

Internet Infrastructure

K. U. Leuven 2005-2006

Domain Naming System

Prof. A. Mariën

Domain Naming System

Domain Naming System	2
Introducing names: local configuration files.....	2
Network information systems	3
Domain Naming System	3
Producers and consumers	3
Definition of DNS	3
Domain name syntax	4
Hierarchical namespace.....	4
Distributed responsibility	5
Choice of names	5
Domain name and ISP	5
Naming management.....	6
Domain structure	6
Zone.....	6
DNS performance.....	7
Other type of requests	7
Query options	8
DNS carrier	9
DNS server cooperation	9
Root name servers	9
Set up primary DNS server	9
Secondary DNS server	10
Internal – external DNS.....	10
Tools for DNS querying.....	10
Nslookup	10
Exercise	11
DNSSecure	12
Mechanism: signatures.....	12
Signatures	12
Trust	12
NOT FOUND authentication	12
Multiple keys.....	13
KEY RR	13
References	14

Introducing names: local configuration files

IP addresses consist of 4 digits. Most people can remember a few of these, but in general people prefer names instead of numbers.

Initial system solutions to use names had to be configured host per host. This option still exists, even if it is mostly used for either fail-back or security reasons. On both Unix and Windows systems the file is called “hosts”. The file is a property file linking names with hosts. The following is the default on some systems:

```
#      102.54.94.97      rhino.acme.com      # bronserver
#      38.25.63.10      x.acme.com          # x clienthost
127.0.0.1      localhost
```

Each host has to have a copy, and any changes must be propagated to all the systems. When the Internet contained few hosts, and most systems had a limited set of other hosts with which they interacted, this work more or less. Nowadays it is unthinkable to work like that again.

Network information systems

Dynamic host configuration gives a solution for the configuration on the network level. Such configurations are local. Initially, the main concerns for name resolution were also local. Therefore it is logical that the first solutions for both problems were similar.

The main problem is the maintenance of a copy per host of all the information, in other words: a change management problem. To address this problem, the solution was the use of a (LAN) network information service.

As an example, Yellow pages (NIS) uses a set of programs to query a Network Information Server. The most important commands are:

- `yycat` : shows the values of all the keys present in the NIS map
- `yymatch` : shows the values of one or more keys present in an NIS map
- `yypwhich` : returns the name of the NIS server.

Domain Naming System

Up to now, we have worked with technical addresses: IP or MAC. These addresses are great for machines and the whole Internet runs on top of them.

But people prefer names. It is much easier to remember that the webserver for company WeirdRus is www.weirdrus.com than to remember the IP address. Second, IP addresses may change.

Applications prefer a name-based interface too. Example: `URLConnection` vs. `socket` interface. This higher level interface makes the program a lot easier to adapt to different situations and changes.

The idea behind DNS is to replace hardware-like addresses by a nice, logical name space. The names should be independent from the underlying network. It should be perfectly possible to use a totally different naming system without any network impact.

Producers and consumers

In DNS you have two types of participants: managers of information and consumers of information, or, information providers and consumers.

The managers of information come in two types: the registration authorities and the information servers. The registration authorities control the name to IP address mapping. The information servers run name services software, like `Bind`.

The consumers of information are the workstations or programs that either want to do a name look-up, a name to IP translation, or a reverse look-up, an IP to name translation.

Definition of DNS

- RFC 1034: STD 13: Domain names - Concepts and Facilities (note: November 1987!)
- RFC 1035: STD 13: Domain Names - Implementation and Specification
- RFC 2065: Domain Name System Security Extensions

- RFC 2181: Clarifications to the DNS Specification

Domain name syntax

Dns = <name> <domain>

Domain = [“.” <name>] <parent domain>

Parentdomain = <domain> | “”

The names are hierarchical, with the top level domain at the rightmost end. The address:

www.cs.kuleuven.ac.be

can be read as: the system “www” in the domain “cs” in the domain “kuleuven” in the domain “ac” in the top level domain “be”.

Hierarchical namespace

The name space is a forest with a limited number of trees. The trees correspond to the top level domains of DNS. There are two types of trees:

- US names: .gov, .mil, .edu, .org, .com, .net, .int
- ISO country code names: .be, .fr, .uk, .ca, ...

The .com root looks like this:

```
com
    primary name server = a.gtld-servers.net
    responsible mail addr = nstld.verisign-grs.com
    serial = 1077298072
    refresh = 1800 (30 mins)
    retry = 900 (15 mins)
    expire = 604800 (7 days)
    default TTL = 900 (15 mins)

com    nameserver = J.gtld-servers.net
com    nameserver = K.gtld-servers.net
com    nameserver = L.gtld-servers.net
com    nameserver = M.gtld-servers.net
com    nameserver = a.gtld-servers.net
com    nameserver = B.gtld-servers.net
com    nameserver = C.gtld-servers.net
com    nameserver = D.gtld-servers.net
com    nameserver = E.gtld-servers.net
com    nameserver = F.gtld-servers.net
com    nameserver = G.gtld-servers.net
com    nameserver = H.gtld-servers.net
com    nameserver = I.gtld-servers.net
J.gtld-servers.net    internet address = 192.48.79.30
K.gtld-servers.net    internet address = 192.52.178.30
L.gtld-servers.net    internet address = 192.41.162.30
M.gtld-servers.net    internet address = 192.55.83.30
a.gtld-servers.net    internet address = 192.5.6.30
B.gtld-servers.net    internet address = 192.33.14.30
C.gtld-servers.net    internet address = 192.26.92.30
D.gtld-servers.net    internet address = 192.31.80.30
E.gtld-servers.net    internet address = 192.12.94.30
F.gtld-servers.net    internet address = 192.35.51.30
G.gtld-servers.net    internet address = 192.42.93.30
H.gtld-servers.net    internet address = 192.54.112.30
```

I.gtld-servers.net internet address = 192.43.172.30

The .be root looks like this:

```
be
    primary name server = a.ns.dns.be
    responsible mail addr = tech.dns.be
    serial = 2004022101
    refresh = 3600 (1 hour)
    retry = 1800 (30 mins)
    expire = 3600000 (41 days 16 hours)
    default TTL = 600 (10 mins)

be      nameserver = a.ns.dns.be
be      nameserver = brussels.ns.dns.be
be      nameserver = paris.ns.dns.be
be      nameserver = amsterdam.ns.dns.be
be      nameserver = c.ns.dns.be
be      nameserver = london.ns.dns.be
be      nameserver = milano.ns.dns.be
be      nameserver = b.ns.dns.be
a.ns.dns.be      internet address = 193.109.126.140
brussels.ns.dns.be      internet address = 193.190.135.4
paris.ns.dns.be internet address = 192.134.1.1
amsterdam.ns.dns.be      internet address = 193.194.136.30
c.ns.dns.be      internet address = 195.22.139.135
london.ns.dns.be      internet address = 195.66.241.90
milano.ns.dns.be      internet address = 217.29.76.10
b.ns.dns.be      internet address = 193.109.126.12
```

Distributed responsibility

Each top level domain has its own structure.

- United Kingdom: .co.uk, .ac.uk
- Belgium: .ac.be, but no .co.be

Each top level has its own registration authority, that is responsible for the rules for assigning names. In Belgium the URL is <http://www.dns.be/>. There are about 300,000 names beginning of 2004. The rule by which it operates also depends on the local legislation, in this case Belgian law. For instance, there is a law of June 26, 2003 that can be used to guide name assignment.

Choice of names

The multinational organization ACME can choose two options. It can register the domain .acme.com, and use country based subdomains: .be.acme.com, fr.acm.com, ...

Alternatively, it can register multiple times, once per country: acme.be, acme.fr.

The decision is based on balancing between the structure, the length of the names, the cost, the availability of the names, and the complexity of the management.

Domain name and ISP

To connect to the internet, you need at least one member of the internet you can connect to. The core Internet infrastructure is provided by Internet Service Providers (ISP). An ISP rents (possesses) a range of IP addresses. You have to determine how many official addresses you need, and then ask for those to the service provider. The price depends on the amount, and large amounts are just not available anymore.

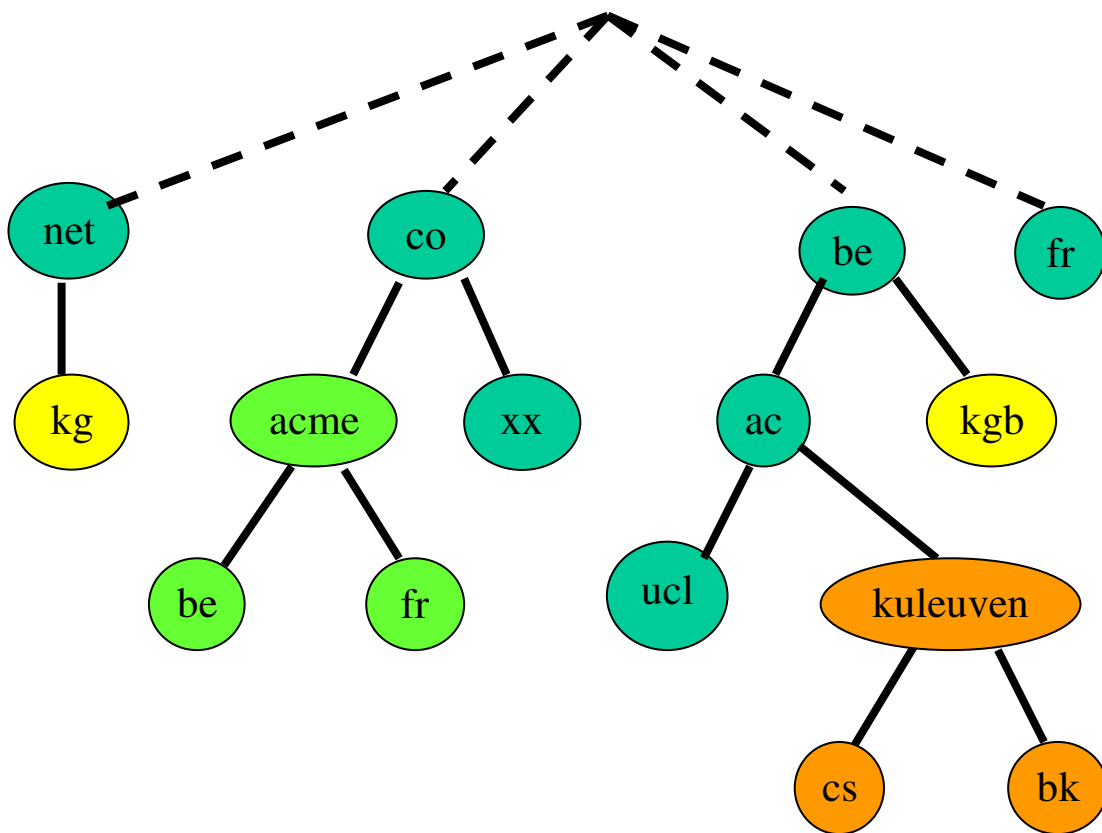
Independent from the selection of the ISP and thus the IP addresses, you need to decide on the parent domain. Are you requesting to become a subzone of .com, .be, or .ac.be? Do you want to register for more than one, like am.com, am.org and am.net? Check the rules to see if you are eligible, and if the name is not already taken. If it is, it may be possible to claim the name anyway. You need to provide the registrar with the DNS server(s) that will be authoritative. You know need to decide to run an own server or use the ISP's server.

Naming management

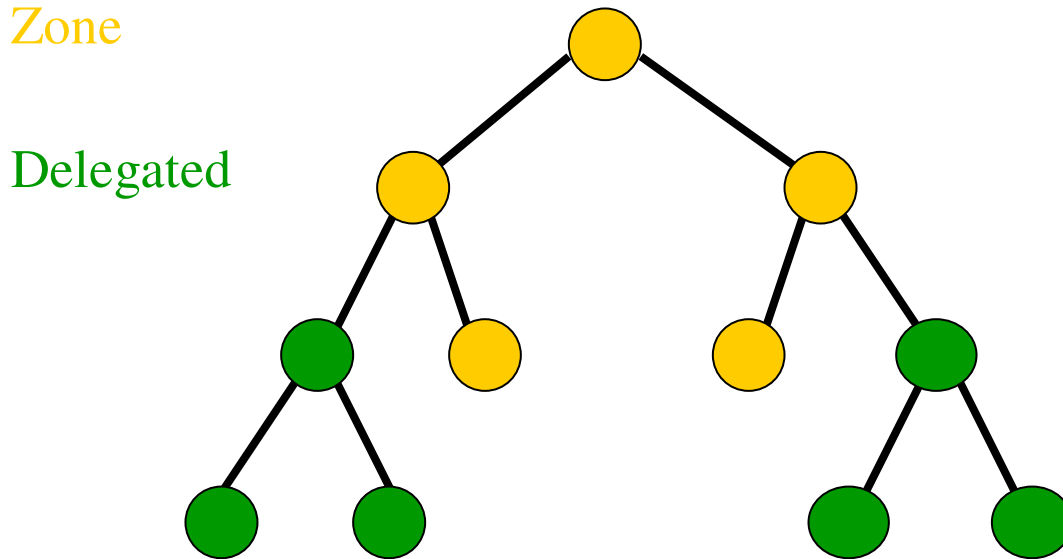
A registration authority is needed. Domain names do have a commercial value. There can be only one www.radio.be, www.tv.be, www.mail.be, etc. These are the obvious, easy to remember names. Other names are directly linked to an established brand, like ibm.com, sub.com, microsoft.com. They have a “natural” owner. The regulations governing a domain are dependent on the local authority. They can differ between .uk, .be and .com. They differ between .be and .ac.be.

The management costs money, so it is expected that owning a domain costs money.

Domain structure



Zone



A zone is the whole naming information within the scope of one name server. Any number of subtrees within a zone can be delegated to other servers.

DNS performance

DNS has become a critical internet infrastructure component. Most applications and people use names. Each name-based request needs resolving: the IP address is necessary or no connection can be made. Not only that, but the service must be high performance, as the resolving is part of the critical path with respect to the time it takes for a request to complete. The client can be anywhere in the world, and the server as well.

The main solutions to fulfill the high demands are:

- ✓ Caching: local, organization, ISP, ...
- ✓ Quick homing into “right” server via the referral mechanism
- ✓ A low-overhead protocol: UDP based, multiple answers per request intelligent prediction of subsequent queries

Other type of requests

DNS supports a number of other requests. The request type that is the closest related to name to IP, is IP to name. The reverse look-up from IP to name can be used to check the source of a request. If it is within .nl, respond with a Dutch page, if it is within .uk, respond with an English page. It is left as an exercise to decide what to do if the request comes from the .com domain.

Reverse look-up is made to look like a regular look-up, but uses inverted tables. Every IP address is converted to a virtual name.

Example:

The IP address is 163.7.23.89, then the corresponding virtual name is 89.23.7.163.in-addr.arpa.

The in-addr.arpa domain hierarchy mimics the IP structure and hierarchy, unlike the regular DNS structure. This “hack” is referred to as the reverse DNS hack.

As the forward and the reverse look-up us somewhat independent trees, it can be use to check the integrity of a name to IP mapping: is $\text{name}(\text{Ip}(\text{name})) == \text{name}$, or is $\text{ip}(\text{name}(\text{ip})) = \text{ip}$? Note that even if they are not equal, that does not mean that there is a real problem.

Another useful type of information are Start-of-authority records.

To find the authoritative nameserver one can also ask for Nameserver records. They describe the nameservers that are authoritative for a given domain.

A strange element is the presence of mail information records: MX records. They are not related to DNS, yet the DNS mechanisms is used to provide this information. It will be discussed when e-mail is tackled.

All the different types of information are called resource records (RRs).

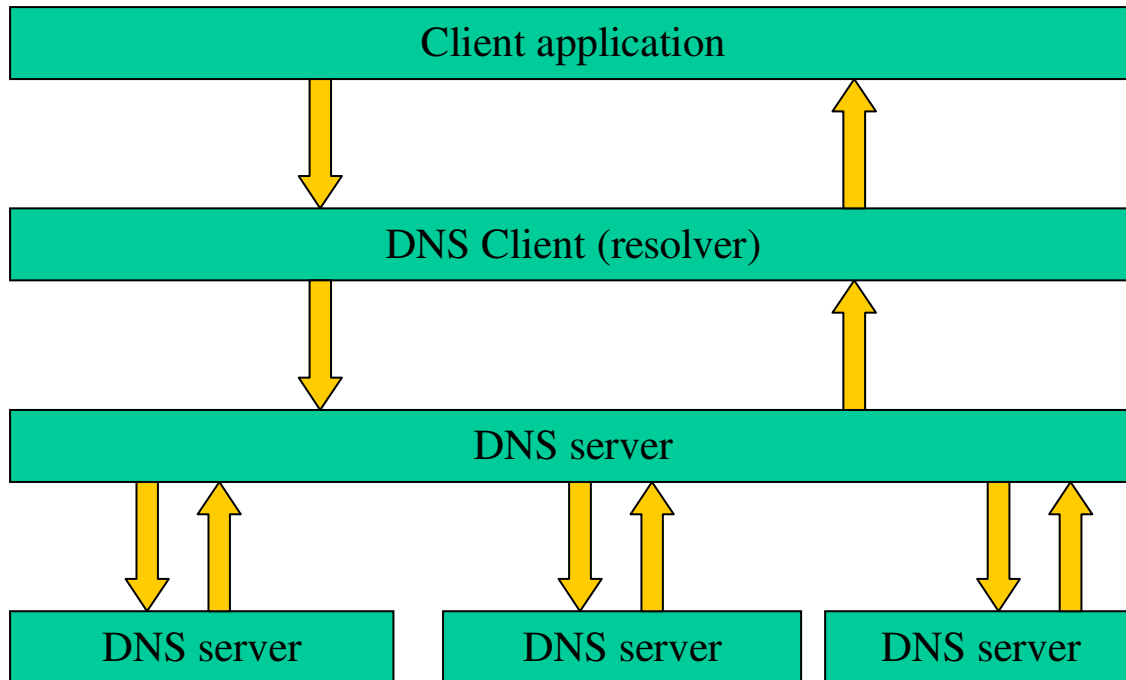
- A: name to address
- PTR: address to name
- CNAME: nick names
- HINFO: host information (security)
- NS: DNS servers (delegation)
- MX: Mail eXchange

Query options

When launching a query the systems address this query at one specific server that is either configured by hand, or via DHCP. It can be the corporate DNS server, or the DNS server of the ISP. That DNS server checks if it knows the answer, because it is either authoritative or is has the requested data in its cache. If it does not know the answer, it will itself launch a query to a server that should know the answer. This is called a recursive query.

Not all servers will allow you to ask a recursive query, as it requires resources. Your company's DNS server and your ISP's DNS server are supposed to do this for you, but other servers will in general refuse it.

The alternative for a client is to look for the right DNS server itself. Starting from the root servers one can obtain information on the right DNS server. For instance, one starts with the DNS server for .be, then for the DNS server for .ac.be, then for the DNS server for kuleuven.ac.be, then for the server for cs.kuleuven.ac.be. Most domain names are not that deep.



DNS carrier

DNS can use either TCP or UDP as protocol. It uses port 53 in both cases. UDP is the typical protocol chosen for name lookup queries. It is fast and adequate for the short requests and responses that are exchanged. TCP is used for bigger queries, like a zone transfer. A zone transfer is a complete dump of all the information in a particular zone. Such a transfer is used to synchronize a secondary server with the primary master server. TCP can also be used if long replies are expected.

DNS server cooperation

Root name servers

There is at least one root server per top level domain. In principle one needs to start from there to find anything. The IP addresses of those servers should be stable: there is no way to find them otherwise. The locations of the root servers must be configured in name servers. There are a number of root servers to avoid availability problems.

Replies from the root servers can be (should be) cached, to avoid having to query them each and every time again.

Set up primary DNS server

To set up a primary DNS server you need to:

- Define server parameters (time-outs, contact information)
- Define name to IP mapping
- Define IP to name mapping
- Configure the top level server locations

Secondary DNS server

The secondary DNS server is typically at a different network location. That way the availability of the DNS information is better available, even if the primary DNS servers location is unreachable.

It copies all data from the primary DNS server. The typical way to achieve this is a zone transfer. The synchronization uses the SOA information of the primary server. It can check if there are any modifications via the SERIAL number of the information. To know the right frequency to refresh its information, it can use the REFRESH information. If there are any connectivity problems, the RETRY information suggest the right wait period. The time-out of cache entries is defined in the EXPIRE information.

Internal – external DNS

DNS can be used by hackers to investigate remote systems. If all your internal systems are listed in the DNS server, as they should be for internal use, outsiders can learn this information. The easiest way is with a zone transfer, but also enumerating may work. Such an enumeration could be done via reverse DNS mapping over all allocated IP addresses.

Zone transfers should not be allowed except for the transfer between primary and secondary DNS server. But that does not solve the second type of discovery.

The advice is to split DNS in internal/external view. The internal DNS server has all the mappings for the internal servers, the external server has only the information about the servers that need to be known outside of the organization. The inner server connect to the outer server for all external queries. This system also matches well with the typical network infrastructure of the DMZ.

Tools for DNS querying

- whois
- nslookup
- nstest: diagnostic tool
- dig

Nslookup

Nslookup is by far the most used tool to diagnose DNS problems. It is widely available. The most basic use is to check if a name can be resolved:

```
nslookup name
```

This command will use your DNS settings to determine the DNS servers to use. It will then try to resolve the name, just like any other client would do. If there is no answer, something is wrong. Trying again may yield other results! This is one of the rare cases where repeating the same action may indeed solve the problem. The reason for this is that requests take time. Your client, nslookup in this case, may have given up, while other servers are still fetching the answer. When they get it, they will update their cache. Your next request will hit the cache entry and provide the answer in time.

You can tell nslookup which server to use:

```
nslookup name dnsserver
```

If you did not get an answer after trying a couple of times, you may use another DNS server to see if the problem is with your DNS server, or with the name you are trying to resolve. A good

idea is to try a probable name server for the domain. One such guess is the server `dns.<domain>`, or `dns.<parent domain>`. `Dns` is a common name for DNS servers, just like `www` for web servers. The other attempt could be with your ISP's DNS server. Often, a link down condition is the cause of non-working DNS.

Keep in mind that recursive queries may not be allowed on other DNS servers than "your" server or the one from your ISP.

For more complex exploration of DNS information, you can start `nslookup` in interactive mode. You do this by starting `nslookup` without name or nameserver arguments.

To ask a zone transfer, use the "ls" command:

```
ls <domain>
```

To resolve a name, just type it at the prompt:

```
name
```

The nameserver that is used is the default one. Names are eventually completed with the default search domain. A "." At the end prevents such completion.

```
name.domain.
```

To select another one for a single query, follow the name with the nameserver name or address:

```
name nameserver
```

To switch to another default nameserver, issue the command "server":

```
server NAME
```

This command sets the default server to `NAME`, using the current default server to resolve the name of the server first. Alternatively, use an absolute IP address. To resolve the name with the initial default servers, use the "lserver" command.

```
lserver NAME
```

If all else fails, it can be necessary to fall back on the root servers. Use this with care. The command is "root":

```
root
```

The standard query mechanism is asking for any records that match, whether it is an alias, a name to pointer, host info, mail information, To select specific RRs, use the "set querytype" command:

```
set querytype=<choose>
```

The query type can be `ANY`, which is the default or one of:

- A(ddress)
- P(oin)T(e)R
- M(ail e)X(change)
- S(tart)O(f)A(uthority)
- N(ame)S(erver)
- C(anonical)NAME
- H(ost)INFO(rmation)

Exercise

Compare DNS and ARP protocols

DNSSecure

Why do we need DNSSecure? DNS is a very insecure protocol. It is UDP based, which means replies can be easily spoofed. There is no authentication, which further facilitates a man-in-the-middle attack. The attacker does not have to be on the path between the requester and the server. He needs to answer at the right moment.

DNSSecure is defined in the RFC 2535: DNS Security extensions. DNSSecure tries to counter the following security risks:

- Denial of Service (DoS)
- Man in the middle (MITM)
- Domain intrusion
 - Authentication via IP, reverse DNS
 - Cookies set for a domain

The security measures DNSSecure introduces are authentication and integrity. It provides authentication of the data and the request, in short: the transaction. It provides integrity protection, indirectly, via the authentication system. It does not provide confidentiality nor authorization.

Mechanism: signatures

DNSSecure uses public key technology to achieve the required protection. It uses itself for key distribution. It introduces two new Resource record types:

- KEY RR: signed public keys
- SIG RR: signatures

Signatures

The signatures are used to sign Resource Record sets but also response sets. Resource records can be pre-signed. That signature need not be recomputed each and every time, saving time and computing cycles. Answers contain varying sets of RRs. These must be signed with the zone key to protect tampering with the answer.

Trust

A trust hierarchy is built. The superzone signs the subzone keys.

If there are untrusted subzones, the zone signs 'no key' KEY RR for that zone.

NOT FOUND authentication

To avoid domain intrusion spoofing, it is necessary to know that certain entries do not belong to the domain. If an attacker can insert a fake server within the domain, name based authentication can be broken. For instance, a server can be thought to be in .us, while it is in .be. A domain cookie can be sent to hrus.xxx.com, but there is no hrus server in the domain.

The mechanism to rule out such rogue members is to use a chain of authenticated data. signed response: before - after RR indicates data not there. A new RR type, NXT RR, chains the records together. The order in the chain is based on canonical ordering of names. The end-of-chain marker, as well as the first name, is the zone itself.

Multiple keys

There is a difference between the zone keys, which are used for “static” data authentication, per RR, and host keys, which are used to sign a transaction or to perform request authentication.

KEY RR

In the KEY RR, the keys are labeled for use: zone key, server key or user key.

- zone key: x.y : zone x.y
- server key: www.x.y : server www in zone x.y
- user key: a.x.y : user [a@x.y](#)

Also key usage and key algorithm are specified.

- key used in protocol: DNSSec, IPSec, ...
- keying algorithm: RSA/MD5, DH, DSA, ...

References

DNS and BIND, 4th Edition
By Paul Albitz, Cricket Liu
4th Edition April 2001
0-596-00158-4
622 pages

DNS on Windows 2000
By Matt Larson, Cricket Liu
2nd Edition September 2001
0-596-00230-0, 349 pages

<http://www.dns.net/dnsrd/rfc/>: DNS related RFCs
<http://www.domtools.com/dns/>
<http://www.sampade.org/ssw/features.html>