DNS / DNSSEC Workshop

bdNOG5

7-11 April 2016, Dhaka, Bangladesh

Issue Date: 03 November 2015

Revision: 2.0-draft4

APNIC

Overview

- DNS Overview
- BIND DNS Configuration
- Recursive and Forward DNS
- Reverse DNS

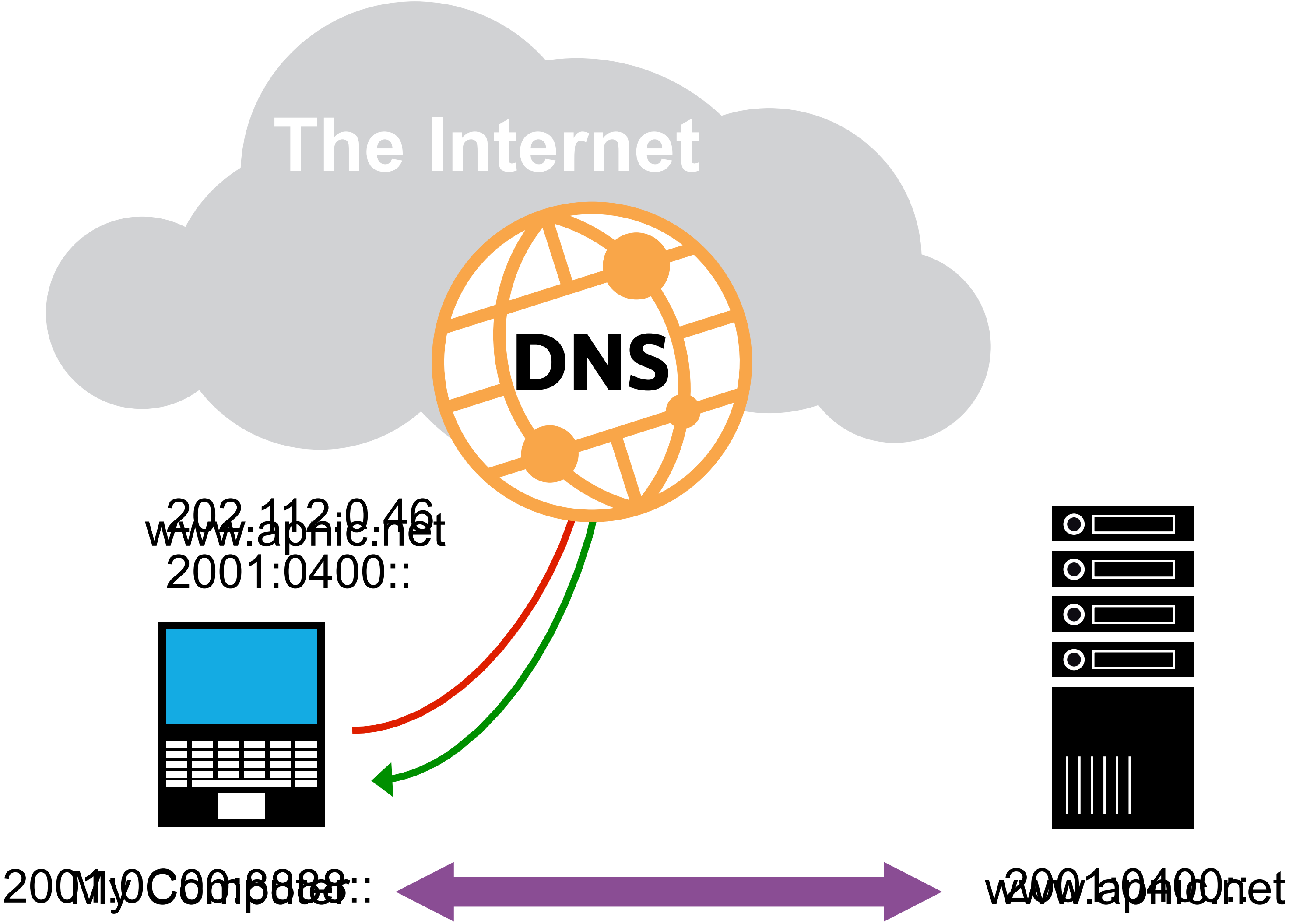
Overview

- **DNS Overview**
- BIND DNS Configuration
- Recursive and Forward DNS
- Reverse DNS

Domain Name System

- A lookup mechanism for translating objects into other objects
  - Mapping names to numbers and vice versa
- A globally distributed, loosely coherent, scalable, reliable, dynamic database
- Comprised of three components
  - A “name space”
  - Servers making that name space available
  - Resolvers (clients) query the servers about the name space
- A critical piece of the Internet infrastructure

IP Addresses vs Domain Names



Old Solution: hosts.txt

- A centrally-maintained file, distributed to all hosts on the Internet
- Issues with having just one file
  - Becomes huge after some time
  - Needs frequent copying to ALL hosts
  - Consistency
  - Always out-of-date
  - Name uniqueness
  - Single point of administration

```
// hosts.txt
SERVER1         128.4.13.9
WEBMAIL         4.98.133.7
FTPHOST         200.10.194.33
```

This feature still exists:
[Unix] /etc/hosts
[Windows] c:\windows\hosts

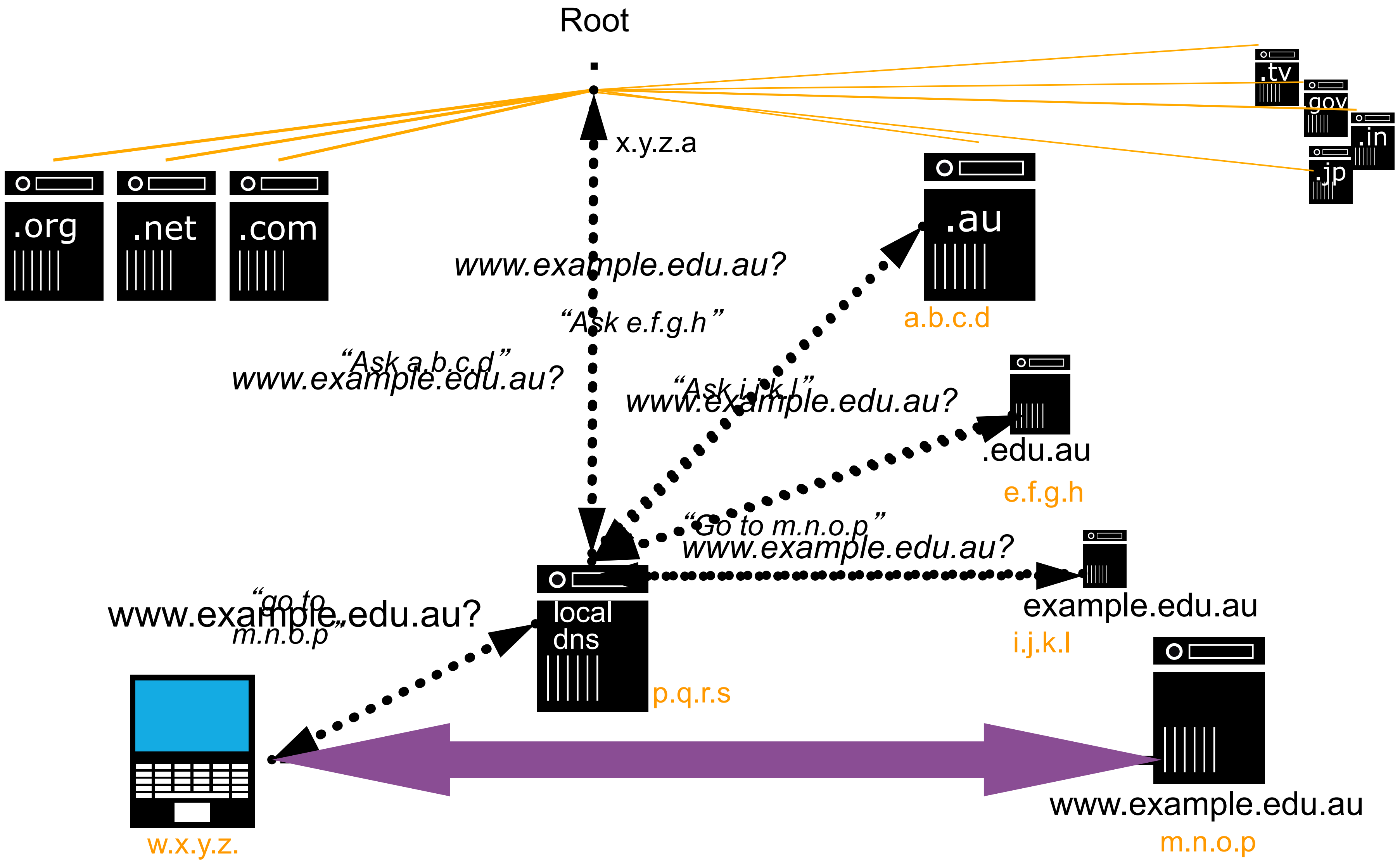
DNS Features

- Global distribution
  - Shares the load and administration
- Loose Coherency
  - Geographically distributed, but still coherent
- Scalability
  - can add DNS servers without affecting the entire DNS
- Reliability
- Dynamicity
  - Modify and update data dynamically

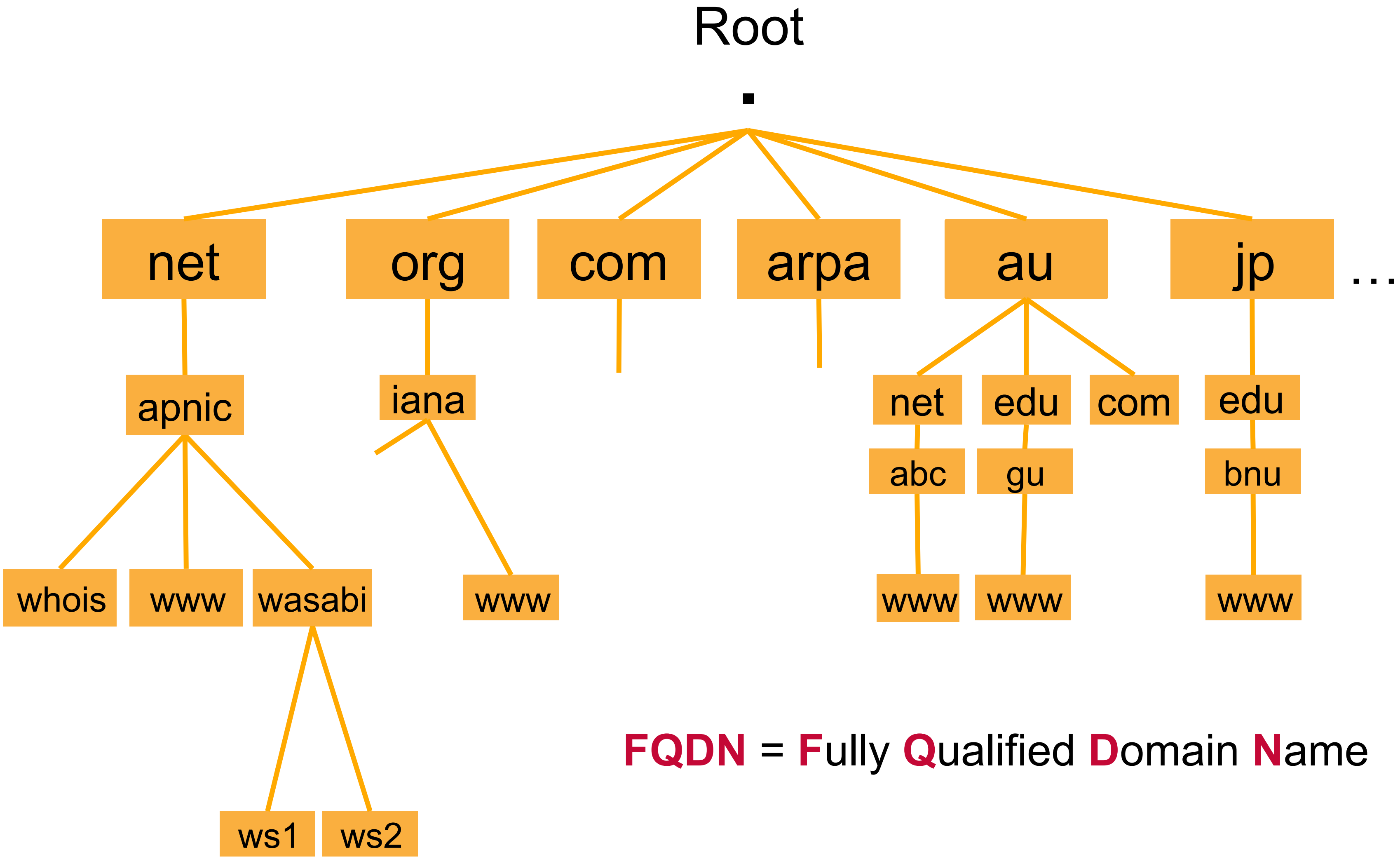
DNS Features

- DNS is a client-server application
- Requests and responses are normally sent in UDP packets, port 53
- Occasionally uses TCP, port 53
  - for very large requests, e.g. zone transfer from master to slave

Querying the DNS – It's all about IP!



The DNS Tree Hierarchy

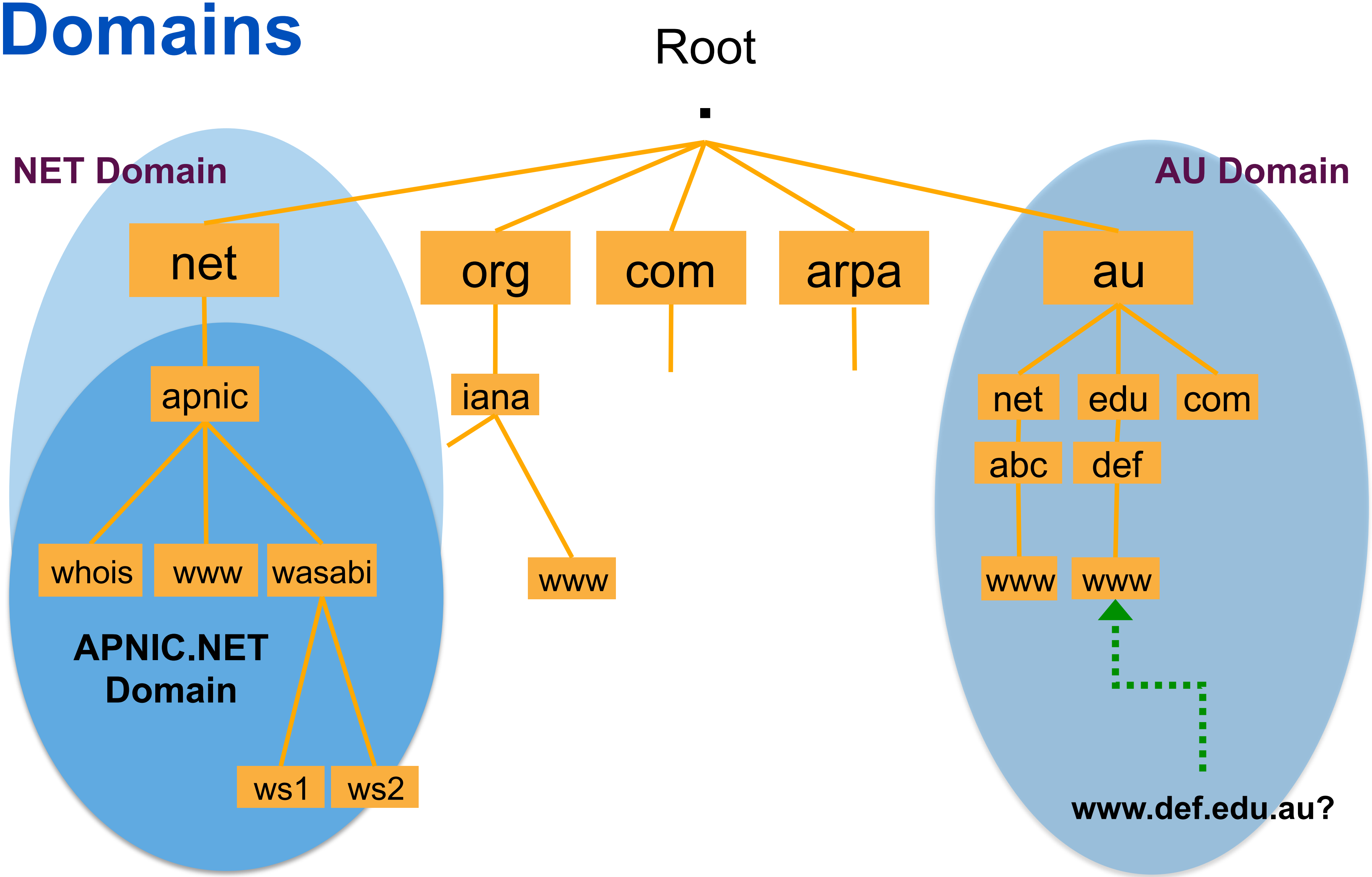


FQDN = **F**ully **Q**ualified **D**omain **N**ame

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

Domains


Root
.
NET Domain
AU Domain
net
org
com
arpa
au
apnic
iana
net
edu
com
abc
def
whois
www
wasabi
www
www
www
APNIC.NET Domain
ws1
ws2
www.def.edu.au?

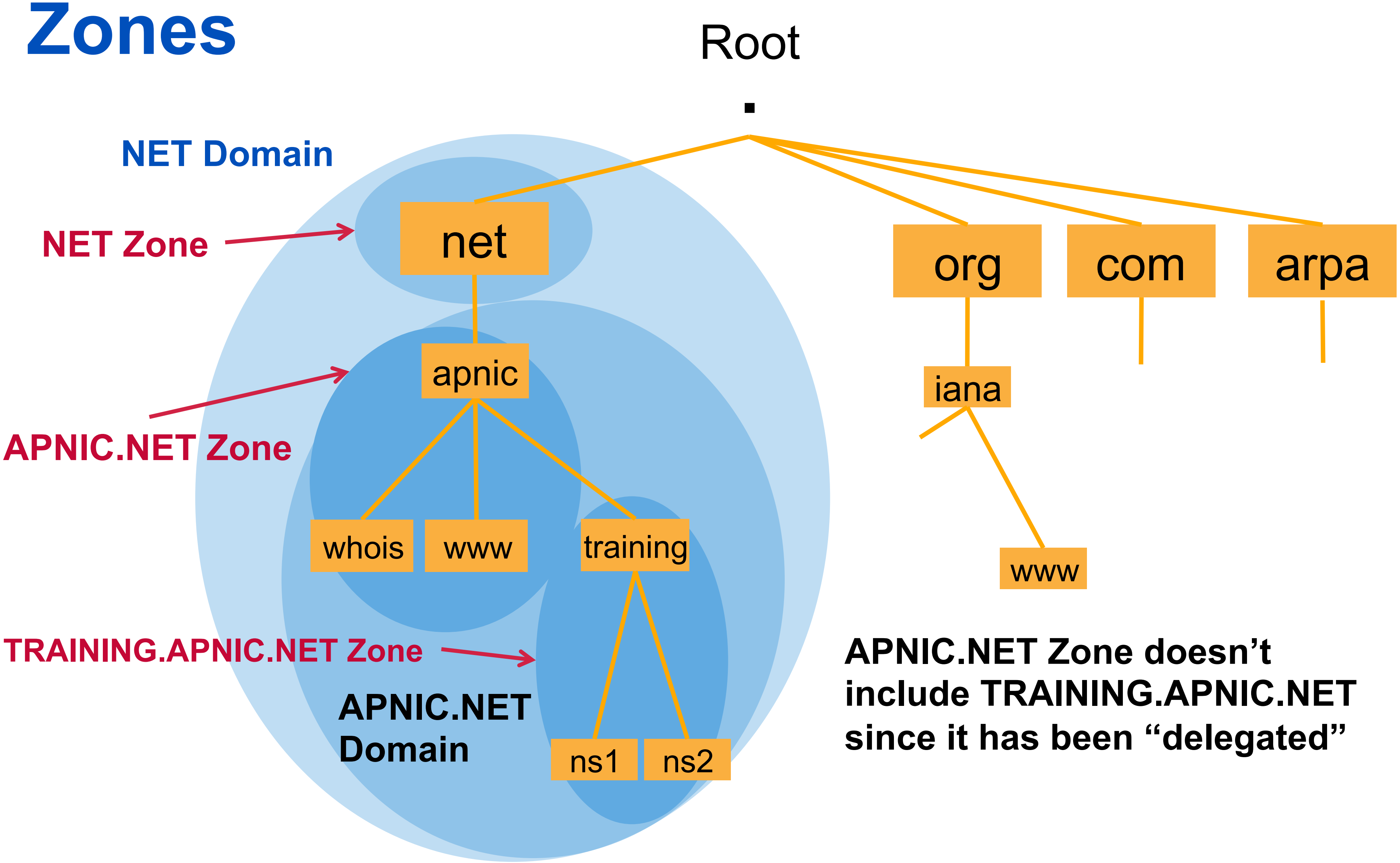
Delegation

- 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” to whom the subdomain is delegated

Zones and Delegations

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

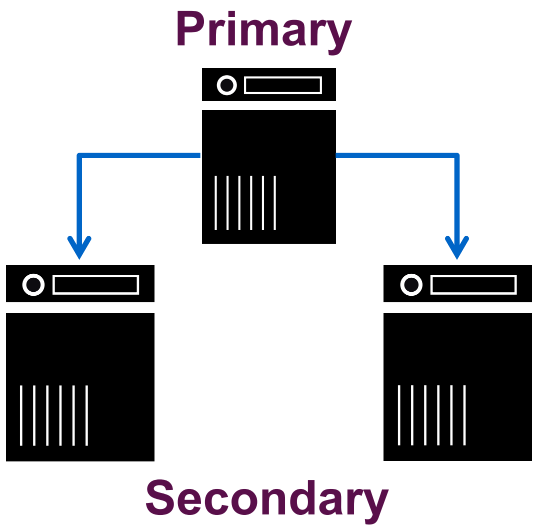
Zones



APNIC.NET Zone doesn’t include TRAINING.APNIC.NET since it has been “delegated”

Name Servers

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



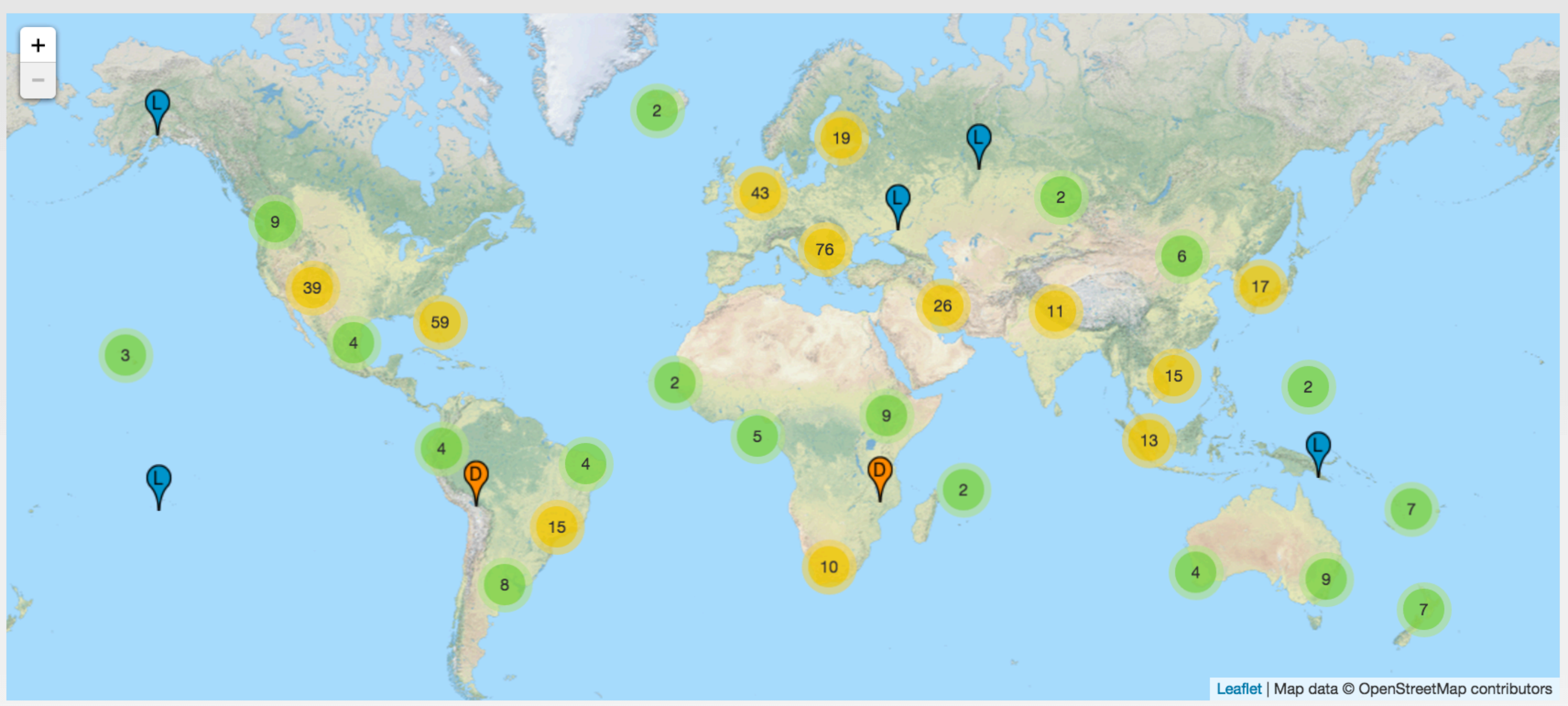
Root Servers

- The top of the DNS hierarchy

- There are 13 root name servers operated around the world
  - `[a-m].root-servers.net`

- There are more than 13 physical root name servers
  - Each rootserver has an instance deployed via anycast

Root Servers



http://root-servers.org/

Root Server Deployment at APNIC

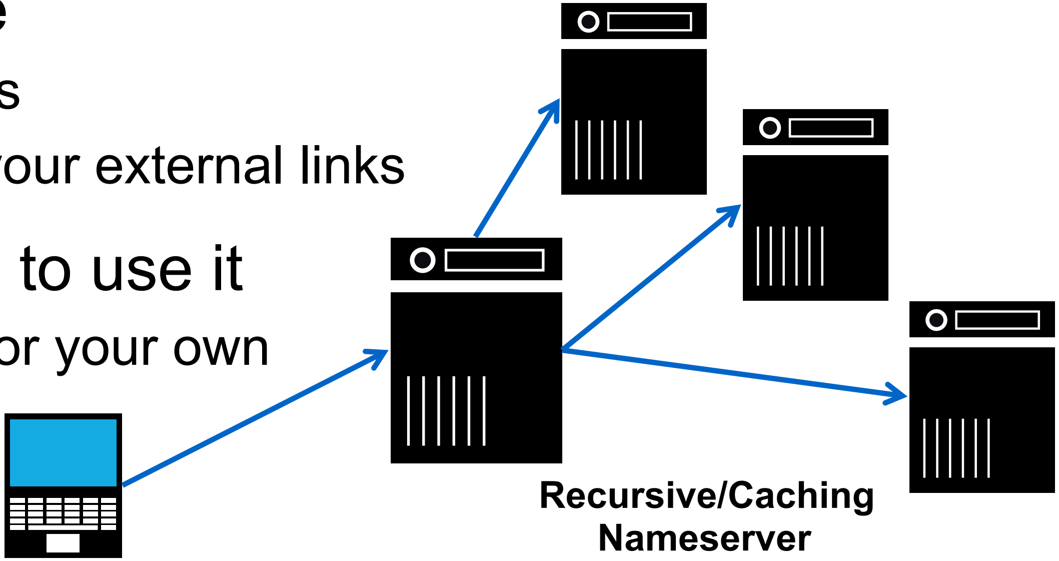
- Started in 2002, APNIC is committed to establish new root server sites in the AP region
- APNIC assists in the deployment providing technical support.
- Deployments of F, K and I-root servers in
  - Singapore, Hong Kong, China, Korea, Thailand, Malaysia, Indonesia, Philippines, Fiji, Pakistan, Bangladesh, Taiwan, Cambodia, Bhutan, and Mongolia

Resolver

- Or “stub” resolver
- A piece of software (usually in the operating system) which formats the DNS request into UDP packets
- A stub resolver is a minimal resolver that forwards all requests to a local recursive nameserver
  - The IP address of the local DNS server is configured in the resolver.
- Every host needs a resolver
  - In Linux, it uses /etc/resolv.conf
- It is always a good idea to configure more than one nameserver

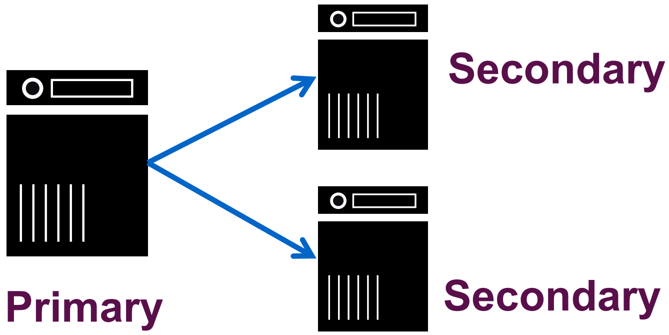
Recursive Nameserver

- The job of the recursive nameserver is to locate the authoritative nameserver and get back the answer
- This process is iterative – starts at the root
- Recursive servers are also usually caching servers
- Prefer a nearby cache
  - Minimizes latency issues
  - Also reduces traffic on your external links
- Must have permission to use it
  - Your ISP's nameserver or your own



Authoritative Nameserver

- A nameserver that is authorised to provide an answer for a particular domain
  - Can be more than one auth nameserver
- Two types based on management method:
  - Primary (Master) and Secondary (Slave)
- Only one primary nameserver
  - All changes to the zone are done in the primary
- Secondary nameserver/s will retrieve a copy of the zonefile from the primary server
  - Slaves poll the master periodically
- Primary server can “notify” the slaves



Resource Records

- Entries in the DNS zone file
- Components:

Resource Record	Function
Label	Name substitution for FQDN
TTL	Timing parameter, an expiration limit
Class	IN for Internet, CH for Chaos
Type	RR Type (A, AAAA, MX, PTR) for different purposes
RDATA	Anything after the Type identifier; Additional data

Common Resource Record Types

RR Type	Name	Functions
A	Address record	Maps domain name to IP address `www.example.com. IN A 192.168.1.1`
AAAA	IPv6 address record	Maps domain name to an IPv6 address `www.example.com. IN AAAA 2001:db8::1`
NS	Name server record	Used for delegating zone to a nameserver `example.com. IN NS ns1.example.com.`
PTR	Pointer record	Maps an IP address to a domain name `1.1.168.192.in-addr.arpa. IN PTR www.example.com.`
CNAME	Canonical name	Maps an alias to a hostname `web IN CNAME www.example.com.`

MX	Mail Exchanger	Defines where to deliver mail for user @ domain
		`example.com. IN MX 10 mail01.example.com.`
		`IN MX 20 mail02.example.com.`

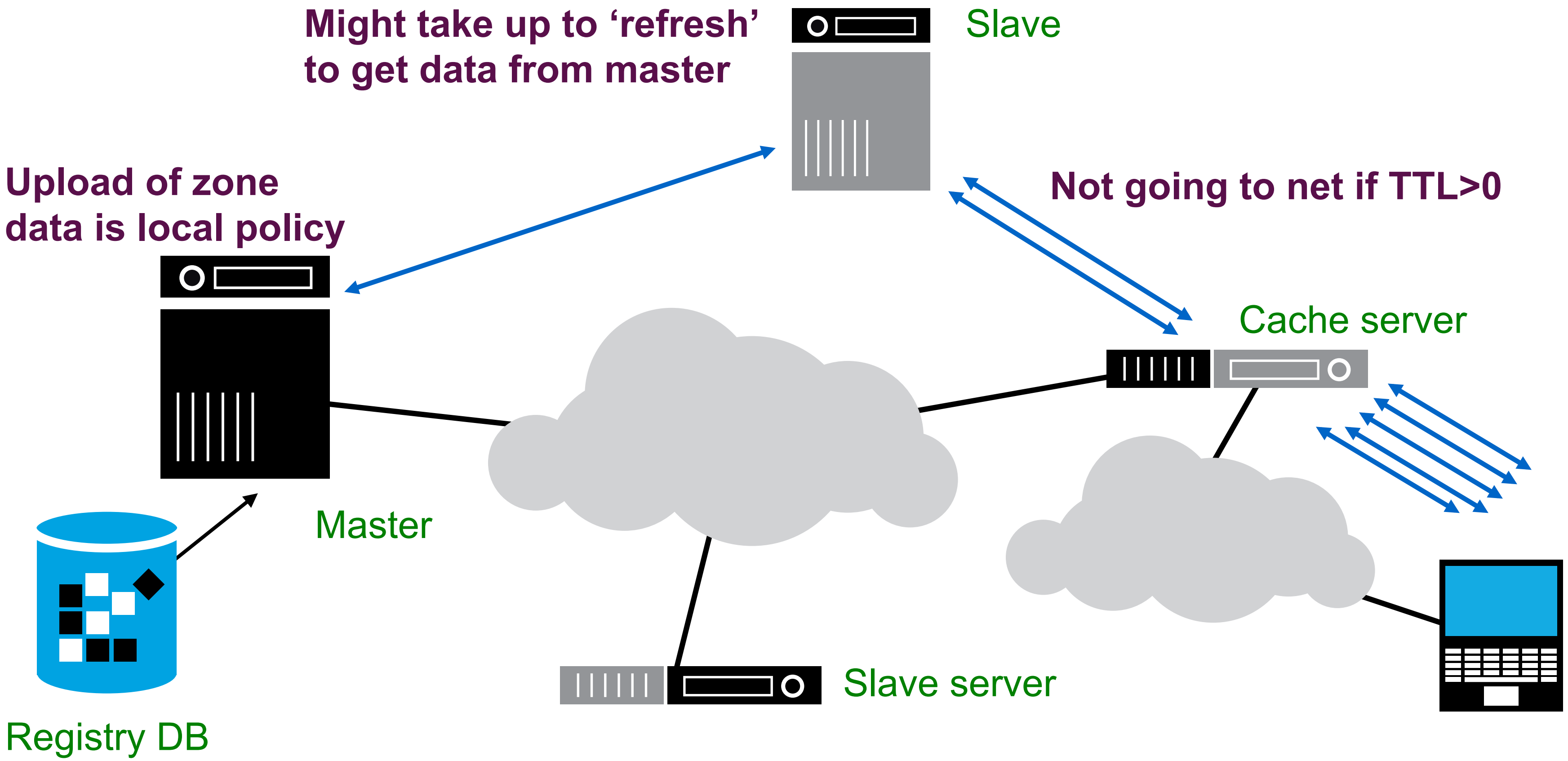
Example: RRs in a zone file

```
apnic.net.         7200 IN     SOA    ns.apnic.net. admin.apnic.net. (
                 2015050501          ; Serial
                 12h                 ; Refresh 12 hours
                 4h                  ; Retry 4 hours
                 4d                  ; Expire 4 days
                 2h                  ; Negative cache 2 hours )

apnic.net.                 7200   IN      NS     ns.apnic.net.
apnic.net.                 7200   IN      NS     ns.ripe.net.
www.apnic.net.             3600   IN      A      192.168.0.2
www.apnic.net              3600   IN      AAAA   2001:DB8::2
```

Label **TTL** **Class** **Type** **Rdata**

Places where DNS data lives



Changes do not propagate instantly

Might take up to ‘refresh’ to get data from master

Upload of zone data is local policy

Not going to net if TTL>0

Delegating a Zone

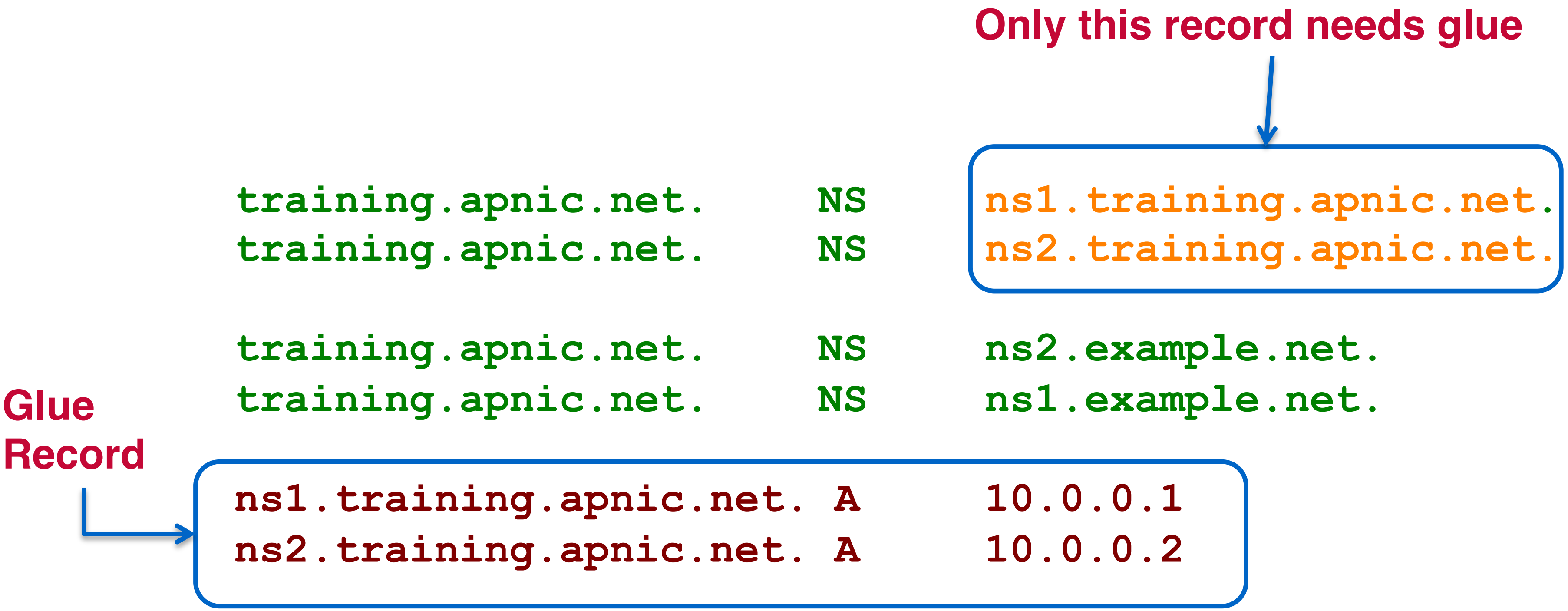
- Delegation is passing of authority for a subdomain to another party
- Delegation is done by adding NS records
  - Ex: if APNIC.NET wants to delegate TRAINING.APNIC.NET

```
training.apnic.net.        NS ns1.training.apnic.net.
training.apnic.net.        NS ns2.training.apnic.net.
```

- Now how can we go to ns1 and ns2?
  - We must add a **Glue Record**

Glue Record

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



Only this record needs glue

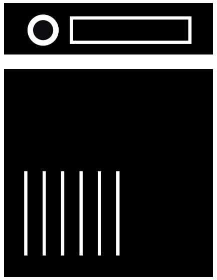
```
training.apnic.net.      NS      ns1.training.apnic.net.
training.apnic.net.      NS      ns2.training.apnic.net.

training.apnic.net.      NS      ns2.example.net.
training.apnic.net.      NS      ns1.example.net.
```

Glue Record

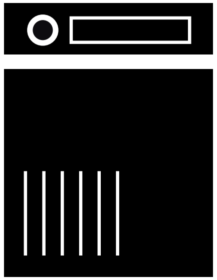
```
ns1.training.apnic.net.  A       10.0.0.1
ns2.training.apnic.net.  A       10.0.0.2
```

Delegating training.apnic.net. from apnic.net.



ns.apnic.net

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



ns.training.apnic.net

1. Setup minimum two servers
2. Create zone file with NS records
3. Add all training.apnic.net data

Remember ...

- Deploy multiple authoritative servers to distribute load and risk
  - Put your name servers apart from each other

- Use cache to reduce load to authoritative servers and response times

- SOA timers and TTL need to be tuned to the needs of the zone
  - For stable data, use higher numbers

Performance of DNS

- Server hardware requirements
- OS and the DNS server running
- How many DNS servers?
- How many zones are expected to load?
- How large are the zones?
- Zone transfers
- Where are the DNS servers located?
- Bandwidth

Performance of DNS

- Are these servers Multihomed?
- How many interfaces are to be enabled for listening?
- How many queries are expected to receive?
- Recursion
- Dynamic updates
- DNS notifications

Questions

Overview

- DNS Overview
- **BIND DNS Configuration**
- Recursive and Forward DNS
- Reverse DNS

DNS Software

- DNS BIND – authoritative + recursive server
- Unbound - caching DNS resolver
- NSD – authoritative only nameserver
- Microsoft DNS – provided with the Windows Server
- Knot DNS – authoritative only nameserver
- PowerDNS – data storage backends

BIND

- **B**erkeley **I**nternet **N**ame **D**omain
- The most widely-used open source DNS software on the Internet
  - Current version is Bind 9.10.3
  - Bind 9.9.8 is also current with Extended Support
  - Bind 9.8.x EOL as of Sep 2014
- Maintained by the Internet Systems Consortium (ISC)
- Bind 10 is in development
  - New architecture
  - Bind 10.1.1 released on June 06 2013
  - Has been concluded and renamed as Bundy (http://bundy-dns.de/)

Where to Get BIND

- Download source from the ISC website
  - http://www.isc.org
  - ftp://ftp.isc.org/isc/bind9
- Install from your distribution’s package manager
- Some packages may also be required
  - OpenSSL is a necessary for DNSSEC

Unpacking BIND9

- When installing BIND from source, decompress the gzip file

```
tar xvfz bind-9.<version>.tar.gz
cd bind-9.<version>
```

- What's in there?
  - A lot of stuff (dig, libraries, etc)
  - Configure scripts
  - Administrator's Reference Manual (ARM)

Building BIND9 from Source

- must be in the BIND 9 directory
- Determine the appropriate includes and compiler settings

```
./configure --with-openssl
```

- Build and compile

```
make
```

- Install the BIND package

```
make install
```

- Verify the installation

```
which named

named -v
```

Building BIND9 with Package Manager

- Redhat/CentOS

```
yum –y install bind9
```

- Ubuntu / Debian

```
apt-get install bind9
```

Location of Executables

`/usr/local/sbin`

- named
- dnssec-keygen, dnssec-makekeyset, dnssec-signkey, dnssec-signzone
- lwresd, named-checkconf, named-checkzone
- rndc, rndc-confgen

`/usr/local/bin`

- dig
- host, isc-config.sh, nslookup
- Nsupdate

Named Configuration

- The BIND configuration file is called “named.conf”
  - Default location is in /etc/named.conf
  - Run named with –c option to specify a different location
- Defines the zones and points to the corresponding zonefile
- Defines global options
- Logging can be turned on for troubleshooting

Named Configuration

- BIND Configuration file

- Options statement contains all global configuration options to be used as defaults by named.

```
options {
   directory “/var/named/recursive”; };
```

- Zone statement defines the zones and any zone-specific option

```
zone “myzone.net” {
   type master;
   file “db.myzone.net”; };
```

Root Hints

- Pointer to the root servers
- Root hints file come in many names
  - db.cache, named.root, named.cache, named.ca
- Get it from ftp://ftp.rs.internic.net/domain/
- Defined as follows in the config file

```
      zone “.” {
         type hint;
         file “root.hints”; };
```

What it looks like

ftp://ftp.rs.internic.net/domain/

```
.                        3600000  IN  NS     A.ROOT-
SERVERS.NET.
A.ROOT-SERVERS.NET.      3600000      A      198.41.0.4
A.ROOT-SERVERS.NET.      3600000      AAAA   2001:503:BA3E::
2:30
; operated by WIDE
.                        3600000      NS     M.ROOT-
SERVERS.NET.
M.ROOT-SERVERS.NET.      3600000      A      202.12.27.33
M.ROOT-SERVERS.NET.      3600000      AAAA   2001:dc3::35
```

Configuring Recursive Server

- The recursive server needs to know how to reach the top of the DNS hierarchy
- It should also stop some queries such as those for localhost (127.0.0.1)
- The following files are required to run a recursive/caching server:
  - named.conf
  - root.hints
  - localhost zone (db.localhost)
  - 0.0.127.in-addr.arpa zone (db.127.0.0.1)
  - ::1 IPv6 reverse zone (db.ip6)

Zones in a Recursive Server

- Loopback name in operating systems
  - Queries for this shouldn't use recursion
  - Configure a file to define the localhost zone
  - Localhost will map to 127.0.0.1 and ::1

```
zone “localhost” {
    type master;
    file db.localhost; };
```

- Reverse zone for the loopback
  - maps 127.0.0.1 (and ::1) to localhost

```
zone “0.0.127.in-addr.arpa” {
    type master;
    file db.127.0.0.1;
    };
```

Zones in a Recursive Server

- Reverse zone for IPv6 link-local address

```
zone “8.B.D.0.1.0.0.2.ip6.arpa” {
   type master;
   file db.2001.db8;
   };
```

- Built-in empty zones will be created for RFC 1918, RFC 4193, RFC 5737 and RFC 6598

https://tools.ietf.org/html/rfc6303

Example named.conf

```
options {
     directory "/var/named/recursive";
     recursion yes;
};
zone "." {
        type hint;
        file "named.root";
};
zone "localhost." {
        type master;
        file "localhost";
};
```

```
zone "0.0.127.in-addr.arpa." {
        type master;
        file ″db.127";
};

zone "8.B.D.0.1.0.0.2.ip6.arpa." {
        type master;
        file ″db.2001.db8";
};
```

Zone Files

- Contain the resource records defined in a particular zone
- begins with a Start of Authority Record (SOA)

```
@     SOA   localhost.  root.localhost.   (
                        20150505 ;serial no.
                        30m      ;refresh
                        15m      ;retry
                        1d       ;expire
                        30m      ;negative cache ttl )
```

- Common Zone File directives
  - $ORIGIN
  - $INCLUDE
  - $TTL
  - @ represents the current origin

Start of Authority (SOA) record

```
Domain_name. CLASS  SOA  hostname.domain.name. mailbox.domain.name (
                                Serial Number
                                Refresh
                                Retry
                                Expire
                                Minimum TTL )
```

- **Serial Number** – must be updated if any changes are made in the zone file
- **Refresh** – how often a secondary will poll the primary server to see if the serial number for the zone has increased
- **Retry** - If a secondary was unable to contact the primary at the last refresh, wait the retry value before trying again
- **Expire** - How long a secondary will still treat its copy of the zone data as valid if it can't contact the primary.
- **Minimum TTL** - The default TTL (time-to-live) for resource records

TTL Time Values

- The right value depends on your domain
- Recommended time values for TLD (based on RFC 1912)

Refresh	86400 (24h)
Retry	7200 (2h)
Expire	2592000 (30d)
Min TTL	345600 (4d)

- For other servers – optimize the values based on
  - Frequency of changes
  - Required speed of propagation
  - Reachability of the primary server
  - (and many others)

localhost zonefile

```
$TTL 86400
@        IN      SOA localhost. root.localhost.  (
                      20150505   ; serial
                      1800       ; refresh
                      900        ; retry
                      69120      ; expire
                      1080       ; negative ttl
                            )
                NS    localhost.
                A     127.0.0.1
                AAAA  ::1
```

0.0.127.in-addr.arpa zonefile

```
$TTL 86400
@        IN     SOA localhost. root.localhost. (
                     20150505  ; serial
                     1800      ;refresh
                     900       ;retry
                     69120     ;expire
                     1080      ;negative ttl
                     )

                NS  localhost.
         1     PTR  localhost.
```

ip6.arpa zonefile

```
$TTL 86400
@        IN     SOA localhost. root.localhost. (
                     20150505  ; serial
                     1800      ;refresh
                     900       ;retry
                     69120     ;expire
                     1080      ;negative ttl
                     )

                NS  localhost.
         1     PTR  localhost.
```

Assembling the files

- Create a directory in /var/named/ and copy the files

```
# mkdir recursive
# ls
    0.0.127.in-addr.arpa   db.localhost  root.hints
```

- The directory name and file names will be defined in named.conf
- Now create a named.conf file in the same directory

Running the server

- From the directory

```
named -g -c named.conf

where:
      -c  path to the configuration file
      -g  run in the foreground
```

Testing the server

```
% dig @127.0.0.1 www.google.com

; <<>> DiG 9.8.3-P1 <<>> www.google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 213
;; flags: qr rd ra; QUERY: 1, ANSWER: 5, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;www.google.com.                    IN       A

;; ANSWER SECTION:
www.google.com.          156        IN       A        74.125.237.115
www.google.com.          156        IN       A        74.125.237.113
www.google.com.          156        IN       A        74.125.237.116
www.google.com.          156        IN       A        74.125.237.114
www.google.com.          156        IN       A        74.125.237.112

;; Query time: 27 msec
;; SERVER: 127.0.0.1#53(203.119.98.119)
;; WHEN: Thu Jul 11 13:46:29 2013
;; MSG SIZE  rcvd: 112
```

Questions

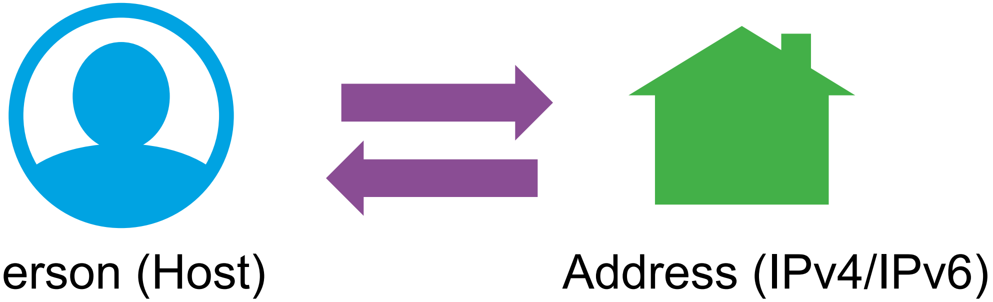
Overview

- DNS Overview
- BIND DNS Configuration
- Recursive and Forward DNS

What is ‘Reverse DNS’?

- ‘Forward DNS’ maps names to numbers
  - svc00.apnic.net ➔192.168.1.100
  - svc00.apnic.net ➔2001:DB8::1

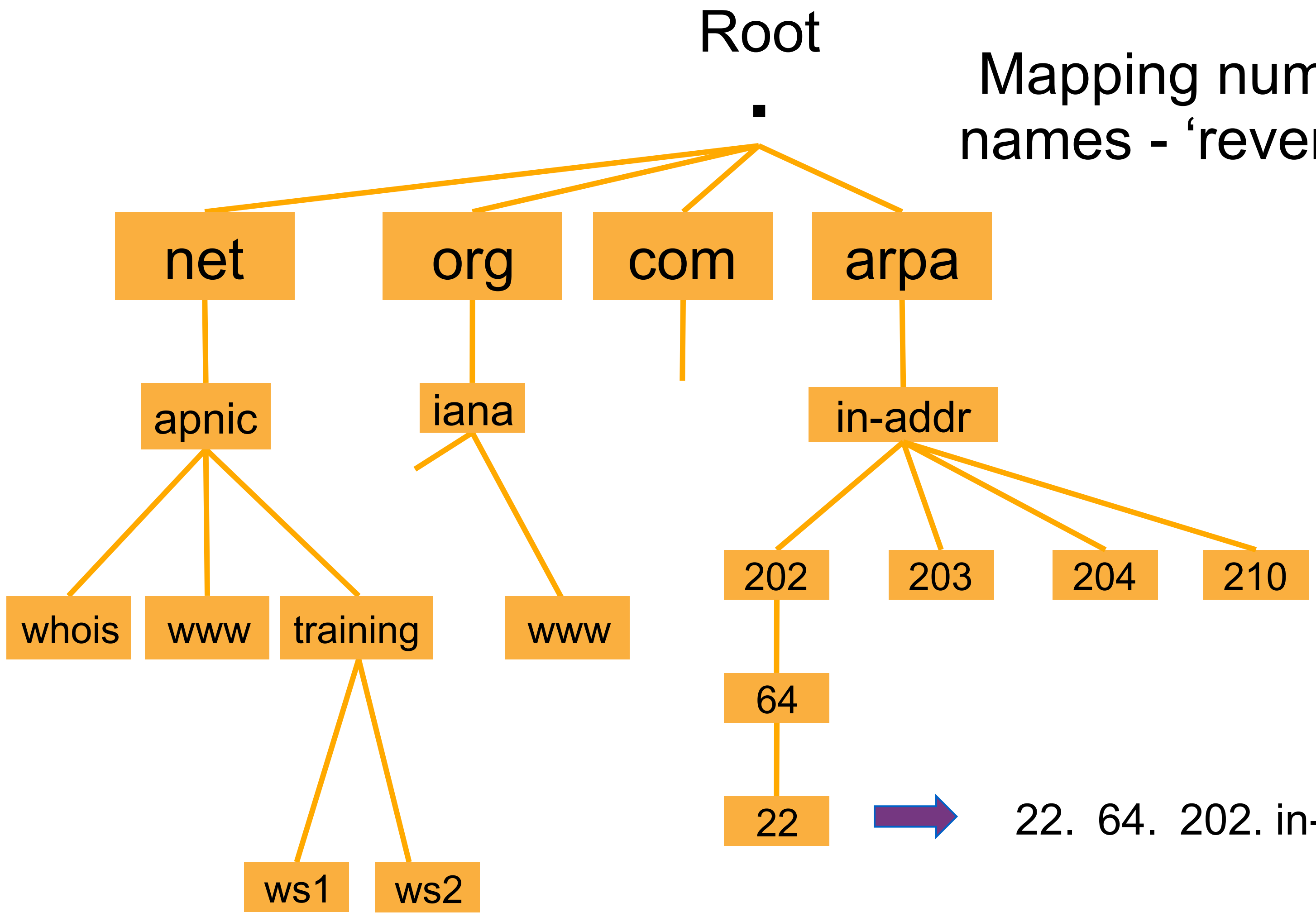
- ‘Reverse DNS’ maps numbers to names
  - 192.168.1.100 ➔ svc00.apnic.net
  - 2001:DB8::1 ➔ svc00.apnic.net



Reverse DNS - why bother?

- Service denial
  - only allow access when fully reverse delegated
  - Example: anonymous ftp
- Diagnostics
  - Used in tools such as traceroute
- Spam identifications
  - Failed reverse lookup results in a spam penalty score
- Registration responsibilities
  - APNIC members must make sure that all their address space are properly reverse delegated

Principles – DNS Tree



Mapping numbers to names - ‘reverse DNS’

22. 64. 202. in-addr.arpa.

Creating Reverse Zones

- Same as creating a forward zone file
  - SOA and initial NS records are the same as forward zone
- Create additional PTR records
- In addition to the forward zone files, you need the reverse zone files
  - Ex: for a reverse zone on a 203.176.189.0/24 block, create a zone file and name it as “db.203.176.189” (make it descriptive)

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.
```

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.
```

Reverse Delegation

- /24 Delegations
  - Address blocks should be assigned or allocated
  - At least two name servers

- /16 Delegations
  - Same as /24 delegations
  - APNIC delegates entire zone to member

- < /24 Delegations
  - Read “Classless IN-ADDR.ARPA delegation” (RFC 2317)

RFC
2317

APNIC & LIR Responsibilities

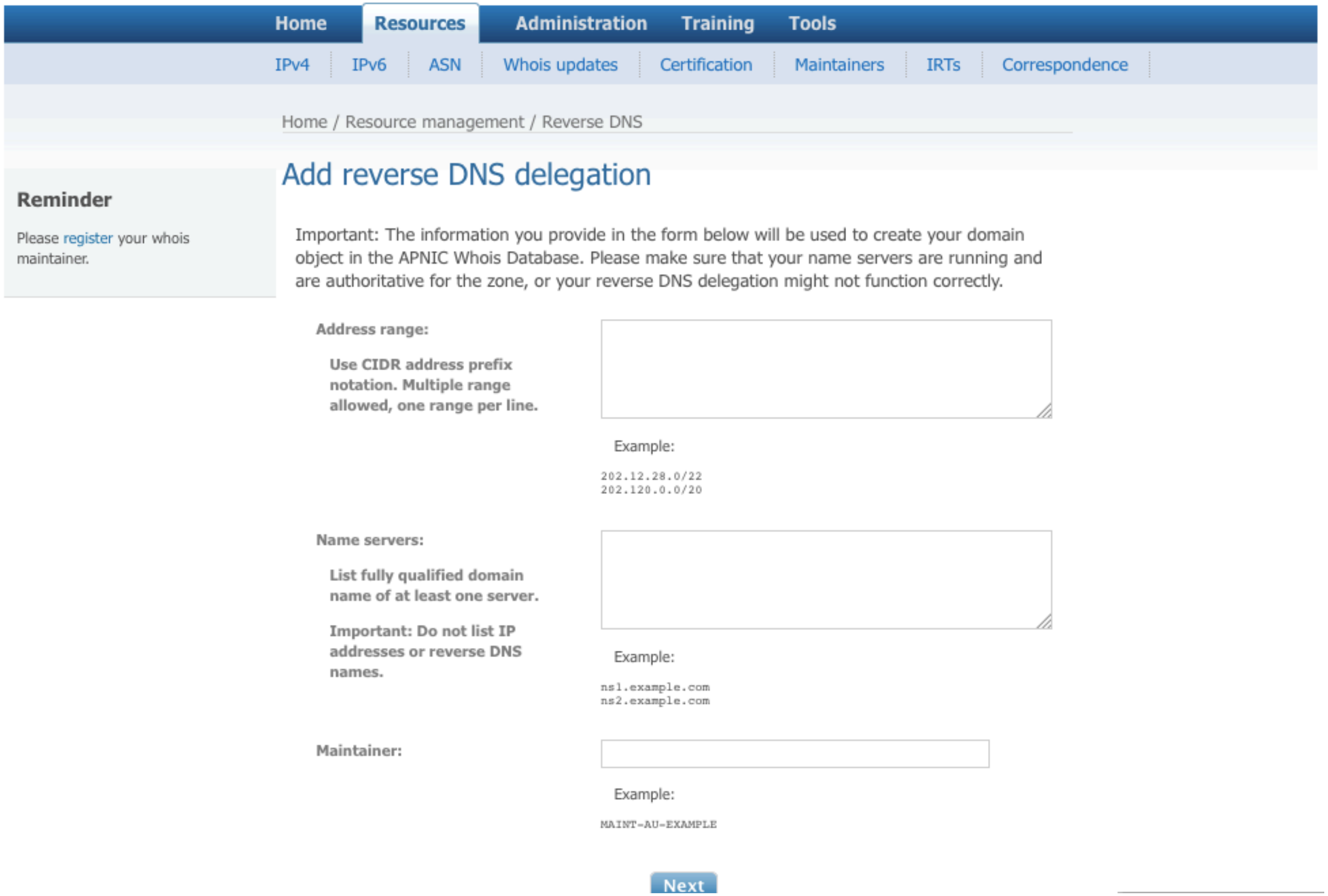
- APNIC
  - Manage reverse delegations of address block distributed by APNIC
  - Process requests for reverse delegation of network allocations

- LIR and members
  - Be familiar with APNIC procedures
  - Ensure that addresses are reverse-mapped
  - Maintain nameservers for allocations
  - Minimize pollution of DNS

Reverse Delegation Procedures

- Create a whois object for the reverse zone
  - This can be done in MyAPNIC
- Verify nameserver and domain set up before submitting to the database
- Provide the FQDN of two nameservers
- Provide the maintainer password
  - Used to protect objects

Reverse Delegation Procedures



Reminder

Please register your whois maintainer.

Add reverse DNS delegation

Important: The information you provide in the form below will be used to create your domain object in the APNIC Whois Database. Please make sure that your name servers are running and are authoritative for the zone, or your reverse DNS delegation might not function correctly.

Address range:

Use CIDR address prefix notation. Multiple range allowed, one range per line.

Example:

```
202.12.28.0/22
202.120.0.0/20
```

Name servers:

List fully qualified domain name of at least one server.

Important: Do not list IP addresses or reverse DNS names.

Example:

```
ns1.example.com
ns2.example.com
```

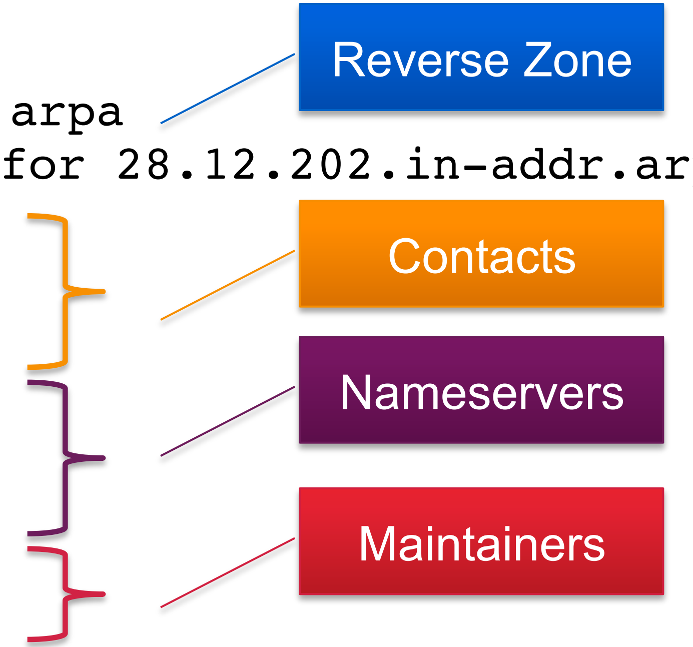
Maintainer:

Example:

```
MAINT-AU-EXAMPLE
```

Whois domain object

```
domain:     28.12.202.in-addr.arpa
Descr:      in-addr.arpa zone for 28.12.202.in-addr.arpa
admin-c:    NO4-AP
tech-c:     AIC1-AP
zone-c:     NO4-AP
nserver:    cumin.apnic.net
nserver:    tinnie.apnic.net
nserver:    tinnie.arin.net
mnt-by:     MAINT-APNIC-AP
mnt-lower:  MAINT-AP-DNS
changed:    inaddr@apnic.net 20021023
changed:    inaddr@apnic.net 20040109
changed:    hm-changed@apnic.net 20091007
changed:    hm-changed@apnic.net 20111208
source:     APNIC
```



Reverse Zone

Contacts

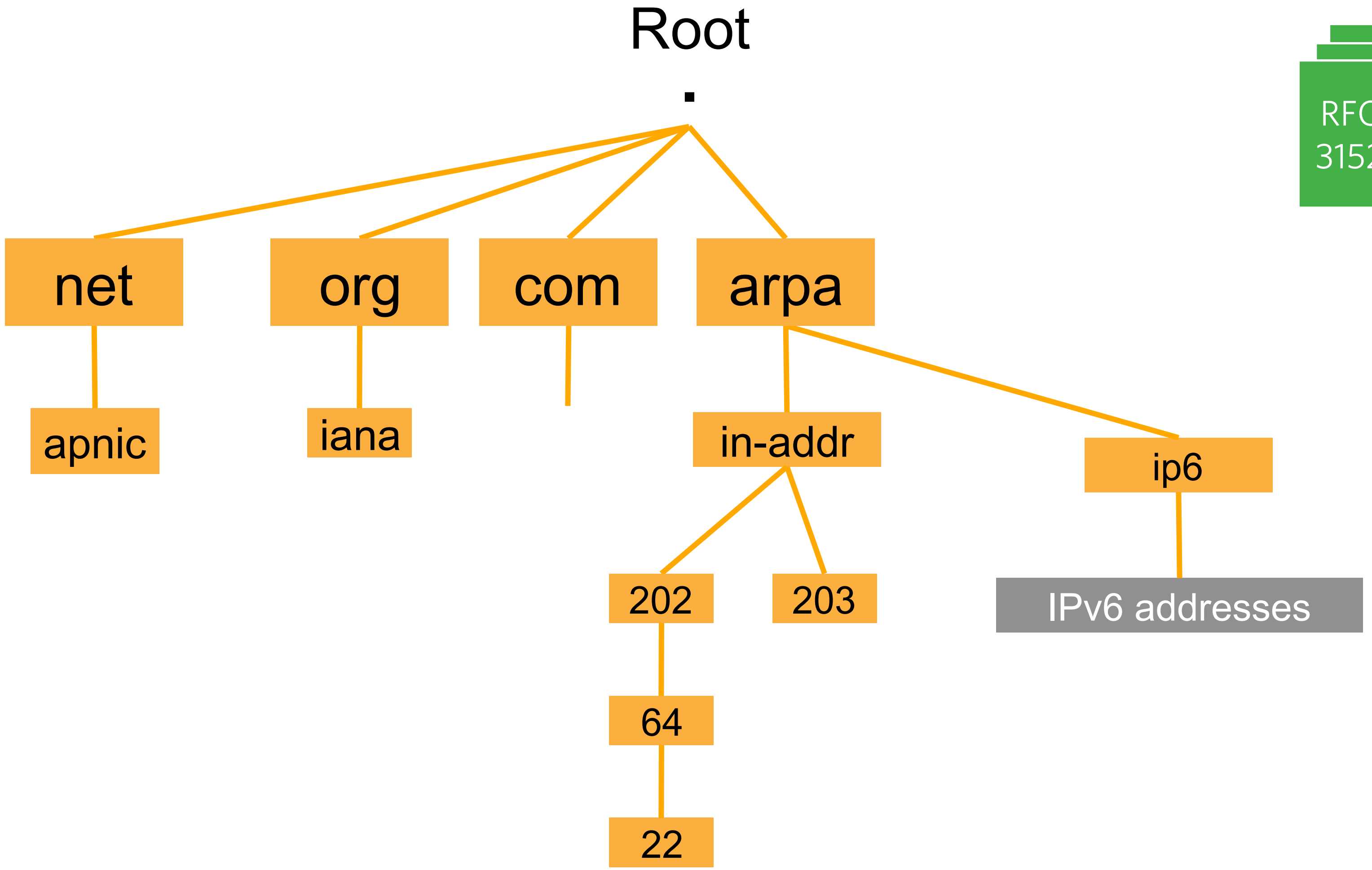
Nameservers

Maintainers

Overview

- DNS Overview
- BIND DNS Configuration
- Recursive and Forward DNS
- **Reverse DNS (for IPv6)**

Reverse DNS Tree – with IPv6



RFC 3152

IPv6 Representation in the DNS

- Forward lookup support: Multiple RR records for name to number
  - AAAA (Similar to A RR for IPv4)

- Reverse lookup support:
  - Reverse nibble format for zone ip6.arpa

IPv6 Reverse Lookups – PTR records

- Similar to the IPv4 reverse record

```
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: The 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.
1.0.0.0.0.0.0.0.0.0.0.0.2.4.0.0  14400  IN PTR
host.example.com.
```

IPv6 Forward and Reverse Mappings

- Existing A record will not accommodate the 128 bit addresses for IPv6
- BIND expects an A record data to be 32-bit address (in dotted-octet format)
- An address record
  - AAAA (RFC 1886)
- A reverse-mapping domain
  - ip6.arpa

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

Example: Forward Zone

```
;; domain.edu
$TTL          86400
@    IN      SOA     ns1.domain.edu. root.domain.edu. (
                           2015050501                ; 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
```

Example: Reverse Zone

```
;; 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. (
                           2015050501        ; 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     IN       PTR      host1.ip6.domain.edu
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0     IN       PTR      host2.domain.edu
```

Questions