deletes the specified resource.
- CONNECT
  - Establishes a tunnel to the server identified by the target resource.
- OPTIONS
  - Used to describe the communication options for the target resource.

HTTP Request (sample)

```
GET /PageText.aspx?id=259 HTTP/1.1\r\n
    Host: www.ua.pt\r\n
    User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-GB; rv:1.9.0.10)
Gecko/2009042523 Ubuntu/9.04 (jaunty) Firefox/3.0.10\r\n
    Accept: text/html,application/xhtml+xml,application/xml;\r\n
    Accept-Language: en-gb,en;q=0.5\r\n
    Accept-Encoding: gzip,deflate\r\n
    Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n
    Keep-Alive: 300\r\n
    Connection: keep-alive\r\n
    Referer: http://www.ua.pt/\r\n
```

- Messages composed in ASCII format.
- Start with a line defining the *method* (GET, POST, HEAD, …).
- Includes a variable number of header lines:
  - Host: identifies the server.
  - Connection: Defines a persistent (keep-alive) or non-persisten connection.
  - User-agent: identifies and describes the client OS and Browser (i.e., Firefox, Linux Ubuntu 9.04).

HTTP Response (sample)

```
HTTP/1.1 200 OK
Connection:close
Date: Thu, 06 Aug 1998 12:00:15 GMT
Server: Apache/1.3.0
Last-Modified: Mon, 22 Jun 1998 09:23:24 GMT
Content-Length: 6821
Content-Type: text/html
(carriage return, line feed)
(data, data, data, …)
```

- Fist line contains the HTTP version and Response Code.
- Includes a variable number of header lines.
- Ends with the requested content.
  - Before defines the length and type of the content.

HTTP Response Codes

- 200 OK
  - Request succeeded, requested object later in this msg.
- 301 Moved Permanently
  - Requested object moved, new location specified later in this msg (Location:).
- 400 Bad Request
  - Request msg not understood by server.
- 404 Not Found
  - Requested document not found on this server.
- 505 HTTP Version Not Supported

HTTP Native Authentication

- HTTP includes natively an authentication process that allows to limit access to files based on a username and password.
- A request message sent by a browser to a protected file is answered by the server with a response message in which the response line is:
  - 401 Authorization Required
- This response includes a header line of the type **WWW-Authenticate** indicating the authentication method to use.
- The new request includes an **Authorization** header line with the username and password information generated by the method requested by the server.
- Typically, the browser stores the username and password information in memory for use in future request messages.
- Nowadays, the authentication is handle at the application level.

HTTP Cookies

- Cookies are a way for the server to identify a terminal in different requests made over time.
- Allows the server to differentiate the information to be made available to the client.
- The first time a terminal sends a request to a server, the server includes in the response a header line of type:
  - Set-Cookie: uu = ad14cc7cf29d446b0b8e73c5606135efcbe4e58c; expires = Wed, 17-Jun-2009 15:47:29 GMT \ r \ n
- If the browser is configured to accept cookies, it saves this number along with the server identifier.
- In future orders, the browser includes the header line:
  - Cookie: uu = ad14cc7cf29d446b0b8e73c5606135efcbe4e58c \ r \ n
- In this way, the server identifies the client.

Conditional GET

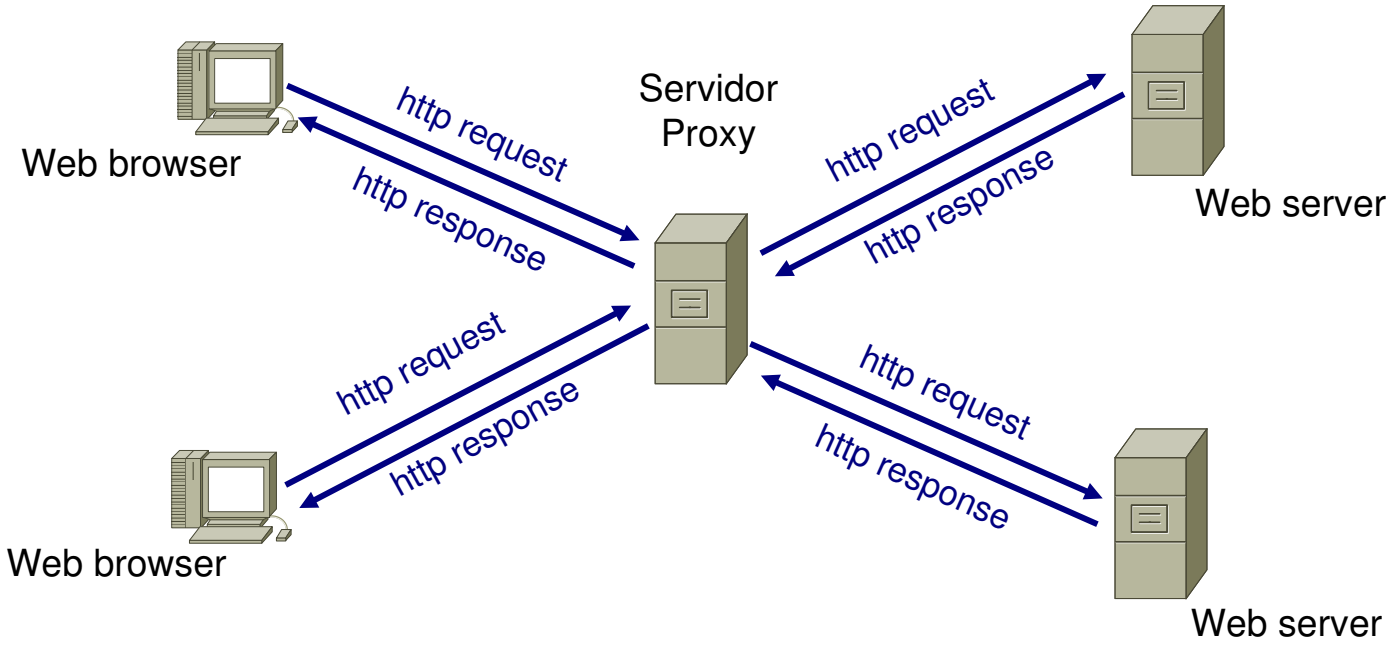
- If the browser supports Web caching, it is possible to:
  - Minimize response times,
  - Minimize network traffic.
- If a file is cached on the client, the browser makes a request with a header line of type **If-modified-since.**
- If not modified, the server just responsd with a **304** response code.

```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
User-agent: Mozilla/4.0
If-modified-since: Mon, 22 Jun 1998 09:23:24 GMT
(carriage return, line feed)
```

```
HTTP/1.1 304 Not Modified
Date: Thu, 19 Aug 1998 12:00:15 GMT
Server: Apache/1.3.0
(carriage return, line feed)
```

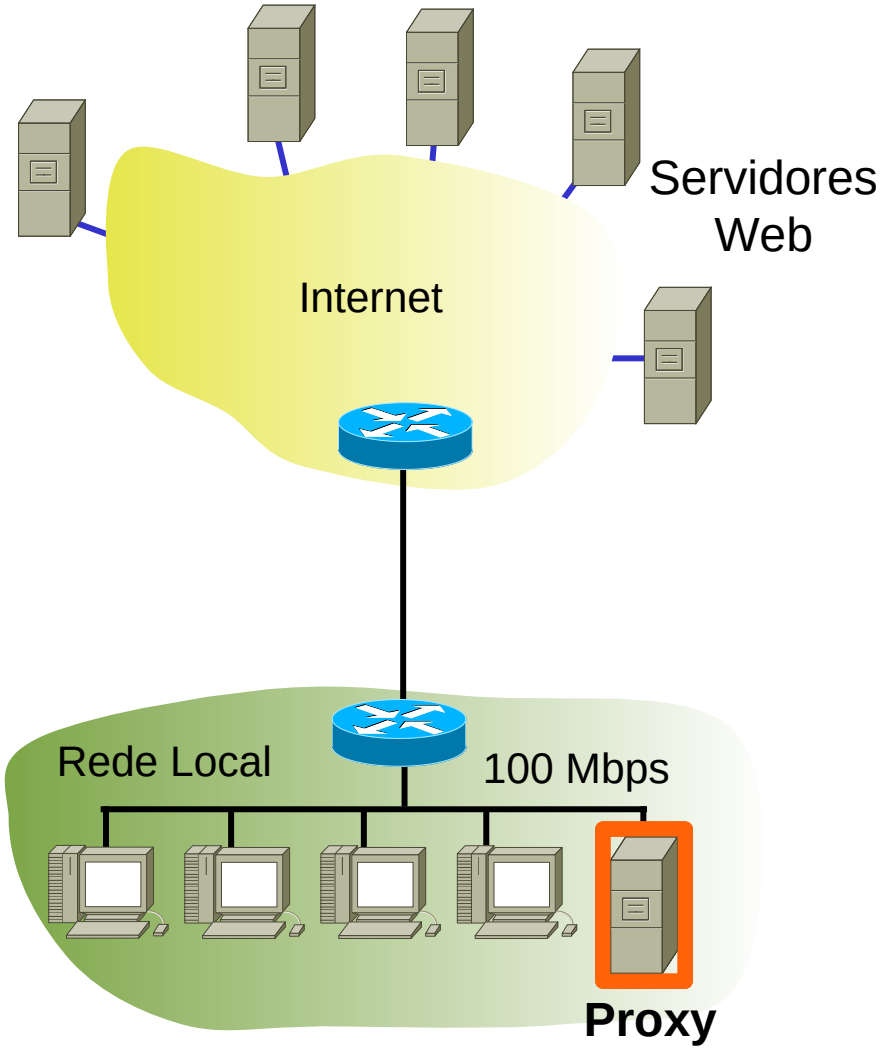
HTTP Proxy Servers (1)

- Acts as an intermediate element between the client and the server:
  - The client interacts with the Proxy Server as if it were the Web server.
  - The Proxy Server interacts with the Web servers on behalf of the clients (for the Web server, the Proxy Server is the client).
- The Proxy server stores all files requested by clients (up to the limit of their storage capacity)
- When a client requests a file that already exists on the Proxy server, it does not have to be requested again from the Web server (it only sends a conditional get message).



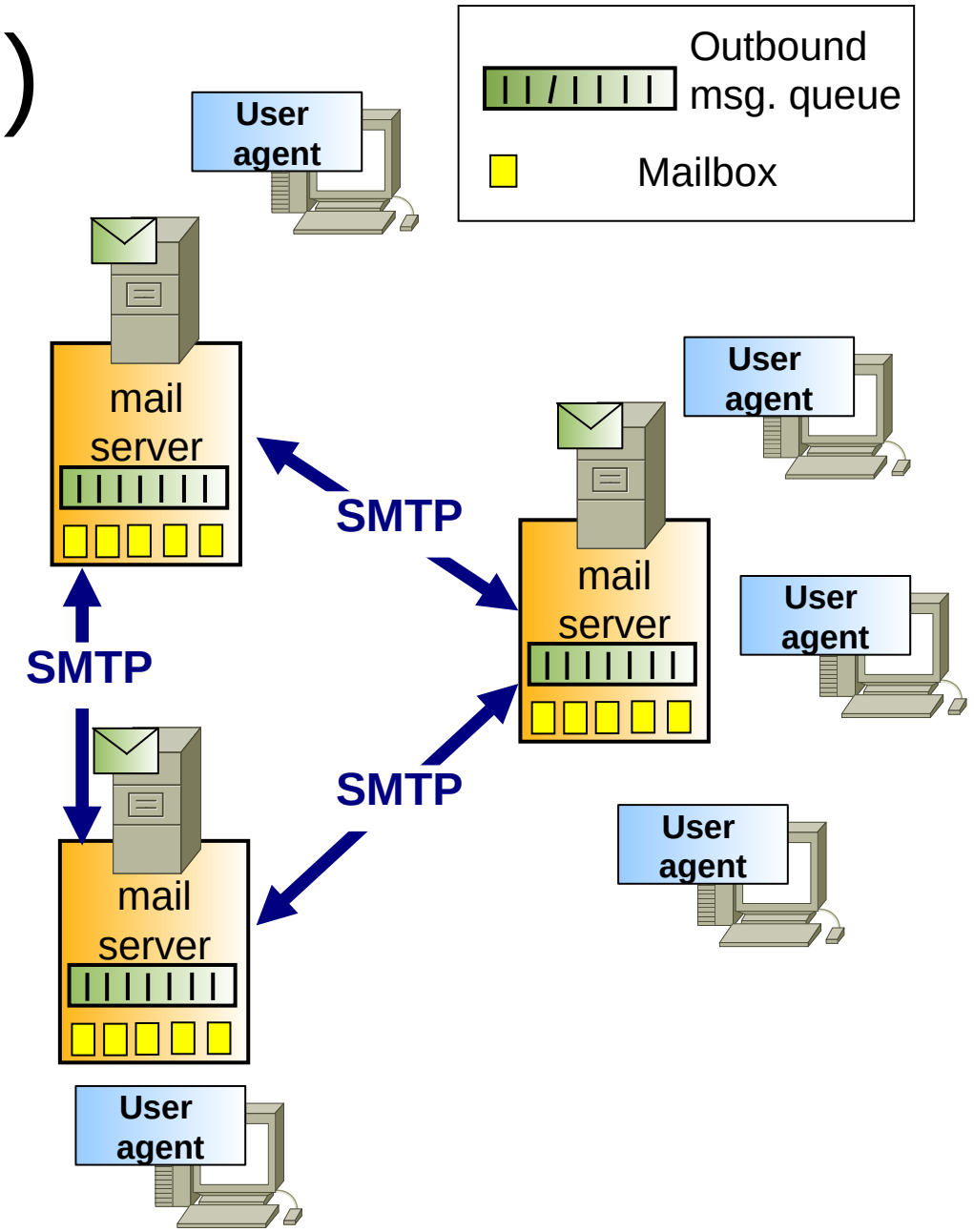
HTTP Proxy Servers (2)

- Proxy servers in companies or institutions:
  - Decrease interaction times.
  - Reduce traffic to the public network.
- Proxy servers on Internet Service Providers (ISPs):
  - They allow an infrastructure of automatic distribution of the most requested Web contents.
  - Usually the ISP Proxy is closer to the client than the server.
    - Response times are lower.
- Nowadays, the more dynamic web contents make proxies less relevant.



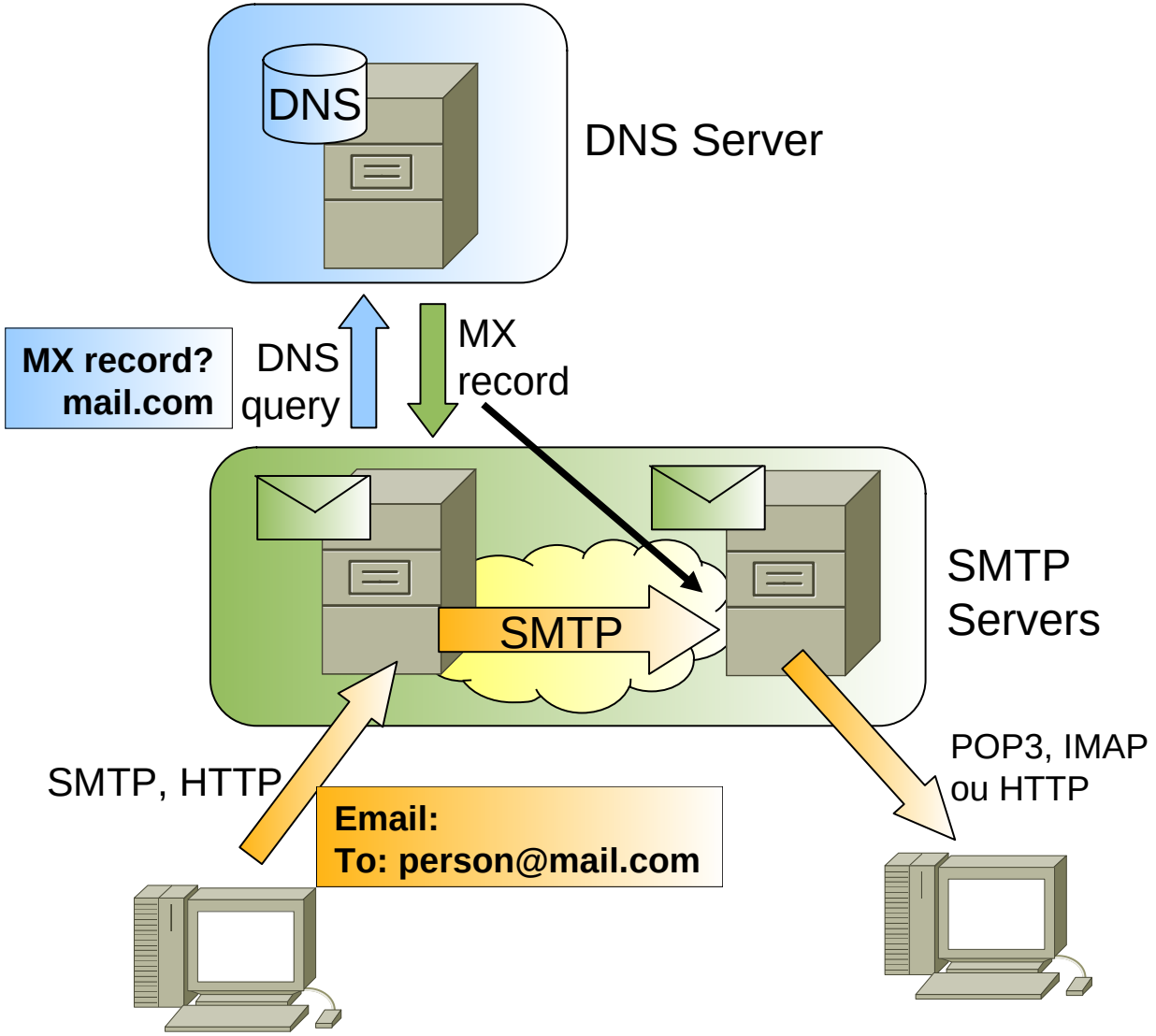
Eletronic Mail (e-Mail)

- User agents: application in which the user writes, sends, receives and reads e-mail messages.
  - The user agent exchanges messages with your mail server.
- Mail server: an e-mail server that sends and receives messages to and from clients, and to and from other servers.
- The mail server includes:
  - A mailbox for each client where the messages are stored.
  - A queue of outgoing messages with messages from its clients customers that have not yet been sent.



E-Mail Protocols

- Message Forwarding (between mail servers)
  - SMTP (Simple Mail Transfer Protocol)
- Message Sending (from user-agent to client server)
  - SMTP (Simple Mail Transfer Protocol)
  - HTTP (Hyper-Text Transport Protocol)
- Mailbox access (from server to user-agent)
  - POP3 (Post Office Protocol – version 3)
  - IMAP (Internet Mail Access Protocol)
  - HTTP (Hyper-Text Transport Protocol)



Simple Mail Transfer Protocol (SMTP)

- It runs over TCP and the default port number of the server is TCP 25.
- The communication is established by the entity that wants to send information (push protocol).
- Direct communications: by default, the sender's mail server sends the messages directly to the recipients mail server.
- The protocol follows a client/server philosophy:
  - The client issues commands.
  - The server responds to commands.
  - Like HTTP, each response consists of a 3-digit code followed by an optional string.
- E-mail in 7-bit ASCII format ending with "CRLF.CRLF"
  - CR- carriage return, LF- line feed

E-Mail Messages Format

```
From: antonio@av.it.pt
To: bruno@ua.pt
Subject: Tens fome?

Boas!
A que horas vamos comer?
Antonio
.
```

- Messages in ASCII format.
- They start with a set of header lines followed by an empty line followed by the body of the message:
  - Some lines (From and To) are required in the original message.
  - Other header lines (Subject, etc ...) are optional.
  - Some header lines (Received, etc ...) are inserted by mail servers.

SMTP Sample Interaction

```
Server: 220 mail.ua.pt
Client: HELO mail.av.it.pt
Server: 250  Hello mail.av.it.pt
Client: MAIL FROM: <antonio@av.it.pt>
Server: 250 antonio@av.it.pt... Sender ok
Client: RCPT TO: <bruno@ua.pt>
Server: 250 junior@det.ua.pt... Recipient ok
Client: DATA
Server: 354 Enter mail, end with "." on a line by itself
Client: Boas!
Client: A que horas vamos comer?
Client: Antonio
Client: .
Server: 250 Message accepted for delivery
Client: QUIT
Server: 221 mail.ua.pt closing connection
```

Multipurpose Internet Mail Extensions (MIME)

- For non-ASCII data.
- Allows you to send messages of different types of information
- Adds the following header lines:
  - Content-Transfer-Encoding – defines the algorithm for encoding the information content in ASCII format
    - Exemples: base64
  - Content-type: type/subtype; parameters – defines the type of information
    - Exemples: text/plain; charset="ISO-8859-1"
      text/html
      image/gif
      image/jpeg
      video/mpeg
      video/quicktime
      application/msword

SMTP Messages with MIME

```
From: antonio@av.it.pt
To: bruno@ua.pt
Subject: Imagem fixe!
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

(base64 encoded data .....
.....................................
..... base 64 encoded data)
```

- At the user agent of the receiver, the content of the message is:
  - Decoded by the base64 algorithm to obtain the original content, and
  - Delivered to a JPEG decoder to view the sent image.

Multipart MIME Extentions

```
From: antonio@av.it.pt
To: bruno@ua.pt
Subject: Imagem fixe!
MIME-Version: 1.0
Content-Type: multipart/mixed; Boundary=98766789

--98766789
Content-Type: text/plain
Olá Bruno,
Junto envio a fotografia combinada.
--98766789
Content-Transfer-Encoding: base64
Content-Type: image/jpeg
(base64 encoded data .....
.....
..... base 64 encoded data)
--98766789
Content-Type: text/plain
Antonio
```

Texto

Imagem

Texto

- The multipart / mixed header line allows to compose a message with multiple types of information.
- Its Boundary parameter identifies the separation between different types of information in the body of the message.

Mensagem SMTP recebida pelo destinatário

```
Received: from mail.meo.pt by mail.ua.pt; 12 Oct 17 15:30:01
GMT
Received: from mail.av.it.pt by mail.meo.pt; 12 Oct 17 15:27:39
GMT
From: antonio@av.it.pt
To: carlos@meo.pt
Subject: Imagem fixe!
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

(base64 encoded data .....
..... base 64 encoded data)
```

- Each mail server inserts a Received header line that identifies the server it sent, the server it received, and the instant of time the message was received.
  - In the example, the carlos user configured his server mail.meo.pt to forward the messages to the server mail.ua.pt.

Post Office Protocol – version 3 (POP3)

- It runs over TCP and the defaults server port number is TCP 110.
- The communication is established by the entity (user agent) that wishes to receive information (pull protocol).
- Message transfer is done in one of two ways:
  - send-and-remove: messages are removed from the server's mailbox after being sent.
  - send-and-store: messages are kept in the mailbox after being sent.
- The protocol is executed in 3 phases:
  - authentication: the user agent sends the user name and password.
  - transaction: the mail server sends the messages that are in the mailbox of the user; the user agent indicates for each message whether or not it should be removed from the mailbox.
  - update: mail server removes from the mailbox the messages indicated by the user agent for removal.

POP3 Commands

- USER userid
- PASS password
- STAT
  - The response contains the number of messages and the total size in bytes of the messages.
  - Sample: +OK 3 345910
- LIST
  - The answer is a list, where each line identifies the number and size in bytes of each message. Ends with a line with only one dot.
  - Sample:

    +OK 3 messages

    1 1205

    2 305

    3 344400

    .
- RETR msg#
  - Retrieves the message with number msg#
  - Example: RETR 2
- DELE msg#
  - Deletes message with number msg#
  - Example: DELE 3
- RSET
  - Clears all marked messages.
- QUIT

POP3 Interaction (sample)

- Authentication phase:
  - user agent Commands:
    - user: username
    - pass: password
  - mail server Responses:
    - +OK
    - -ERR
- Transaction phase, user agent:
  - list: Lists messages; with the number and size of each one.
  - retr: Retrieves a message based on its number.
  - dele: Deletes a message based on its number.
  - quit: Terminates the POP3 interaction.

```
S: +OK POP3 server ready
C: user antonio
S: +OK
C: pass hungry
S: +OK user successfully logged on
```

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 2 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

Internet Mail Access Protocol (IMAP)

- It runs over TCP and the server port number is TCP 143.
- IMAP allows the user additional important functionalities:
  - Create and manage a messaging directory system on the server; do search operations on the directory system.
  - Requesting the sending of portions of the mail messages.
- In one session, the server is in one of 4 states:
  - Unauthenticated status: the initial state before the user agent sends the user's name and password.
  - Authenticated state: The user agent must identify a directory before sending any command that affects mail messages.
  - Status selected: user agent can send message management commands (view, remove, transfer, etc …).
  - Logout status: When the session ends.

IMAP (Client) Commands

- CAPABILITY
- LOGIN
- SELECT
  - Selects a folder/directory within the mailbox.
  - Same as EXAMINE.
- CREATE/DELETE/RENAME
  - Creates/deletes/renames a folder/directory in the mailbox.
- LIST
  - Lists folders/directories within the mailbox.
- SUBSCRIBE/UNSUBSCRIBE
  - Change the active state of a folder/directory.
- STATUS
  - Verifies the state of a folder/directory.
- FETCH
  - Retrieves a message or folder (fully or partially).
- LOGOUT
- Others: APPEND, EXPUNGE, SEARCH, STORE, COPY, ...