The Domain Name System

Author: Paul Mockapetris

1987

The old approach:

- **ARPANET :**
  - Host name to IP address mapping using a centralized database (i.e., HOSTS.TXT file) maintained by the Network Information Center (NIC)
  - HOSTS.TXT file is ftp-ed daily by all hosts from NIC
- **Scalability problems :**
  - Network bandwidth of HOSTS.TXT distribution ~ (no. of hosts)^2
  - Search time of centralized database (HOSTS.TXT) ~ no. of hosts
  - Name updates :
    - Local updates => name conflicts
    - Central updates (by NIC) => delayed name changes
- **Solution => The Domain Name System (DNS)**

DNS design goals :

- Provide a consistent name space through hierarchical, domain-based naming scheme.
- Implement a distributed database (with local caching)
- Allow names to be used in retrieving host or mailbox addresses and other yet unspecified type of data
- Allow the use of the same name space with different protocol families
- Work with datagrams as well as virtual circuits
- Able to be used by PCs as well as mainframes

Assumptions about usage :

- **Data base size :**
  - Initially ~ no. of hosts in the system
  - Eventually ~ no. of users in the system
- **Data change rate:**
  - Most of the data changes very slowly (weeks, month)
  - The system should deal with subsets that change more rapidly
- **Zones :**
  - Name space divided into administrative subdomains (zone).
  - Each zone provides multiple Name Servers (NS).
- **Queries :**
  - A NS may deny any query, but sends the same reply to identical queries
  - If a NS is presented with a query it cannot answer, we have two approaches :
    - recursive
    - iterative

What makes up the DNS ?

- **The Domain Name Space :**
  - provides for hierarchical addressing;
  - it is made up by **zones** (administrative subdomains).

- **Name Servers (NS) :**
  - map names to IP addresses (e.g. **eng.umd.edu** => **129.2.94.5**);
  - maintain the database associated with a subset of the domain space (one or more zones).

- **Resolvers :**
  - sit on the same host as the client
  - translate the client's requests into DNS queries;
  - extract information from NS in response to client requests.

The three views on the DNS :

- **User's :**
  - the name system is accessed through a simple procedure or OS call
  - single tree structure of the name space

- **Resolver's :**
  - the DNS is composed of an unknown no. of Name Servers
  - has knowledge of at least one Name Server

- **Name Server's :**
  - domain name space partitioned in **zones**
  - supports one or possibly more zones
  - zone data :
    - kept in master files
    - edited by the zone's administrator
    - ftp-ed by NSs supporting that zone

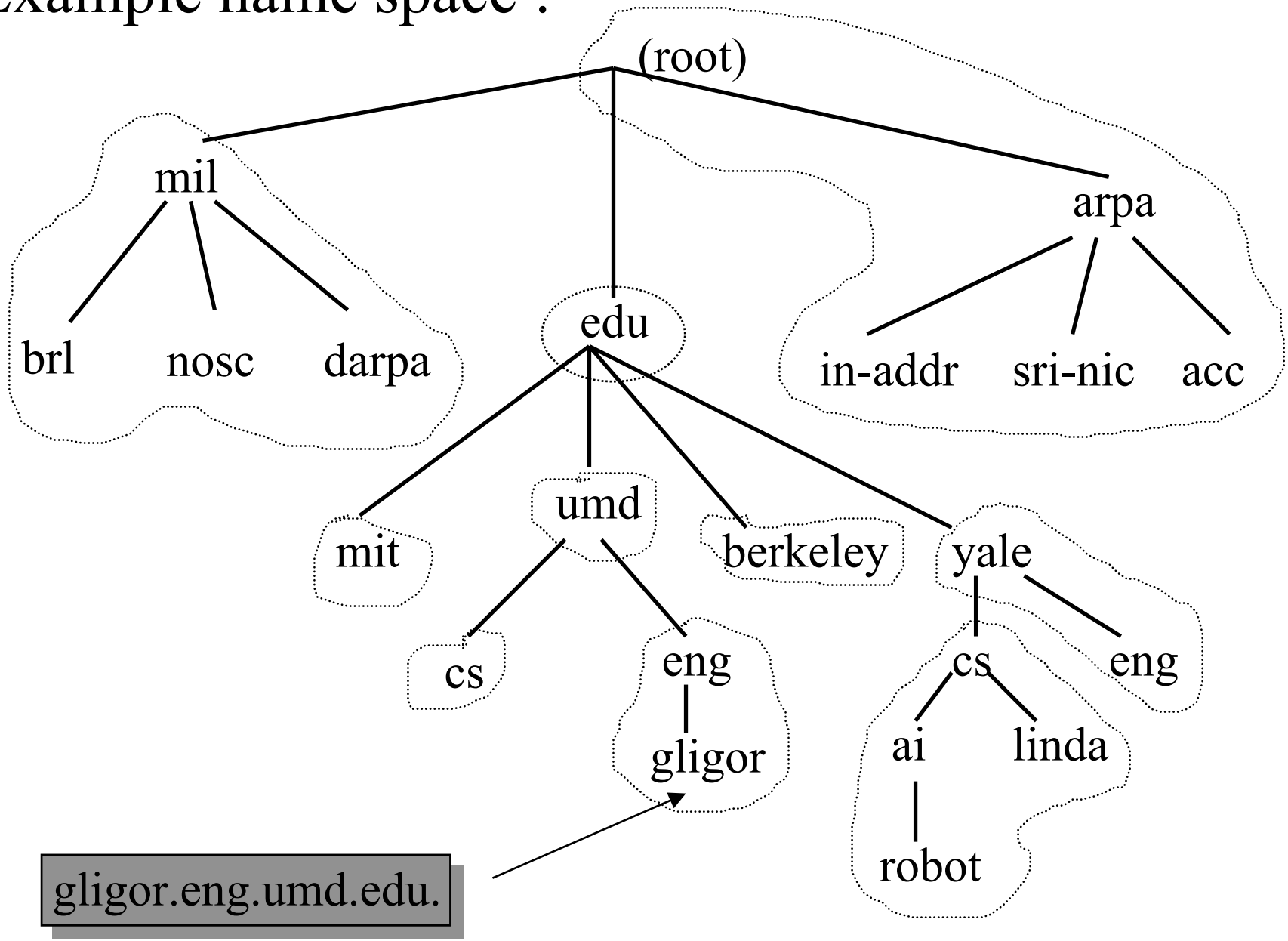
Domain Name Space

- **Tree structure :**
  - each node and leaf corresponds to a resource set
  - no distinction between interior nodes and leaves

- **Each node/leaf has a label :**
  - sibling nodes may not have the same label
  - the null label is reserved for the root

- **The domain name of a node/leaf:**
  - list of labels on the path from the node to the root separated by dots
  - domain names are not case sensitive
  - absolute names end with a "." (e.g., **eng.umd.edu.**)
  - relative names are resolved depending upon context
    (if within Univ. of MD : **eng** => **eng.umd.edu.**)

Example name space :



More about names :

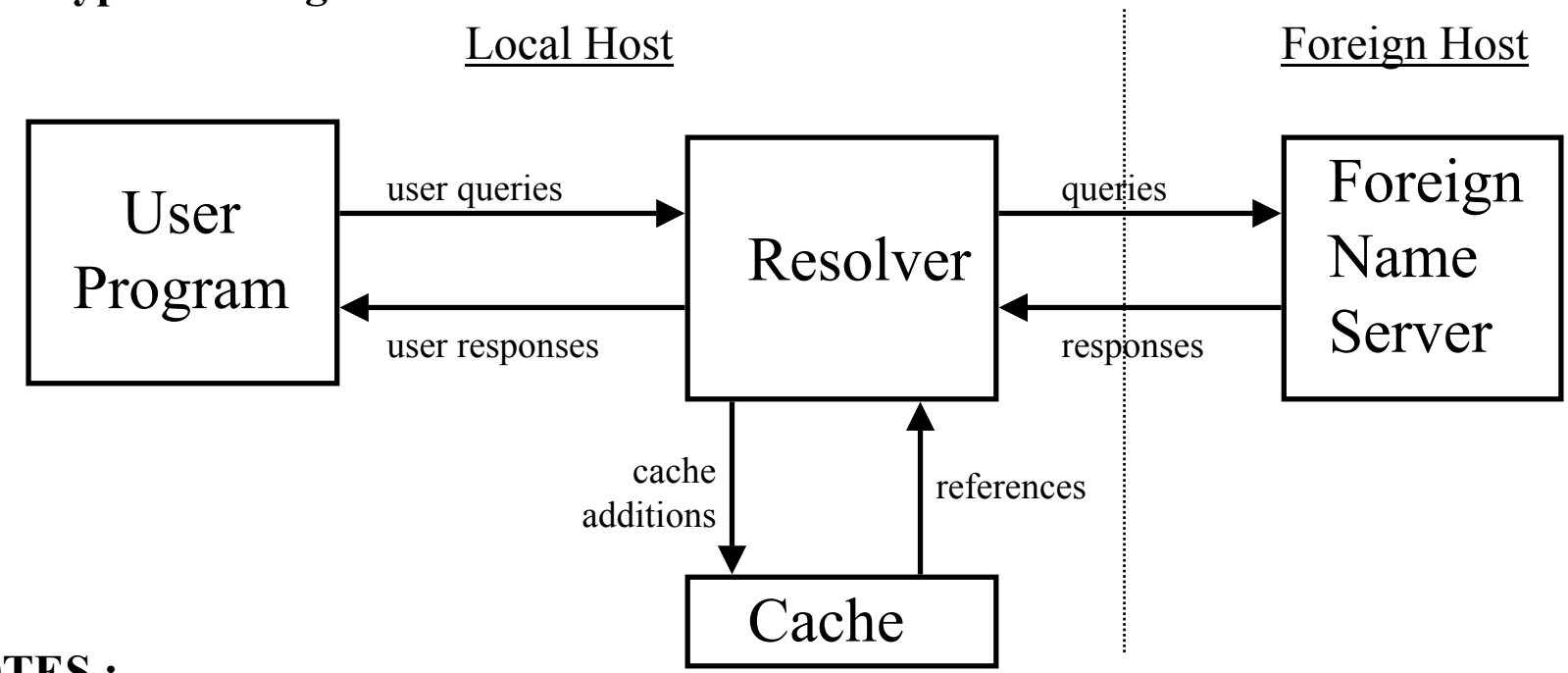
- **Domains :**
  - identified by a domain name;
  - consists of that part of the tree below the domain name which specifies the domain
  - follows the rules for the old HOSTS.TXT file
  - are case insensitive
  - start with a letter, end with a letter or digit
  - contain only letters, digits and hyphen (i.e., "-" and "_")

- **Subdomain :**
  - a domain is a subdomain of another domain if it is contained in it
  - **a.b.c.d.** is a subdomain of **b.c.d.**, **c.d.**, **d.** and " " (the root)

- **Mailbox naming :**
  - concatenation of user name, a dot (","), and the domain name
  - the domain name of **gligor@eng.umd.edu** is **gligor.eng.umd.edu**

Common configurations :

1 **Typical configuration :**

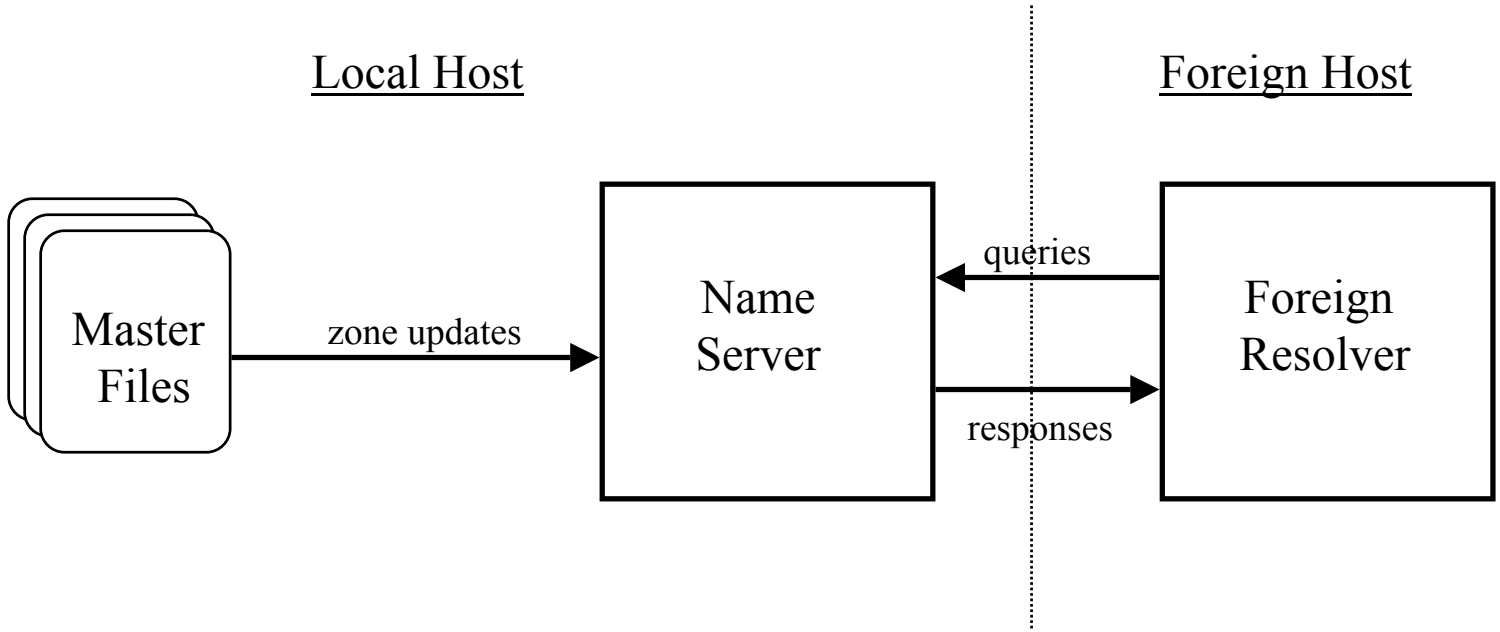


NOTES :

- Format of user queries/responses specific to the local host and its OS
- User queries are generally OS calls => resolver and its cache are part of the OS
- Format of queries/response to/from foreign NS are standard
- Name Server may be a stand alone program on a dedicated machine or a process on a large timeshared machine

Common configurations :

2 Primary NS configuration :

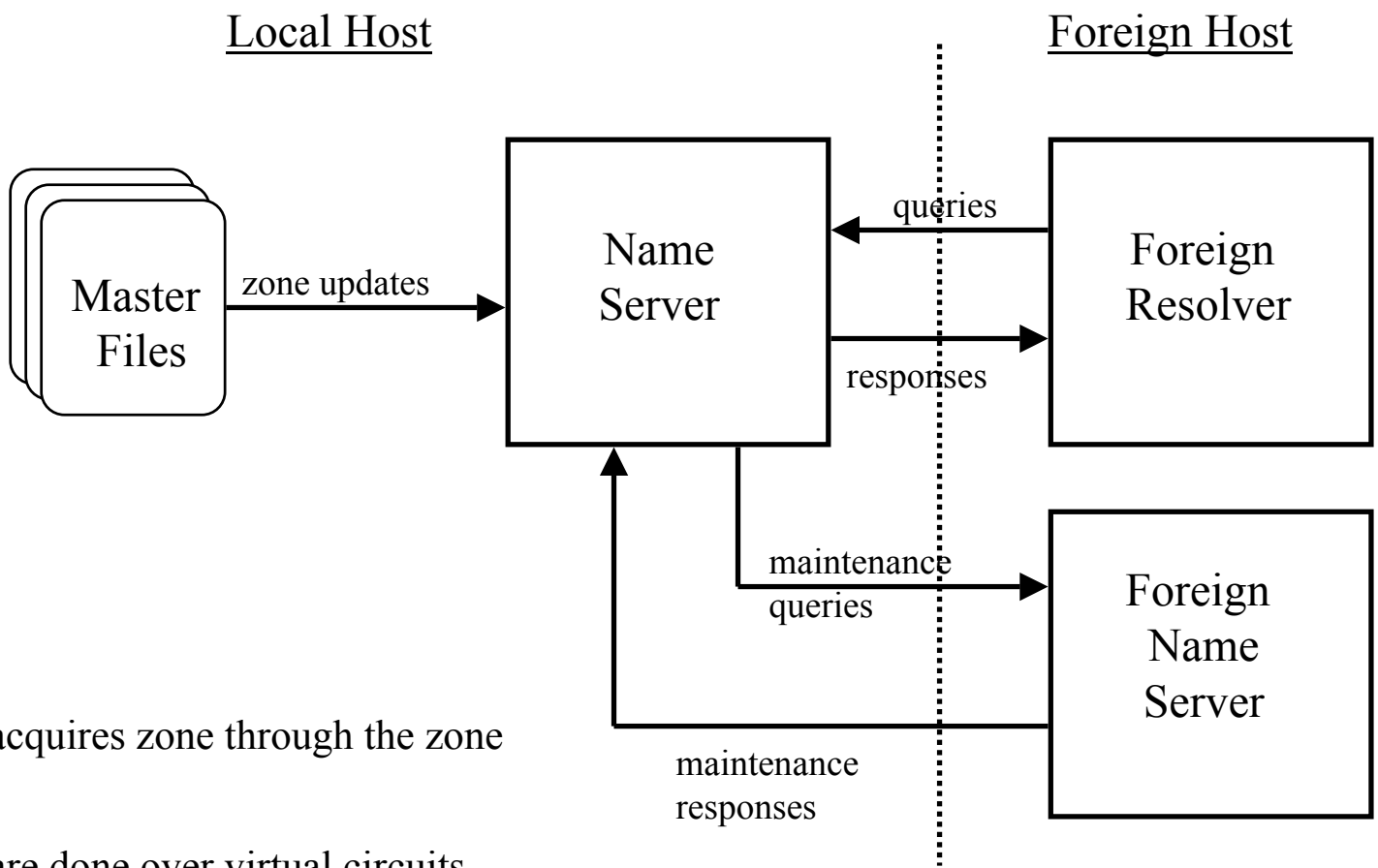


NOTES :

- Here the NS gets information on its zones by reading master files from its local file system
- In this case the NS is called Primary Name Server
- The other (secondary) NSs of the same zone get information on the zone from Primary NS

Common configurations :

3 Secondary NS configuration :

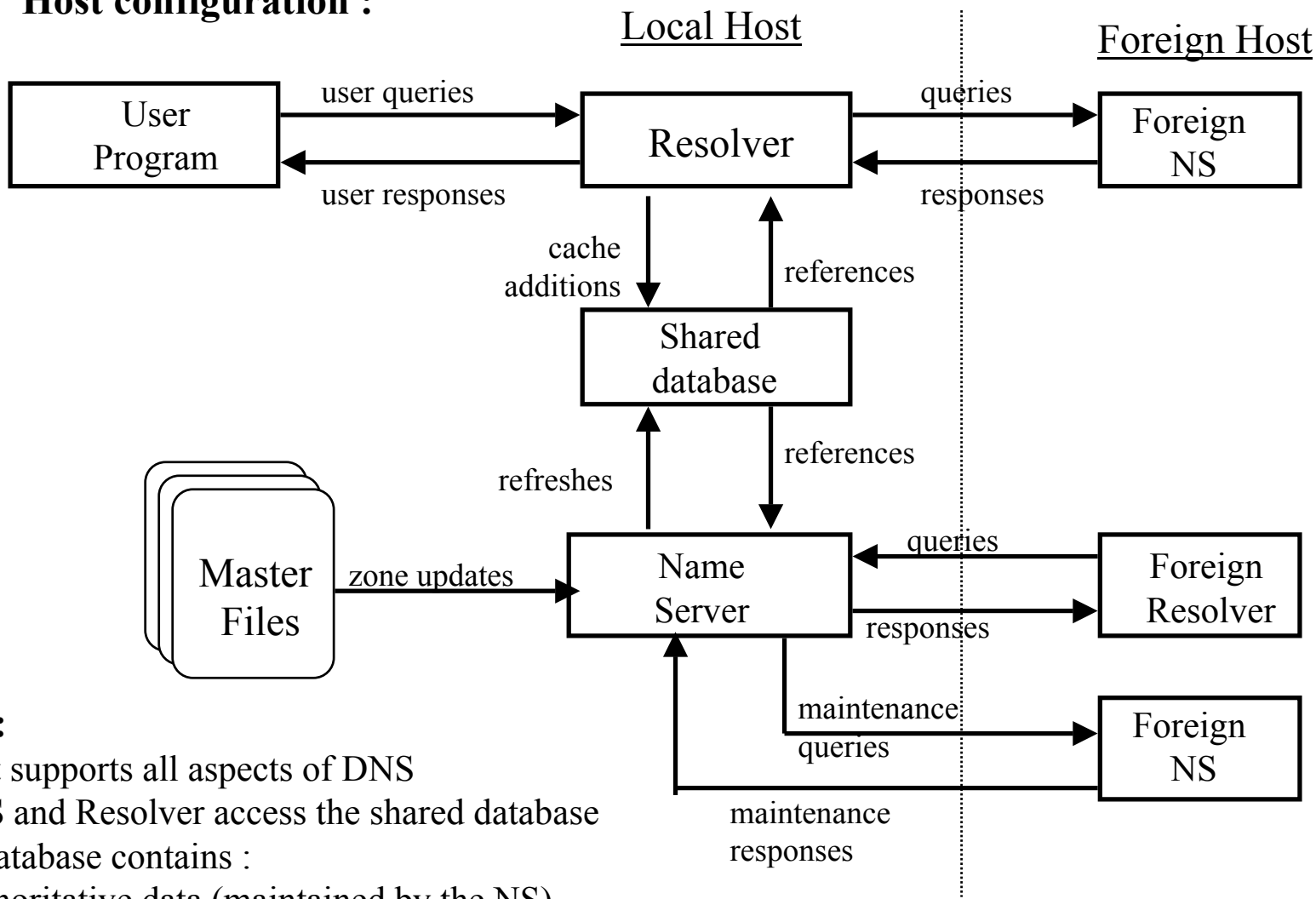


NOTES :

- Secondary NS acquires zone through the zone transfer protocol
- Zone transfers are done over virtual circuits

Common configurations :

4 **Host configuration :**

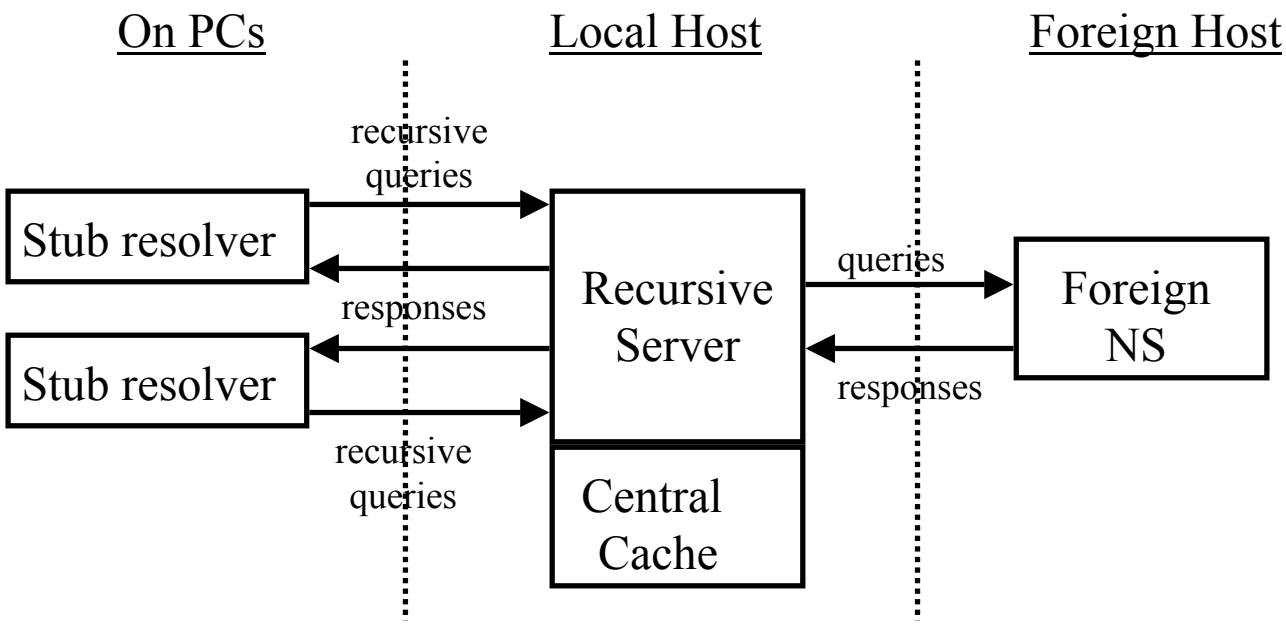


NOTES :

- This host supports all aspects of DNS
- Local NS and Resolver access the shared database
- Shared database contains :
  - authoritative data (maintained by the NS)
  - cached (nonauthoritative) data from previous resolver queries

Common configurations :

5 **PC configuration :**



NOTES :

- PCs don't have the resources to implement full resolvers
- Stub Resolvers = front ends for resolver located on the Recursive Server
- Recursive Server is part of a NS known to provide for recursion
- Centralized caches have a higher hit-ratio

Resource Records :

- Data on zones is stored in Master Files
- Master Files contain **Resource Records**
- Resource Records (RRs) store resource information associated with names
- A domain name identifies a node or leaf (there is no distinction beteewn them)
- When a resolver queries a NS about a name => it gets back one or more RRs

NOTE :

- The order of RRs in Master files is not significant

Resource Records :

- **Contain the following fields :**
  - **owner** - the domain where the RR is found
  - **type** - an encoded 16 bit value that specifies the type of the resource in this RR
    - refer to abstract resources
  - **class** - an encoded 16 bit value which identifies a protocol family or instance of a protocol
  - **TTL** - time to live - 32 bit integer in units of seconds
    - used by resolvers when they cache RRs
    - describes how long a RR may be cached before being discarded
  - **RDATA** - type and sometimes class dependent data which describes the resource

Some Resource Record Types :

- **A** a host address
- **CNAME** the canonical name of an alias
- **HINFO** CPU and OS of the host
- **MX** mail exchange for the domain
- **NS** authoritative name server for the domain
- **PTR** pointer to another part of the domain space
- **SOA** start of a zone of authority

Some Resource Record Classes :

- **IN** the Internet system
- **CH** the Chaos system

Resource Record Owners :

- Often implicit
- If not mentioned, the owner is the same as for the previous RR

Resource Record TTL :

- Time limit on how long the RR can be cached
- Doesn’t apply to authoritative data in zones
- It is timed out by the refreshing policies for the zone

Resource Record RDATA :

For type :	Contains :
• **A**	- for IN class : an IP address - for CH class : a domain name followed by a 16 bit octal Chaos address
• **CNAME**	- a domain name
• **MX**	- a 16 bit preference value (lower is better) followed by a host name willing to act as a mail exchange for the owner domain
• **NS**	- a host name
• **PTR**	- a domain name
• **SOA**	- several fields describing the zone

Aliases :

• Entities have sometimes multiple names
(e.g. **gligor@eng.umd.edu** and **gligor@glue.umd.edu**)

• One of those names is the **canonical** or primary name; the others are aliases

• **CNAME RRs :**
  • the owner is an alias
  • the RDATA specifies the canonical name
  • domain names in RRs that point to other names should point to the canonical name not to an alias (avoid extra indirections)
  • if a CNAME is present at a certain node => no other RR should be present there (consistency reasons)
  • chains of aliases should be followed
  • circular chains of aliases should be detected => error signal

Aliases and Queries :

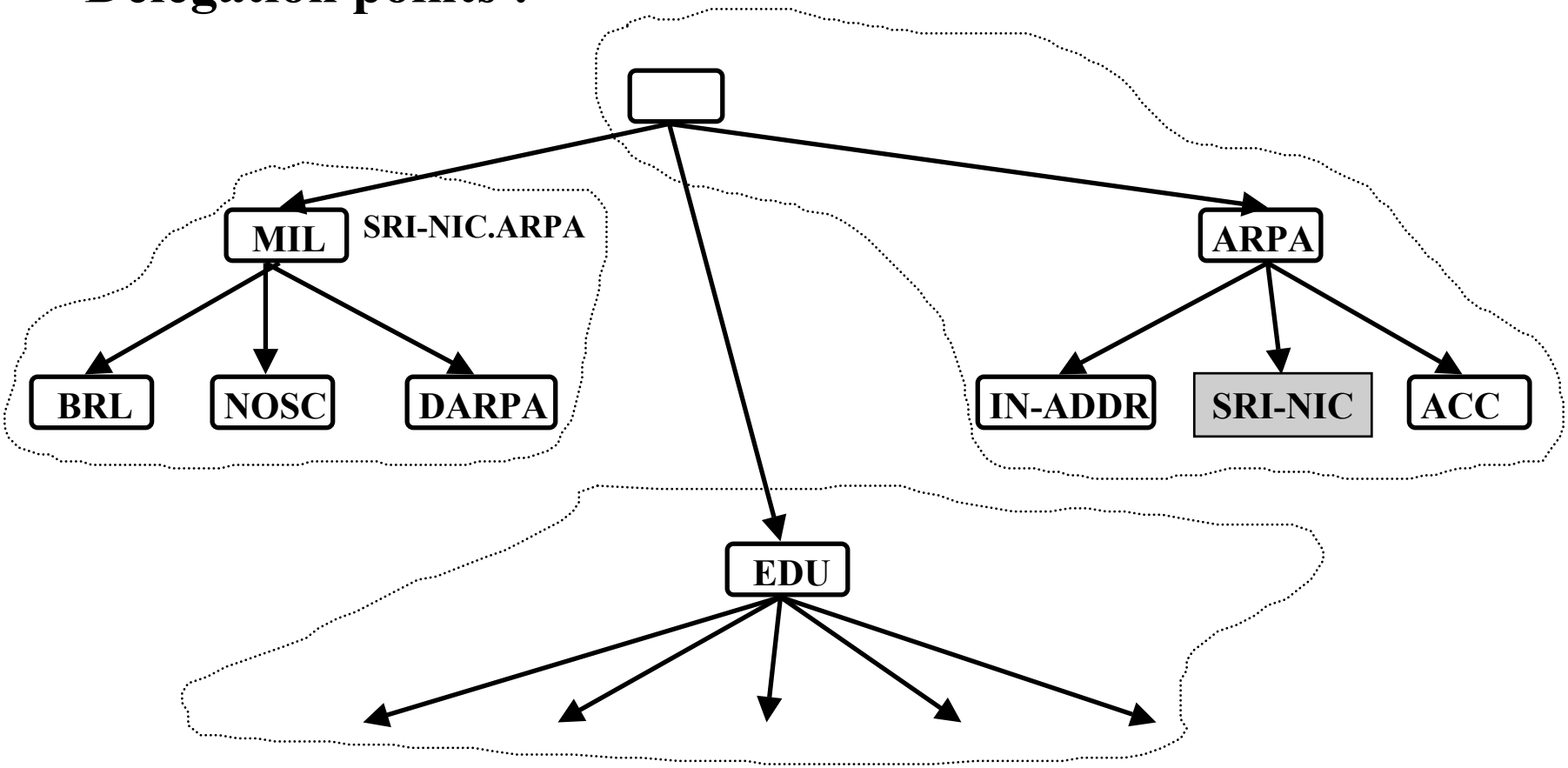
Example :

USC-ISIC.ARPA	IN	CNAME	C.ISI.EDU
C.ISI.EDU	IN	A	10.0.0.52
	IN	MX	C.ISI.EDU

Query : owner = USC-ISIC.ARPA and type = A

- the NS should detect the CNAME at USC-ISIC.ARPA
- it should replace C.ISI.EDU as the owner and restart the query
- one exception to the above rule => when **type = CNAME** the query is not restarted

Delegation points :



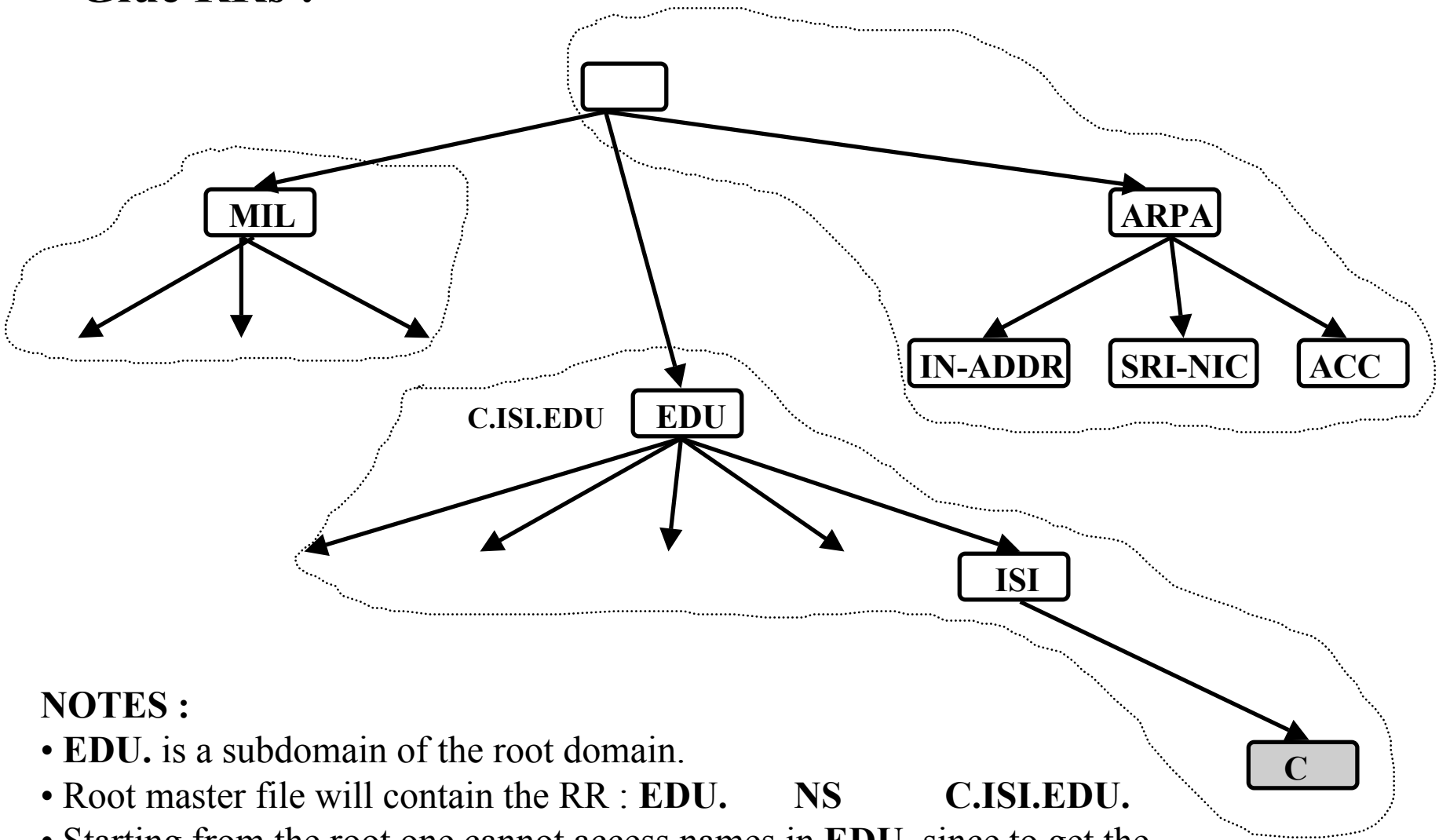
NOTES :

• **MIL.** is a subdomain of the root domain.

• **SRI-NIC.ARPA** is a NS for the **MIL.** domain => the master file for the root domain will contain the RR : **MIL. NS SRI-NIC.ARPA.**

• Since SRI-NIC.ARPA. is a name from the root domain => starting from the root one can easily get name information on entities in the **MIL.** domain.

Glue RRs :



NOTES :

- **EDU.** is a subdomain of the root domain.
- Root master file will contain the RR : **EDU.** **NS** **C.ISI.EDU.**
- Starting from the root one cannot access names in **EDU.** since to get the
- address of **C.ISI.EDU.** we have to go and ask **C.ISI.EDU.** itself !
- Solution : **glue RR** placed in the root master file :

C.ISI.EDU. **A** **10.0.0.52**

NS Resource Records and Glue RRs :

- Location
  - at the top node of a zone and are authoritative
  - at cuts around the bottom, not authoritative (are the NS of some subzone)
  - never in between.

- NS RRs : the domain name of the zone served by the NSs is owner of the RR

- One or more NSs that support a zone can be outside that zone (as far as naming is concerned) => e.g. one of the NS for the **uk** domain is in the US

- A super zone contains NS RRs describing the NSs in the subzone

- If all NSs of the subzone are in the subzone itself => cannot access subzone's NSs

- **Solution** : **GLUE** resource records
  - allow access to NSs for the subzones
  - non-authoritative A type RRs (that RR is owned by the subzone)
  - give the IP address of NSs in the subzone

The IN-ADDR.ARPA domain :

- Provides for gateway location and IP address to host name mapping
- It is structured according to IP addresses, following the Internet addressing structure.

NOTE :

- Network numbers correspond to some non-terminal network nodes in the IN-ADDR.ARPA domain.
- Network nodes : hold pointers to primary host names.
- A gateway sits on multiple networks => multiple network nodes will point to it.
- Gateways also have host level pointers pointing at their fully qualified addresses.

IP address / network number to name mapping :

• IN-ADDR.ARPA domain database content :

10.IN-ADDR.ARPA.	**PTR**	**MILNET-GW.ISI.EDU.**
10.IN.ADDR.ARPA.	**PTR**	**GW.LCS.MIT.EDU.**
18.IN-ADDR.ARPA.	**PTR**	**GW.LCS.MIT.EDU.**
26.IN-ADDR.ARPA.	**PTR**	**MILNET-GW.ISI.EDU.**
22.0.2.10.IN-ADDR.ARPA.	**PTR**	**MILNET-GW.ISI.EDU.**
103.0.0.26.IN-ADDR.ARPA.	**PTR**	**MILNET-GW.ISI.EDU.**
77.0.0.26.IN-ADDR.ARPA.	**PTR**	**GW.LCS.MIT.EDU.**
4.0.10.18.IN-ADDR.ARPA.	**PTR**	**GW.LCS.MIT.EDU.**
6.0.0.10.IN-ADDR.ARPA.	**PTR**	**MULTICS.MIT.EDU.**

• Query : QTYPE=PTR, QCLASS=IN, QNAME=10.IN-ADDR.ARPA.
  • locates the gateways on net 10.
• Response :

10.IN-ADDR.ARPA.	**PTR**	**MILNET-GW.ISI.EDU.**
10.IN.ADDR.ARPA.	**PTR**	**GW.LCS.MIT.EDU.**

• Query : QTYPE=PTR, QCLASS=IN, QNAME=6.0.0.10.IN-ADDR.ARPA.
  • find the host name corresponding to IP address 10.0.0.6
• Response :

6.0.0.10.IN-ADDR.ARPA.	**PTR**	**MULTICS.MIT.EDU.**

The SOA RR :

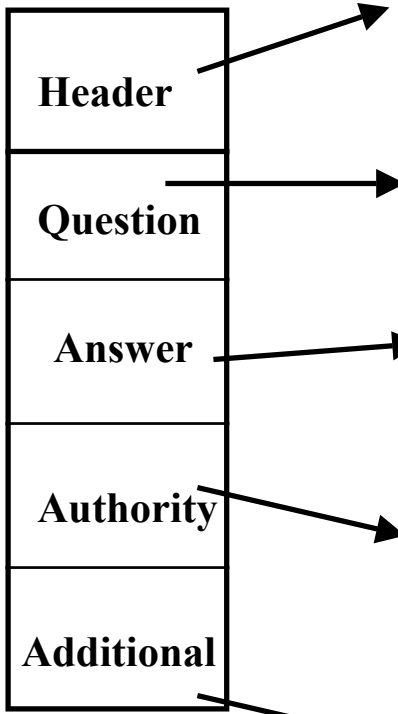
RDATA content :

- MNAME domain name of the NS the is the original or primary source of data for this zone.
- RNAME domain name specifying the mailbox of person responsible for this zone.
- SERIAL version number of the original copy of the zone.
- REFRESH time interval before zone should be refreshed
- RETRY time interval that should elapse before a failed refresh should be retried.
- EXPIRE time value specifying the upper limit on the time interval before a zone is no longer authoritative.
- MINIMUM minimum TTL field that should be exported with any RR from this zone.

Queries :

- Carried over UDP datagrams or TCP connections.

- Standard message (query or response) format :



Type of query, and other flags.
(ID, OPcode, response/query, authoritative/nonauthoritative response, etc.)

Query name and other parameters.
(QNAME, QTYPE, QCLASS)

RRs which directly answer the query.

RRs which describe other authoritative servers. May optionally carry the SOA RR for the authoritative data in the answer section.

RRs which may be helpful in using the RRs in the other sections.

Standard queries :

- Ask for RRs that match the specified QNAME, QTYPE and QCLASS.
- Opcode = QUERY (0).
- QNAME : domain name
- QTYPE :
  - a single type(e.g., A, PTR)
  - AXFR => zone transfer
  - MAILB => matches all mail box related types (e.g., MX, MB, MG)
  - * => matches all RR types
- QCLASS
  - a single class (e.g., IN, CH)
  - * => all classes (NS cannot know all existing classes => response cannot be authoritative).

Other query types :

- **Inverse :**
  - Opcode = 1 (IQUERY).
  - Map a particular resource to a domain name that has the resource (e.g., an IP address to a host name).
  - Used for debugging and database maintenance.
  - Implementation of this service is optional.
  - Completeness or uniqueness of responses to inverse queries cannot be guaranteed (domain name is organized by name rather than any other resource type).
  - Responses to inverse queries should never be cached.
  - Not an acceptable method for mapping host addresses to host names.

- **Status :**
  - Experimental.
  - Opcode = 2 (STATUS).

- **Completion :** obsolete.

Name Servers :

- Are repositories of information that make up the domain database.
- Support one or more zones (for which are authoritative).
- Can reside on hosts with names not in the zone supported.
- Contain :
  - Authoritative information about all names in the supported zone(s)
  - Nonauthoritative information (cached, or Glue RR)
- Are primary (have original copy of the zone) or secondary for a zone.
- Support :
  - Iterative queries (always).
  - Recursive queries (optional); may choose to restrict the clients which can use the recursive service.

Wildcards :

- Are RRs whose owner names start with *.
- When the necessary conditions are met, the NS creates RRs with the owner name equal to the QNAME, and content taken from the wildcard RRs.
- Most often used to create a zone to forward mail from the Internet to another system.
- A query with **QNAME=*.domain.** applies to descendants of **domain.**, but not to **domain.** itself.
- Wildcard RRs do not apply :
    - when the query is in another zone (delegation cancels the wildcard defaults) ????
    - when the query name, or a name between the wildcard domain and the query name is known to exist (e.g. if a zone contains RRs with owners X., *.X., B.X. the the RRs with owner *.X would be returned for queries with QNAME=Z.X., but not QNAME=B.X. or X. or A.B.X.)
- A * label in a query name :
    - has no special effect.
    - can be used to test for wildcard RRs in an authoritative zone
    - is the only way to get RRs with owner name containing *.
    - generates responses that should not be cached.

Example of wildcard RRs use :

• A company X.COM has a mail gateway (A.X.COM). Let the COM. zone contain :

X.COM	**MX**	**10**	**A.X.COM**
***.X.COM**	**MX**	**10**	**A.X.COM**
A.X.COM	**A**	**1.2.3.4**	
A.X.COM	**MX**	**10**	**A.X.COM**
***.A.XCOM**	**MX**	**10**	**A.X.COM**

Comments :

• Any MX query for a domain name ending in X.COM returns an MX RR pointing at A.X.COM.

• Two wildcard RRs are needed (the wildcard at *.X.COM is inhibited at A.X.COM)

• Explicit MX data at X.COM and A.X.COM is required (won't match any wildcard)

Resolvers :

- **Possible implementations :**
  - Full featured resolver => able to carry out iterative queries.
  - Stub resolvers => the resolution function is on a NS that supports recursive queries.

- **Resources :**
  - SNAME the domain name we are searching for.
  - STYPE the QTYPE of the search request.
  - SCLASS the QCLASS of the search request.
  - SLIST structure describing the NSs and the zone which the resolver is currently trying to query.
  - SBELT "safety belt" of the same structure as the SLIST, which is initialized from a configuration file.
  - CACHE structure in which results from previous queries are stored.

Resolver internals :

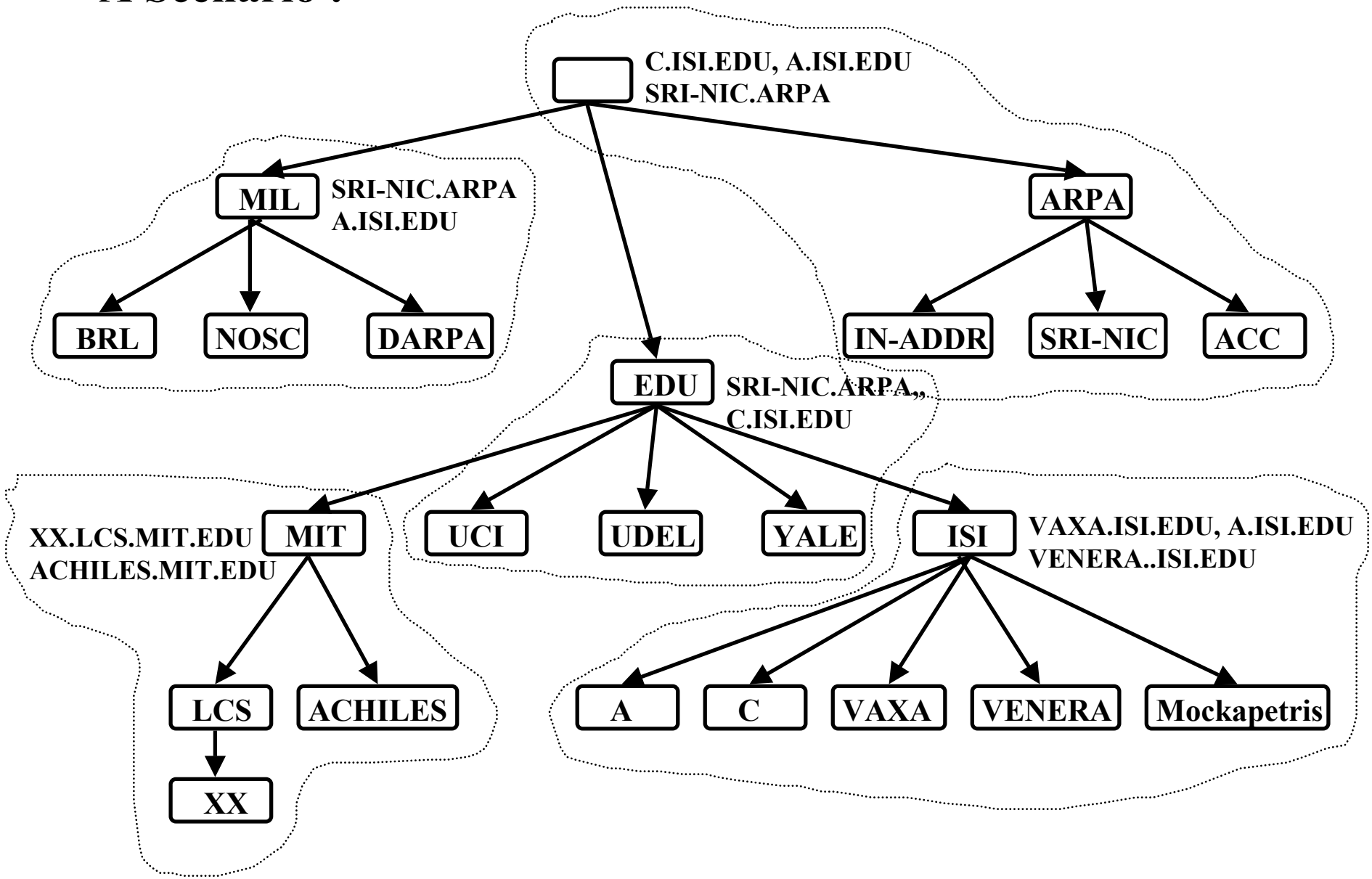
SLIST :

- keeps track of the resolver's current best guess about which NSs should be queried.
- is updated when arriving information changes the guess.
- includes :
    - label match count between the zone being queried and SNAME (equivalent of a zone name the measures how "close" the resolver is to SNAME).
    - NSs for that zone, and their addresses.
    - history information which suggests which NS is the best one to try next.

SBELT :

- lists server that the resolver should use when it doesn't have any information to guide NS selection.
- lists a match count of -1 (indicates no labels match).

A Scenario :


C.ISI.EDU, A.ISI.EDU
SRI-NIC.ARPA
MIL
SRI-NIC.ARPA
A.ISI.EDU
ARPA
BRL
NOSC
DARPA
IN-ADDR
SRI-NIC
ACC
EDU
SRI-NIC.ARPA,,
C.ISI.EDU
XX.LCS.MIT.EDU
ACHILES.MIT.EDU
MIT
UCI
UDEL
YALE
ISI
VAXA.ISI.EDU, A.ISI.EDU
VENERA..ISI.EDU
LCS
ACHILES
A
C
VAXA
VENERA
Mockapetris
XX

The master file for the root domain :

```
.           IN      SOA     SRI-NIC.ARPA.   HOSTMASTER.SRI-NIC.ARPA. (
                                            870611          ;serial
                                            1800            ;refresh every 30 min
                                            300             ;retry every 5 min
                                            604800          ;expire after a week
                                            86400)          ;minimum of a day
                    NS      A.ISI.EDU.
                    NS      C.ISI.EDU.
                    NS      SRI-NIC.ARPA.

MIL.        86400   NS      SRI-NIC.ARPA.
            86400   NS      A.ISI.EDU.

EDU.        86400   NS      SRI-NIC.ARPA.
            86400   NS      C.ISI.EDU.

SRI-NIC.ARPA.       A       26.0.0.73
                    A       10.0.0.51
                    MX      0       SRI-NIC.ARPA.
                    HINFO   DEC-2060 TOPS20
```

```
ACC.ARPA.          A       26.6.0.65
                   HINFO   PDP-11/70 UNIX
                   MX      10      ACC.ARPA.

USC-ISIC.ARPA.     CNAME   C.ISI.EDU.

73.0.0.26.IN-ADDR.ARPA.    PTR     SRI-NIC.ARPA.
65.0.6.26.IN-ADDR.ARPA.    PTR     ACC.ARPA.
51.0.0.10.IN-ADDR.ARPA.    PTR     SRI-NIC.ARPA.
52.0.0.10.IN-ADDR.ARPA.    PTR     C.ISI.EDU.
103.0.3.26.IN-ADDR.ARPA.   PTR     A.ISI.EDU.

A.ISI.EDU.         86400   A       26.3.0.103
C.ISI.EDU.         86400   A       10.0.0.52
```

The master file for the EDU domain :

```
EDU.        IN    SOA     SRI-NIC.ARPA.  HOSTMASTER.SRI-NIC.ARPA. (
                          870729                  ;serial
                          1800                    ;refresh every 30 minutes
                          300                     ;retry every 5 minutes
                          604800                  ;expire after a week
                          86400                   ;minimum of a day
                          )
                    NS    SRI-NIC.ARPA.
                    NS    C.ISI.EDU.

UCI                 172800  NS    ICS.UCI
                    172800  NS    ROME.UCI

ICS.UCI             172800  A     192.5.19.1
ROME.UCI            172800  A     192.5.19.31

ISI                 172800  NS    VAXA.ISI
                    172800  NS    A.ISI
                    172800  NS    VENERA.ISI.EDU.
```

VAXA.ISI	172800	A	10.2.0.27
	172800	A	128.9.0.33
VENERA.ISI.EDU.	172800	A	10.1.0.52
	172800	A	128.9.0.32
A.ISI	172800	A	26.3.0.103
UDEL.EDU.	172800	NS	LOUIE.UDEL.EDU.
	172800	NS	UMN-REI-UC.ARPA.
LOUIE.UDEL.EDU.	172800	A	10.0.0.96
	172800	A	192.5.39.3
YALE.EDU.	172800	NS	YALE.ARPA.
YALE.EDU.	172800	NS	YALE-BULLDOG.ARPA.
MIT.EDU.	43200	NS	XX.LCS.MIT.EDU.
	43200	NS	ACHILLES.MIT.EDU.
XX.LCS.MIT.EDU.	43200	A	10.0.0.44
ACHILLES.MIT.EDU.	43200	A	18.72.0.8