# DHCP and NAT in complex networks

**Arquitetura e Gestão de Redes**

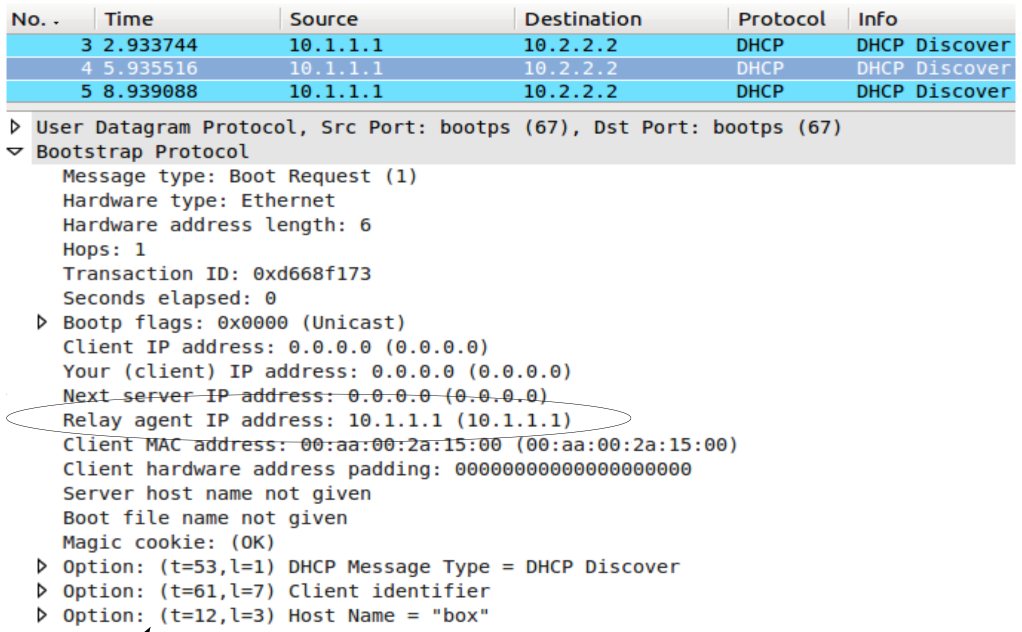
**DETI-UA**

universidade de aveiro

deti.ua.pt

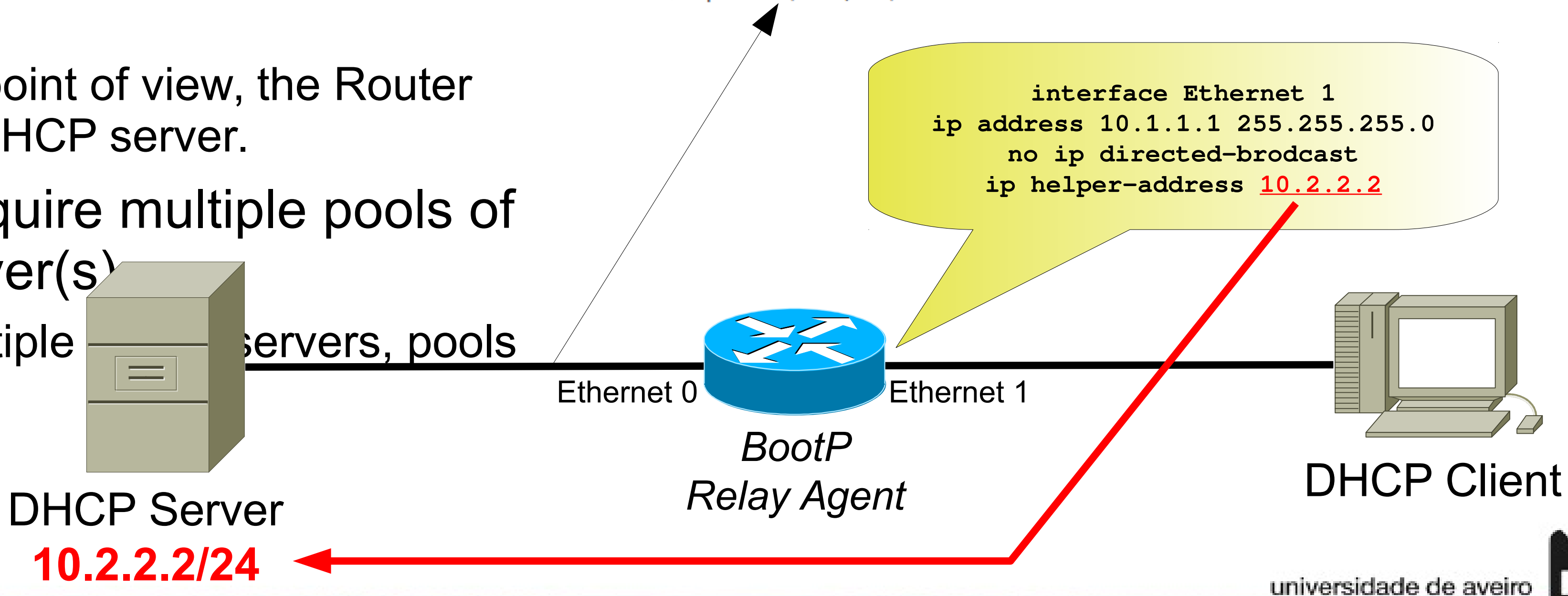
# DHCP in Complex Environments

- In complex network environments where one (or more) DHCP server provide addresses to multiple (V)LAN.
  - Router must have a “BootP Relay Agent” configured and active.
  - Router redirects the client DHCP (broadcast) packets to DHCP server(s) using unicast,
    - Append information of the network/interface where it received the DHCP packet from client.
  - Router redirects server responses to the client.
  - From the client point of view, the Router behaves like a DHCP server.
- Multiple VLAN require multiple pools of addresses at server(s).
  - When using multiple servers, pools must be disjoint.



| No. | Time | Source | Destination | Protocol | Info |
|---|---|---|---|---|---|
| 3 | 2.933744 | 10.1.1.1 | 10.2.2.2 | DHCP | DHCP Discover |
| 4 | 5.935516 | 10.1.1.1 | 10.2.2.2 | DHCP | DHCP Discover |
| 5 | 8.939088 | 10.1.1.1 | 10.2.2.2 | DHCP | DHCP Discover |

```
▷ User Datagram Protocol, Src Port: bootps (67), Dst Port: bootps (67)
▽ Bootstrap Protocol
    Message type: Boot Request (1)
    Hardware type: Ethernet
    Hardware address length: 6
    Hops: 1
    Transaction ID: 0xd668f173
    Seconds elapsed: 0
  ▷ Bootp flags: 0x0000 (Unicast)
    Client IP address: 0.0.0.0 (0.0.0.0)
    Your (client) IP address: 0.0.0.0 (0.0.0.0)
    Next server IP address: 0.0.0.0 (0.0.0.0)
    Relay agent IP address: 10.1.1.1 (10.1.1.1)
    Client MAC address: 00:aa:00:2a:15:00 (00:aa:00:2a:15:00)
    Client hardware address padding: 00000000000000000000
    Server host name not given
    Boot file name not given
    Magic cookie: (OK)
  ▷ Option: (t=53,l=1) DHCP Message Type = DHCP Discover
  ▷ Option: (t=61,l=7) Client identifier
  ▷ Option: (t=12,l=3) Host Name = "box"
```

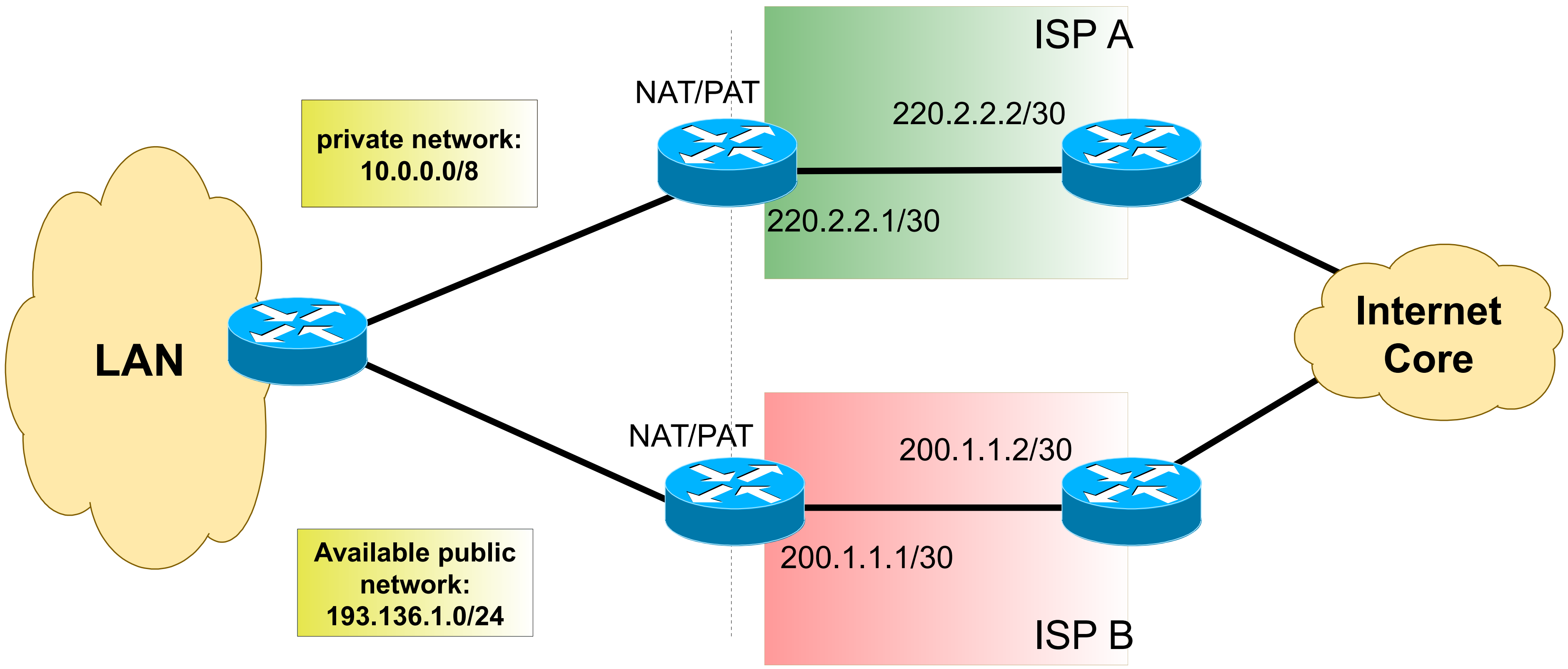


```
interface Ethernet 1
ip address 10.1.1.1 255.255.255.0
no ip directed-brodcast
ip helper-address 10.2.2.2
```

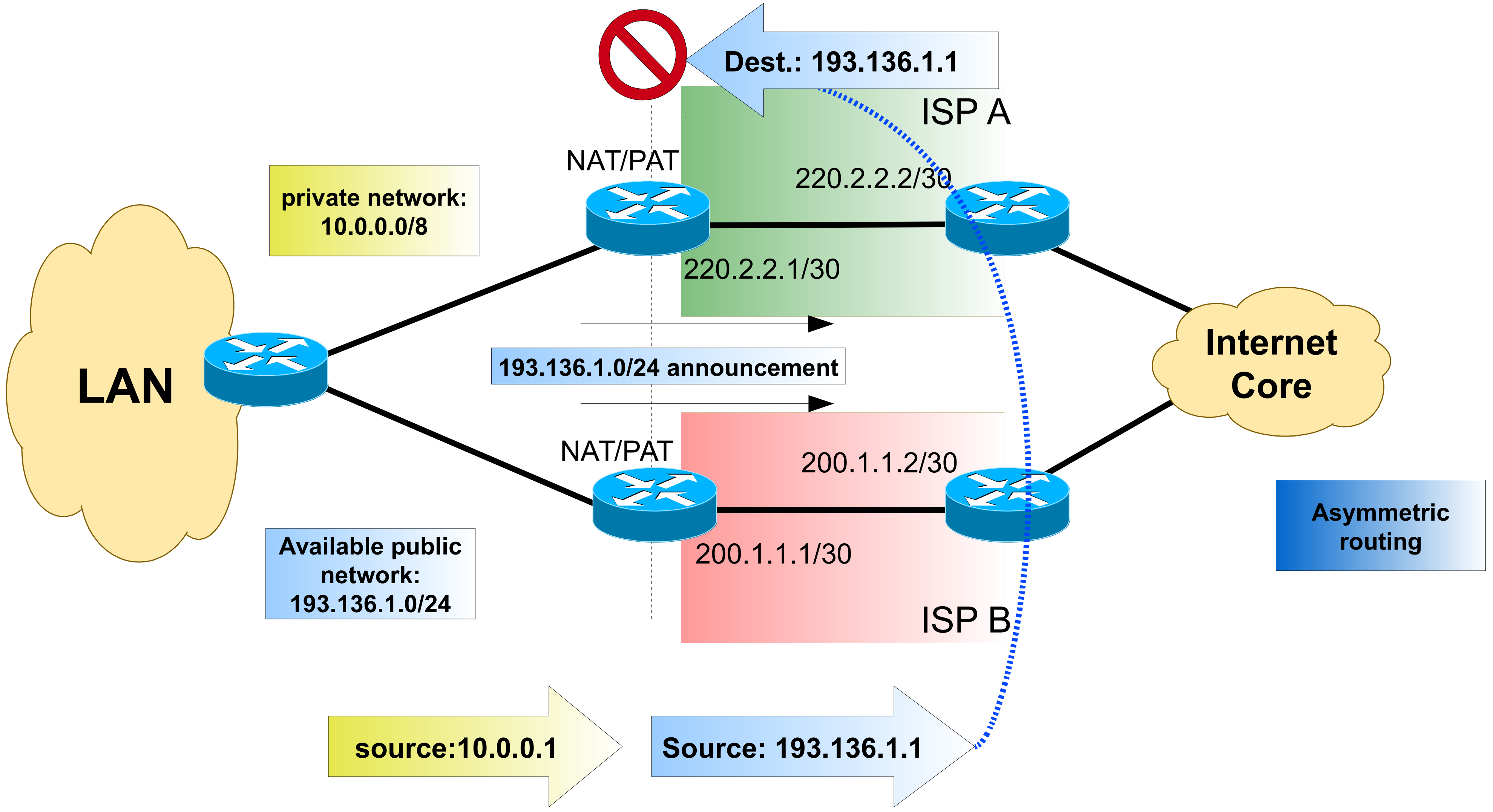
# Static vs. Dynamic Addressing

- Static addressing is many times a must
  - Servers, printers, …
  - Remote management of terminals requires a known static name or address
    - MAC addresses of all equipments must be mapped to an IP address.
  - Only “visiting” equipments get dynamic addresses.
  - This can be extremely complex to manage!
- Alternative: complex management system that reports DHCP dynamic allocations and reconfigures DNS.
- Next Step: DNS... We will come back to this later!

# NAT/PAT with Multiple ISP



# NAT/PAT with Multiple ISP and Asymmetric Routing



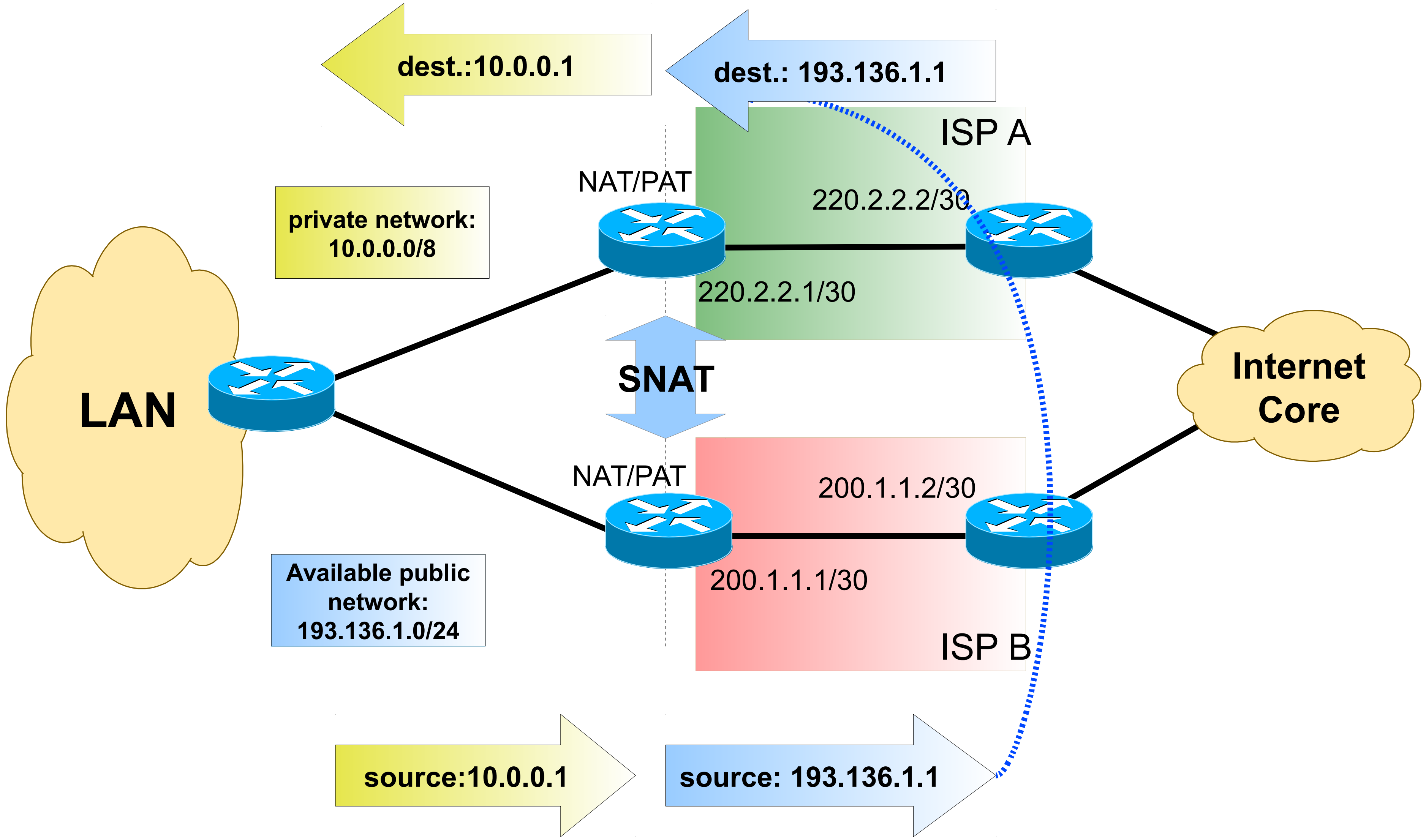
**NAT/PAT tables are not synchronized**

A translation is only known by the router that routed the first packet

# NAT State Synchronization

- Internet Draft proposals (work in progress)
  - NAT State Synchronization Using SCSP (v02 - August 2010)
    - Server Cache Synchronization Protocol (SCSP)
  - Redundancy and Load Balancing Framework for Stateful Network Address Translators (NAT) (v06 – October 2010)
- Cisco Solution (already in Cisco's equipments)
  - Stateful NAT
  - Proprietary solution, details not known
    - Data synchronization over TCP
    - Necessary to define a primary server (other are backup servers)

# Stateful NAT



## Advanced DNS & DNSSEC

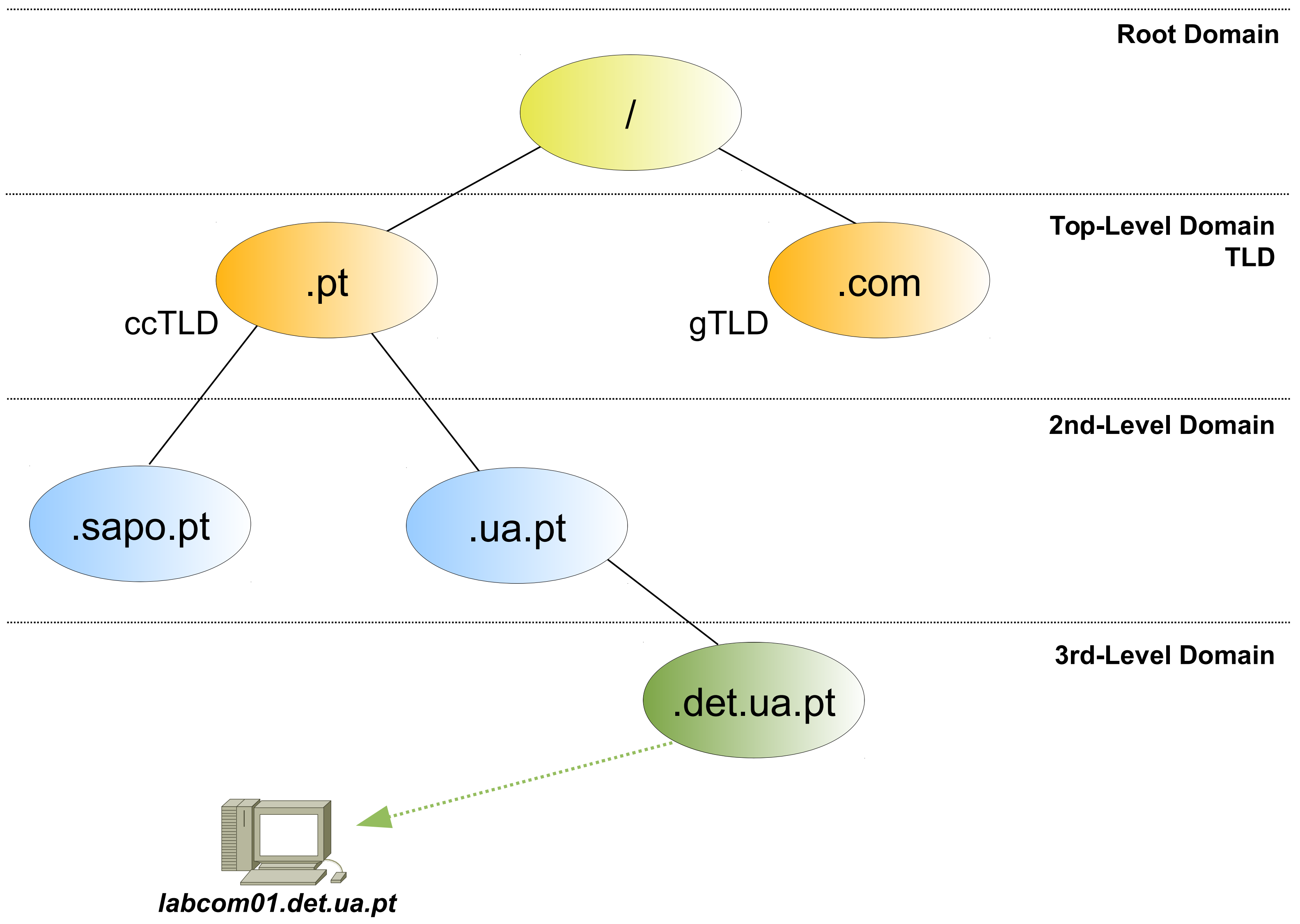
universidade de aveiro

deti.ua.pt

## Domain Name System (DNS)

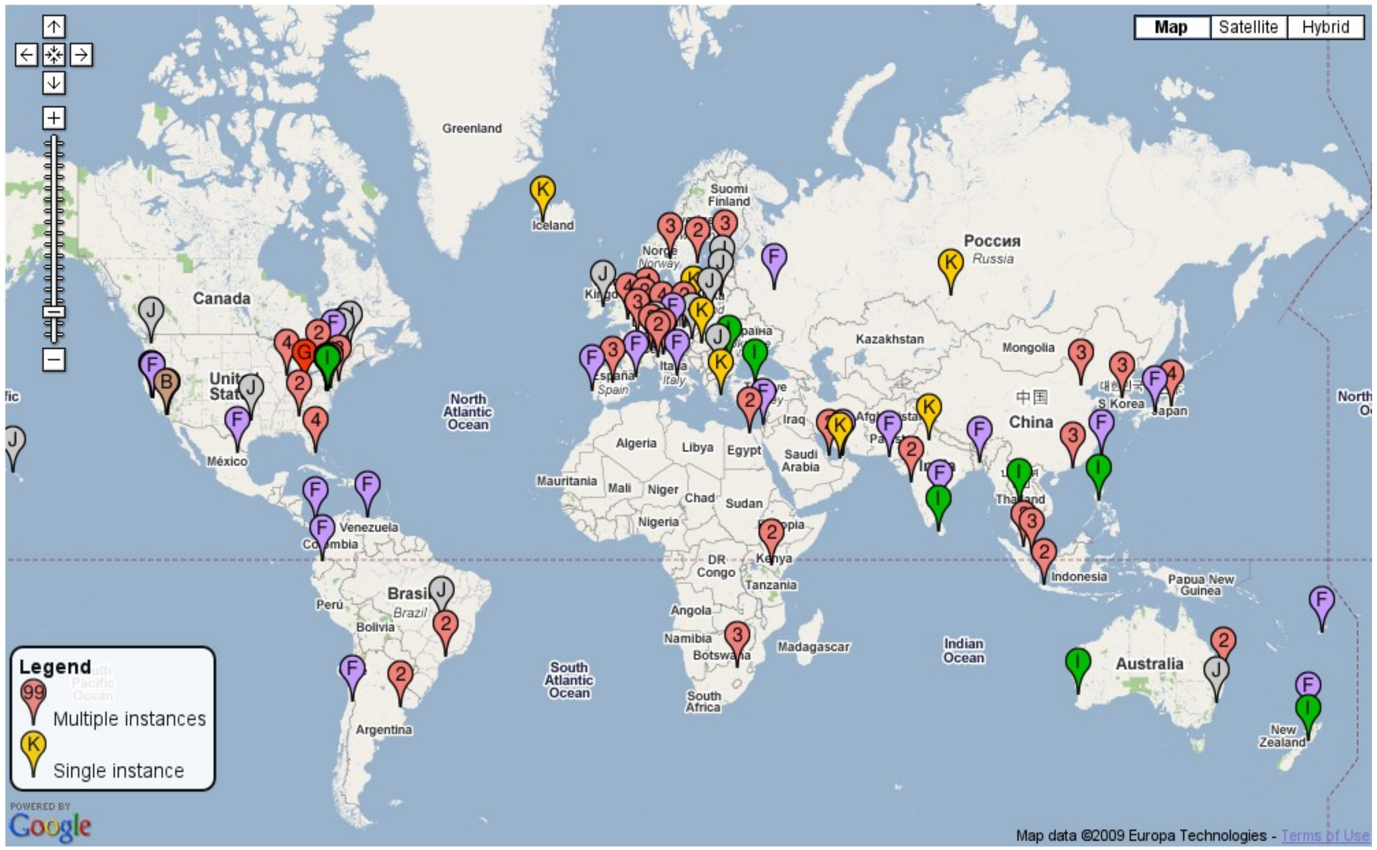
- Distributed database system that facilitates a translation service (resolution) between host names and IP addresses.
- Allows also the translation/resolution between IP addresses and host names
  - The name ”DD.CC.BB.AA.**in-addr.arpa**” allows the resolution of the IPv4 address AA.BB.CC.DD
  - The 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 it has 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 (under approval)
  - **.amazon**, .app, **.apple**, .bank, .bet, .blog, .book, .cars, **.goog**, **.goggle**, .hotel, ...

## TLD Zone Files (sample)

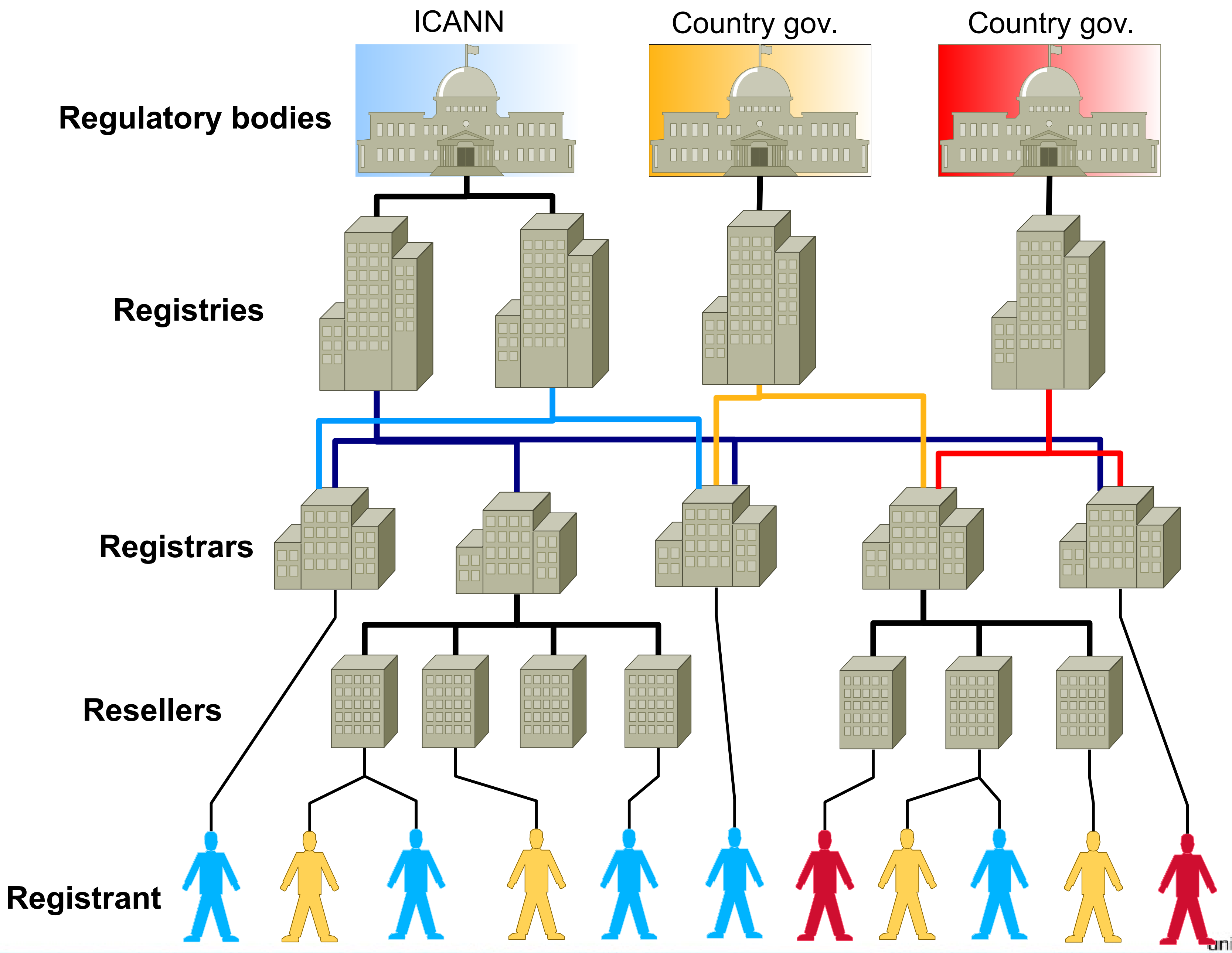
- .ORG (Public Interest Registry)

```
..........................................
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)

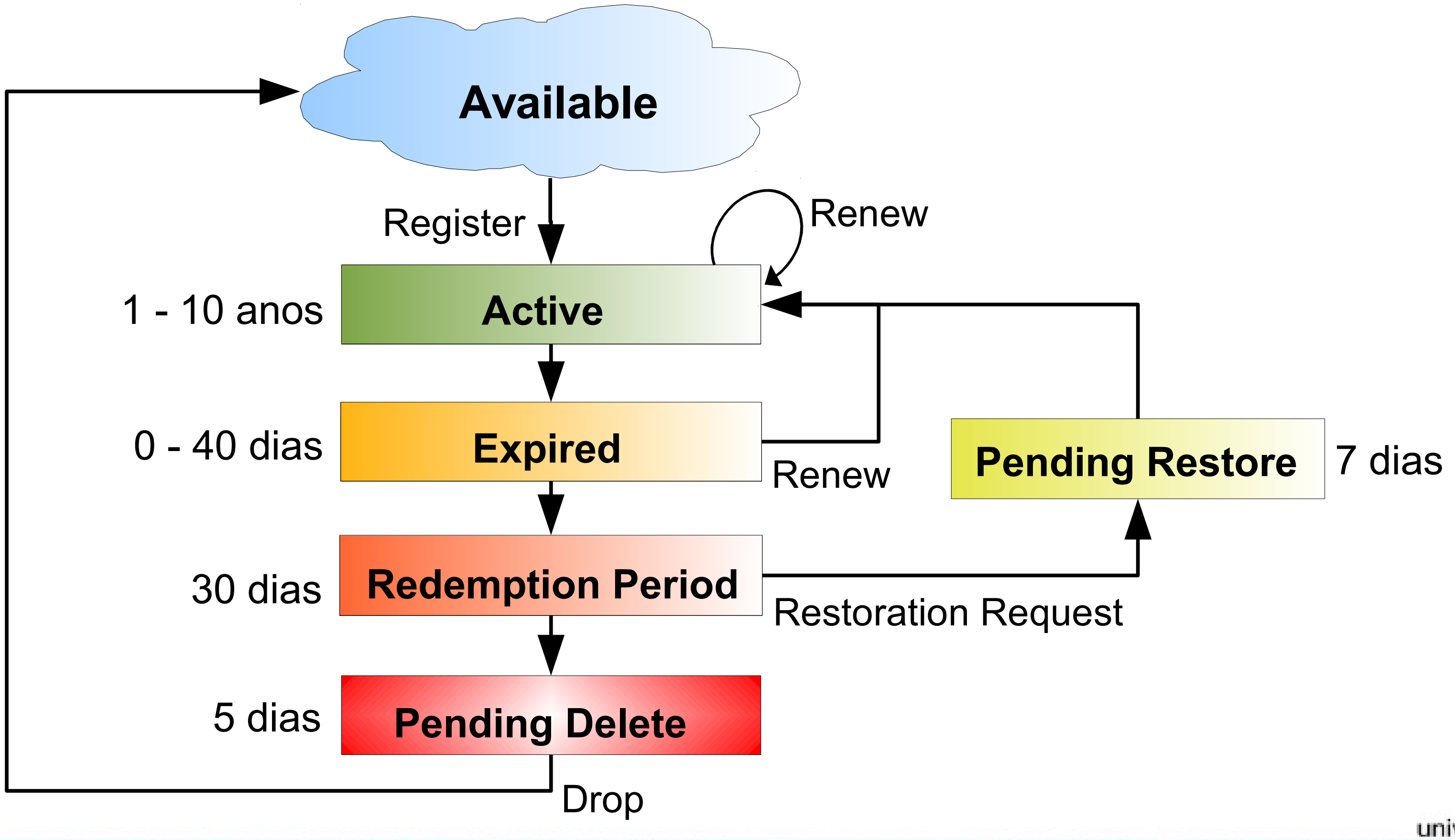


# Domain Management Model (2)

- 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 wants 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, the process of deletion of the domain name from the DNS database is initiated.
  - 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

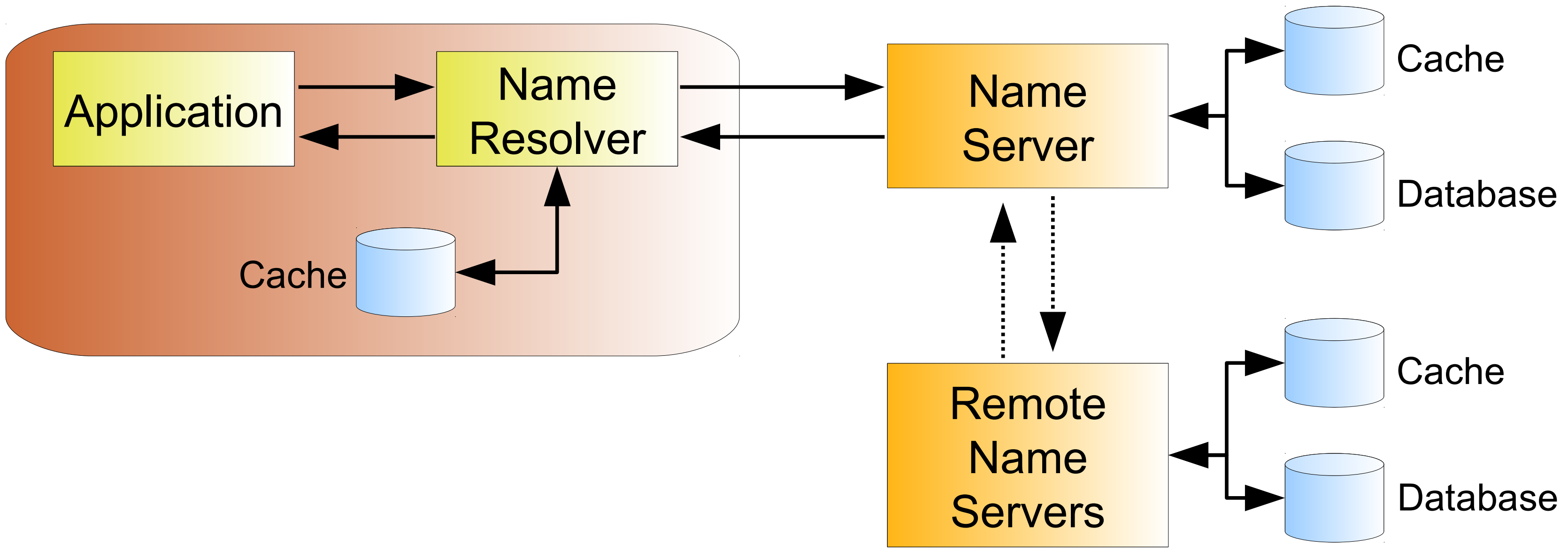
- Contains information about the registrant of a domain
  - Name servers
  - Status of the domain
    - Registry-Registrar Protocol (RPP)
    - Extensible Provisioning Protocol (EPP)
  - Creation, expiration and last update dates.
  - Registrant contacts
    - General
    - Administrative
    - Technical
    - 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



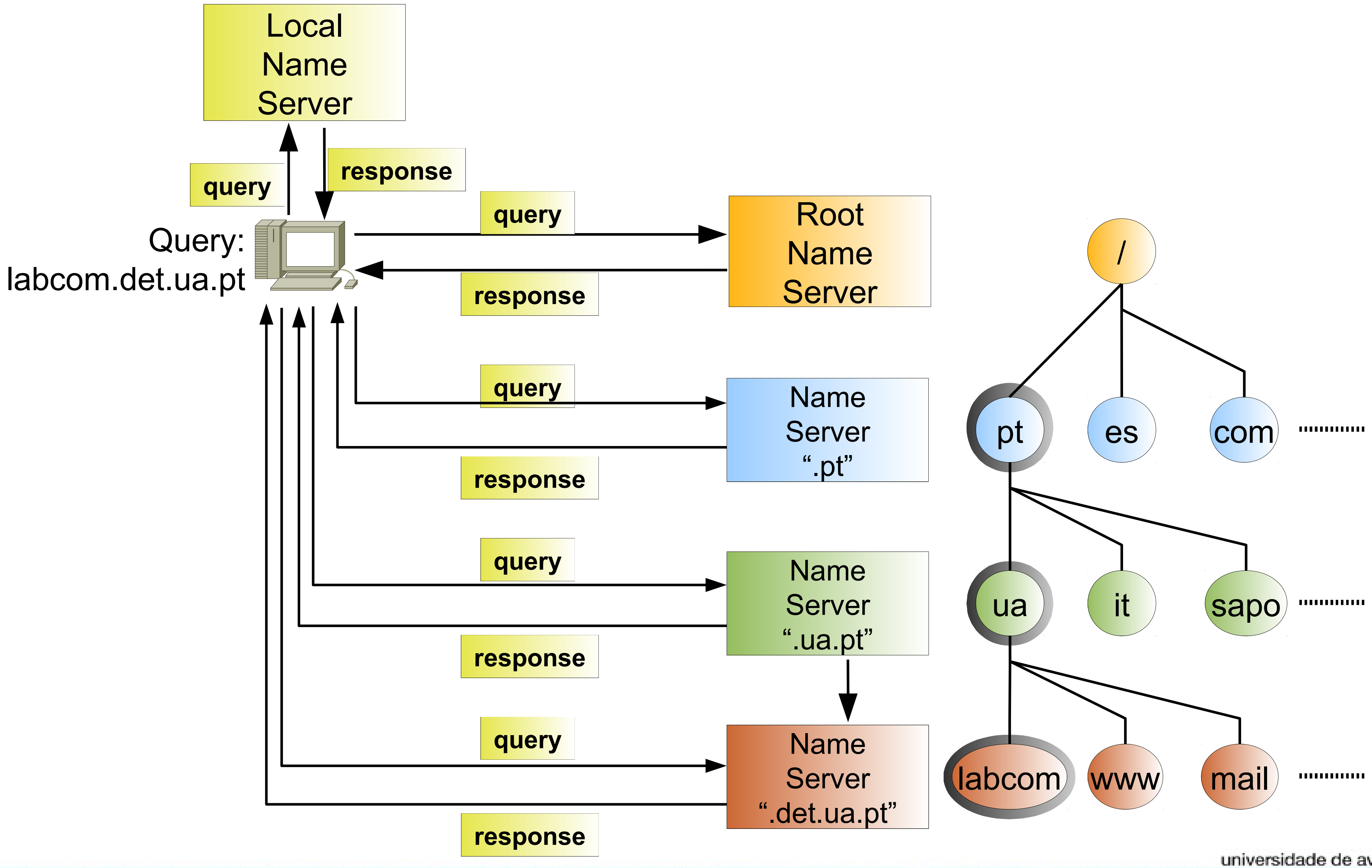
- Received answers are (may be) temporarily stored in cache (have an associated TTL)
  - Can be reused in future queries to speed up answers.
- Cache use improves the system efficiency by eliminating unnecessary external queries.

# DNS Query & DNS Response

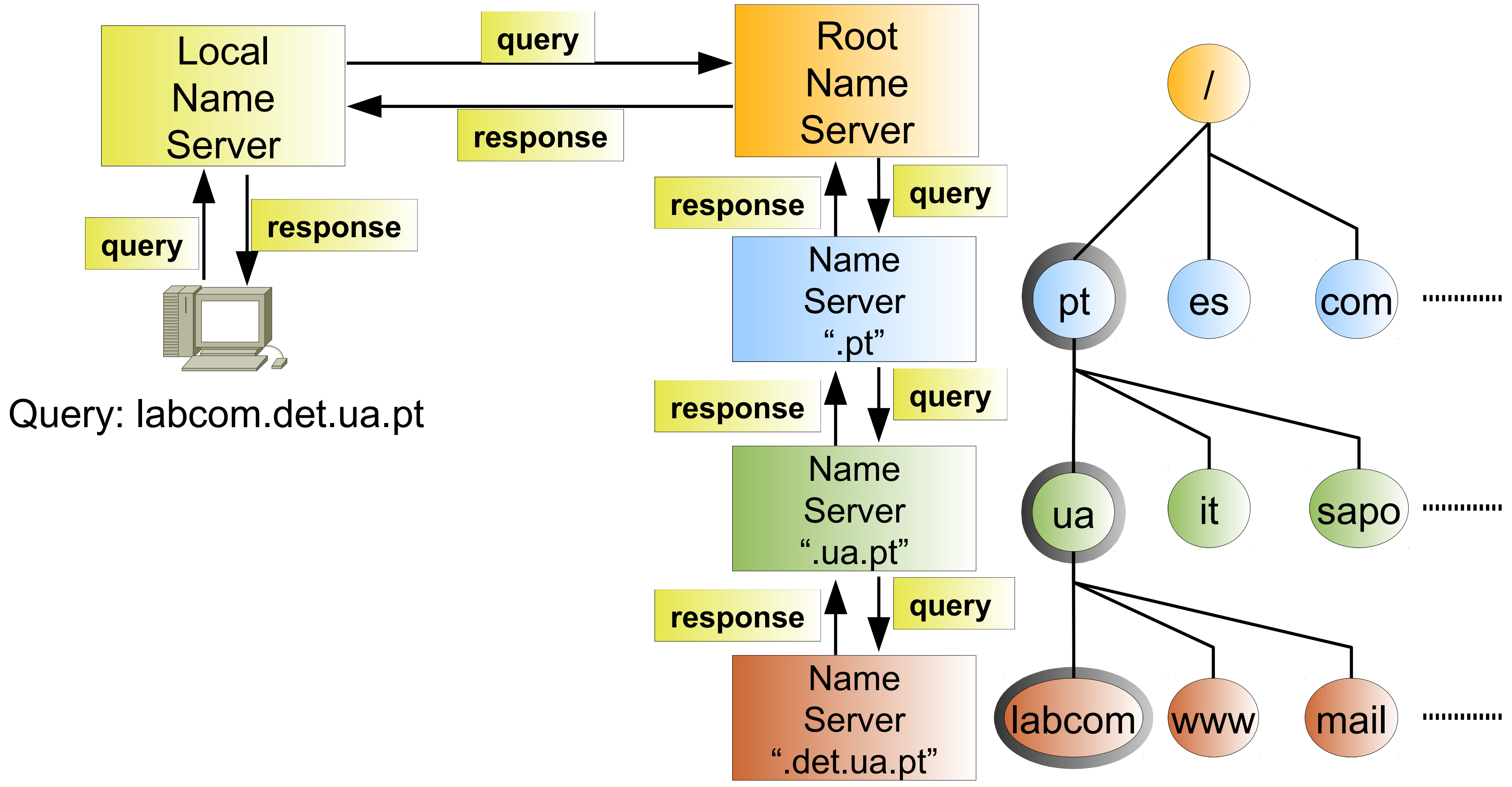
```
Frame 1928 (69 bytes on wire, 69 bytes captured)
Ethernet II, Src: 00:15:f2:9f:38:9d, Dst:
00:60:08:1f:b8:26
Internet Protocol, Src: 193.136.92.160, Dst:
193.136.92.65
User Datagram Protocol, Src Port: 54277, Dst Port: 53
    Source port: 54277 (54277)
    Destination port: 53 (53)
    Length: 35
    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.

## Advanced DNS

universidade de aveiro

deti.ua.pt

## DNS Configuration Types (1)

- Master (Primary) Name Servers
  - A Master DNS server defines one or more zone files for which this DNS is Authoritative.
  - The zone has been delegated (via an NS Resource Record) to this DNS.
  - Master DNS gets its zone data from a local file system.
- Slave (Secondary) Name Servers
  - A Slave DNS gets its zone data using a zone transfer operation (typically from a zone master)
  - Responds as authoritative for those zones for which it is defined to be a Slave and for which it has a currently valid zone configuration.
    - It is impossible to determine from a query result that it came from a zone Master or Slave.
  - A Slave DNS server gets its zone data via zone transfers/updates.
  - In most scenarios a Slave DNS server is updated from the Master DNS server.
    - However, in some scenarios a (Second) Slave DNS server can be updated from a (First) Slave server.
- Caching (Hint) Name Servers
  - A Caching Server obtains information from another server (a Zone Master) in response to a host query and then saves (Caches) the data locally.
    - On a subsequent request for the same data the Caching Server will respond with its locally cached data until the time-to-live (TTL) value of the response expires
      - After any entry expiration the server will refresh the data from the zone master.
    - If the caching server obtains its data directly from a zone master it will respond as 'Authoritative',
    - If the data is supplied from its cache the response is 'Non-Authoritative'.

# DNS Configuration Types (2)

- Forwarding (Proxy) Name Servers
  - A forwarding (Proxy, Client, Remote) server is one which simply forwards all requests to another DNS and caches the results.
  - A forwarding DNS server can be very useful when the access is slow or expensive
    - Reduces external access, speeds up responses and removes unnecessary traffic.
  - Forwarding servers also can be used to ease local administration by providing a single point at which changes to remote name servers may be managed, rather than having to update all hosts.
- Stealth (Split, DMZ, Hidden Master) Name Server
- Authoritative Only Server

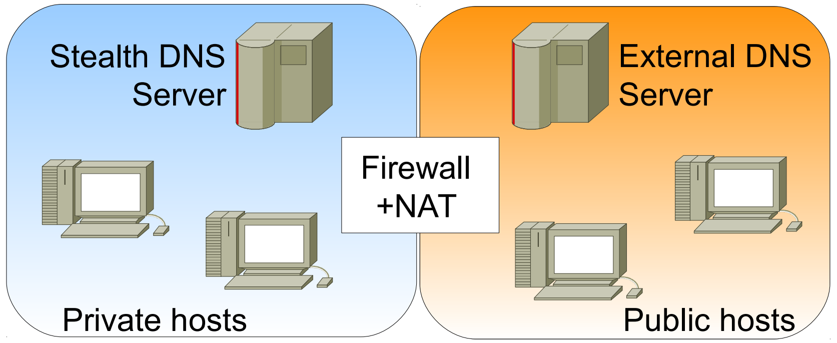
- DNS servers should provide only a single function, for instance, authoritative only, or caching only, not both capabilities in the same server.
  - This may be possible only in larger organizations.
  - Usually, DNS servers run in mixed mode.

# Authoritative Only Server

- The term Authoritative Only is normally used to describe two concepts:
  - The server will deliver Authoritative Responses - it is a zone master or slave for one or more domains.
  - The server will NOT cache.
- There are two scenarios in which Authoritative Only servers are typically used:
  - As the public or external server in a Stealth DNS setup
    - Provides perimeter security.
  - High Performance DNS servers.

```
options {
  directory "/var/named";

  // version statement - inhibited for security (avoids hacking any known weaknesses)
  version "not currently available";

  // disable all recursion - authoritative only – indirectly defines “no caching”
  recursion no;

  // disables all zone transfer requests in this case
  // for performance not security reasons
  allow-transfer{none;};
};
```

# Stealth (Split) DNS Server (1)



- Stealth (or Split) DNS system is one where there is a clear line of frontier between the 'Internal' server(s) and the 'External' or Public DNS servers(s).
  - 'Stealth' Servers will provide a set of services to internal users to include caching and recursive queries and would be configured as a typical Master DNS
  - External server may provide limited services and would typically be configured as an Authoritative Only DNS server.
- The zone files in 'Stealth' servers will contain both public and private hosts.
- 'Public' server's master zone file will contain only public hosts.
- Note: to preserve the 'Stealth' nature it is vital that the Public DNS configuration does not include options with references to the IP of the 'Stealth' server.
  - If not, an attacker could gain more knowledge about the organization

# Zone Updates/Transfers

- Slave (or secondary) DNS servers can 'poll' the Master servers.
  - The time between such 'polling' is determined by the REFRESH value on the domain's SOA Resource Record.
- Zone transfers are carried out using TCP on port 53 whereas normal DNS query operations use UDP on port 53.
- Full Zone Update (AXFR)
  - The Slave sends a query to the Master requesting the latest SOA record
    - If the SERIAL number of this record is different from the current one maintained by the Slave, a zone transfer (AXFR) is requested.
    - It is vital to update the SOA serial number every time anything changes in any of the zone records.
- Incremental Zone Update (IXFR)
  - Allows the Slave and Master to transfer only those records that have changed.
  - The process works as for AXFR.
  - When a Slave requests a Zone Transfer indicates whether or not it is capable of accepting an Incremental Transfer (IXFR).
    - If both Master and Slave support the feature an Incremental Transfer (IXFR) takes place
    - Otherwise, a Full Zone Transfer (AXFR) takes place.
- Notify (NOTIFY)
  - RFC 1912 recommends a REFRESH interval of up to 12 hours.
  - Changes in the Master DNS may not be visible at the Slave DNS for up to 12 hours.
  - In a dynamic environment this may be unacceptable.
  - RFC 1996 introduced a scheme whereby the Master sends a NOTIFY message to the Slave indicating that a change MAY have occurred in the domain records.
  - The Slave on receipt of the NOTIFY will request the latest SOA Resource Record and if necessary will perform a AXFR or IXFR.

## 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 Types

- Master: The server reads the zone data direct from local storage (a zone file) and provides authoritative answers for the zone.
- Slave: A slave zone is a replica of the master zone and obtains its zone data by zone transfer operations.
    - The slave will respond authoritatively for the zone as long as it has valid (not timed out) zone data.
- Hint: The initial set of root-servers is defined using a hint zone.
    - When the server starts up it uses the hints zone file to find a root name server and get the most recent list of root name servers.
- Forward: A zone of type forward is simply a way to configure forwarding on a per-domain or per zone basis.
    - To be effective both a forward and forwarders statements should be included.
- Stub: A stub zone is similar to a slave zone except that it replicates only the NS records of a master zone instead of the entire zone.
- Delegation-only: Provides referrals to DNS servers further down the domain tree and not direct resolution. This zone type was introduced primarily for TLD zones (e.g., .com, .net, etc.).

# BIND – Zone Declaration Examples

```
zone "domain.com" {
        type master;
        file "zones/domain.com";
};

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

zone "example.com" in {
    type slave;
    file "slave.example.com";
    masters {192.168.2.7; 10.2.3.15 port 1127; 2001:db8:0:1::15;};
};
```

# 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 exis
  - This
  - Sma

```
@  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
```

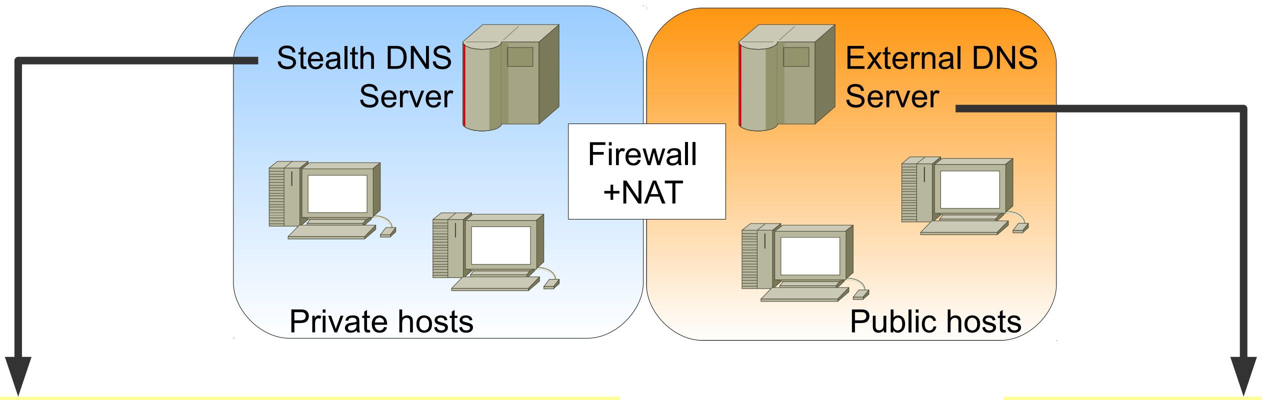
Correct!

```
            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
```

# Stealth (Split) DNS Server (2)



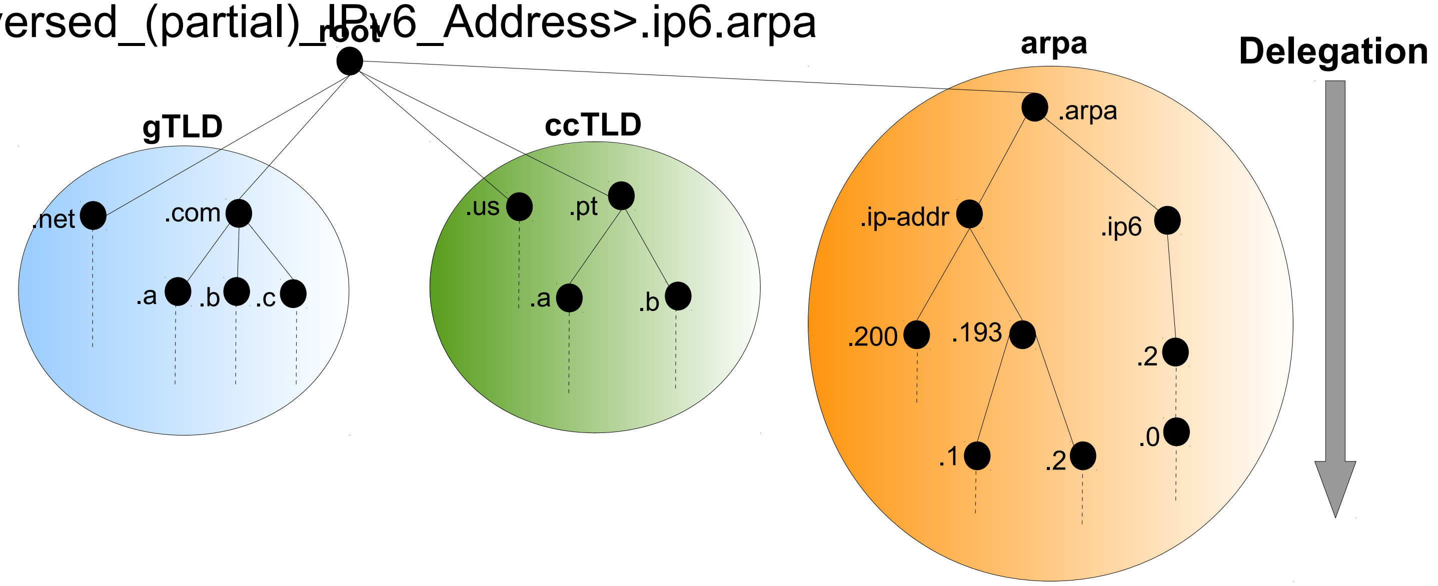
```
example.com.  IN      SOA   ns.example.com. root.example.com. (
                              2003080800 ; se = serial number
                              3h         ; ref = refresh
                              15m        ; ret = update retry
                              3w         ; ex = expiry
                              3h         ; min = minimum
                              )
              IN      NS      ns1.example.com.
              IN      NS      ns2.example.com.
              IN      MX   10 mail.example.com.
; public hosts
ns1           IN      A       193.1.1.1
ns2           IN      A       193.1.1.2
mail          IN      A       193.1.1.3
www           IN      A       193.1.1.4
ftp           IN      A       193.1.1.5
; private hosts
joe           IN      A       192.168.254.6
bill          IN      A       192.168.254.7
fred          IN      A       192.168.254.8
....
accounting    IN      A       192.168.254.28
payroll       IN      A       192.168.254.29
```

```
options {
        directory "/var/named";
        version "not currently available";
        recursion no;
};
```

```
example.com.  IN      SOA   ns.example.com. root.example.com. (
                              2003080800 ; se = serial number
                              3h         ; ref = refresh
                              15m        ; ret = update retry
                              3w         ; ex = expiry
                              3h      ; min = minimum
                              )
              IN      NS      ns1.example.com.
              IN      NS      ns2.example.com.
              IN      MX   10 mail.example.com.
ns1           IN      A       193.1.1.1
ns2           IN      A       193.1.1.2
mail          IN      A       193.1.1.3
www           IN      A       193.1.1.4
ftp           IN      A       193.1.1.5
```

# 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
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.0.0.0         IN      PTR     ns1.example.com.
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.0.0.0         IN      PTR     mail.example.com.
; hosts in subnet ID 2
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.0.0         IN      PTR     joe.example.com.
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.0.0         IN      PTR     www.example.com.
```

## Static Naming vs. Dynamic Addresses

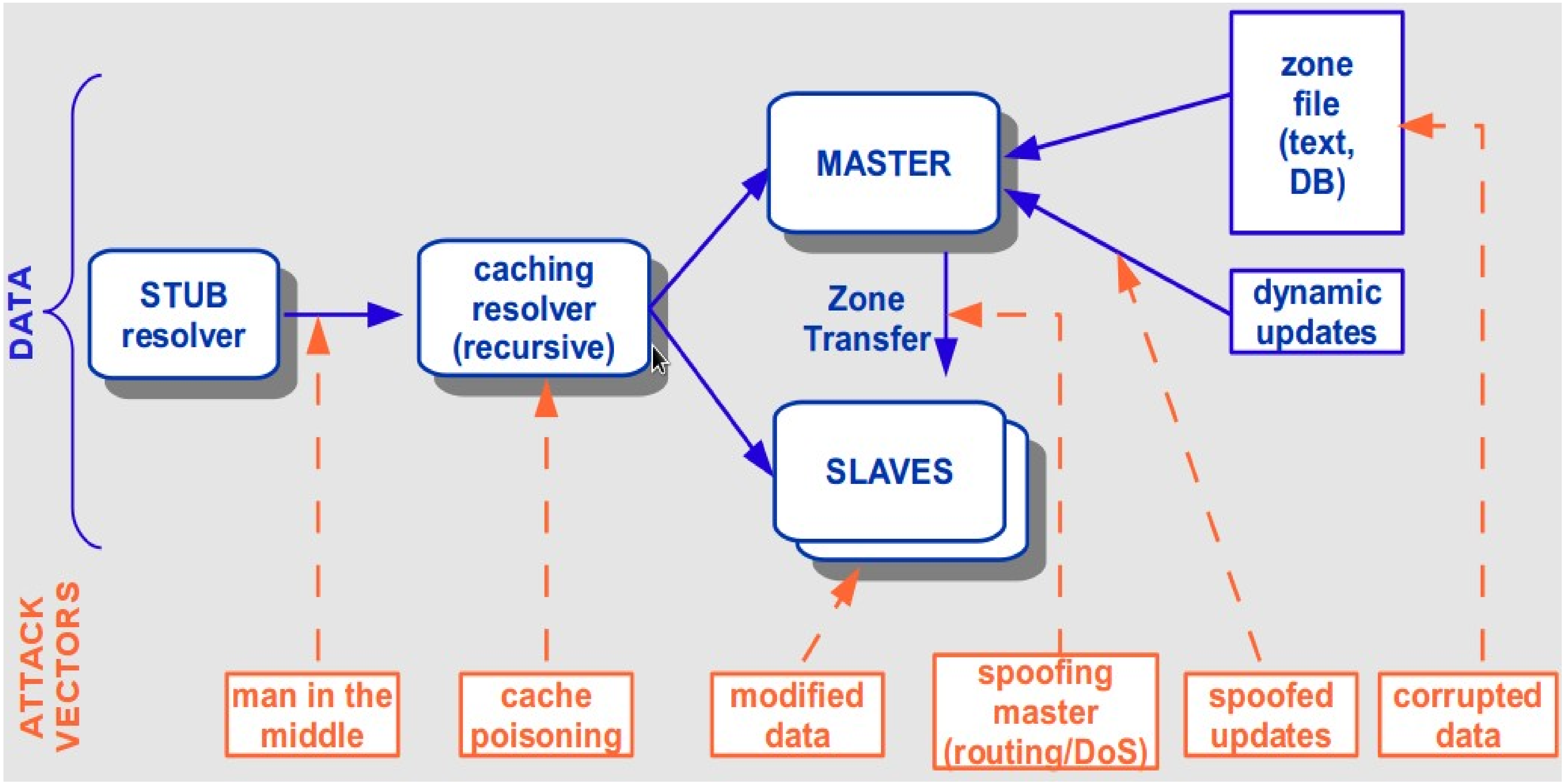
- For management purposes static name is mandatory in complex environments!
  - e.g., floor1printerA, PC1room220, etc...
- Incompatible with dynamic assignment
  - DHCP and Stateless IPv6 Assignment.
- Requires management software to handle simultaneously <u>static naming</u> and <u>dynamic address assignment</u>:
  - 1. Map MAC address to assigned IP address.
    - (with DHCP) Process DHCP logs to map MACs into IPs.
    - (with Stateless IPv6) Retrieve from routers ARP tables (MACs/IPs).
  - 2. Identify location of terminal
    - By MAC address and know location.
    - By retrieving Switching tables of switches (requires traffic from terminal).
  - 1&2 alternative: Terminal reporting to central server (webservice, SNMP,...) upon IP assigment.
  - 3. Rewrite DNS zone files
    - Location ↔ Name ↔ IP

## DNSSEC (DNS Security Extensions)

universidade de aveiro

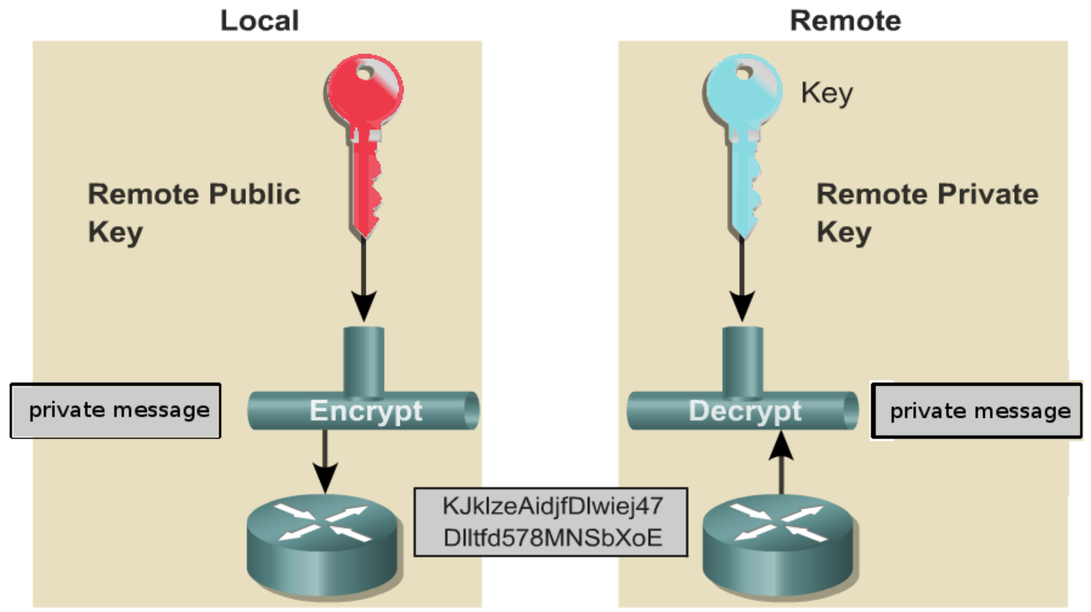
deti.ua.pt

# Points/types of attack



# Public Key Cryptography

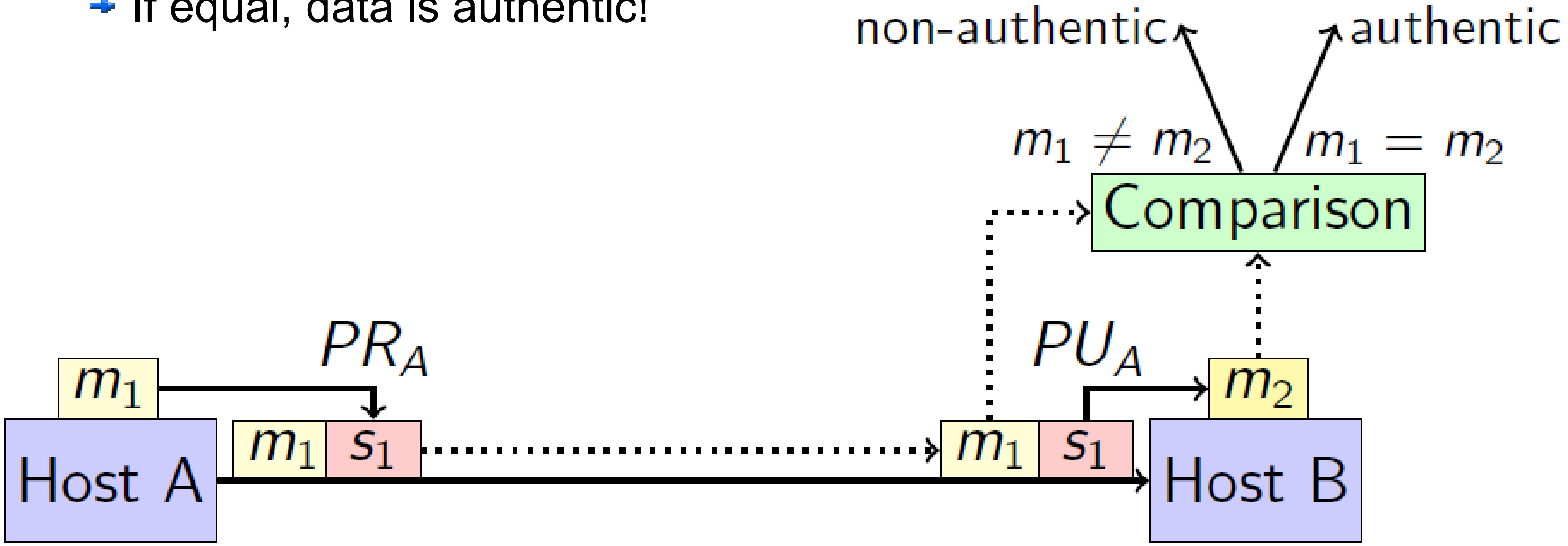
- Public Key Cryptography involves a pair of keys
- **A public key**
  - May be known by anybody, and can be used to encrypt messages, and verify signatures
- **A private key**
  - Known only to the recipient, used to decrypt messages, and sign (create) signatures



- Each public key is published, and the corresponding private key is kept secret
- Is asymmetric because those who encrypt messages or verify signatures cannot decrypt messages or create signatures

# Public Key
# Digital Signatures for Authentication

- To authenticate data from A to B
  - Host A creates a signature by “encrypting” data with Host A private key (PR_A),
  - Host A sends data and signature to host B,
  - Host B verifies the authenticity of data by decrypting signature with Host A public key (PU_A) and comparing the result with the message data.
    - If equal, data is authentic!



# DNSSEC

- DNS Security Extensions are a collection of DNS resource records and protocol modifications that provide source authentication for DNS.
- DNSSEC is based on public key (asymmetrical) cryptography
  - Private key is used to sign DNS data.
  - Public key is published via DNS so that resolvers can retrieve it.
    - The public key is then used to validate the signatures, and thereby, DNS data
- DNSSEC provides cryptographic proof (authentication) that the data received in response to a query is unmodified.
  - DNSSEC protects the system from forged DNS data
- DNSSEC **does not**
  - Encrypt data.
  - Protect your servers from denial of service attacks.
  - Protect user from phishing attacks.

## New Resource Records

- DNSKEY (RFC 4034): DNS public KEY record.
- DS (RFC 4034): Delegated Signer record.
- RRSIG (RFC 4034): Record with SIGnature of a set of records.
- NSEC (RFC 4034): Next SECure record.

# DNSKEY Record

- The DNSKEY record contains the public key (of an asymmetric encryption algorithm) used in zone signing operations.
- DNSKEY records may either define a Zone Signing Key (ZSK) or a Key Signing Key (KSK).
  - Zone Signing Key (ZSK) - used to sign the data within the zone
  - Key Signing Key (KSK) - used to sign the Zone signing key and to create the “Secure Entry Point” for the zone
    - The Secure Entry Point (SEP) flag it is used to distinguish between them during operations.
    - We sign the zone with our ZSK, and we then sign the ZSK with our KSK. The DS record that our parent zone contains, is the fingerprint of our KSK.
- Consists of a Flags field, a Protocol field, an Algorithm field, and the Public Key field.
  - **Flag** field (2 bytes)
    - Bit 7 of the Flags field is the Zone Key flag.
    - If bit 7 has value 1, then the DNSKEY record holds a DNS zone key and CAN be used to verify RRSIG records.
    - If bit 7 has value 0, then the DNSKEY record holds some other type of DNS public key and MUST NOT be used to verify RRSIG records.
    - Bit 15 of the Flags field is the Secure Entry Point (SEP) flag.
    - Typical values: in ZSK 256 (bit 7=1, bit 15=0), in KSK 257 (bit 7=1, bit 15=1).
  - **Protocol** field (1 byte)
    - The Protocol Field MUST have value 3, and the DNSKEY record MUST be treated as invalid during signature verification if it is found to be some value other than 3.
  - **Algorithm** field (1 byte)
    - Identifies the public key's cryptographic algorithm.
    - Determines the format of the Public Key field.
    - 2 for Diffie-Hellman [DH], 3 for DSA/SHA-1 [DSA], 4 for Elliptic Curve [ECC], 5 for RSA/SHA-1 [RSASHA1].

# DNSKEY Record

Resource Record that stores a zone's public key

```
                     1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Flags            |    Protocol   |   Algorithm   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                                                               /
/                            Public Key                         /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

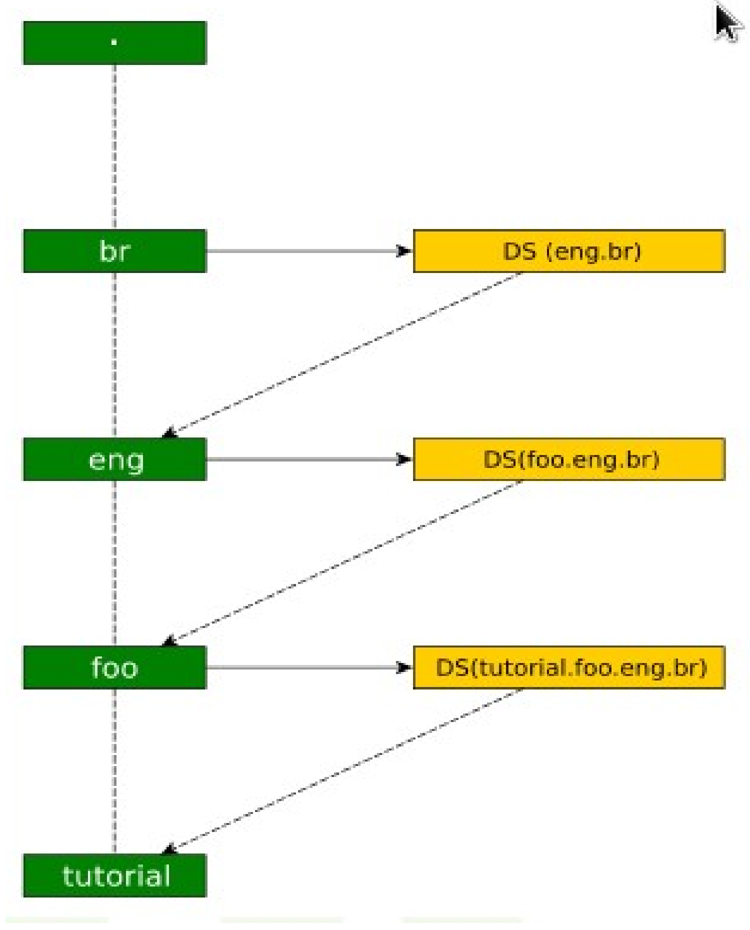
## Example

```
foo.eng.br.           900 IN DNSKEY 256 3 5 (
                          AwEAAeZPN2yMs9q6kgYjFUblEwjCnWWcPq+TGcJrD5ga
                          XXAbP5MAqIkgZ5J4TU1mmpL1A8gMfd/wUmBkVipXR8FK
                          HRajBZSRfgeKnKaQtrxNZ32Ccts2F6Yl v9WaLXtiqebg
                          OZtuJFpQr6pnIt/FoOI+I7BUSMrX28VTq4jXu/qTrmM/
                          ) ; key id = 62745
```

# DS Record

- The DS record refers to a DNSKEY record and is used in the DNSKEY authentication process.
  - Refers to a DNSKEY record by storing the key tag, algorithm number, and a digest (hash) of the DNSKEY record.
  - The digest should be sufficient to identify the public key, storing the key tag and key algorithm helps make the identification process more efficient.
- Used to create a chain of trust
  - The parent zone stores the DS record of a specific child zone
  - The resolver compares the DS record (given by a trusted parent zone) with the hash of the DNSKEY of a child zone.
    - If equal, the DNSKEY (child zone) is trustful.
- Consists of a Key Tag field, an Algorithm field, a Digest Type field, and the Digest field.
  - **Key Tag** field (2 bytes)
    - Contains a tag that identifies the referred DNSKEY. Is calculated based on the content of the DNSKEY record.
  - **Algorithm** field (1 byte)
    - Has the same value of the Algorithm field of the referred DNSKEY record.
  - **Digest Type** field (1 byte)
    - Identifies the algorithm used to calculate the digest (hash).

# DS Record



```
                     1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Key Tag             |  Algorithm    |  Digest Type  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                                                               /
/                            Digest                             /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

**Example**

```
foo.eng.br.          IN DS 817 5 1 EAEC29E4B0958D4D3DFD90CC70C6730AD5880DD3
```

# RRSIG Record

- The RRSIG record contains the signature for a record with a particular name, class, and type.
- Specifies a validity interval for the signature and uses the Algorithm, the Signer's Name, and the Key Tag to identify the DNSKEY record containing the public key that a resolver can use to verify the signature.
- Consists of a Type Covered field, an Algorithm field, a Labels field, a Original TTL field, a Signature Expiration field, a Signature Inception field, a Key tag, the Signer's Name field, and the Signature field.
  - Type Covered field (2 bytes)
    - Identifies the type of the record covered by this RRSIG record.
  - Algorithm field (1 byte)
    - Has the same value of the Algorithm field of the respective DNSKEY record.
  - Labels field (1 byte)
    - Specifies the number of labels in the original RRSIG record owner name.
  - Original TTL field (4 bytes)
    - Specifies the TTL of the covered record as it appears in the authoritative zone.
  - The Signature Expiration (4 bytes) and Signature Inception (4 bytes) fields specify a validity period for the signature. The RRSIG record MUST NOT be used for authentication prior to the inception date and MUST NOT be used for authentication after the expiration date.
    - Default value for Signature Expiration is (Signature time + 30 days).
  - Key tag (2 bytes)
    - Contains the key tag value of the DNSKEY record that validates this signature.
  - Signer's Name field
    - Identifies the owner name of the DNSKEY record that a resolver is supposed to use to validate this signature.

# RRSIG Record

```
                     1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Type Covered           |  Algorithm    |     Labels    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Original TTL                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Signature Expiration                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Signature Inception                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Key Tag            |                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+         Signer's Name         /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                                                               /
/                            Signature                          /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

## Exemplo

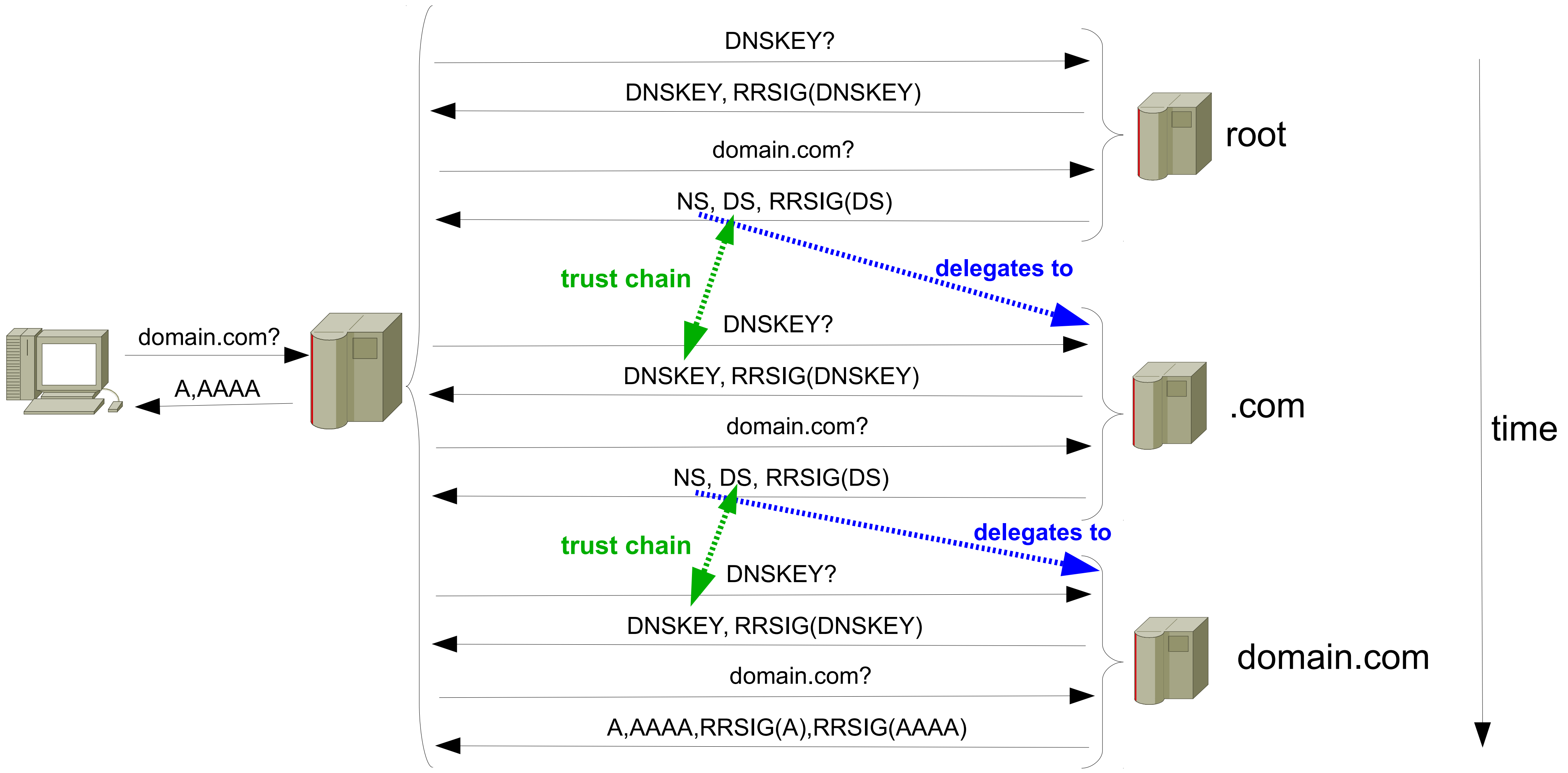
```
foo.eng.br.            900 IN RRSIG SOA 5 3 900 20070617200428 (
                            20070518200428 62745 foo.eng.br.
                            glEeCYyd/CCBfzH64yORAQf9OxYDsI4xuBNaam+8DZQZ
                            xeoSLQEEtwmp6wBtQ7G10wSM9nEjRRhbZdNPNKJMp2PE
                            lLLgLI+BLwdlzOt8MypcpLOaTm9rc7pP7UR5XLzU1k8D
                            m6ePW1bNkId7iOIPSghyoHM7tPVdL2GW51hCujA= )
```

# NSEC Record

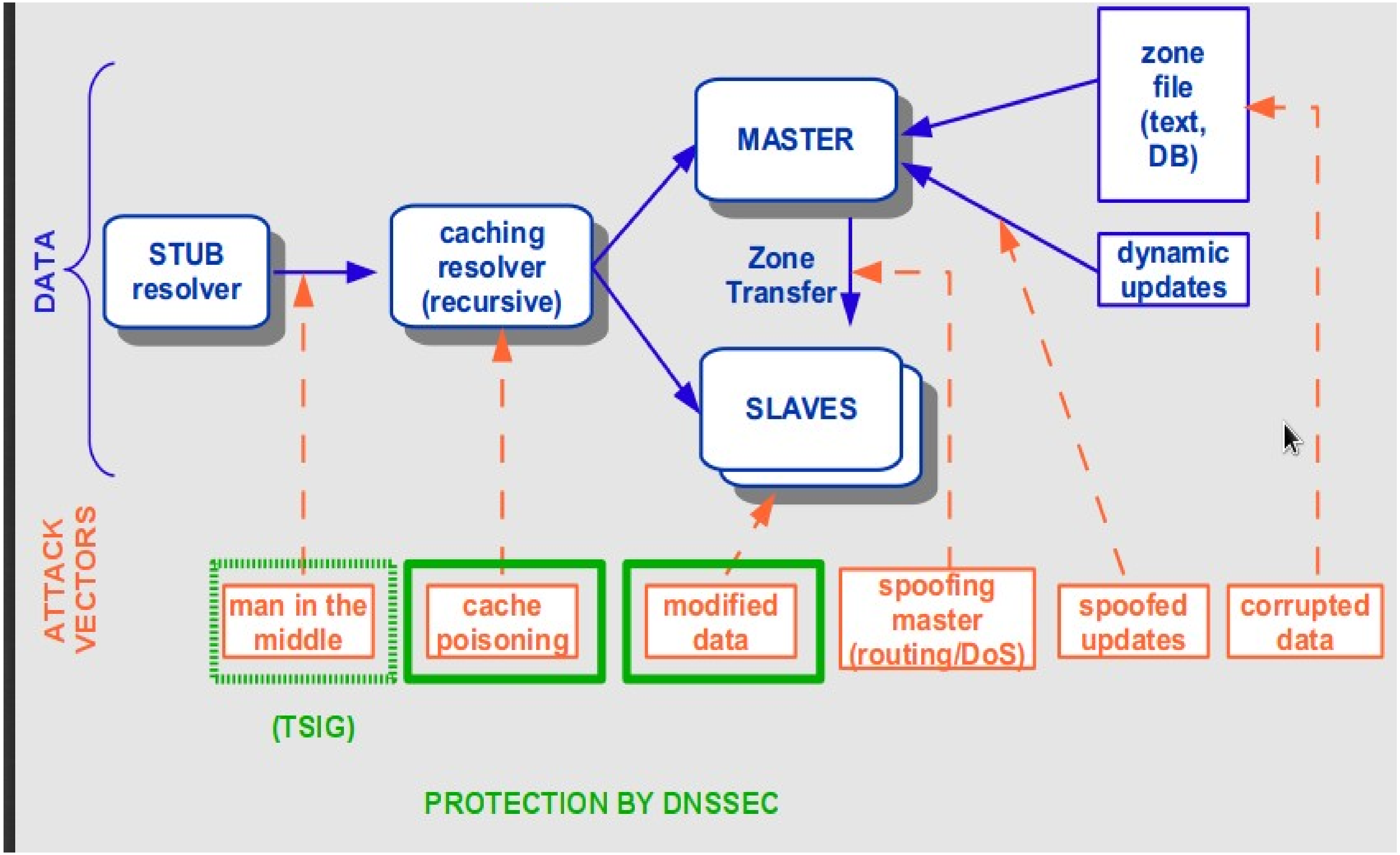
- Points to the next valid name in the zone file and is used to provide proof of non-existence of any name within a zone.
- The NSEC resource record lists two separate things:
  - The next record name (in the canonical ordering of the zone) in the zone that contains authoritative data or a delegation point (NS record).
  - The set of record types that exist for the name of this NSEC record.
- The complete set of NSEC records in a zone indicates which authoritative records exist and also form a chain of authoritative owner names in the zone.
- This information is used to provide authenticated denial of existence of some DNS data (a negative answer).
- To compute the canonical ordering of a set of DNS names, start by sorting the names according to their most significant (rightmost) labels.
  - For names in which the most significant label is identical, continue sorting according to their next most significant label, and so forth.
- For example, the following names are sorted in canonical DNS name order.
  - The most significant label is "example". At this level, "example" sorts first, followed by names ending in "a.example", then by names ending "z.example". The names within each level are sorted in the same way.

```
example
a.example
yljkjljk.a.example
Z.a.example
zABC.a.EXAMPLE
z.example
a.z.example
b.z.example
xxx.z.example
```

# DNSSEC


DNSKEY?
DNSKEY, RRSIG(DNSKEY)
domain.com?
NS, DS, RRSIG(DS)
root
trust chain
delegates to
domain.com?
A,AAAA
DNSKEY?
DNSKEY, RRSIG(DNSKEY)
domain.com?
NS, DS, RRSIG(DS)
.com
time
trust chain
delegates to
DNSKEY?
DNSKEY, RRSIG(DNSKEY)
domain.com?
A,AAAA,RRSIG(A),RRSIG(AAAA)
domain.com

# What does DNSSEC protect?



## Trust Anchors

- DNSSEC is based on chains of trust.
- At the top of chains are "trust-anchors".
  - One (signed) root, one trust-anchor,
  - Until all TLDs are signed, it's not so easy,
  - Trust anchors must be gathered and added to DNS configuration.
- Individual trust anchors do not scale well
- To help solve this problem, ISC created the DLV “Domain Lookaside Validation” record and registry concept.

# Domain Lookaside Validation (DLV)

- DLV (DNSSEC Look-aside Validation) is an extension to the DNSSECbis protocol.
  - It was designed to assist in early DNSSEC adoption by simplifying the configuration of servers.
  - Tries to solve the problem of parent zone servers without DNSSEC support.
    - Without a fully signed trust chain from root to a zone, users wishing to enable DNSSEC-aware server would have to configure and maintain multiple trusted keys.
- DLV records provides an additional entry point (besides the root zone) from which to obtain DNSSEC validation (trust chain).
- DLV records are stored in a central server (DLV Registry).
  - DLV records have the same information has DS records (hash of DNS keys).
- When validating, a resolver looks in the parent zone server for a DS record to validate a DNSKEY,
  - If it does not exist, it performs a query for a DLV record in the DLV Registry.
  - If successful, the DLV Record is used as the DS for that given zone.

## Domain Lookaside Validation (DLV)

- Temporary/transitional model.
- Even if the root or a TLD isn't ready to support DNSSEC and sign its zone, perhaps a trusted third party can collect, authenticate and deliver the required keys.
- Someone attempting to do DNSSEC then has only to configure the DLV server or servers as an anchor of trust, thereafter automatically trusting domains that are anchored/validated via the DLV.
- Assuming you need to trust the data that a DLV Registry secures, you will want to be extremely careful when adding trusted DLV Registries.

# Deploying DNSSEC

- For each zone, two pairs of keys must be created
  - Zone Signing Key (ZSK) - used to sign the data within the zone
  - Key Signing Key (KSK) - used to sign the Zone signing key and to create the "Secure Entry Point" for the zone
  - Operational Practices
    - ZSK should have at least a length of 1024 bits and rolled quarterly.
    - KSK should have at least a length of 2048 bits and rolled every two years.
- Include keys into zone file
- Sign the zone
  - Create a RRSIG for all records
  - Calculate DS records
- Provide parent zone with DS records
  - In the case of a DNSSEC unaware parent, provide DLV Registry with DLV records

## Creating the ZSK

- dnssec-keygen -a RSASHA1 -b 1024 -n ZONE teste.com
  - Uses the RSASHA1 algorithm
  - 1024 bits in length
  - This is a DNSSEC ZONE key
- Generates two files
  - Public key:
    - Kteste.com+005+<keyid>.key
  - Private key:
    - Kteste.com+005+<keyid>.private

```
Private-key-format: v1.3
Algorithm: 5 (RSASHA1)
Modulus: zYWAozVgXzNeDF+7pUnvnCCOcn2zYhtmB4DcbI
5CONbyRYVcIMIaF1PM38pgxv30HpZGXzun39bB/TJhW26h
OiDWLu2mYfqNydWGYtFTn3t4gRz2pXM7CE2D88oW2u4aE
QfSSvRnmDiEKQYt45euOsVloFUgKoFM8xK8ZYi6ws8=
PublicExponent: AQAB
PrivateExponent: V79RLd2jOOJInEXfavHHUMSBrRZIGXITP
M7izaJrWbNEGzSWIhJda2pHBaF1cmCI8uo8P3rL8QDTMFR
rRstUTI2S291y4gk5Mm+iSzcP+7JBZYd8Kie5+NnVBHLL1wa
XlhlxXq5jyGB9S7q+sMtzKvQ32G64ySHSGGir7sqwz7E=
Prime1: 5zx365EUcvkc7T+m7LrBjBHPzXhTgCDD/MwNwbYb
qA/tf86sv5n9BmoUKFm5TjaiTQhlvDGJL1Xp2qchGFgBKQ==
Prime2: 44gLRKnMAVyhoMm49kV1BUjsbHio/qzNWg+NnixFD
nG8ij4BleqOVrqEfBg4W2GiVqbWGU9RnfuEVxYMx4/LNw==
Exponent1: 3j0smFfwimvIFHFHsl/vovp/eN/7iLp1AuLvGc03m
XftcBenyeJq355WT02knno91MGMXapSSg9NWdulICt8uQ==
Exponent2: gEI9D9yjShU8axWrNO/cUjlURKUTplQkgeMIkqQ
j3UuR21+upyKMUCld61H/2stDppLvV18WA/c3F5wvYgMVqw==
Coefficient: 4gP76ppmvGzKjTTX04+8/E+Uhs6PGwudf4VF2TxL
SoJdqJQ2YBhCr2P0cnG1FczS8nX9IWJmkLyKzHWEAYWzEg==
```

```
; This is a zone-signing key, keyid 64575, for teste.com.
; Created: 20120222113328 (Wed Feb 22 11:33:28 2012)
; Publish: 20120222113328 (Wed Feb 22 11:33:28 2012)
; Activate: 20120222113328 (Wed Feb 22 11:33:28 2012)
teste.com. IN DNSKEY 256 3 5 AwEAAc2FgKM1YF8zXgx
fu6VJ75wgjnJ9s2IbZgeA3GyOQjjW8kWFXCDC GhdTzN/K
YMb99B6WRl87p9/Wwf0yYVtuoaIg1i7tpmH6jcnVhmLRU597
eIEc9qVzOwhNg/PKFtruGhEH0kr0Z5g4hCkGLeOXrjrFZaB
VICqBTPMS vGWIusLP
```

# Creating the KSK

- dnsseckeygen a RSASHA1 b 2048 n ZONE f KSK teste.com
  - Uses the RSASHA1 algorithm
  - 2048 bits in length
  - This is a DNSSEC ZONE key
  - Has the Secure Entry Point (KSK) bit set
- Generates two files
  - Public key:
    - Kteste.com+005+<keyid>.key
  - Private key:
    - Kteste.com+005+<keyid>.private

```
Private-key-format: v1.3
Algorithm: 5 (RSASHA1)
Modulus:
26Reg1cwwnQboUi9bf2mz/h2bu4oUspJ0iN1mmDBcZ2HLQ6...
PublicExponent: AQAB
PrivateExponent:
LnliEjJhu9NrgT3t7xcLs9ej36b+2z24TwF3wdmVNOAbGRq...
Prime1:
8nGTcuM+UaC5w/ge4Ewt/O+59VBmsEvg91dNa964zox4zBW...
Prime2:
5+xnOh/bQ3GBMVHz2ZpzYI7XDzg7GYJk7BbEY0kdDYx6qkk...
Exponent1:
TObh0mtqdRH6WsL1aEhBvh18aufZ6snmzg4PLMw06q98EaB...
Exponent2:
5JuKRSkRoLFJf6wgieZMxGkIY+AxoTt+75ihjJyNHsXSQ/h...
Coefficient:
z7+JRfh0/d3AimGosXHFqrSKLiH8dlmfppho3TRrdTHfK8j...
Created: 20120222114632
Publish: 20120222114632
Activate: 20120222114632
```

```
; This is a key-signing key, keyid 2368, for teste.com.
; Created: 20120222114632 (Wed Feb 22 11:46:32 2012)
; Publish: 20120222114632 (Wed Feb 22 11:46:32 2012)
; Activate: 20120222114632 (Wed Feb 22 11:46:32 2012)
teste.com. IN DNSKEY 257 3 5 AwEAAdukXoNXMMJ0G6FIvW39ps
/4dm7uKFLKSdIjdZpgwXGdhy0OrkSO Bivs1JyK+YIH2GnIcVTgE7Im
q1KD8Xh1gkz2lD2OLQu3rbVB/P/WXGN4mHQEM8DXGG15rnkWvoDFcra
3ebEjisgVEkLT591bDVpDIAtAn1jxnFx8 qYsxOIBca96FmgeifcZTm
CHkjXnu6bOwAZava6/H+cVNCWrYNvZa5QwS GmirrANMrafN2yBOkzo
luF+Ppr5nY6iOWaO787JkxdKJoCqcj5GOETRq U3qWv1MTbrEqcWYgg
K0NuFaPYPSEBwR37a2zNUSJ2AdJqlSqA/AWV3tb EZyBNWAIxHM=
```

# Zone File Changes

- Add the public portions of both KSK and ZSK to the zone to be signed.
  - cat Kteste.com+005+*.key >> db.teste.com

```
@       IN      SOA     teste.com. adm.teste.com. (
                        199609206       ; 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
...
ws-177201       A       206.6.177.201
; This is a key-signing key, keyid 2368, for teste.com.
; Created: 20120222114632 (Wed Feb 22 11:46:32 2012); Publish: 20120222114632 (Wed Feb 22 11:46:32 2012) ; Activate: 20120222114632
(Wed Feb 22 11:46:32 2012)
teste.com. IN DNSKEY 257 3 5
AwEAAdukXoNXMMJ0G6FIvW39ps/4dm7uKFLKSdIjdZpgwXGdhy0OrkSOBivs1JyK+YIH2GnIcVTgE7Imq1KD8Xh1gkz21D2OLQu3rbVB/P/WXGN4mHQEM8DXGG15rnkWvoDFcra
3ebEjisgVEkLT591bDVpDIAtAn1jxnFx8qYsxOIBca96FmgeifcZTmCHkjXnu6bOwAZava6/H+cVNCWrYNvZa5QwSGmirrANMrafN2yBOkzoluF+Ppr5nY6iOWaO787JkxdKJoC
qcj5GOETRq U3qWv1MTbrEqcWYggK0NuFaPYPSEBwR37a2zNUSJ2AdJqlSqA/AWV3tb EZyBNWAIxHM=

; This is a zone-signing key, keyid 64575, for teste.com.
; Created: 20120222113328 (Wed Feb 22 11:33:28 2012); Publish: 20120222113328 (Wed Feb 22 11:33:28 2012) ; Activate: 20120222113328
(Wed Feb 22 11:33:28 2012)
teste.com. IN DNSKEY 256 3 5
AwEAAc2FgKM1YF8zXgxfu6VJ75wgjnJ9s2IbZgeA3GyOQjjW8kWFXCDCGhdTzN/KYMb99B6WRl87p9/Wwf0yYVtuoaIg1i7tpmH6jcnVhmLRU597eIEc9qVzOwhNg/PKFtruGhE
H0kr0Z5g4hCkGLeOXrjrFZaBVICqBTPMS vGWIusLP
```

# Signing the Zone File

- dnssec-signzone -g -l dlv.isc.org -o teste.com -N INCREMENT db.teste.com

```
teste.com.      86400 IN SOA    teste.com. adm.teste.com. (
                          199609207   ; serial
                          28800       ; refresh (8 hours)
                          7200        ; retry (2 hours)
                          2419200     ; expire (4 weeks)
                          86400       ; minimum (1 day)
                          )
                86400 RRSIG SOA 5 2 86400 20120323112617 (
                          20120222112617 64575 teste.com.
                          H4nA373UPtXKqeaY73mmuLfaAvugr6Bo1/st9udF4ogCLfiW2riS/+1DqiEECes654Llc3li
                          536wPTWLRPyB+hH1f5IS3JdfAbhTO4Gcwgb/HOv0G+tgzQ/NcPOKO9ipkC+dvdO/TBsbHgE
                          zPMzEUKSlfcv6EUC5ctCbPoEYX0= )
                86400 NS    ns1.teste.com.
                86400 NS    ns2.teste.com.
                86400 RRSIG NS 5 2 86400 20120323112617 (
                          20120222112617 64575 teste.com.
                          pHg4bzdsfh9kY9LW+/uppHokX7Gb0pxmU306IJw4/JsMAhDCr3M4TvEjJXFE9C4ikqWfL2Rb
                          rSx5+ZtQIXHgZtGjGsng3r/7RgdJ575LhOpBSgwNYTS2sTvWGiKqoR/26J6LyI1EtQ
                          /E9PBU6BZwBqPTVkUppJ9qOtjUpApiUzg= )
                86400 A     206.6.177.2
                86400 RRSIG A 5 2 86400 20120323112617 (
                          20120222112617 64575 teste.com.
                          Hc5ugtZ7tR1VgUz2SDXRWoWPjdENNb92DPgYX/W/3wvyjL0OKJORQaWTHBuCxbGqDm+kYsg
                          jbKMFC9kfnBX8MNLyhQx3xNDDw6CSjGY8cd7ONRE28KICwxTjXLkJUDXxTBuoRVrAF86Re8ol
                          T+pUEjAw+CxB8OK+xpCBFd3UV+Y= )
                86400 MX    10 teste.com.
                86400 RRSIG MX 5 2 86400 20120323112617 (
                          20120222112617 64575 teste.com.
                          G2x5SlnwAVCNGk/O+HrFa+ltBQ/t4SUYzn0rU1c2RkZtu4mlVAB5B0Dv0pq6ghbVbAiEBGZV
                          DZrLTsmIvrEp/RoHDvpyArz97ah6vR+WDArEKrwFV6Bqhhzsb/bu7BcHg1IjVvfGW/8JGzFE
                          74+TsJbkvInstcktFkbI7DFTsOQ= )
                ..........................
```

## DS and DLV Records

- DS records to be exported to parent zone
  - Hash of each DNSKEY (KSK and ZSK)

```
teste.com.        IN DS 2368 5 1 A29EA609B1D12A8E04D6AFE5636ED37BF3CA6EA5
teste.com.        IN DS 2368 5 2 868F4A8BFE2D5C096F2FB7A469A30A9D9B4848395
                  84C50495AE6E408 7830255E
```

- Your parent zone must now insert the DS Records to create a chain-of-trust.
  - Procedures will differ between, but this must be done securely.

- DLV records (same as DS records) to be exported to ISC DLV Registry
  - Easy upload process (via web @ dlv.isc.org).

```
teste.com.dlv.isc.org. IN DLV 2368 5 1 A29EA609B1D12A8E04D6AFE5636ED37
                       BF3CA6EA5
teste.com.dlv.isc.org. IN DLV 2368 5 2 868F4A8BFE2D5C096F2FB7A469A30A9
                       D9B48483958 4C50495AE6E408 7830255E
```

## Periodic Maintenance Issues

- Signatures have lifespans
  - Expired signatures lead to zones that will not validate!
- Every time time you modify a zone or at least before the Signature Expiration of RRSIG records (minus TTL) it is necessary to re-sign the zone.
  - 30 days is the default RRSIG record validity period.
- Keys need to be rotated
  - The longer a key is in public view, the more likely it is to be compromised
- Automation exists! :-)