Reverse DNS

Overview

- Principles
- Creating reverse zones
- Setting up nameservers
- Reverse delegation procedures

What is ‘Reverse DNS’?

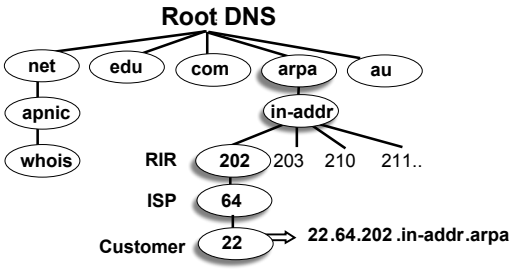
- ‘Forward DNS’ maps names to numbers
  - svc00.apnic.net -> 202.12.28.131
- ‘Reverse DNS’ maps numbers to names
  - 202.12.28.131 -> svc00.apnic.net

Reverse DNS - why bother?

- Service denial
  - That only allow access when fully reverse delegated eg. anonymous ftp
- Diagnostics
  - Assisting in trace routes etc
- SPAM identifications
- Registration responsibilities

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

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

253.253.192.in-addr.arpa.

Pointer (PTR) records

- Create pointer (PTR) records for each IP address

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

```
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.
```

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>"; };
```

-
  - 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”

APNIC & ISPs responsibilities

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

- Example: an organisation given a /16
  - 192.168.0.0/16 (one zone file and further delegations to downstreams)
  - 168.192.in-addr.arpa 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 /20
  - 192.168.0.0/20 (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.
:
:
15.168.192.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 for 192.253.254/24

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

```
$ORIGIN 254.253.192.in-addr.arpa.

0-63      NS      ns1.organisation1.com.
0-63      NS      ns2.organisation1.com.

1         CNAME   1.0-63
2         CNAME   2.0-63

64-127    NS      ns1.organisation2.com.
64-127    NS      ns2.organisation2.com.

65        CNAME   65.64-127
66        CNAME   66.64-127
```

Classless in-addr for 192.253.254/24

- Using $GENERATE (db.192.253.254 file)

```
$ORIGIN 254.253.192.in-addr.arpa.

0-63                    NS      ns1.organisation1.com.
0-63                    NS      ns2.organisation1.com.

$GENERATE 1-63$         CNAME   $.0-63

64-127                  NS      ns1.organisation2.com.
64-127                  NS      ns2.organisation2.com.

$GENERATE 65-127$       CNAME   $.64-127
```

Classless in-addr for 192.253.254.0/26

- 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

```
$ORIGIN 0-63.254.253.192.in-addr.arpa.
$TTL 1d
@   SOA  ns1.organisation1.com. Root.ns1.organisation1.com. (
                            1       ; Serial
                            3h      ; Refresh
                            1h      ; Retry
                            1w      ; Expire
                            1h )    ; Negative caching TTL
             NS      ns1.organisation1.com.
             NS      ns2.organisation1.com.

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

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 myAPNIC, 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

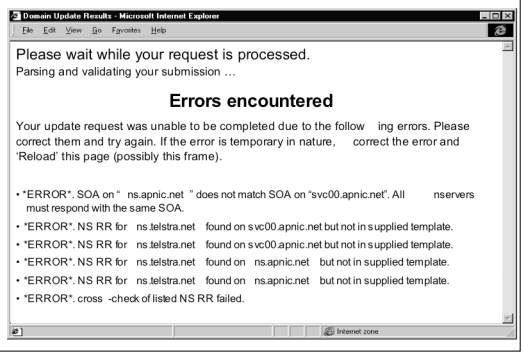
Reverse delegation procedures

- Use MyAPNIC to create 'domain' objects
  - Highly recommended
- Or use the web form
  - 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

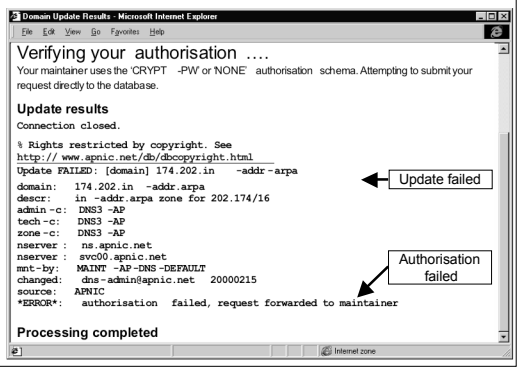
Evaluation procedures

- 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

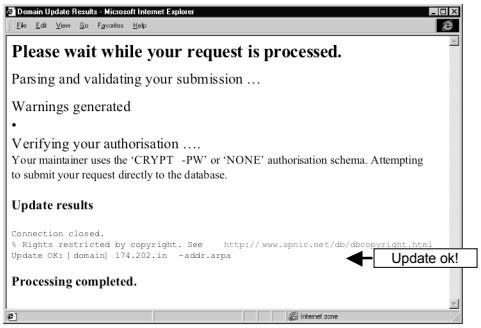
Online errors (also via email)



Request submission error



Successful update



Creation of domain objects

- Two options
  - Ask APNIC hostmasters to create the dummy domain objects at the time when the IP allocation is made
  - Do it yourself
- Domain objects protected by maintainers
  - hierarchical protection using "mnt-lower"
  - APNIC protects all the /8's by MAINT-AP-DNS

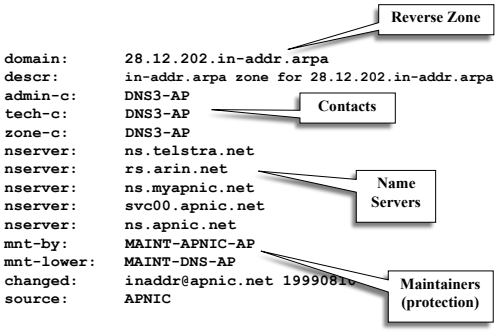
Creation of domain objects

- If you opt to create the domain objects yourself
  - Either you can use MyAPNIC
  - Or use web/email templates
- Using web/email templates will result in initial errors
  - As the /8 is hierarchically maintained by MAINT-AP-DNS
  - Contact <helpdesk@apnic.net>

Creation of domain objects

- APNIC highly recommend you to use MyAPNIC when creating domain objects
  - MyAPNIC parser will check the maintainer of 'inetnum' object
  - If the password matches no errors will be returned
- Can use MyAPNIC to create multiple domain objects at once
  - ex: If you are allocated a /19, you can provide the full IP range and 32 domain objects can be created in one go

Whois domain object



Removing lame delegations

- Objective
  - To repair or remove persistently lame DNS delegations
- DNS delegations are lame if:
  - Some or all of the registered DNS nameservers are unreachable or badly configured
- APNIC commenced formal implementation of the lame DNS reverse delegation procedures

IPv6 Reverse delegations

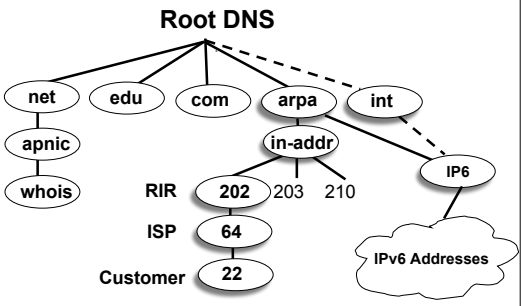
IPv6 representation in the DNS

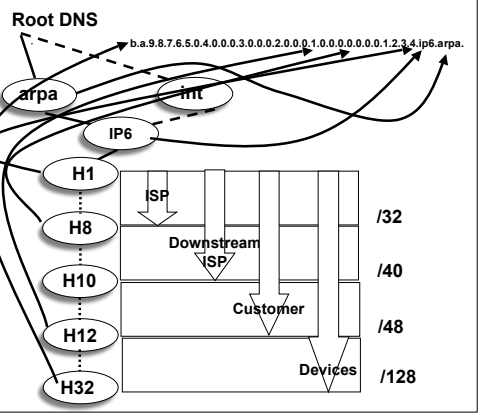
- Forward lookup support: Multiple RR records for name to number
  - AAAA (Similar to A RR for IPv4)
- Reverse lookup support:
  - Reverse nibble format for zone ip6.arpa

IPv6 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.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
- The ip6.int domains has been deprecated
  - Now using ip6.arpa for reverse

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

```
$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.
```

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.
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0   IN      PTR     host1.ip6.domain.edu
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0   IN      PTR     host2.domain.edu
;;
;; Can delegate to other nameservers in the usual way
;;
```

Questions ?