ADVANCED OPERATING SYSTEMS AND NETWORKS

Computer Science Engineering

Universidad Complutense de Madrid

1.3. Network Services

PROFESSORS:
Rubén Santiago Montero
Eduardo Huedo Cuesta

OTHER AUTHORS:
Rafael Moreno Vozmediano
Juan Carlos Fabero Jiménez

Packet Filtering

Firewalls and Packet Filtering

- A **firewall** is a hardware-software security component that analyzes network traffic and determines if it should be allowed or not. Functions:
  - Network packet filtering
  - Activity logging
  - Network address translation
- Firewall types:
  - By state management: They can be based on the characteristics of individual packets (packet filtering or screening) or they can also consider the connection status (Stateful Packet Inspection, SPI)
  - By protocol layer: They can be based on packet headers (network firewall) or on they can also consider packet data belonging to application protocols (application firewall/gateway or Deep Packet Inspection, DPI)
- Packet filtering in Linux (Netfilter/iptables):
  - Based on rules stored in tables
  - Packet filtering and table storage provided by the OS kernel (Netfilter)
  - User-space program for rule management (iptables)

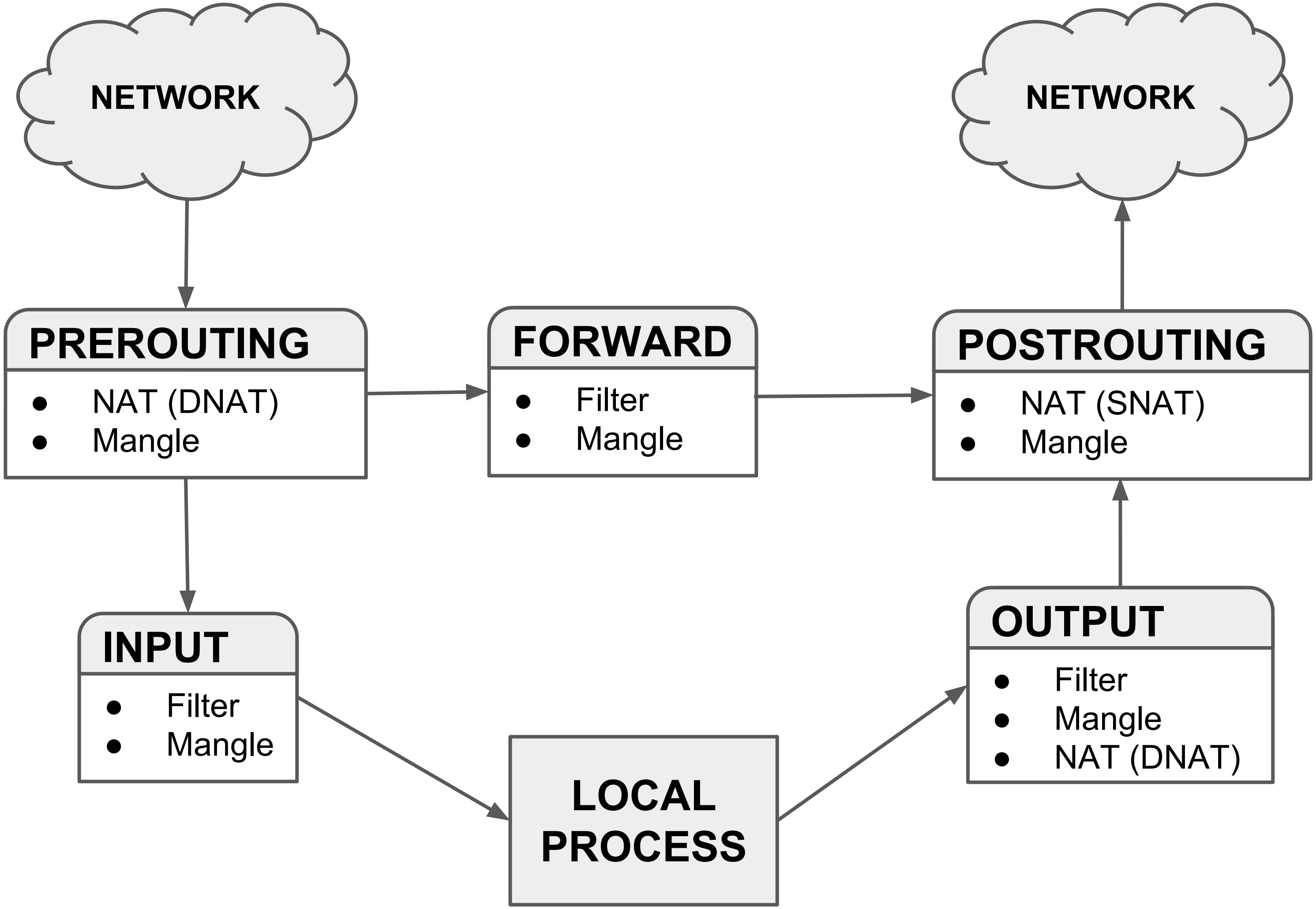
iptables: Tables, Chains and Rules

- **Rules** specify what to do (e.g. drop or accept) with a packet that matches some criteria (e.g. source port, destination address...)
- **Chains** are lists of rules that are applied to packets in order at some point of their processing
  - A rule can move a packet to another chain
  - All input or output packets in the system traverses at least one chain
  - If a packet doesn’t match any rule, the chain’s default policy is applied
- **Tables** are groups of chains dealing with some type of processing

iptables: Predefined Tables and Chains

- **Filter table**
  - Default table used to block or allow packets
    - INPUT chain: applied to packets destined to the system
    - OUTPUT chain: applied to packets generated by the system
    - FORWARD chain: applied to packets being routed through the system
- **NAT table**
  - Used to rewrite source or destination addresses and ports
    - PREROUTING chain: applied to input packets before routing
      - Used for DNAT (Destination NAT)
    - POSTROUTING chain: applied to output packets after routing
      - Used for SNAT (Source NAT)
    - OUTPUT chain: applied to locally-generated packets before routing
- **Mangle table**
  - Used for specialized packet alteration (e.g. change TOS/DS or MSS in TCP)
  - It has all the five previous chains

iptables: Predefined Tables and Chains



NETWORK

PREROUTING

- NAT (DNAT)
- Mangle

FORWARD

- Filter
- Mangle

POSTROUTING

- NAT (SNAT)
- Mangle

NETWORK

INPUT

- Filter
- Mangle

LOCAL PROCESS

OUTPUT

- Filter
- Mangle
- NAT (DNAT)

Simplified version of tables and chains

iptables: Rule Definition

- Rules can be defined in terms of packet information and connection status
- A rule must include the chain to which is added and must include a target

Option/Example	Meaning

`-A INPUT`
`-A OUTPUT`
`-A FORWARD`
	Add a rule to the INPUT chain
	Add a rule to the OUTPUT chain
	Add a rule to the FORWARD chain (only for routers)

`-s 192.168.1.1`
`-d 140.10.15.1`
	Filter by source IP address
	Filter by destination IP address

`-p tcp`
`-p udp`
`-p icmp`
	Filtering of TCP packets
	Filtering of UDP packets
	Filtering of ICMP packets

`--sport 3000`
`--dport 80`
`--icmp_type 8`
	Filter by source port number (for TCP or UDP)
	Filter by destination port number (for TCP or UDP)
	Filter by ICMP type (for ICMP)

`-i eth0`
`-o eth1`
	Filter by network interface the packet was received
	Filter by network interface the packet is going to be sent

iptables: Rule Definition

- Rule definition in terms of connection status:

Option	Meaning
`-m state --state NEW`	Filter packets starting new connections (first one)
`-m state --state ESTABLISHED`	Filter packets from established connections
`-m state --state RELATED`	Filter packets from new connections related to other established connections
`-m state --state INVALID`	Filter packets from connections in other state

- Rule targets (jumps) for packet filtering:
  - `-j DROP`
  - `-j ACCEPT`
  - `-j LOG`
  - `-j REJECT`, like `-j DROP` but sends an ICMP packet (the type can be defined with `--reject-with`, e.g. `connection-administratively-filtered` or `icmp-port-unreachable`)

iptables: Rule Examples

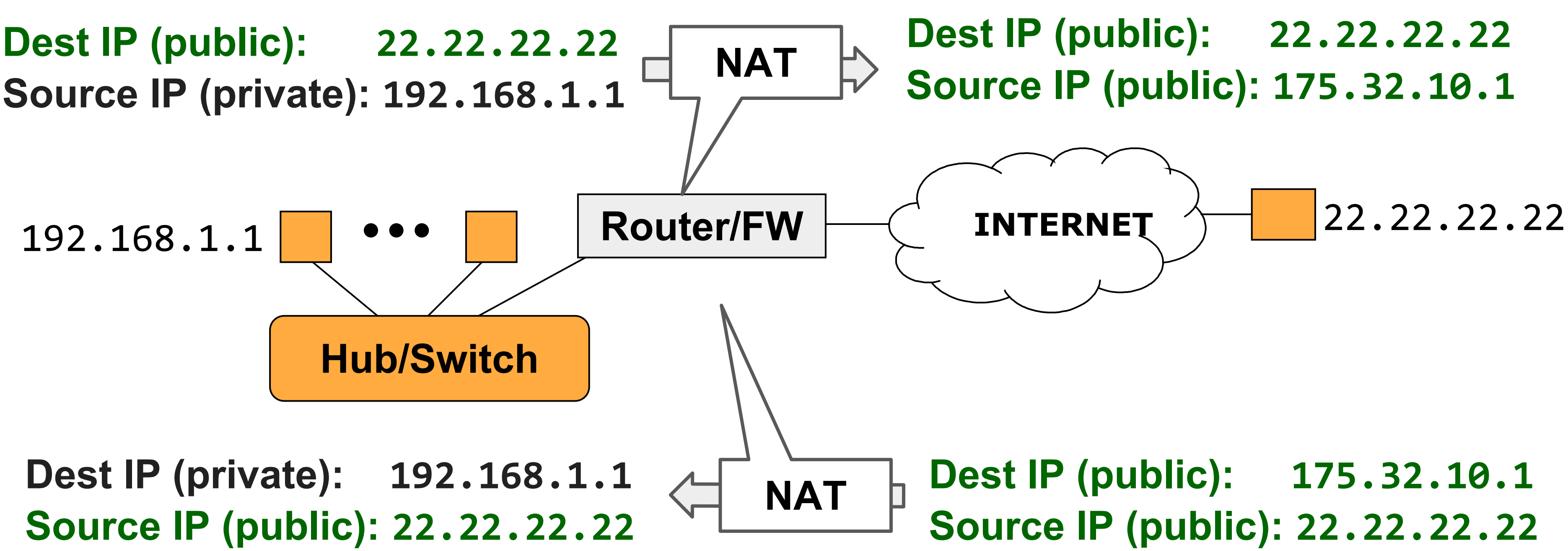
```
# Default policy for INPUT, OUTPUT and FORWARD chains
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
# Allow incoming or outgoing packets from established or related
#connections
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
# Allow incoming SSH connections (tcp/22) from home PC
iptables -A INPUT -s 200.1.1.1 -p tcp --dport 22 -m state \
      --state NEW -j ACCEPT
# Allow outgoing web connections (tcp/80) to any destination
iptables -A OUTPUT -p tcp --dport 80 -m state --state NEW -j ACCEPT
# Allow outgoing POP3 connections (tcp/110) to mail server
iptables -A OUTPUT -d 22.1.1.1 -p tcp --dport 110 -m state \
      --state NEW -j ACCEPT
# Allow outgoing DNS connections (udp/53) to DNS server
iptables -A OUTPUT -d 22.1.1.2 -p udp --dport 53 -m state \
      --state NEW -j ACCEPT
```

Network Address Translation

Network Address Translation

IPv4 Private Networks

- Alleviates the problem of the limited number of IPv4 addresses
- The objective of NAT is to provide Internet access to hosts in private networks



Dest IP (public): 22.22.22.22
Source IP (private): 192.168.1.1
NAT
Dest IP (public): 22.22.22.22
Source IP (public): 175.32.10.1
192.168.1.1
•••
Router/FW
INTERNET
22.22.22.22
Hub/Switch
Dest IP (private): 192.168.1.1
Source IP (public): 22.22.22.22
NAT
Dest IP (public): 175.32.10.1
Source IP (public): 22.22.22.22

Static Translation

- Assignment of N private addresses to N public addresses
- Fixed assignment

- Example for N=7:

Private IP	Public IP
192.168.1.3	147.96.80.132
192.168.1.23	147.96.80.12
192.168.1.2	147.96.80.122
192.168.1.5	147.96.81.2
192.168.1.4	147.96.81.23
192.168.1.7	147.96.81.77
192.168.1.56	147.96.81.4

Dynamic Translation

- Assignment of N private addresses to M public addresses (M < N)
- Dynamic assignment, only M machines can access the Internet at a given time

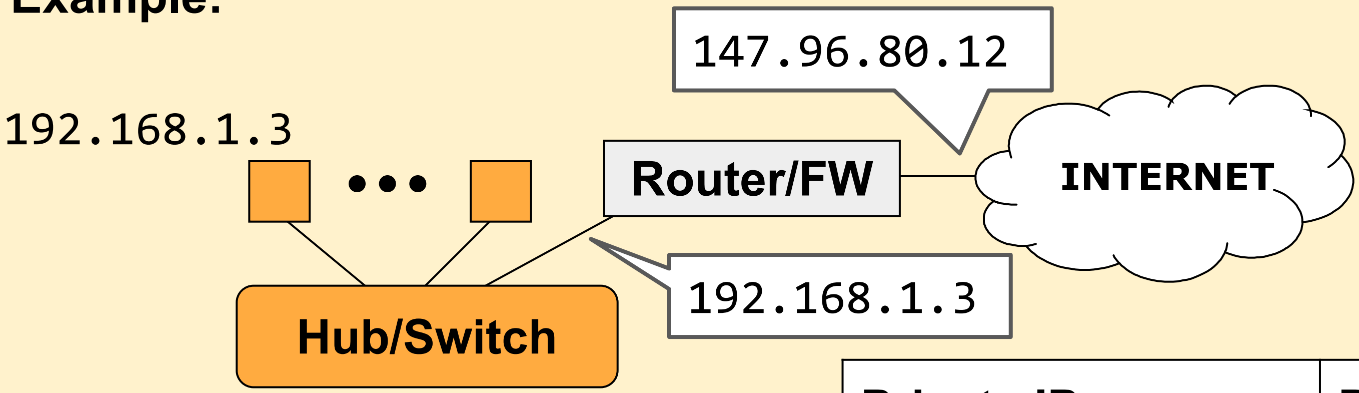
- Example for N=7 and M=3:

Private IP	**Public IP**
`192.168.1.3`	`147.96.80.132`
`192.168.1.23`	`147.96.80.12`
`192.168.1.2`	`147.96.80.122`
`192.168.1.5`	No access to Internet until a public IP address is released
`192.168.1.4`	
`192.168.1.7`	
`192.168.1.56`	

NAPT - Masquerading

- NAPT (Network Address and Port Translation)
- Assignment of N private addresses to **1 public address**
- **Operation:**
  - The only available public IP address is the router’s one
  - The client port number of the source host is translated to a free port in the router

Example:



Private IP	**Public IP**
192.168.1.3:3453	**147.96.80.12**:6782
192.168.1.7:2380	147.96.80.12:3342
192.168.1.5:6790	147.96.80.12:4390

Source NAT (SNAT)

- SNAT target in NAT table changes the source address of the first packet
  - SNAT is done after routing, in the POSTROUTING chain, just before the packet is finally sent out
  - The result is applied to all subsequent packets of the same connection
  - Provides NAPT with a static public IP address

```
iptables -t nat -A POSTROUTING -o ppp0 -j SNAT --to 175.20.12.1
```

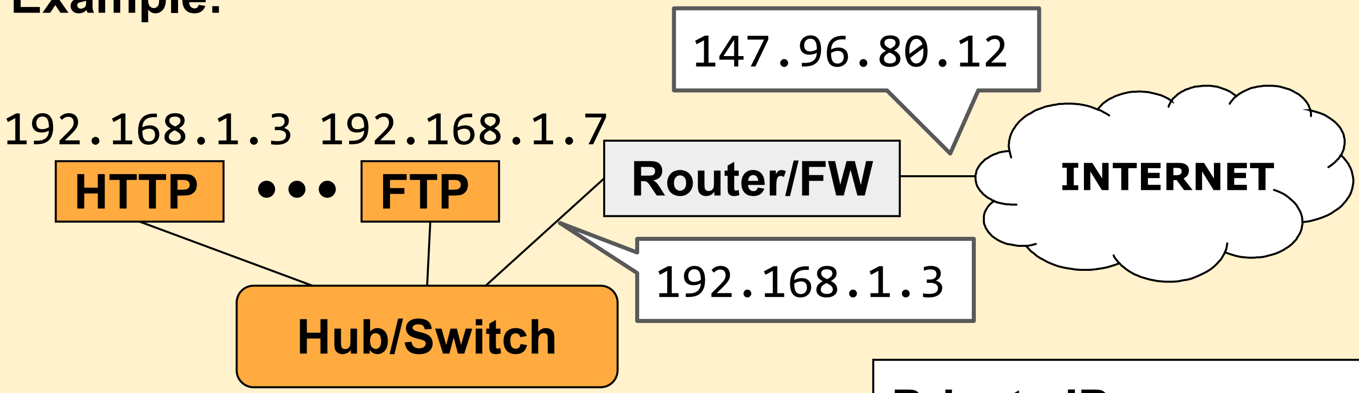
- The MASQUERADE target can be used when the public IP address is dynamic
  - It uses the IP address of the interface as source address
  - Being dynamic, this public IP address can change between connections, so it also keeps track of active connections to apply the change

```
iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
```

Port Forwarding - Virtual Servers

- Assignment of **1 public address** to N private addresses
- Allows servers in the private network to be accessed from the Internet
- **Operation:**
  - All servers are accessed using the same public IP address (the router's one)
  - The router redirects packets to the actual server in the private network

Example:



Private IP	Public IP
192.168.1.3:8080	**147.96.80.12**:80
192.168.1.7:20	147.96.80.12:20
192.168.1.7:21	147.96.80.12:21

Destination NAT (DNAT)

- Target DNAT in NAT table can modify the destination address of the first packet and, optionally, its destination port
  - DNAT is done before routing, in the PREROUTING chain, just as the packet comes in
  - It can also be done in the OUTPUT chain (also before routing) to translate locally generated packets
  - The result is applied to all subsequent packets of the same connection

```
iptables -t nat -A PREROUTING -d 175.20.12.1 -p tcp --dport 80 \
            -j DNAT --to 192.168.1.1:8080

iptables -t nat -A PREROUTING -d 175.20.12.1 -p tcp --dport 25 \
            -j DNAT --to 192.168.1.2

iptables -t nat -A PREROUTING -d 175.20.12.1 -p tcp --dport 20 \
            -j DNAT --to 192.168.1.3

iptables -t nat -A PREROUTING -d 175.20.12.1 -p tcp --dport 21 \
            -j DNAT --to 192.168.1.3
```

Domain Name System (DNS)

Domain Name System (DNS)

- DNS keeps, among other things, the mapping between domain names and IP addresses
- DNS is implemented as a distributed database
  - Each site stores information about its systems only
  - Sites interchange and share information with other sites
  - DNS receives and performs queries about domain names
- DNS is a very complex system
  - Defined in approximately 108 RFCs request for comments
  - Several implementations with different functionality, for example:
    - BIND (80%)
    - Microsoft DNS (15%)
    - djbdns (3%), NSD, Unbound, PowerDNS (<1%)
- DNS defines:
  - A hierarchical name space of domain names and IP addresses
  - A distributed database and client tools (resolvers) to query it
  - A mechanism to find network services
  - A protocol to interchange information

Zones and Domains

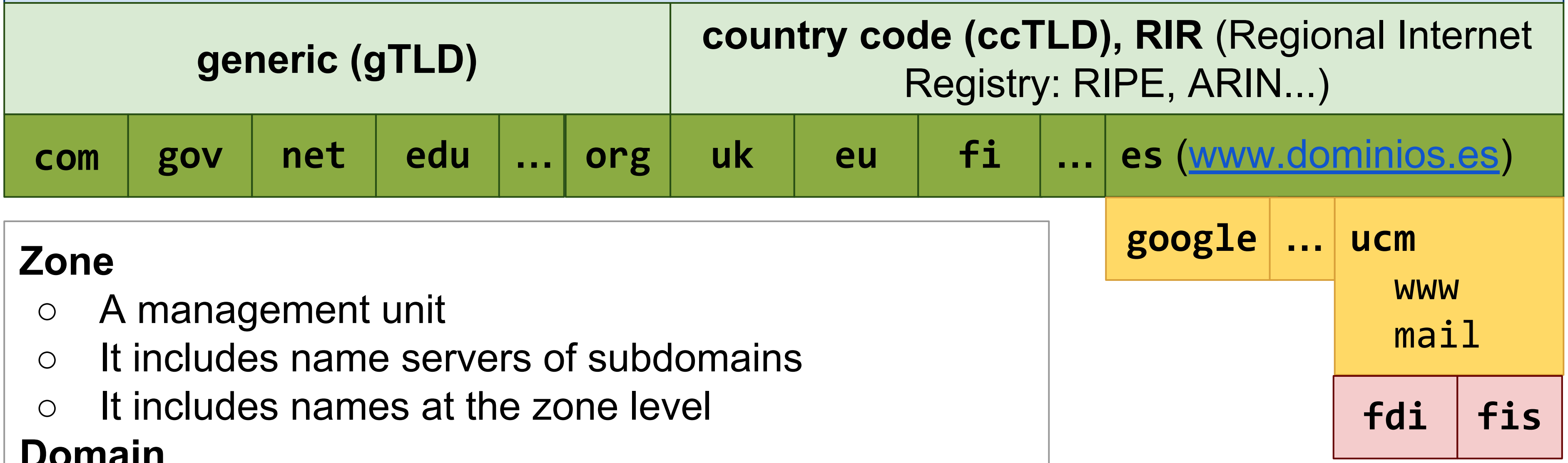
Root domain "."

- Contains a reference to the name servers of the top level domains
- 13 name servers `[a-m].root-servers.net` (several physical hosts, anycast)

each mapped to one IP, but each IP is assigned to several machines

Top Level Domains (TLDs)

- Managed by ICANN
- Full list in http://www.iana.org/domains/root/db
- Each zone contains the authoritative name servers for TLD and referrals to name servers of subdomains



generic (gTLD)

country code (ccTLD), RIR (Regional Internet Registry: RIPE, ARIN...)

Zone

○ A management unit
○ It includes name servers of subdomains
○ It includes names at the zone level

Domain

○ Delegated management into several organizations
○ All subdomains (subtree)

Domain Names

- **Fully Qualified Domain Name (FQDN)**
  - List of node names or domain labels (e.g. `www`, `printer-server`...) representing the hierarchy from the lowest relevant level to the root (although it is usually omitted), using the dot character as a separator between labels typically ended with a dot ".", but it is mostly omitted
    - Example: `www.ucm.es.` (most significant part at right, `"."`)
- **Reverse lookup**
  - Get the domain name associated to an IP address
  - IPv4 address space in `in-addr.arpa.`
  - The IP address is reversed to have the most significant part on the right
    - Example: `63.173.189.1` → `1.189.173.63.in-addr.arpa.`
- **Restrictions in domain names**
  - No limit in the number of subdomains in the hierarchy
  - Maximum of 255 characters per FQDN (including dots)
  - Maximum of 63 characters per label in FQDN
  - Valid characters are numbers (`"0"` to `"9"`), letters (`"a"` to `"z"`, not case sensitive) and dash symbols (`"-"`)

Operation: Resource Records

- Database structured in **Resource Records** (RR)
- Servers store records in **zone files** (text format)
- Different records to store name servers, name-to-IP and IP-to-name mappings, mail servers...
- Records are standard and implementation independent
- Basic information that is interchanged and cached in servers
- Example: `piscis.mydnsdomain.com` ←→ `147.96.80.1`

(address) relates the node name to the IP in the domain below

```
piscis   IN  A 147.96.80.1
         IN MX mailserver.mydnsdomain.com.
```

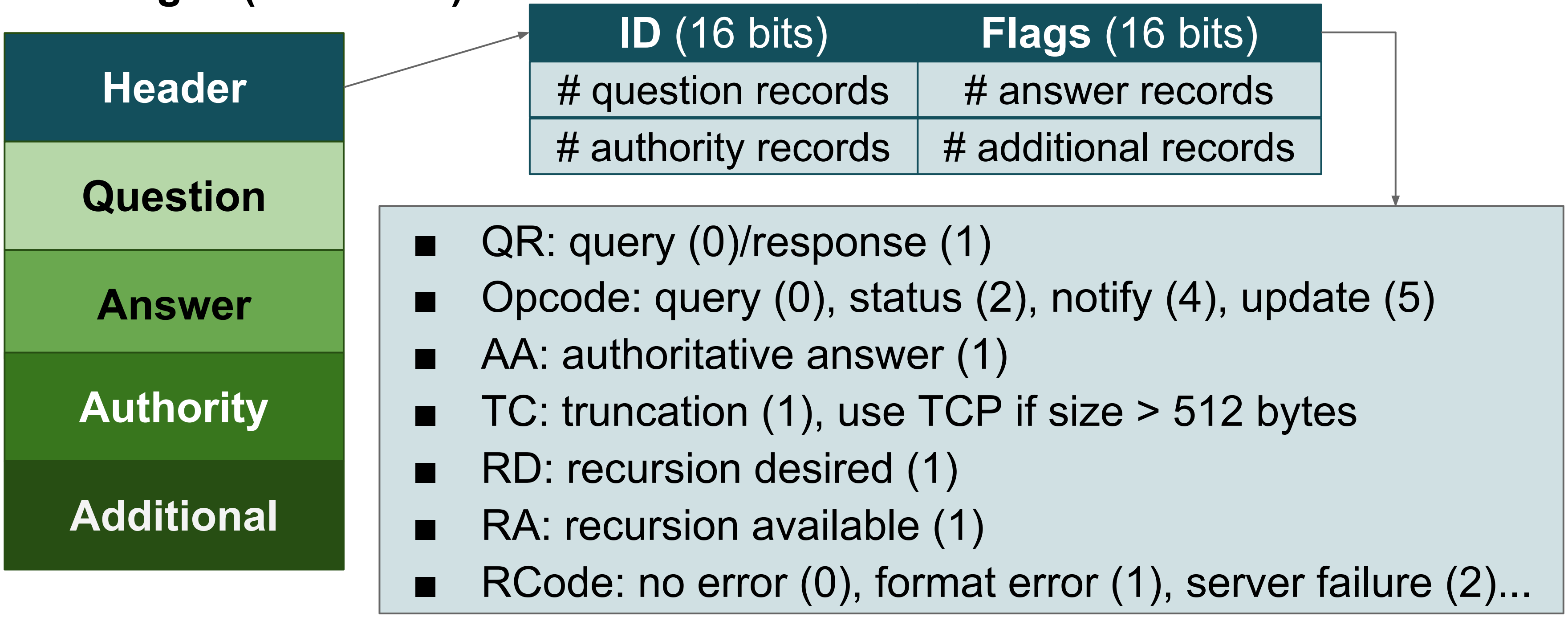
```
1        IN PTR piscis.mydnsdomain.com.
```

1 bc the name belongs to the
domain 80.96.147.in-addr.arpa.

DNS record type

Operation: DNS Protocol

- **Transport Protocol**
  - Mainly UDP using port 53 UDP bc it has to be very efficient
  - TCP for zone transfers or long answers (more than 512 bytes, RFC 5966)
- **Messages (RFC 1035)**



Header

Question

Answer

Authority

Additional

ID (16 bits)	**Flags** (16 bits)
# question records	# answer records
# authority records	# additional records

- ■ QR: query (0)/response (1)
- ■ Opcode: query (0), status (2), notify (4), update (5)
- ■ AA: authoritative answer (1)
- ■ TC: truncation (1), use TCP if size > 512 bytes
- ■ RD: recursion desired (1)
- ■ RA: recursion available (1)
- ■ RCode: no error (0), format error (1), server failure (2)...

  - Question section (both in questions and answers) includes the domain name and the record type for which it is asking
  - Authority section specifies the authoritative servers for the domains
  - Additional section includes records that may help the client (resolver)

Operation: DNS Protocol

answer from a DNS server

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 19305
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 7, ADDITIONAL: 14
;; WARNING: recursion requested but not available
```

flag ra is not present, this server doesn't allow recusion

```
;; QUESTION SECTION:
;informatica.ucm.es.        IN  A
```

if the server knows the answer, there would be an answer section

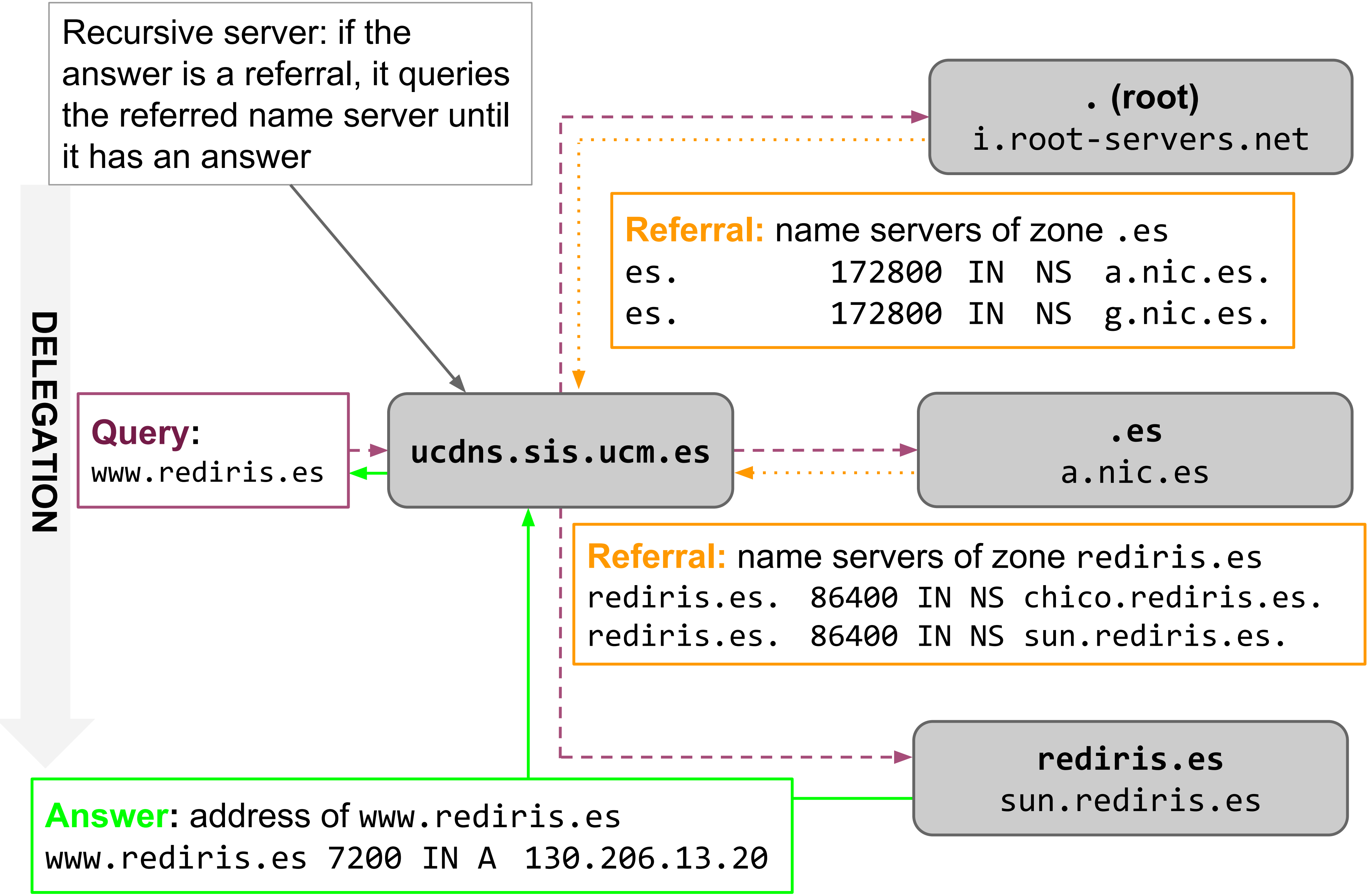
```
;; AUTHORITY SECTION:
es.         172800  IN  NS  f.nic.es.
es.         172800  IN  NS  g.nic.es.
es.         172800  IN  NS  a.nic.es.
...
```

the authoritive name servers (ns) are these

```
;; ADDITIONAL SECTION:
a.nic.es.       172800 IN  A   194.69.254.1
a.nic.es.       172800 IN  AAAA   2001:67c:21cc:2000::64:41
...
```

provides the IP addresses of the servers above

Operation: Delegation and Resolution



Recursive server: if the answer is a referral, it queries the referred name server until it has an answer

DELEGATION

Query: www.rediris.es

ucdns.sis.ucm.es

. (root)
i.root-servers.net

Referral: name servers of zone .es

```
es.          172800  IN  NS  a.nic.es.
es.          172800  IN  NS  g.nic.es.
```

.es
a.nic.es

Referral: name servers of zone rediris.es

```
rediris.es.  86400 IN NS chico.rediris.es.
rediris.es.  86400 IN NS sun.rediris.es.
```

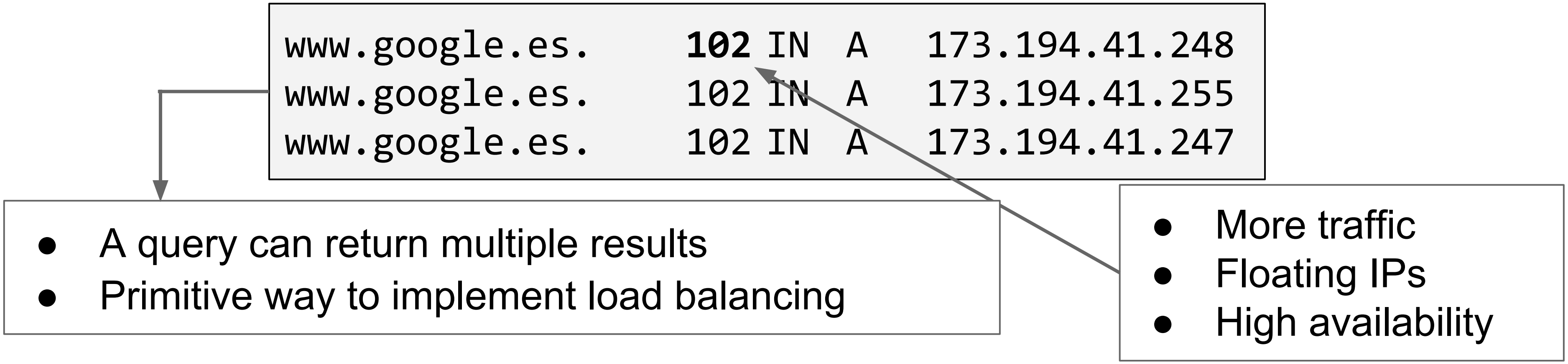
rediris.es
sun.rediris.es

Answer: address of www.rediris.es

```
www.rediris.es 7200 IN A  130.206.13.20
```

Operation: Caching

- Caching address resolution notably improves efficiency
- Name-to-IP relationship is practically static
- Answers are cached for TTL (“time-to-live”)
- The TTL of each entry varies depending of its level in the hierarchy, which is associated to its probability of change. In the previous example:
    - Name servers of zone `.es`: 2 days (172800 seconds)
    - Name servers of zone `.rediris.es`: 1 day (86400 seconds)
    - IP address of `www.rediris.es`: 2 hours (7200 seconds)
- Failed queries are also cached (negative caching)
    - No host or domain matches the requested domain name
    - The requested record doesn’t exist for the domain name
    - The server doesn’t answer or is not reachable due to network problems
- Cache clients and servers can observe TTL or not



```
www.google.es.    102 IN  A   173.194.41.248
www.google.es.    102 IN  A   173.194.41.255
www.google.es.    102 IN  A   173.194.41.247
```

- A query can return multiple results
- Primitive way to implement load balancing

- More traffic
- Floating IPs
- High availability

Name Servers

- **Authoritative Servers (primary and secondary)**
  - They officially represent the zone
  - Primary server, or master, gets the official DB from disk
  - Secondary server, or slave, get the DB from the primary server through a zone transfer
  - DNS specification requires at least one secondary server per zone
- **Caching-only Servers**
  - They store results of queries done, starting from the root servers
  - They don’t store any record of its own, and are not authoritative for any zone
  - Used to reduce query latency and DNS traffic on the network
- **Recursive and Non-recursive Servers**
  - Non-recursive servers return a referral to the name server that could have the requested record, if they don’t have it
  - Recursive servers resolve any referral until they return a positive or negative answer to the client
  - Usually, authoritative servers are non-recursive (and they should be)
  - Recursive resolvers should be provided for client configuration

DNS Database

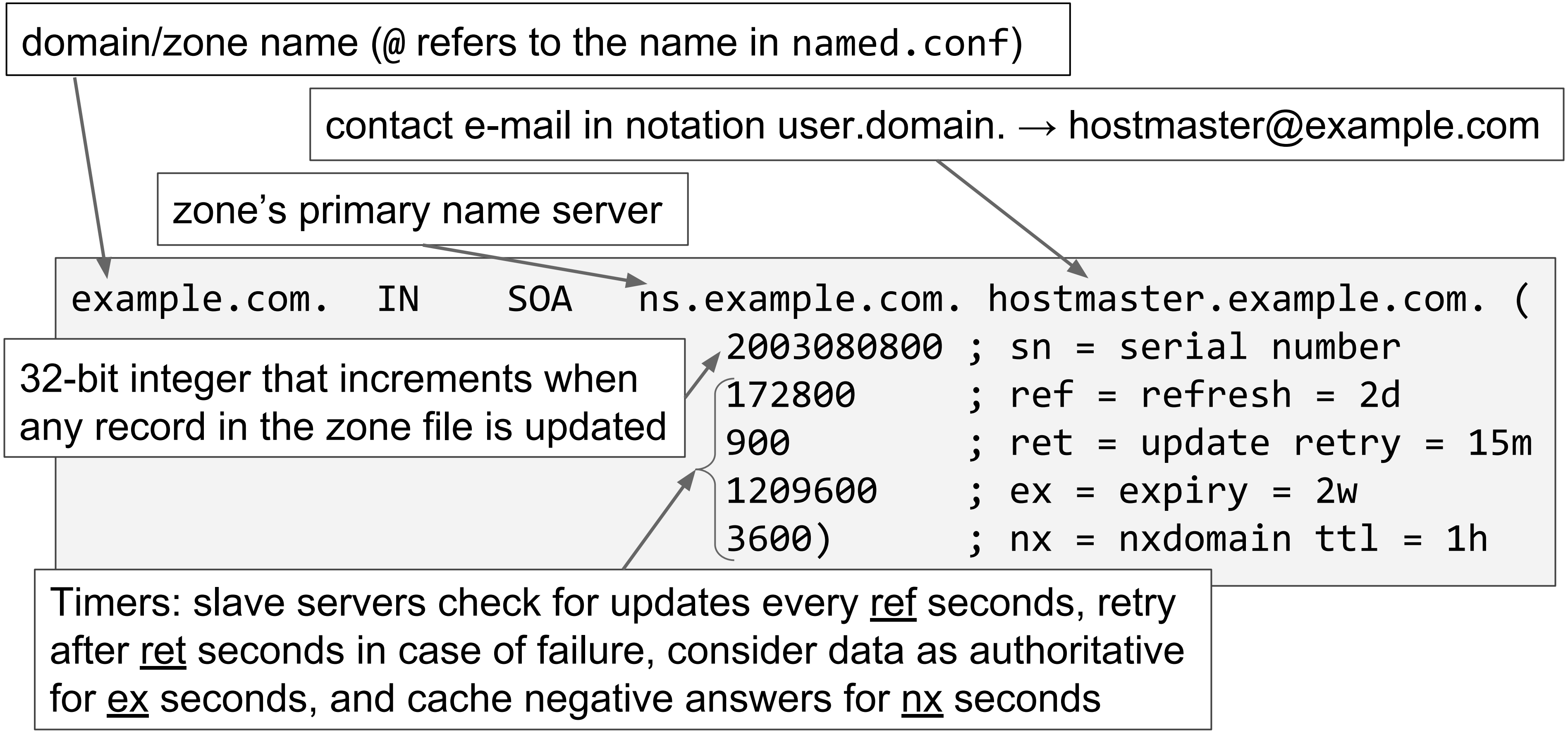
- Zone files in text format maintained in the zone’s primary server
- **Commands** specify how to interpret the records. Standard commands:
    - `$ORIGIN`: default domain added to all names that are not FQDN
    - `$INCLUDE`: to include a file, so that information can be kept in separate files
    - `$TTL`: default TTL value
- **Resource Records (RR)** are associated to the domain names of the zone
    - Format (RFC 1034 and 2181):

```
[name]  [ttl]  [class]  type  data
```

        - `name`: identifies the record, usually a host or domain name
        - `ttl`: time in seconds the record can be cached and considered valid
        - `class`: `IN` (internet), `HS` (Hesoid, used internally in some sites) and `CH` (Chaosnet, now provides information of BIND server)
        - `type`: classified in 4 groups (Zone, Basics, Security and Optional), there are many types, but only a few are used regularly
        - `data`: Depends on the record type

DNS Database: SOA Record

- SOA (Start of Authority) record must be the first record in the zone and defines the global parameters for the zone
- There are usually two zones:
  - Forward zone: name → IP
  - Reverse zone: IP → name



domain/zone name (@ refers to the name in `named.conf`)

contact e-mail in notation user.domain. → hostmaster@example.com

zone’s primary name server

```
example.com.  IN   SOA  ns.example.com. hostmaster.example.com. (
                           2003080800 ; sn = serial number
                           172800     ; ref = refresh = 2d
                           900        ; ret = update retry = 15m
                           1209600    ; ex = expiry = 2w
                           3600)      ; nx = nxdomain ttl = 1h
```

32-bit integer that increments when any record in the zone file is updated

Timers: slave servers check for updates every ref seconds, retry after ret seconds in case of failure, consider data as authoritative for ex seconds, and cache negative answers for nx seconds

DNS Database: NS Record

- NS (Name Server) records specify the authoritative name servers for the zone
- Also, include the name servers of subdomains delegated to other organizations
- Usually added after SOA record (name can be omitted as being the same)

Refers to `example.com` in SOA record

```
       NS  ns.example.com.
       NS  ns1.example.com.
       NS  ns-ha.example.com.
sub    NS  ns.sub.example.com.
       NS  ns.example.com.
```

Notice the final "`.`" for FQDNs

Subdomains included for the delegation to work, although the information corresponds to the zone of the subdomain (**glue records**). Similarly, `com.` must include the name servers listed in `example.com.`

DNS Database: A and PTR Records

- Address (A for IPv4 or AAAA for IPv6) records are the basis of DNS as they provide forward translation (hostname → IP)

```
ns                IN  A     63.175.177.1
                  IN  A     63.175.177.4
                  IN  AAAA  2001:501:2f::a01b
ns1.example.com.  IN  A     63.175.177.2
```

- No FQDN, completed with `$ORIGIN`
- Several records for `ns.example.com.`

- Pointer (PTR) records provide reverse translation (IP → hostname)
- Organized in different zones for each subnetwork (or redefining `$ORIGIN`)

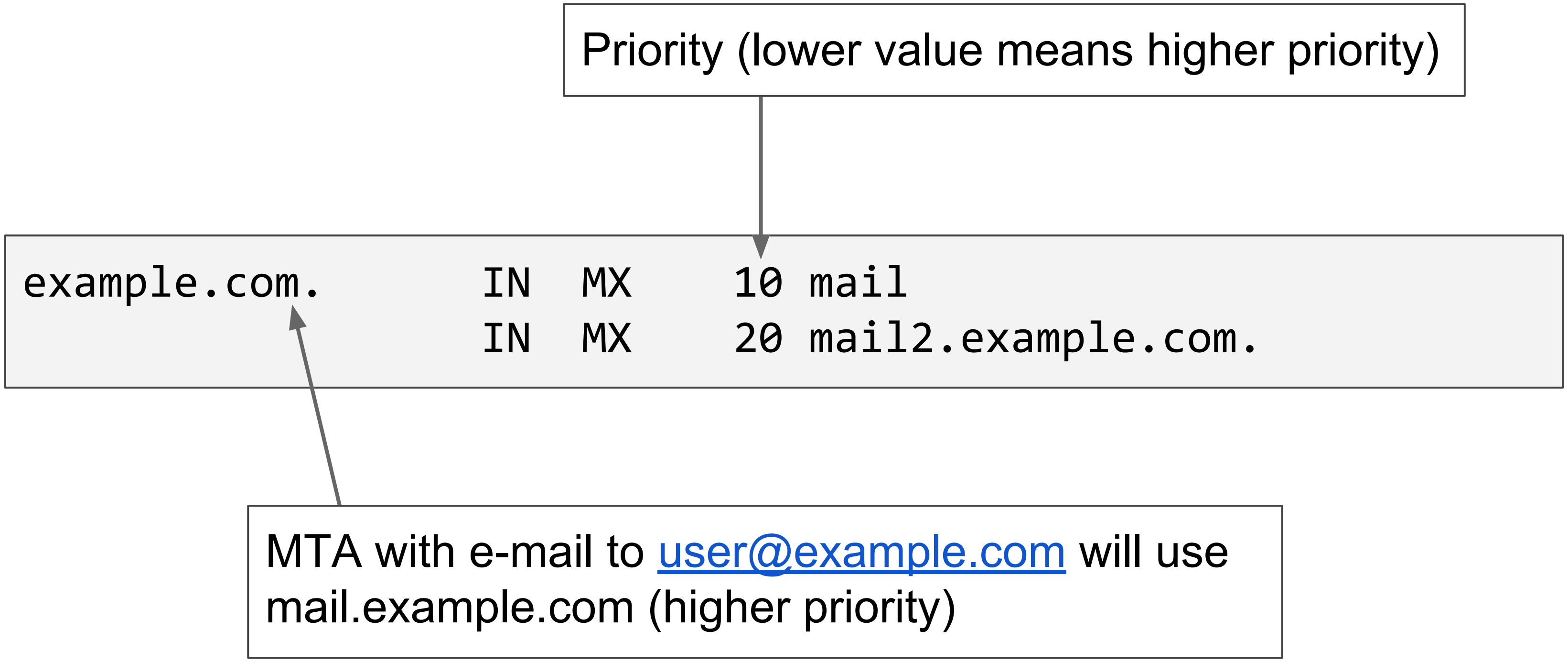
```
1.177             IN  PTR  ns.example.com.
```

Relative to `175.63.in-addr.arpa`

FQDN used, `$ORIGIN` not added

DNS Database: MX Record

- Mail eXchanger (MX) records are used by e-mail systems to route messages efficiently
- Allow receiving all e-mail of an organization in a centralized way and perform centralized operations (e.g. SPAM filtering)



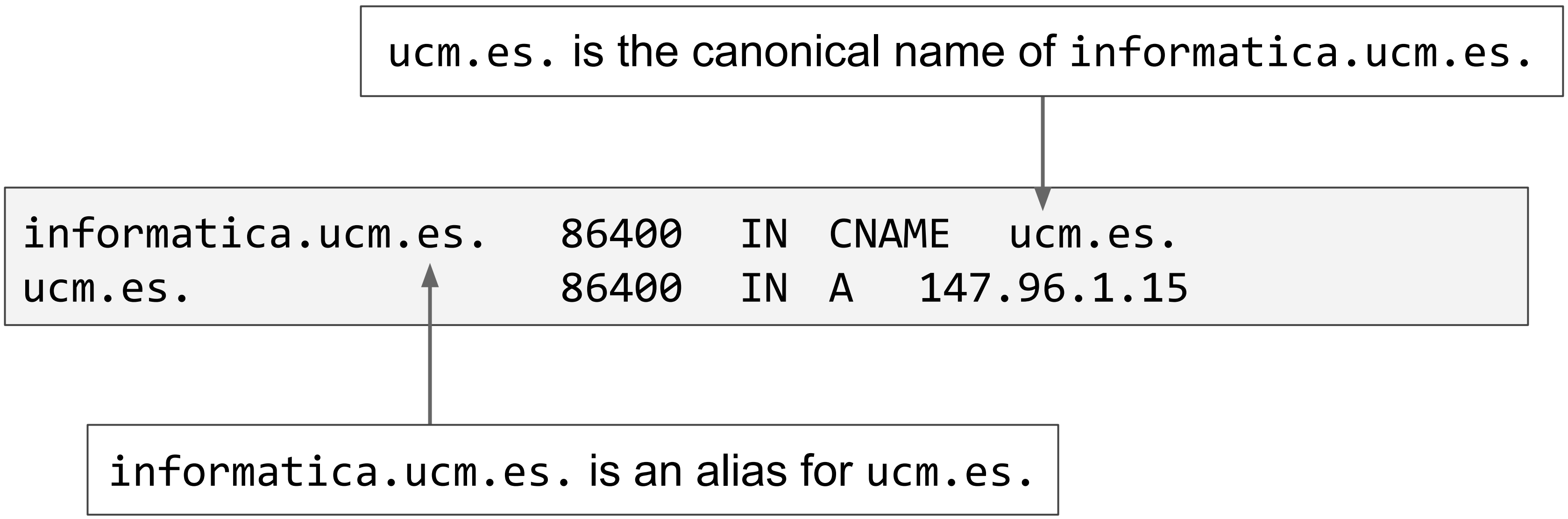
Priority (lower value means higher priority)

```
example.com.      IN  MX    10 mail
                  IN  MX    20 mail2.example.com.
```

MTA with e-mail to user@example.com will use mail.example.com (higher priority)

DNS Database: CNAME Record

- Canonical Name (CN) records provide an alias for a domain name
- They point to a domain name (the canonical name)
- An *alias* defined by a CNAME record must not have any other records
- MX and NS can not point to a CNAME record
- Resolvers provide the address of the canonical name in the additional section



`ucm.es.` is the canonical name of `informatica.ucm.es.`

```
informatica.ucm.es.   86400  IN  CNAME  ucm.es.
ucm.es.               86400  IN  A   147.96.1.15
```

`informatica.ucm.es.` is an alias for `ucm.es.`

DNS Database: Example

```
; Example for zone example.com
$TTL 2d ; TTL default = 2 days or 172800 seconds
$ORIGIN example.com.
example.com.  IN    SOA  ns.example.com. admin.example.com. (
                    2003080800 ; serial number (year,month,day,seq)
                    3h         ; refresh
                    15M        ; update retry = 15 minutes
                    3W12h      ; expiry = 3 weeks + 12 hours
                    2h20M      ; nx ttl = 2 hours + 20 minutes
                    )
              IN    NS   ns
              IN    NS   ns-backup
              IN    MX   10 mail ; equivalent to mail.example.com.
              IN    MX   20 mail2.example.com. ; failover server
; the local servers need an A record
ns            IN    A    192.168.0.10
ns-backup     IN    A    192.168.0.11
mail          IN    A    192.168.0.12
mail2         IN    A    192.168.0.13
www           IN    A    192.168.0.50
```

BIND

- Berkeley Internet Name Domain (BIND) is an open source implementation of the DNS specification
- Common versions are BIND9 and BIND10
- Components:
  - Name server: `named`
  - Remote Name Daemon Control program: `rndc`
  - Client programs: `dig`, `nslookup` and `host`
  - Client libraries associated to DNS server query
- Configuration files:
  - `named.conf`: specifies server configuration (server type, access control…)
  - Text files with the zone’s database