Asia Pacific Network Information Centre

APNIC

DNS Concepts

APNIC 16, Seoul, Korea
19, August 2003

Acknowledgements

- Bill Manning
- Olaf M. Kolkman
- Ed Lewis
- Joe Abley

Overview

- Introduction to the DNS system
- DNS features & concepts
- Writing zone files
- Reverse DNS
- APNIC procedures

Purpose of naming

- Addresses are used to locate objects
- Names are easier to remember than numbers
- You would like to get to the address or other objects using a name
- **DNS provides a mapping from names to resources of several types**

Names and addresses in general

- An address is how you get to an endpoint
  - Typically, hierarchical (for scaling):
    - 950 Charter Street, Redwood City CA, 94063
    - 204.152.187.11, +1-650-381-6003
- A “name” is how an endpoint is referenced
  - Typically, no structurally significant hierarchy
    - “David”, “Tokyo”, “itu.int”

Naming History

- 1970's ARPANET
  - Host.txt maintained by the SRI-NIC
  - pulled from a single machine
  - Problems
    - traffic and load
    - Name collisions
    - Consistency
- DNS created in 1983 by Paul Mockapetris (RFCs 1034 and 1035), modified, updated, and enhanced by a myriad of subsequent RFCs

DNS

- A lookup mechanism for translating objects into other objects
- A globally distributed, loosely coherent, scalable, reliable, dynamic database
- Comprised of three components
  - A “name space”
  - Servers making that name space available
  - Resolvers (clients) which query the servers about the name space

DNS Features: Global Distribution

- Data is maintained locally, but retrievable globally
  - No single computer has all DNS data
- DNS lookups can be performed by any device
- Remote DNS data is locally cachable to improve performance

DNS Features: Loose Coherency

- The database is always internally consistent
  - Each version of a subset of the database (a zone) has a serial number
    - The serial number is incremented on each database change
- Changes to the master copy of the database are replicated according to timing set by the zone administrator
- Cached data expires according to timeout set by zone administrator

DNS Features: Scalability

- No limit to the size of the database
  - One server has over 20,000,000 names
    - Not a particularly good idea
- No limit to the number of queries
  - 24,000 queries per second handled easily
- Queries distributed among masters, slaves, and caches

DNS Features: Reliability

- Data is replicated
  - Data from master is copied to multiple slaves
- Clients can query
  - Master server
  - Any of the copies at slave servers
- Clients will typically query local caches

DNS Features: Dynamicity

- Database can be updated dynamically
  - Add/delete/modify of any record
- Modification of the master database triggers replication
  - Only master can be dynamically updated
    - Creates a single point of failure

Concept: DNS Names

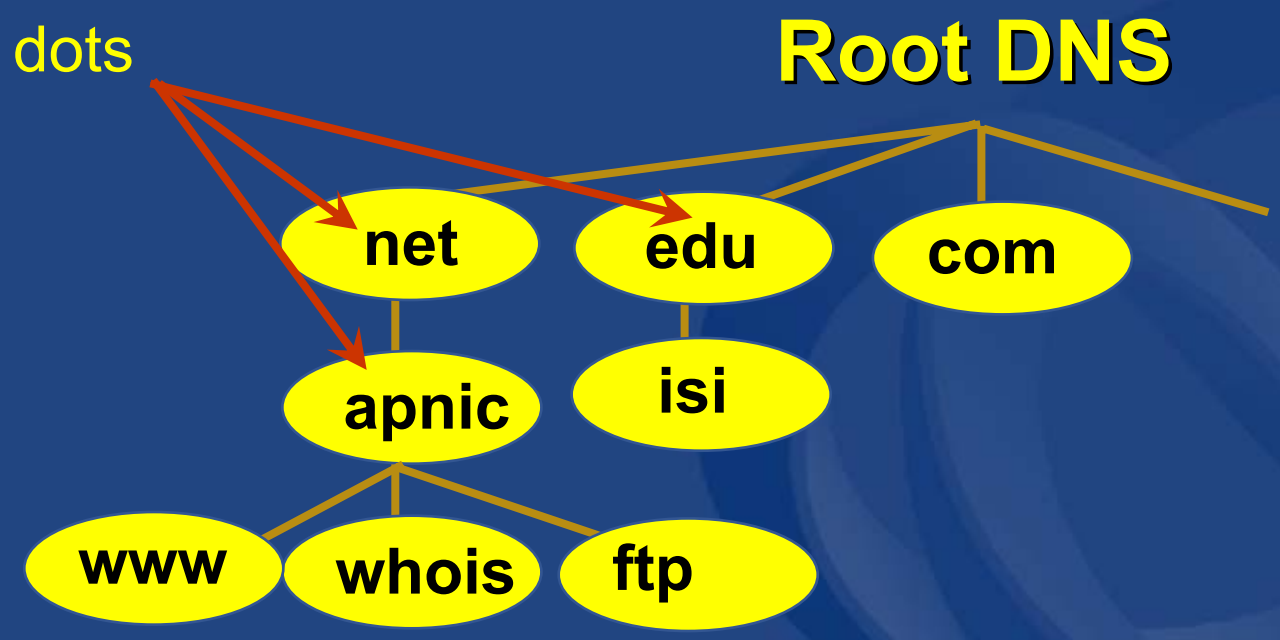
- The namespace needs to be made hierarchical to be able to scale.
- The idea is to name objects based on
  - location (within country, set of organizations, set of companies, etc)
  - unit within that location (company within set of company, etc)
  - object within unit (name of person in company)

Concept: DNS Names contd.

- How names appear in the DNS
  - Fully Qualified Domain Name (FQDN)
    - `WWW.APNIC.NET.`
  - labels separated by dots
- DNS provides a mapping from FQDNs to resources of several types
- Names are used as a key when fetching data in the DNS

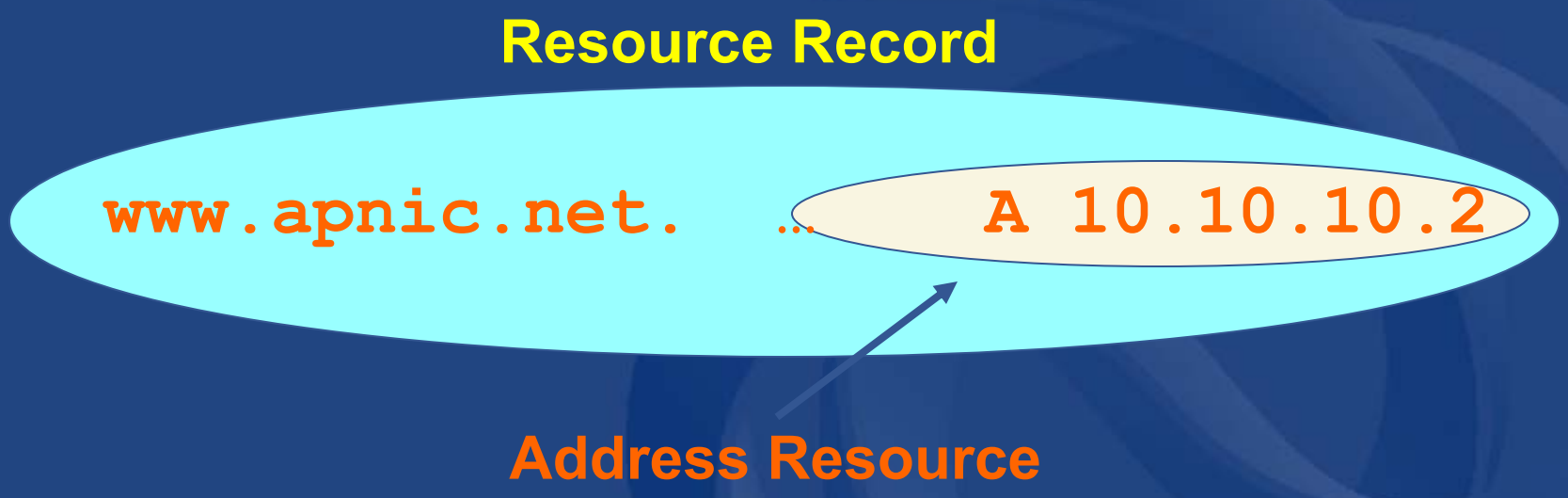
Concept: DNS Names contd.

- Domain names can be mapped to a tree
- New branches at the ‘dots’



Concept: Resource Records

- The DNS maps names into data using Resource Records.

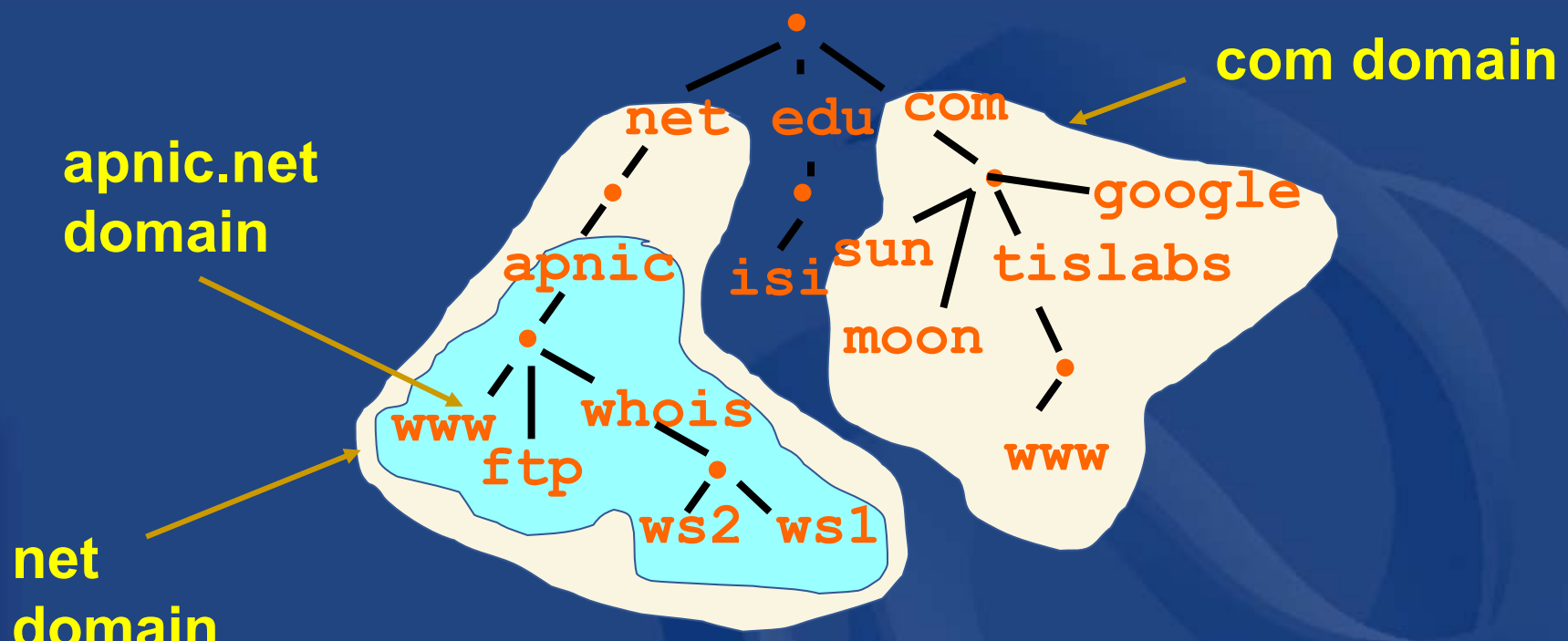


- More detail later

Concept: Domains

- Domains are “namespaces”
- Everything below *.com* is in the com domain
- Everything below *apnic.net* is in the apnic.net domain and in the net domain

Concept: Domains



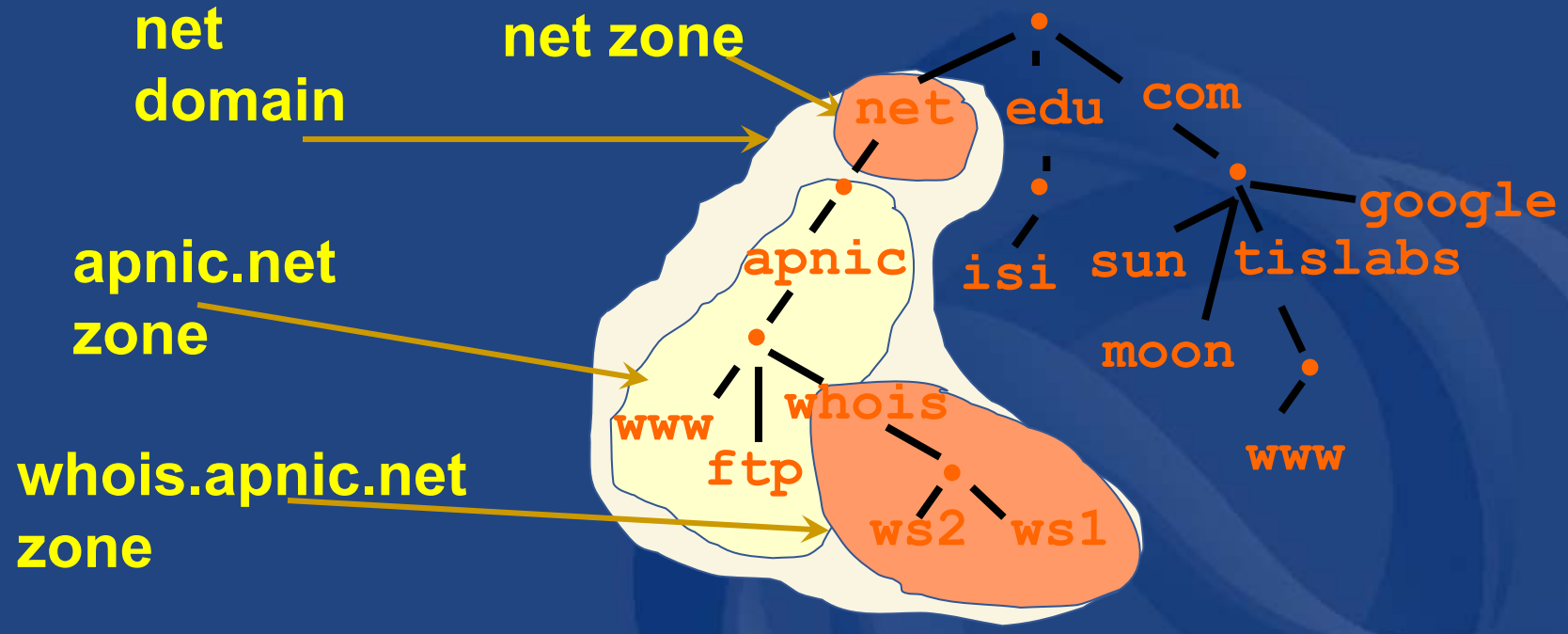
Delegation

- Administrators can create subdomains to group hosts
  - According to geography, organizational affiliation or any other criterion
- An administrator of a domain can delegate responsibility for managing a subdomain to someone else
  - But this isn't required
- The parent domain retains links to the delegated subdomain
  - The parent domain "remembers" who it delegated the subdomain to

Concept: Zones and Delegations

- Zones are “administrative spaces”
- Zone administrators are responsible for portion of a domain’s name space
- Authority is delegated from a parent and to a child

Concept: Zones and Delegations



Concept: Name Servers

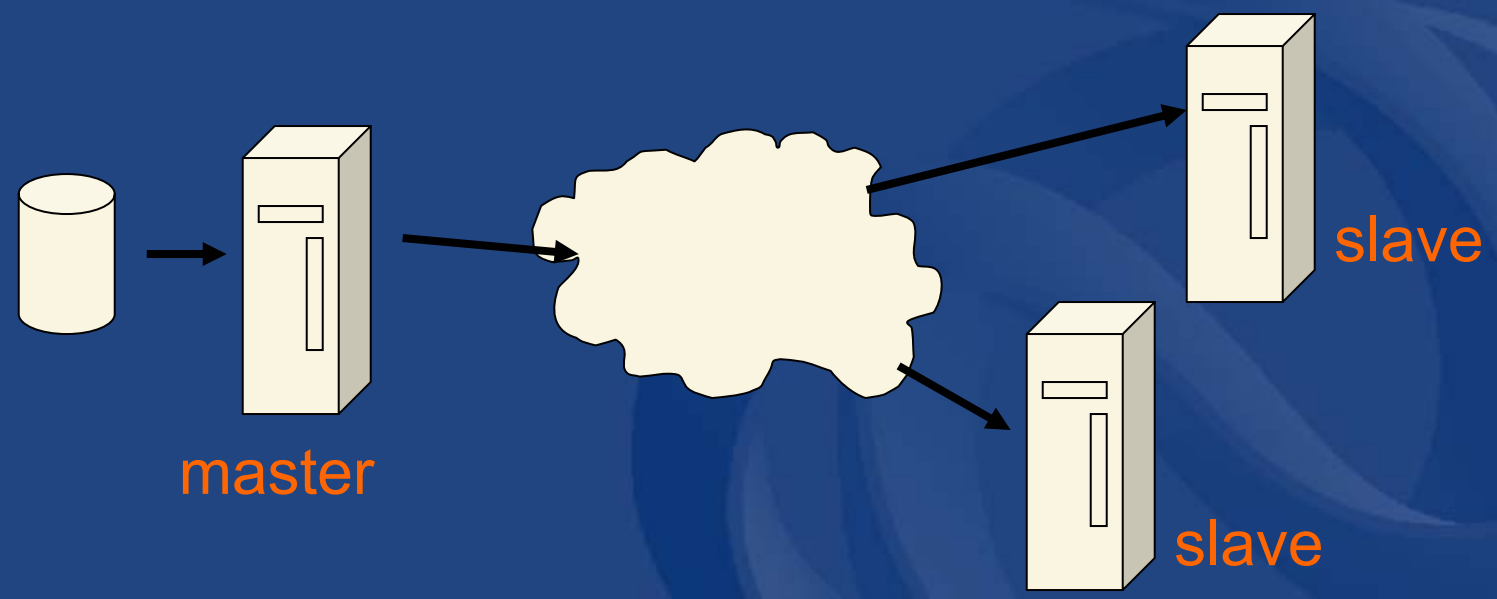
- Name servers answer ‘DNS’ questions
- Several types of name servers
  - Authoritative servers
    - master (primary)
    - slave (secondary)
  - (Caching) recursive servers
    - also caching forwarders
  - Mixture of functionality

Concept: Name Servers contd.

- Authoritative name server
  - Give authoritative answers for one or more zones
  - The master server normally loads the data from a zone file
  - A slave server normally replicates the data from the master via a zone transfer

Concept: Name Servers contd.

- Authoritative name server



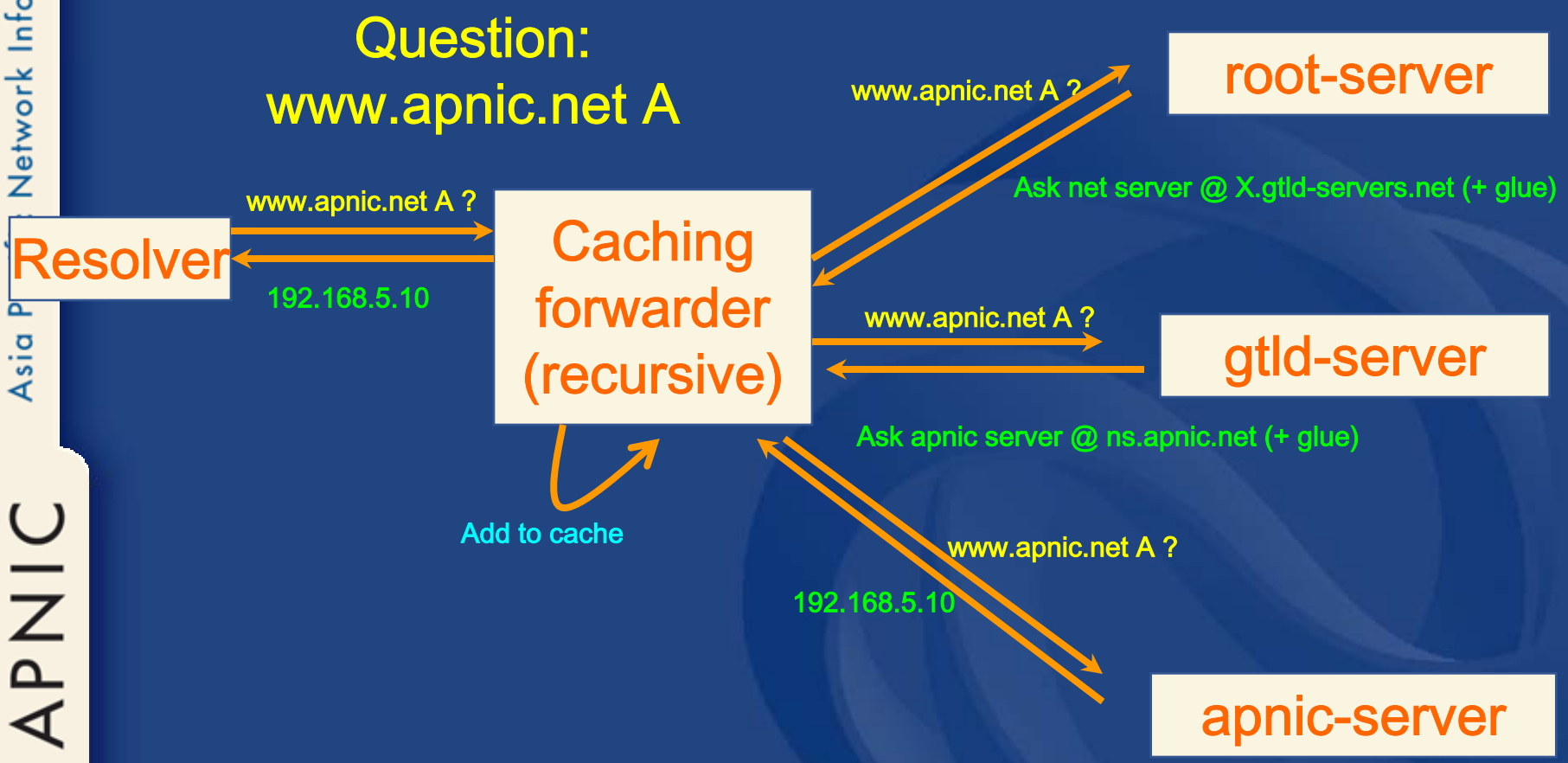
Concept: Name Servers contd.

- Recursive server
  - Do the actual lookups; ask questions to the DNS on behalf of the clients
  - Answers are obtained from authoritative servers but the answers forwarded to the clients are marked as not authoritative
  - Answers are stored for future reference in the cache

Concept: Resolvers

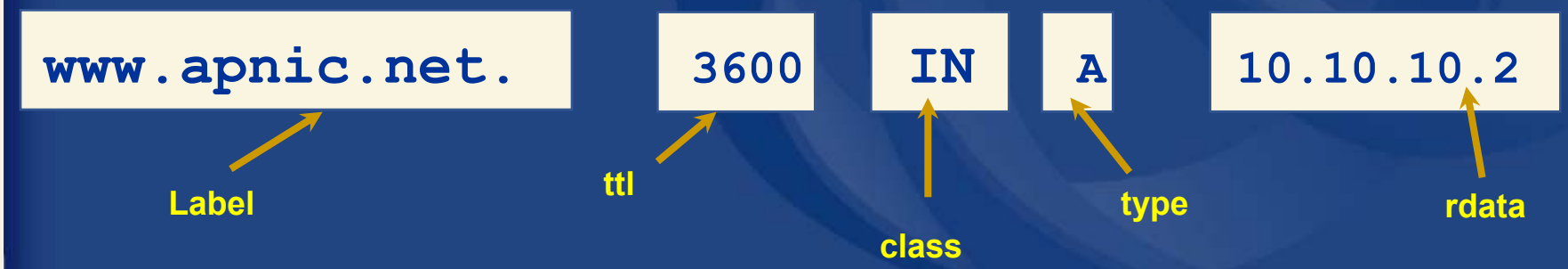
- Resolvers ask the questions to the DNS system on behalf of the application
- Normally implemented in a system library (e.g, libc)

Concept: Resolving process & Cache



Concept: Resource Records

- Resource records consist of it's name, it's TTL, it's class, it's type and it's RDATA
- TTL is a timing parameter
- IN class is widest used
- There are multiple types of RR records
- Everything behind the type identifier is called rdata



Example: RRs in a zone file

```
apnic.net. 7200 IN      SOA     ns.apnic.net.
   admin.apnic.net.        (
                           2001061501     ; Serial
                           43200  ; Refresh 12 hours
                           14400  ; Retry 4 hours
                           345600 ; Expire 4 days
                           7200 ; Negative cache 2
                                  hours    )

apnic.net.          7200  IN    NS      ns.apnic.net.
apnic.net.          7200  IN    NS      ns.eu.net.

whois.apnic.net.    3600  IN    A       193.0.1.162
host25.apnic.net.   2600  IN    A       193.0.3.25
```

Label ttl class type rdata

Resource Record: SOA and NS

- The SOA and NS records are used to provide information about the DNS itself
- The NS indicates where information about a given zone can be found
- The SOA record provides information about the start of authority, i.e. the top of the zone, also called the APEX

Resource Record: SOA

Contact address

Master server

```
net. 3600 IN SOA   A.GTLD-SERVERS.net.   nstld.verisign-grs.com.  (
                                          2002021301      ; serial
                                          30M             ; refresh
                                          15M             ; retry
                                          1W              ; expiry
                                          1D )            ; neg.answ.ttl
```

Version number

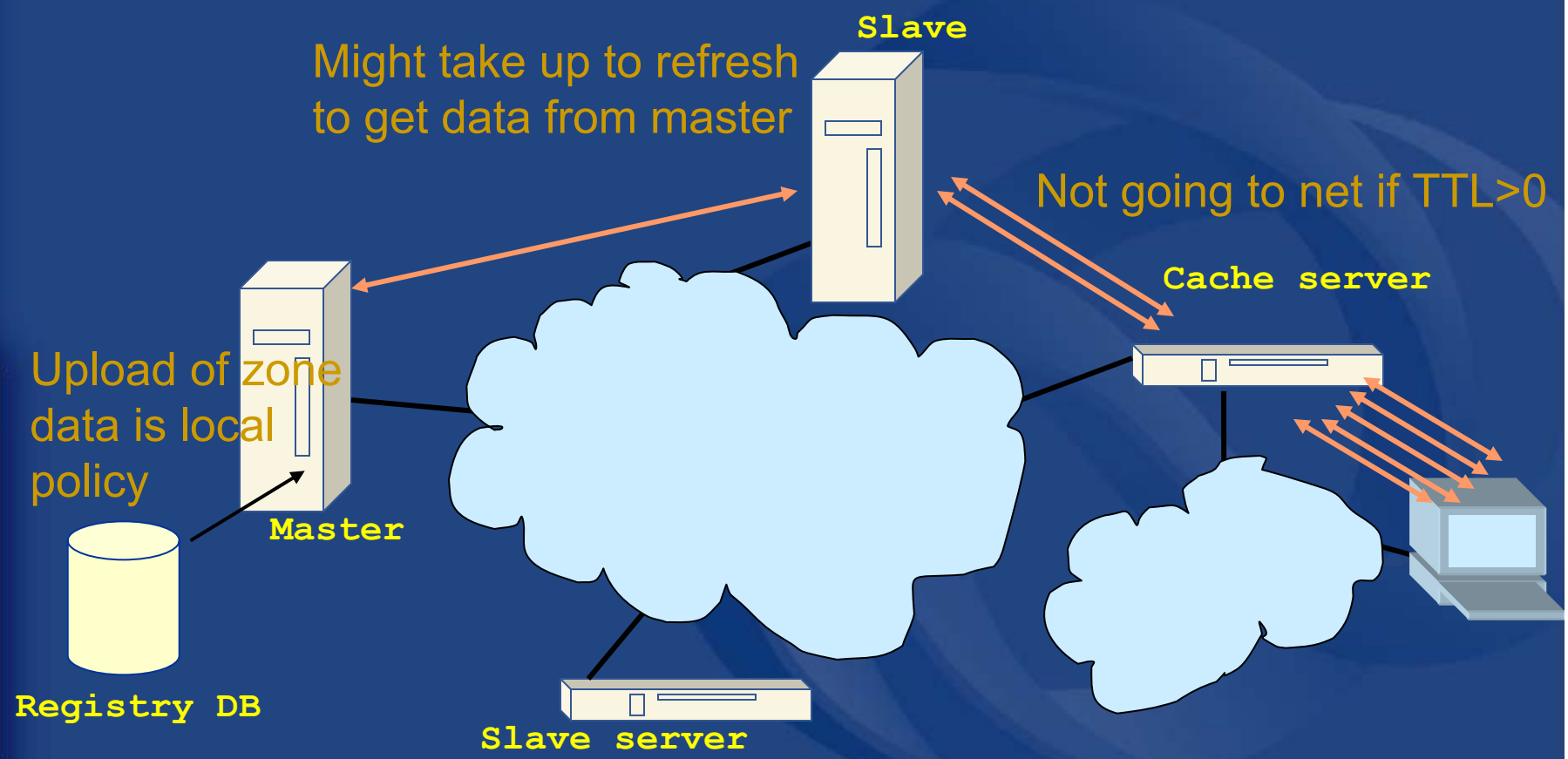
Timing parameter

Concept: TTL and other Timers

- TTL is a timer used in caches
  - An indication for how long the data may be reused
  - Data that is expected to be ‘stable’ can have high TTLs
- SOA timers are used for maintaining consistency between primary and secondary servers

Places where DNS data lives

- Changes do not propagate instantly



To remember...

- Multiple authoritative servers to distribute load and risk:
  - Put your name servers apart from each other
- Caches to reduce load to authoritative servers and reduce response times
- SOA timers and TTL need to be tuned to needs of zone. Stable data: higher numbers

What have we learned so far

- We learned about the architectures of
  - resolvers,
  - caching forwarders,
  - authoritative servers,
  - timing parameters

- We continue writing a zone file

Writing a zone file

- Zone file is written by the zone administrator
- Zone file is read by the master server and it's content is replicated to slave servers
- What is in the zone file will end up in the database
- Because of timing issues it might take some time before the data is actually visible at the client side

First attempt

- The ‘header’ of the zone file
  - Start with a SOA record
  - Include authoritative name servers and, if needed, glue
  - Add other information
- Add other RRs
- Delegate to other zones

The SOA record

Comments

```
apnic.net. 3600 IN SOA ns.apnic.net.
   admin\.email.apnic.net. (
                        2002021301      ; serial
                        1h              ; refresh
                        30M             ; retry
                        1W              ; expiry
                        3600 )          ; neg. answ. ttl
```

- admin.email@apnic.net admin\.email.apnic.net
- Serial number: 32bit circular arithmetic
  - People often use date format
  - To be increased after editing
- The timers above qualify as reasonable

Authoritative NS records and related A records

```
netsa.org.          3600 IN NS  IN.netsa.org.
netsa.org.          3600 IN NS  LK.netsa.org.
IN.netsa.org.       3600 IN A  193.0.0.4
LK.netsa.org.       3600 IN A  193.0.0.202
```

- NS record for all the authoritative servers
  - They need to carry the zone at the moment you publish
- A records only for “in-zone” name servers
  - Delegating NS records might have glue associated

Other data in the zone

```
localhost.netsa.org. 3600  IN    A      127.0.0.1
IN.netsa.org.        4500  IN    A      193.0.0.4
www.netsa.org.       3600  IN    CNAME  IN.netsa.org.
```

- Add all the other data to your zone file
- Some notes on notation
  - Note the fully qualified domain name including trailing dot
  - Note TTL and CLASS

Zone file format short cuts nice formatting

```
netsa.org.          3600  IN SOA IN.netsa.org.
admin\.email.netsa.org. (
                                  2002021301    ; serial
                                  1h            ; refresh
                                  30M           ; retry
                                  1W            ; expiry
                                  3600 )        ; neg. answ. Ttl

netsa.org.              3600 IN NS   IN.netsa.org.
netsa.org.              3600 IN NS   LK.netsa.org.
netsa.org.              3600 IN MX   50   mailhost.netsa.org.
netsa.org.              3600 IN MX   150 mailhost2.netsa.org.

netsa.org.              3600 IN TXT  "Demonstration and test zone"
IN.netsa.org.           4500 IN A    193.0.0.4
LK.netsa.org.           3600 IN A    193.0.0.202
localhost.netsa.org.    3600 IN A    127.0.0.1

IN.netsa.org.           3600 IN A     193.0.0.4
www.netsa.org.          3600 IN CNAME IN.netsa.org.
```

Zone file format short cuts: repeating last name

```
netsa.org.          3600  IN SOA IN.netsa.org.
admin\.email.netsa.org. (
                                 2002021301      ; serial
                                 1h              ; refresh
                                 30M             ; retry
                                 1W              ; expiry
                                 3600 )          ; neg. answ. Ttl
                        3600 IN NS   IN.netsa.org.
                        3600 IN NS   LK.netsa.org.
                        3600 IN MX   50   mailhost.netsa.org.
                        3600 IN MX   150 mailhost2.netsa.org.

                        3600 IN TXT  "Demonstration and test zone"
IN.netsa.org.           3600 IN A    193.0.0.4
LK.netsa.org.           3600 IN A    193.0.0.202

localhost.netsa.org.    4500 IN A    127.0.0.1

IN.netsa.org.           3600 IN A    193.0.0.4
www.netsa.org.          3600 IN CNAME IN.netsa.org.
```

Zone file format short cuts: default TTL

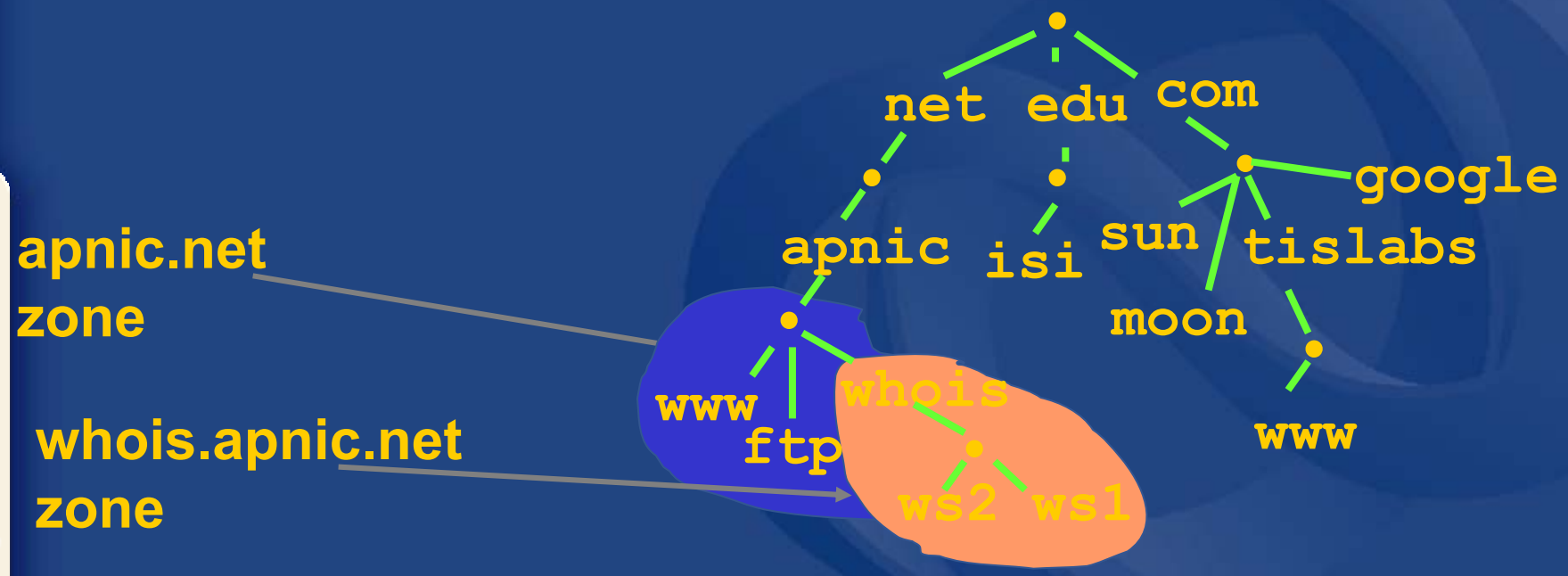
```
$TTL    3600 ; Default TTL directive
netsa.org.      IN SOA IN.netsa.org. admin\.email.netsa.org. (
                                2002021301    ; serial
                                1h            ; refresh
                                30M           ; retry
                                1W            ; expiry
                                3600 )        ; neg. answ. Ttl
                        IN NS   IN.netsa.org.
                        IN NS   LK.netsa.org.
                        IN MX   50 mailhost.netsa.org.
                        IN MX   150 mailhost2.netsa.org.

                        IN TXT  "Demonstration and test zone"
IN.netsa.org.           IN A   193.0.0.4
LK.netsa.org.           IN A   193.0.0.202

localhost.netsa.org.    IN A    127.0.0.1

IN.netsa.org.    4500   IN A    193.0.0.4
www.netsa.org.          IN CNAME IN.netsa.org.
```

Zone file format short cuts: ORIGIN

```
$TTL    3600 ; Default TTL directive
$ORIGIN netsa.org.
@               IN SOA IN admin\.email.netsa.org. (
                                2002021301     ; serial
                                1h             ; refresh
                                30M            ; retry
                                1W             ; expiry
                                3600 )         ; neg. answ. Ttl
                IN NS    IN
                IN NS    LK
                IN MX    50  mailhost
                IN MX    150 mailhost2


                IN TXT   "Demonstration and test zone"
IN              IN A     193.0.0.4
LK              IN A     193.0.0.202

localhost       IN A    127.0.0.1

IN         4500 IN A     193.0.0.4
www             IN CNAME IN
```

Delegating a zone (becoming a parent)

- Delegate authority for a sub domain to another party (splitting of *whois.apnic.net* from *apnic.net*)



Concept: Glue

- Delegation is done by adding NS records:

```
whois.apnic.net.      NS ns1.whois.apnic.net.
whois.apnic.net.      NS ns2.whois.apnic.net.
```

- How to get to ns1 and ns2… We need the addresses

- Add glue records to so that resolvers can reach ns1 and ns2

```
ns1.whois.apnic.net. A 10.0.0.1
ns2.whois.apnic.net. A 10.0.0.2
```

Concept: Glue contd.

- Glue is ‘non-authoritative’ data
- Don’t include glue for servers that are not in sub zones

```
whois.apnic.net.     NS    ns1.whois.apnic.net.
whois.apnic.net.     NS    ns2.apnic.net.
whois.apnic.net.     NS    ns1.apnic.net.
ns1.whois.apnic.net. A     10.0.0.1
```

Only this record needs glue

Delegating whois.apnic.net. from apnic.net.

whois.apnic.net

- Setup minimum two servers
- Create zone file with NS records
- Add all whois.apnic.net data

apnic.net

- Add NS records and glue
- Make sure there is no other data from the whois.apnic.net. zone in the zone file

Reverse DNS

Overview

- Principles
- Creating reverse zones
- Setting up nameservers
- Reverse delegation procedures
- IPv6 reverse delegations
- Current status

What is ‘Reverse DNS’?

- ‘Forward DNS’ maps names to numbers
  - svc00.apnic.net -> 202.12.28.131
- ‘Reverse DNS’ maps numbers to names
  - 202.12.28.131 -> svc00.apnic.net

Reverse DNS - why bother?

- Service denial
  - That only allow access when fully reverse delegated eg. anonymous ftp
- Diagnostics
  - Assisting in trace routes etc
- Registration
  - Responsibility as a member and Local IR

In-addr.arpa

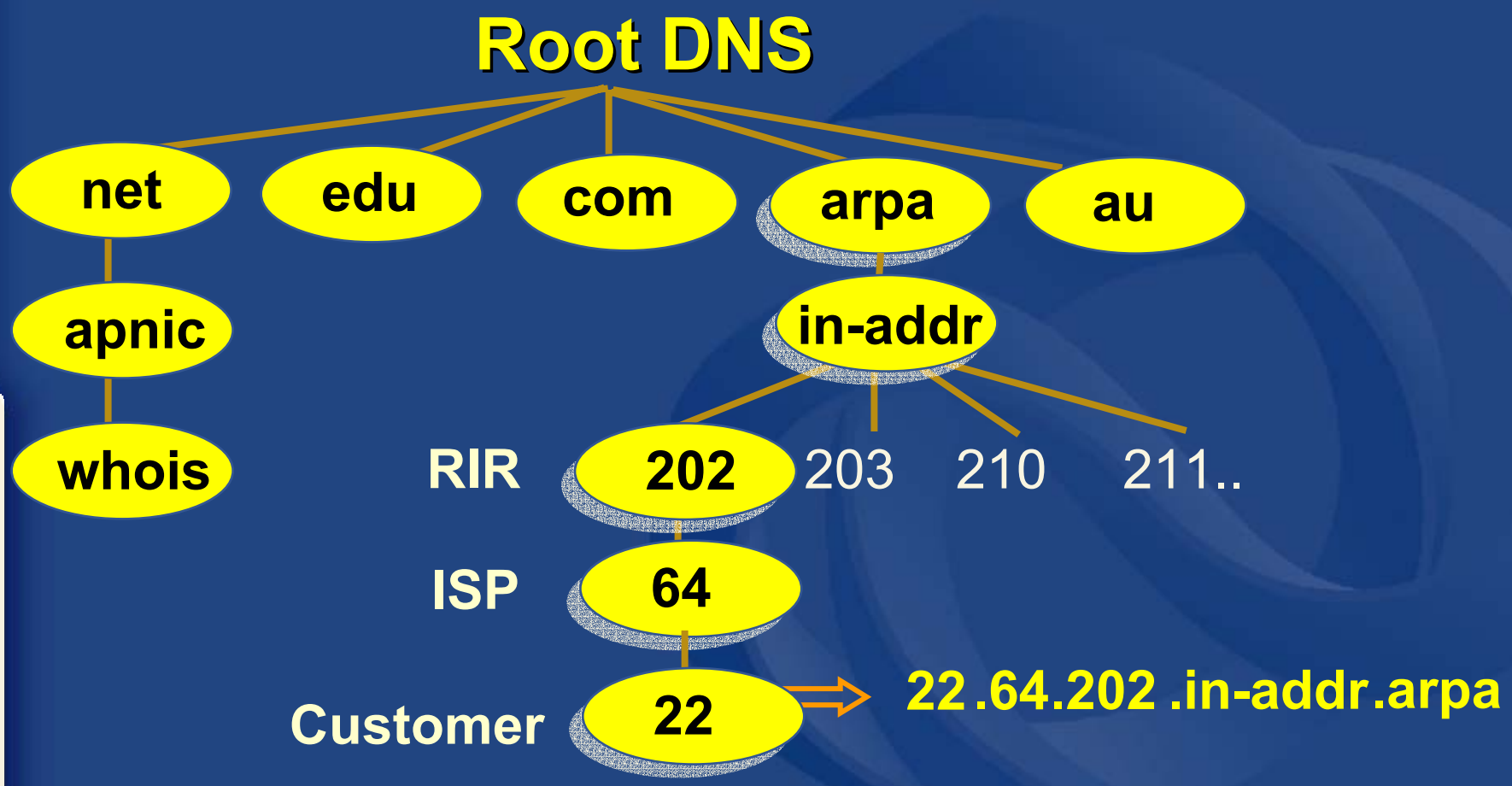
- Hierarchy of IP addresses
  - Uses ‘in-addr.arpa’ domain
    - INverse ADDRess
- IP addresses:
  - Less specific to More specific
    - 210.56.14.1
- Domain names:
  - More specific to Less specific
    - delhi.vsnl.net.in
  - Reversed in in-addr.arpa hierarchy
    - 14.56.210.in-addr.arpa

Principles

- Delegate maintenance of the reverse DNS to the custodian of the address block
- Address allocation is hierarchical
  - LIRs/ISPs -> Customers -> End users

Principles – DNS tree

- Mapping numbers to names - 'reverse DNS'



Creating reverse zones

- Same as creating a forward zone file
  - SOA and initial NS records are the same as normal zone
  - Main difference
    - need to create additional PTR records
- Can use BIND or other DNS software to create and manage reverse zones
  - Details can be different

Creating reverse zones - contd

- Files involved
  - Zone files
    - Forward zone file
      - e.g. db.domain.net
    - Reverse zone file
      - e.g. db.192.168.254
  - Config files
    - <named.conf>
  - Other
    - Hints files etc.
      - Root.hints

Start of Authority (SOA) record

admin.test-domain.net

ns.test-domain.net.

```
<domain.name.>          CLASS   SOA     <hostname.domain.name.>
   <mailbox.domain.name> (
                                          <serial-number>
                                          <refresh>
                                          <retry>
                                          <expire>
                                          <negative-caching> )
```

253.253.192.in-addr.arpa.

IN

(<2003033101>
<10800>
<3600>
<604800>
<10800>)

Nameserver (NS) records

- Declares the nameservers that serve a given zone

```
<domain.name.> IN NS <hostname.domain.name.>
```

ns.apnic.net.

Pointer (PTR) records

- Create pointer (PTR) records for each IP address

```
131.28.12.202.in-addr.arpa. IN PTR svc00.apnic.net.
```

or

```
131          IN     PTR          svc00.apnic.net.
```

A reverse zone example

```
$ORIGIN 1.168.192.in-addr.arpa.
@       3600  IN SOA test.company.org. (
                     sys\.admin.company.org.
                     2002021301     ; serial
                     1h             ; refresh
                     30M            ; retry
                     1W             ; expiry
                     3600 )         ; neg. answ. ttl

        NS      ns.company.org.
        NS      ns2.company.org.

1       PTR     gw.company.org.
                router.company.org.

2       PTR     ns.company.org.
;auto generate:  65 PTR host65.company.org
$GENERATE 65-127 $ PTR host$.company.org.
```

What we covered so far

- Why Reverse DNS ?
- The DNS tree ?
- Files involved
- Essential Resource Records
- How to create reverse zones

Setting up the primary nameserver

- Add an entry specifying the primary server to the ***named.conf*** file

```
zone "<domain-name>" in {
type master;
file "<path-name>"; };
```

- <domain-name>
  - Ex: 28.12.202.in-addr.arpa.
- <type master>
  - Define the name server as the primary
- <path-name>
  - location of the file that contains the zone records

Setting up the secondary nameserver

- Add an entry specifying the primary server to the ***named.conf*** file

```
zone "<domain-name>" in {
type slave;
file "<path-name>";
Masters { <IP address> ; }; };
```

- <type slave> defines the name server as the secondary
- <ip address> is the IP address of the primary name server
- <domain-name> is same as before
- <path-name> is where the back-up file is

Reverse delegation requirements

- /24 Delegations
  - Address blocks should be assigned/allocated
  - At least two name servers
- /16 Delegations
  - Same as /24 delegations
  - APNIC delegates entire zone to member
  - Recommend APNIC secondary zone
- < /24 Delegations
  - Read “classless in-addr.arpa delegation”

RFC 2317

APNIC & ISPs responsibilities

- APNIC
  - Manage reverse delegations of address block distributed by APNIC
  - Process members requests for reverse delegations of network allocations
- ISPs
  - Be familiar with APNIC procedures
  - Ensure that addresses are reverse-mapped
  - Maintain nameservers for allocations
    - Minimise pollution of DNS

Subdomains of in-addr.arpa domain

- Subnetting on an Octet Boundary
  - Similar to delegating subdomains of forward-mapping domains
- Mapping problems
  - In IPv4 the mapping is done on 8 bit boundaries (class full), address allocation is classless
  - Zone administration does not always overlap address administration

Subdomains of in-addr.arpa domain

- Example: an organisation given a /16
  - 192.168.0.0/16 (one zone file and further delegations to downstreams)
  - Zone file should have:

```
0.168.192.in-addr.arpa.     NS ns1.organisation0.com.
0.168.192.in-addr.arpa.     NS ns2.organisation0.com.
1.168.192.in-addr.arpa.     NS ns1.organisation1.com.
1.168.192.in-addr.arpa.     NS ns2.organisation1.com.
2.168.192.in-addr.arpa.     NS ns1.organisation2.com.
2.168.192.in-addr.arpa.     NS ns2.organisation2.com.
        :
        :
```

Subdomains of in-addr.arpa domain

- Example: an organisation given a /19
  - 192.168.0.0/19 (a lot of zone files!) – have to do it per /24)
  - Zone files

0.168.192.in-addr.arpa.
1.168.192.in-addr.arpa.
2.168.192.in-addr.arpa.
:
:
31.168.192.in-addr.arpa.

Subdomains of in-addr.arpa domain

- Example: case of a /24 subnetted with the mask 255.255.255.192
  - In-addr zone – 254.253.192.in-addr.arpa
  - Subnets
    - 192.253.254.0/26
    - 192.253.254.64/26
    - 192.253.254.128/26
    - 192.253.254.192/26
  - If different organisations has to manage the reverse-mapping for each subnet
    - Solution to follow…

Classless in-addr

- CNAME records for each of the domain names in the zone
  - Pointing to domain names in the new subdomains

```
 1.254.253.192.in-addr.arpa.          IN  CNAME   1.0-63.254.253.192.in-addr.arpa.
2.254.253.192.in-addr.arpa.           IN  CNAME   2.0-63.254.253.192.in-addr.arpa.
:
0-63.254.253.192.in-addr.arpa.        86400   IN  NS          ns1.organisation1.com.
0-63.254.253.192.in-addr.arpa.        86400   IN  NS          ns2.organisation1.com.

65.254.253.192.in-addr.arpa.          IN CNAME    65.64-127.254.253.192.in-addr.arpa.
66.254.253.192.in-addr.arpa.          IN CNAME    66.64-127.254.253.192.in-addr.arpa.
:
64-127.254.253.192.in-addr.arpa.      86400  IN   NS          ns1.organisation2.com.
64-127.254.253.192.in-addr.arpa.      86400  IN   NS          ns2.organisation2.com.

129.254.253.192.in-addr.arpa.         IN CNAME     129.128-191.254.253.192.in-addr.arpa.
130.254.253.192.in-addr.arpa.         IN CNAME     130.128-191.254.253.192.in-addr.arpa.
:
128-191.254.253.192.in-addr.arpa.     86400  IN  NS          ns1.organisation3.com.
128-191.254.253.192.in-addr.arpa.     86400  IN  NS          ns2.organisation3.com.
:
:
```

Classless in-addr

- Using $GENERATE (db.192.253.254 file)

```
$GENERATE 1-63  $   IN  CNAME            $.0-63.254.253.192.in-addr.arpa.

0-63.254.253.192.in-addr.arpa.   86400   IN  NS        ns1.organisation1.com.
0-63.254.253.192.in-addr.arpa.   86400   IN  NS        ns2.organisation1.com.

$GENERATE 65-127 $   IN CNAME            $.64-127.254.253.192.in-addr.arpa.

64-127.254.253.192.in-addr.arpa.         86400  IN   NS        ns1.organisation2.com.
64-127.254.253.192.in-addr.arpa.         86400  IN   NS        ns2.organisation2.com.

$GENERATE 129-191 $   IN CNAME           $.128-191.254.253.192.in-addr.arpa.

128-191.254.253.192.in-addr.arpa.        86400  IN  NS        ns1.organisation3.com.
128-191.254.253.192.in-addr.arpa.        86400  IN  NS        ns2.organisation3.com.
:
:
```

Classless in-addr

- Now, the zone data file for 0-63.254.253.192.in-addr.arpa can contain just PTR records for IP addresses 192.253.254.1 through 192.253.154.63

```
$TTL 1d
@    IN   SOA   ns1.organisation1.com. Root.ns1.organisation1.com.
(
                                1        ; Serial
                                3h       ; Refresh
                                1h       ; Retry
                                1w       ; Expire
                                1h )     ; Negative caching TTL
        IN    NS         ns1.organisation1.com.
        IN    NS         ns2.organisation1.com.

      1 IN    PTR        org1-name1.organisation1.com.
      2 IN    PTR        org1-name2.organisation1.com.
      3 IN    PTR        org1-name3.organisation1.com.
```

APNIC reverse delegation procedures

- Upon allocation, member is asked if they want /24 place holder domain objects with member maintainer
  - Gives member direct control
- Standard APNIC database object,
  - can be updated through online form or via email.
- Nameserver/domain set up verified before being submitted to the database.
- Protection by maintainer object
  - (current auths: CRYPT-PW, PGP).
- Zone file updated 2-hourly

APNIC reverse delegation procedures

- Complete the documentation
  - http://www.apnic.net/db/domain.html
- On-line form interface
  - Real time feedback
  - Gives errors, warnings in zone configuration
    - serial number of zone consistent across nameservers
    - nameservers listed in zone consistent
  - Uses database ‘domain’ object

Reverse delegation request form

Create Domain Object - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address http://www.apnic.net/apnic-bin/creform.pl

Asia Pacific Network Information Centre

APNIC Info & FAQ | Resource services | Training | Meetings | Membership | Documents | Whois & Search | Internet community

Create Domain Object

Domain Object

What is this form to be used for?
This form assists in the creation and maintenance of domain objects. The domain class:

(* indicates mandatory field)

*Domain:

*Descr:

Please change this field - This is added by http://www.apnic.net/db/domain.html The reverse delegation zone for the

Country:

*Admin-c:

An admin must be someone physically located at the site of the network.

Reverse delegation request form

Create Domain Object - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address http://www.apnic.net/apnic-bin/creform.pl Go

*Nserver:

dns.vsnl.net.in
giasbm01.vsnl.net.in

Remarks:

Notify:

This email address will be notified by the APNIC database when this object changes

*Mnt-by:

MAINT-WF-EX

*Password:

You must supply a password for one of the maintainers listed in this field

Mnt-lower:

This stops ad-hoc additions beneath this zone.

Online errors (also via email)

Domain Update Results - Microsoft Internet Explorer

File Edit View Go Favorites Help

Please wait while your request is processed.

Parsing and validating your submission …

Errors encountered

Your update request was unable to be completed due to the following errors. Please correct them and try again. If the error is temporary in nature, correct the error and 'Reload' this page (possibly this frame).

- *ERROR*. SOA on "ns.apnic.net" does not match SOA on "svc00.apnic.net". All nservers must respond with the same SOA.
- *ERROR*. NS RR for ns.telstra.net found on svc00.apnic.net but not in supplied template.
- *ERROR*. NS RR for ns.telstra.net found on svc00.apnic.net but not in supplied template.
- *ERROR*. NS RR for ns.telstra.net found on ns.apnic.net but not in supplied template.
- *ERROR*. NS RR for ns.telstra.net found on ns.apnic.net but not in supplied template.
- *ERROR*. cross-check of listed NS RR failed.

Internet zone

Request submission error

Domain Update Results - Microsoft Internet Explorer

File Edit View Go Favorites Help

Verifying your authorisation ….

Your maintainer uses the 'CRYPT-PW' or 'NONE' authorisation schema. Attempting to submit your request directly to the database.

Update results

```
Connection closed.

% Rights restricted by copyright. See
http://www.apnic.net/db/dbcopyright.html
Update FAILED: [domain] 174.202.in-addr-arpa

domain:    174.202.in-addr.arpa
descr:     in-addr.arpa zone for 202.174/16
admin-c:   DNS3-AP
tech-c:    DNS3-AP
zone-c:    DNS3-AP
nserver:   ns.apnic.net
nserver:   svc00.apnic.net
mnt-by:    MAINT-AP-DNS-DEFAULT
changed:   dns-admin@apnic.net 20000215
source:    APNIC
*ERROR*:   authorisation failed, request forwarded to maintainer
```

Update failed

Authorisation failed

Processing completed

Internet zone

Asia Pacific Network Information Centre

APNIC

APNIC reverse delegation procedures - Evaluation

- Parser checks for
  - ‘whois’ database
    - IP address range is assigned or allocated
    - Must be in APNIC database
  - Maintainer object
    - Mandatory field of domain object
  - Nic-handles
    - zone-c, tech-c, admin-c

APNIC reverse delegation procedures - Evaluation

- Nameserver checks
  - Minimum 2 nameservers required
  - Check serial versions of zone files are the same
  - Check NS records in zones are the same as listed on form
  - Nameserver can resolve itself, forward and reverse

Successful update

Domain Update Results - Microsoft Internet Explorer

File Edit View Go Favorites Help

Please wait while your request is processed.

Parsing and validating your submission …

Warnings generated

•

Verifying your authorisation ….

Your maintainer uses the 'CRYPT-PW' or 'NONE' authorisation schema. Attempting to submit your request directly to the database.

Update results

```
Connection closed.
% Rights restricted by copyright. See http://www.apnic.net/db/dbcopyright.html
Update OK: [domain] 174.202.in-addr.arpa
```

Update ok!

Processing completed.

Internet zone

Whois domain object

```
domain:       28.12.202.in-addr.arpa
descr:        in-addr.arpa zone for 28.12.202.in-addr.arpa
admin-c:      DNS3-AP
tech-c:       DNS3-AP
zone-c:       DNS3-AP
nserver:      ns.telstra.net
nserver:      rs.arin.net
nserver:      ns.myapnic.net
nserver:      svc00.apnic.net
nserver:      ns.apnic.net
mnt-by:       MAINT-APNIC-AP
mnt-lower:    MAINT-DNS-AP
changed:      inaddr@apnic.net 19990810
source:       APNIC
```

Reverse Zone

Contacts

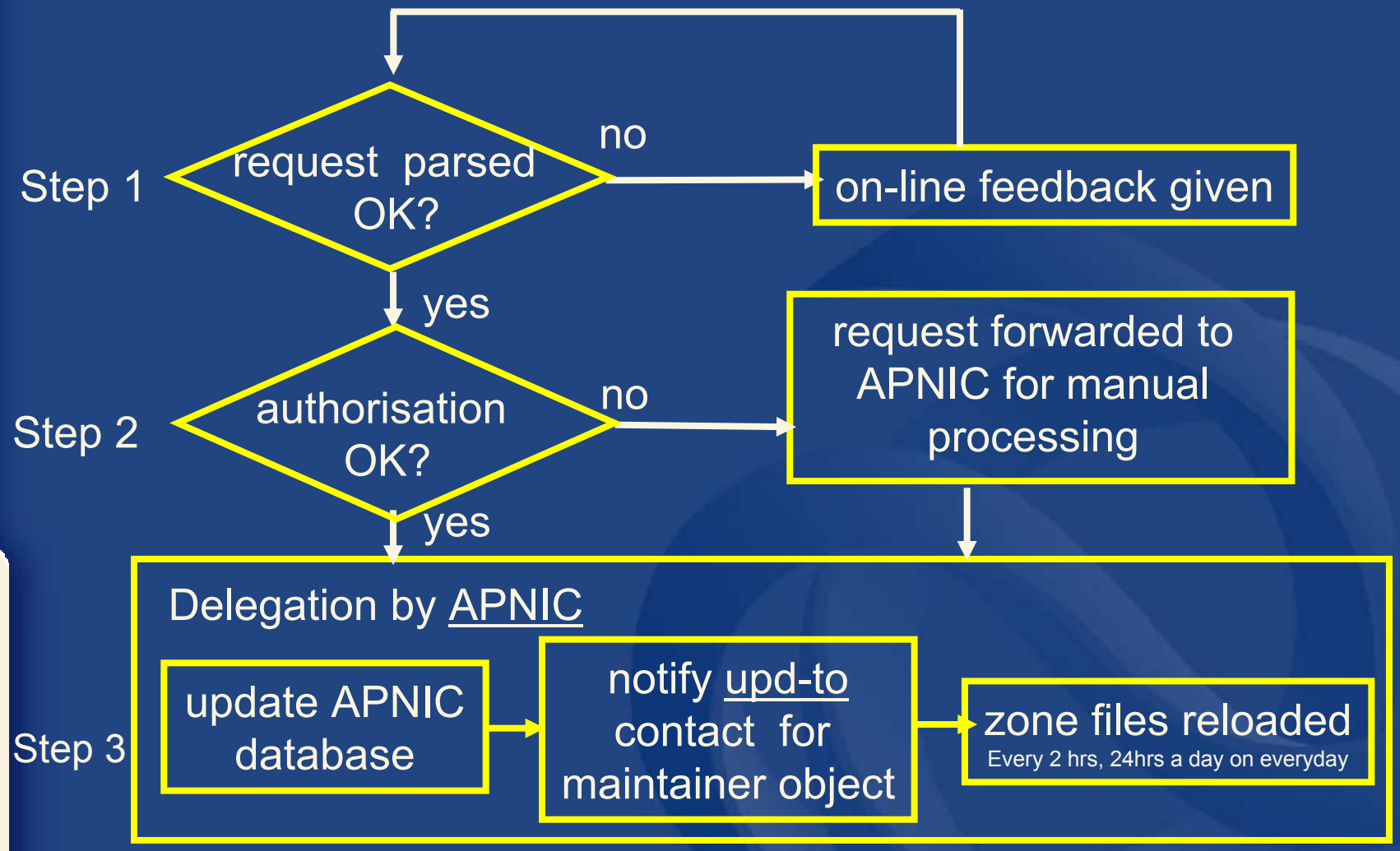
Name Servers

Maintainers (protection)

Use of maintainer object

- Domain objects protected by maintainers
  - hierarchical protection using “mnt-lower”
- Bootstrap period
  - ‘MAINT-AP-DNS-DEFAULT’ for all objects imported by APNIC from existing zone files
    - Changing delegations requires valid maintainer
    - Maintainer creation & authorisation is manual
      - Turnaround time 2 days
  - /24 place holder objects created upon allocation gives members direct control
    - No need to contact APNIC when changing nservers

Delegation process summary



Reverse DNS Troubleshooting Guide:
http://www.apnic.net/services/help/rd/troubleshooting.html

What we covered so far

- Why Reverse DNS ?
- The DNS tree
- Files involved
- Essential Resource Records
- How to create reverse zones

- Setting up nameservers – config files
- APNIC reverse delegation requirements
- Classless in-addr.arpa
- APNIC reverse delegation procedures

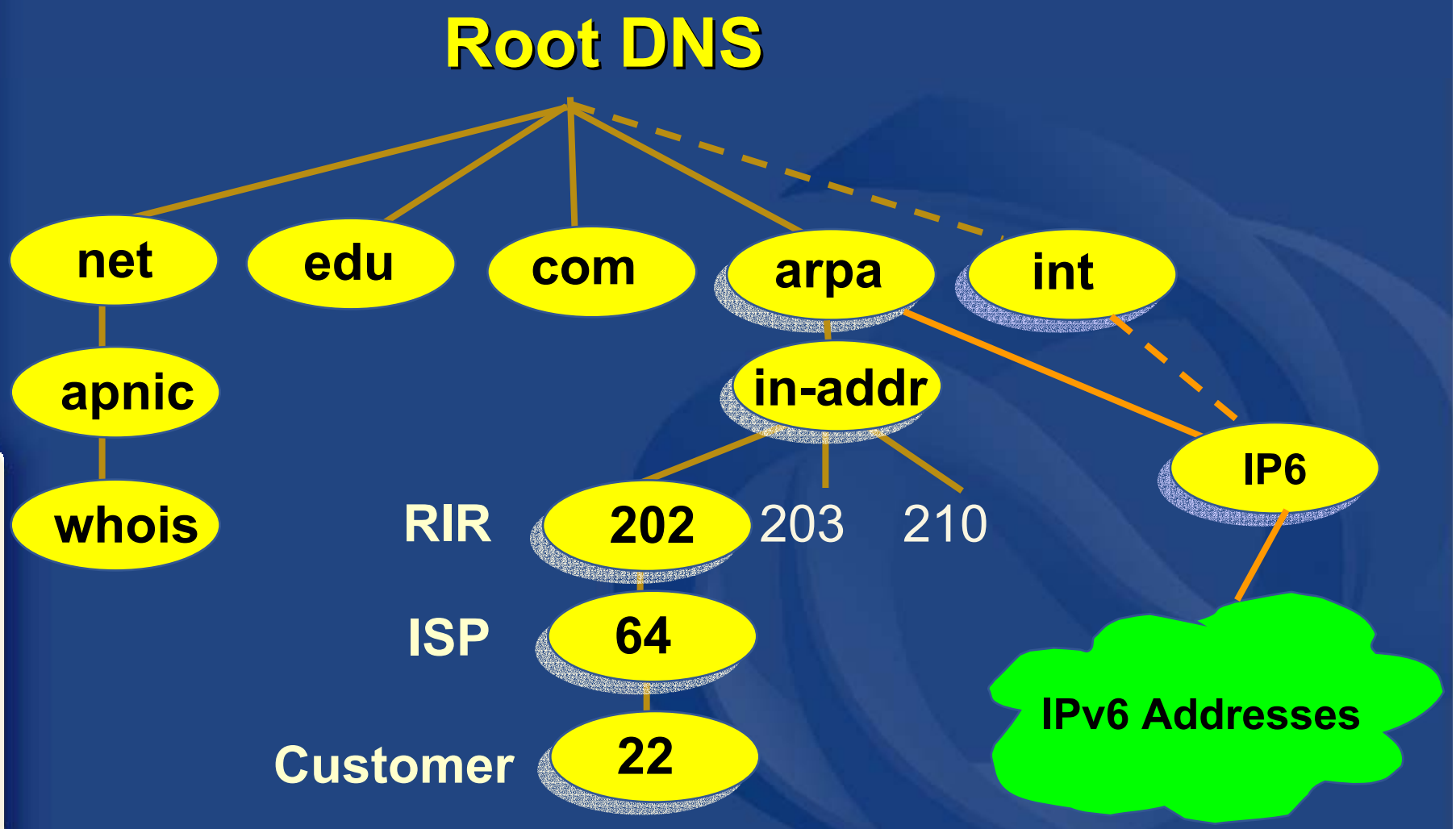
IPv6 representation in the DNS

- Forward lookup support: Multiple RR records for name to number
  - AAAA (Similar to A RR for IPv4)
  - A6 without chaining (prefix length set to 0)
- Reverse lookup support:
  - Reverse nibble format for zone ip6.int
  - Reverse nibble format for zone ip6.arpa

IPv6 forward and reverse mappings

- Existing A record will not accommodate IPv6's 128 bit addresses
- BIND expects an A record's record-specific data to be a 32-bit address (in dotted-octet format)
- An address record
  - AAAA (RFC 1886)
- A reverse-mapping domain
  - Ip6.int (now replaced by ip6.arpa)

The reverse DNS tree – with IPv6



IPv6 forward lookups

- Multiple addresses possible for any given name
  - Ex: in a multi-homed situation
- Can assign A records and AAAA records to a given name/domain
- Can also assign separate domains for IPv6 and IPv4

Sample forward lookup file

```
;; domain.edu
$TTL            86400
@     IN       SOA     ns1.domain.edu. root.domain.edu. (
               2002093000    ; serial - YYYYMMDDXX
               21600         ; refresh - 6 hours
               1200          ; retry - 20 minutes
               3600000       ; expire - long time
               86400)        ; minimum TTL - 24 hours
;; Nameservers
               IN     NS     ns1.domain.edu.
               IN     NS     ns2.domain.edu.

;; Hosts with just A records
host1          IN     A      1.0.0.1

;; Hosts with both A and AAAA records
host2          IN     A      1.0.0.2
               IN     AAAA   2001:468:100::2
```

IPv6 reverse lookups

- IETF decided to restandardize IPv6 PTR RRs
  - They will be found in the IP6.ARPA namespace rather than under the IP6.INT namespace
- The ip6.int domains has been deprecated, but some hosts still use them
  - Supported for backwards compatiblity
- Now using ip6.arpa for reverse

IPv6 reverse lookups - AAAA and ip6.arpa

- Address record four times longer than A
  - Quad A (AAAA)
- AAAA record is a parallel to the IPv4 A record
- It specifies the entire address in a single record

IPv6 reverse lookups - AAAA and ip6.arpa

- **Example**

```
Ipv6-host     IN     AAAA     4321:0:1:2:3:4:567:89ab
```

  – Each level of subdomain
    - Represents 4 bits

IPv6 reverse lookups - PTR records

- Similar to the in-addr.arpa

```
b.a.9.8.7.6.5.0.4.0.0.0.3.0.0.0.2.0.0.0.1.0.0.0.0.0.0.0.1.2.3.4.ip6.arpa.
        IN    PTR        test.ip6.example.com.
```

- Example: reverse name lookup for a host with address 3ffe:8050:201:1860:42::1

```
$ORIGIN 0.6.8.1.1.0.2.0.0.5.0.8.e.f.f.3.ip6.arpa.

1.0.0.0.0.0.0.0.0.0.0.0.0.0.2.4.0.0  14400  IN PTR host.example.com.
```

Sample reverse lookup file

```
;; 0.0.0.0.0.0.1.0.8.6.4.0.1.0.0.2.rev
;; These are reverses for 2001:468:100::/64)
;; File can be used for both ip6.arpa and ip6.int.
$TTL          86400
@     IN      SOA     ns1.domain.edu. root.domain.edu. (
                          2002093000                ; serial - YYYYMMDDXX
                          21600             ; refresh - 6 hours
                          1200              ; retry - 20 minutes
                          3600000           ; expire - long time
                          86400)            ; minimum TTL - 24 hours
;; Nameservers
                  IN      NS      ns1.domain.edu.
                  IN      NS      ns2.domain.edu.
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0    IN      PTR     host1.ip6.domain.edu
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0    IN      PTR     host2.domain.edu
;;
;; Can delegate to other nameservers in the usual way
;;
```

Sample configuration file

```
// named.conf

zone “domain.edu” {
        type master;
        file “master/domain.edu”;
}
zone “0.0.0.0.0.0.1.0.8.6.4.0.1.0.0.2.ip6.int" {
        type master;
        file "master/0.0.0.0.0.0.1.0.8.6.4.0.1.0.0.2.rev";
};
zone “0.0.0.0.0.0.1.0.8.6.4.0.1.0.0.2.ip6.arpa" {
        type master;
        file "master/0.0.0.0.0.0.1.0.8.6.4.0.1.0.0.2.rev";
};
```

Reverse delegation for existing /35 holders

- Reverse tree has 4bit ‘boundary’
  - /35 allocation needs two /36 delegations

```
FP |                    /35 allocations|
|3 |                                /32 |
|--|-------------------------------|--|----……
0010000000000001000000100011100000?
------------- 35 bits ---------------
   2   0   0   1:  0   2   3   8   0/35
```

Can be 1 or 0

- Delegation for two /36

```
0.8.3.2.0.1.0.0.2.ip6.arpa
1.8.3.2.0.1.0.0.2.ip6.arpa
```

Current Status – IPv6 in DNS

- A6 and Bit label specifications has been made experimental
  - RFC3363
- IETF standardized 2 different formats
  - AAAA and A6
  - Confusions on which format to deploy
  - More than one choice will lead to delays in the deployment of IPv6

AAAA Vs A6 – IETF WG consensus

- AAAA records are preferable at the moment for production deployment of IPv6
- A6 records have interesting properties that need to be better understood before deployment
- It is not known if the benefits of A6 outweigh the costs and risks

What we covered so far

- Why Reverse DNS ?
- The DNS tree
- Files and essential Resource Records
- How to create reverse zones

- Setting up nameservers – config files
- APNIC reverse delegation requirements
- Classless in-addr.arpa
- APNIC reverse delegation procedures

- IPv6 representation in the DNS
- IPv6 forward and reverse mappings
- AAAA and A6 records
- Current status

Questions ?

References

- DNS and BIND by Paul Albitz & Cricket Liu
  - O'Reilly
- Request Forms
  - http://www.apnic.net/db/revdel.html
  - http://www.apnic.net/db/domain.html
- Classless Delegations
  - http://ftp.apnic.net/ietf/rfc/rfc2000/rfc2317.txt
- Common DNS configuration errors
  - http://ftp.apnic.net/ietf/rfc/rfc1000/rfc1537.txt

- Domain name structure and delegation
  - http://ftp.apnic.net/ietf/rfc/rfc1000/rfc1591.txt
- Domain administrators operations guide
  - http://ftp.apnic.net/ietf/rfc/rfc1000/rfc1033.txt
- Taking care of your domain
  - ftp://ftp.ripe.net/ripe/docs/ripe-114.txt
- Tools for DNS debugging
  - http://ftp.apnic.net/ietf/rfc/rfc2000/rfc2317.txt

Domain object template

domain:	[mandatory]	[single]	[primary/look-up key]
descr:	[mandatory]	[multiple]	[ ]
country:	[optional]	[single]	[ ]
admin-c:	[mandatory]	[multiple]	[inverse key]
tech-c:	[mandatory]	[multiple]	[inverse key]
zone-c:	[mandatory]	[multiple]	[inverse key]
nserver:	[mandatory]	[multiple]	[inverse key]
sub-dom:	[optional]	[multiple]	[inverse key]
dom-net:	[optional]	[multiple]	[ ]
remarks:	[optional]	[multiple]	[ ]
notify:	[optional]	[multiple]	[inverse key]
mnt-by:	[mandatory]	[multiple]	[inverse key]
mnt-lower:	[optional]	[multiple]	[inverse key]
refer:	[optional]	[single]	[ ]
changed:	[mandatory]	[multiple]	[ ]
source:	[mandatory]	[single]	[ ]