Reverse DNS

Presented by Joe Abley
SANOG 4, 2004

Outline

- General introduction
- IPv4 reverse DNS
  - Revere mapping and relation to address allocation
  - Problems and solutions for reverse mapping
- IPv6 reverse DNS

Addresses in the DNS

- Mapping from numbers to names
- It is just ordinary DNS
  - No different standards
  - No different operation
- But you might need a little background
  - There are some conventions
  - IPv6 is a moving/developing target
- First IPv4

Mapping of addresses to reverse

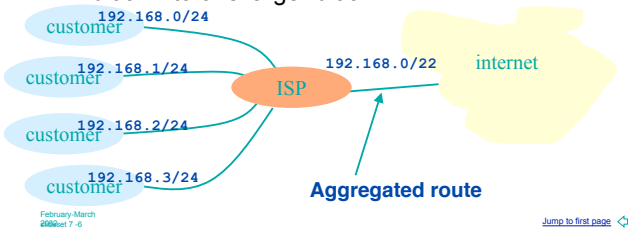
- Mapping from names to addresses is common:
  `bert.secret-wg.org  A 193.0.0.4`
- Sometimes one wants to know which name comes with a given address. If you can translate the address to a FQDN one can use the DNS
- Design goal: Delegate maintenance of the reverse DNS to the owner of the address block

Mapping the IPv4 address into the DNS: address allocation

- Address allocation is hierarchical:
  - blocks of addresses are allocated to LIRs/ISPs
  - smaller blocks are allocated to client
  - clients will assign address blocks to end users
- Routing is based on destinations for given address blocks
  - Historically on 8 bit boundaries (Class A,B,C)
  - Classless Inter Domain Routing (CIDR)

Classless inter domain routing (CIDR)

- Routing table size (router memory) is a limited resource
- Goal of CIDR: aggregate many small address block into one larger block

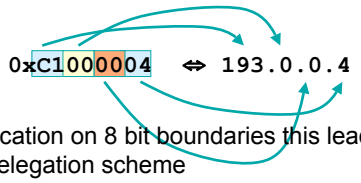


Mapping the IPv4 address into the DNS: address blocks

- Address block notation:
  <address>/<number of significant bits>
  For instance:
  `193.0.0.0/8 or short 193/8`
  `193.165.64/19=`
  `0xc1a54000/19 =`
  `1100 0001 1001 0101 0010 0000 0000 0000`
  19 bits

IPv4 address format

- An IP address is a 4 byte number normally represented by the decimal representation of the 4 bytes separated by dots



- With allocation on 8 bit boundaries this leads to a simple delegation scheme

Mapping the IPv4 address into the DNS

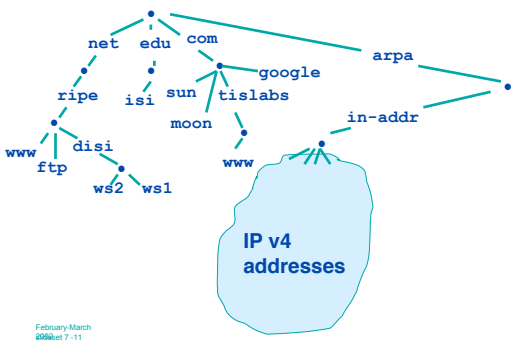
- Example 192.26.1.3
  - 192/8 is allocated to a RIR
  - 192.26/16 is allocated by RIR to LIR/ISP
  - 192.26.1/24 is assigned by ISP to a company.
- Delegation in the DNS:
  - root delegates 192 domain to RIR
  - RIR delegates “26” sub-zone to ISP
  - ISP delegates “1” sub-zone to company.
- Name that makes this possible: 1.26.192

Mapping addresses to names

- Revert the decimal representation:
  - 192.26.1.3 maps to 3.1.26.192 and put this under a top level domain.
  - For IPv4 this TLD is in-addr.arpa
- In the DNS one publishes PTR records to point back to the name:

```
4.0.0.193.in-addr.arpa 3600 IN PTR bert.secret-wg.org.
```


The reverse tree



Outline

- General introduction
- IPv4 reverse DNS
  - Revere mapping and relation to address allocation
  - Problems and solutions for reverse mapping
- IPv6 reverse DNS

Mapping address to names: mapping problems

- In IPv4 the mapping is done on 8 bit boundaries (class full), address allocation is class less
- Zone administration does not always overlap address administration
- If you have a /19 of address space: divide it in /24s and request a delegation for each one of them as soon as you use the address space
- /25 and smaller we will cover later

Setting your reverse zones

- The reverse zone file is a regular zone file.
  - SOA and NS RRs in the APEX
  - Mostly PTR records in the zone itself
- Make sure the zone is served by the masters and slaves
- Bind9 has a $GENERATE directive that might be handy

A reverse zone example

```
$ORIGIN 1.168.192.in-addr.arpa.
@       3600  IN SOA bert.secret-wg.org. (
                                olaf\.kolkman.ripe.net.
                                2002021301   ; serial
                                1h           ; refresh
                                30M          ; retry
                                1W           ; expiry
                                3600 )             ; neg. answ. ttl
        NS      ns.secret-wg.org.
        NS      ns2.secret-wg.org.

1       PTR     gw.secret-wg.org.
                router.secret-wg.org.
2       PTR     ns.secret-wg.org.
; BIND9 auto generate:  65 PTR host65.secret-wg.org
$GENERATE 65-127 $ PTR host$.secret-wg.org.
```

Note trailing dots

Getting a reverse delegation

- The procedure is registry dependent
  - For APNIC region read:
    http://www.apnic.net/db/revdel.html
    http://www.apnic.net/services/dns_guide.html
- Get a delegation from APNIC by filling adding a whois domain object:
  http://www.apnic.net/db/domain.html
- Only /16 and /24 delegations

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
```


Allocations smaller than /24

- Imagine a /25 address block delegated to a company by an ISP
- The company wants to maintain the reverse mapping of the address they use
- In the reverse DNS one can not delegate
- Use the 'classless inaddr' technique described in RFC 2317
- Based on the use of CNAME RRs
  - CNAME provide a means to alias names to another namespace

RFC2317 explained (1)

- **192.0.2.0/25 to organization A,**
- **192.0.2.128/26 to organization B and**
- **192.0.2.192/26 to organization C**

```
$ORIGIN 2.0.192.in-addr.arpa.
  ;
  1      PTR     host1.organizationA.com.
  2      PTR     host2.organizationA.com.
  3      PTR     host3.organizationA.com.
  ;
  129    PTR     host1.organizationB.com.
  130    PTR     host2.organizationB.com.
  131    PTR     host3.organizationB.com.
  ;
  193     PTR     host1.organizationC.com.
  194     PTR     host2.organizationC.com.
  195     PTR     host3.organizationC.com.
```


RFC2317 explained (2)

- Generate a 'sub domain' for each address block and delegate these to the children
  - Name the sub domain after the address block
    - 0/25, 128/26, and 190/26
    - 0-127, 128-189, 190-255
    - orgA, orgB, orgC
- For each name in the zone create a CNAME that points into the delegated namespace e.g.:

```
1 CNAME    1.orgA.2.0.193.inaddr-arpa.
```


RFC2317 explained(3) Parent zone

```
$ORIGIN 2.0.192.in-addr.arpa.
  @       IN      SOA     my-ns.my.domain. (
                          hostmaster.my.domain.
                          ...)
  ; ...
  orgA      NS ns1.organizationA.com.
            NS ns2.organizationA.com.
  1               CNAME   1.orgA
  2               CNAME   2.orgA
  ; ...
  orgB      NS ns1.organizationB.com.
            NS ns2.organizationB.com.
  129             CNAME   129.orgB
  130             CNAME   130.orgB

  ;
```


RFC2317 explained(4) Children's zone

```
$ORIGIN orgA.2.0.192.in-addr.arpa.
  @       IN      SOA     ns1.organizationA.com. (
                          hostmaster.organizationA.com.
                          ...)
  ; ...
  @          NS ns1.organizationA.com.
             NS ns2.organizationA.com.
  1          PTR    host1.organizationA.com.
  2          PTR    host2.organizationA.com.
```


RFC2317 explained(5)

- You could also delegate to a forward zone
  - Eases maintaining consistency in mapping

```
$ORIGIN 1.168.192.in-addr.arpa
;
;
24      CNAME   in24.foo.net.
25      CNAME   in25.foo.net.
26      CNAME   in26.foo.net.
27      CNAME   in27.foo.net.
28      CNAME   in28.foo.net.
;
; etc
```

```
$ORIGIN foo.net.
;
www     A   192.168.1.24
in24  PTR  www.foo.net.
ftp     A   192.168.1.25
in25  PTR  ftp.foo.net.
silver A   192.168.1.26
in26    PTR    silver.foo.net.
;
; etc
```


Outline

- General introduction
- IPv4 reverse DNS
- IPv6 reverse DNS
  - IPv6 addresses
  - IPv6 in the forward tree
  - IPv6 in the reverse tree

IPv6 addresses

- 128 bits
  - 64 low order bits “host” identifier
    - e.g. a mapping of the hosts’ Ethernet address
  - 64 high order bits “network” identifier
    - Further subdivision inside network id.
- Let’s look at notation first, then at further subdivision

IPv6 address Notation

- 16 bit integers (in Hex) separated by colons

```
FEDC:BA98:7654:3210:FEDC:BA98:7654:3210
1080:0000:0000:0000:0008:0800:200C:417A
```

- Leading zeros can be skipped

```
1080:0:0:0:8:800:200C:417A
```

- Consecutive NULL 16-bit numbers → “::”

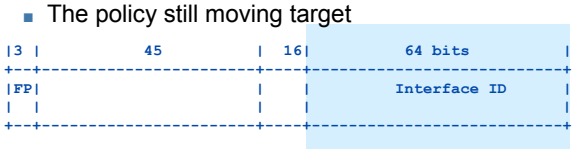
```
1080::8:800:200C:417A
```


IPv6 addresses

- For globally routable unicast addresses the 1st 3 bits are set to “001”
- Unicast addresses are further subdivided in “aggregates”
- More address classes available like:
  - Link local: fe80/10
  - Multicast: ff00/8
  - Mapped IPv4 address: 0::ffff:0:0:0:0/96

Globally routable unicast addresses

- 1st 3 bits are format prefix
- last 16 bits of network ID are used for ‘sites’
  - /48 assigned to end-users.
- RIRs minimum allocation size: /32 blocks
- The policy still moving target



Outline

- General introduction
- IPv4 reverse DNS
- IPv6 reverse DNS
  - IPv6 addresses
  - IPv6 in the forward tree
  - IPv6 in the reverse tree

IPv6 address representation in the DNS

- Multiple RR records for name to number
  - AAAA
  - A6 (Deprecated)
- Multiple ways to map address to DNS name
  - nibble notation
  - bit strings and nibbles (Deprecated)

AAAA RR

- Name to number mapping
- Similar to A RR for IPv4
- Uses the 'common' representation of the address

```
$ORIGIN example.com.
host   3600 IN AAAA 2001:238:f00:80:230:65ff:fe28:40f5
```


Outline

- General introduction
- IPv4 reverse DNS
- IPv6 reverse DNS
  - IPv6 addresses
  - IPv6 in the forward tree
  - IPv6 in the reverse tree
    - nibbles in ip6.arpa

Reverse DNS

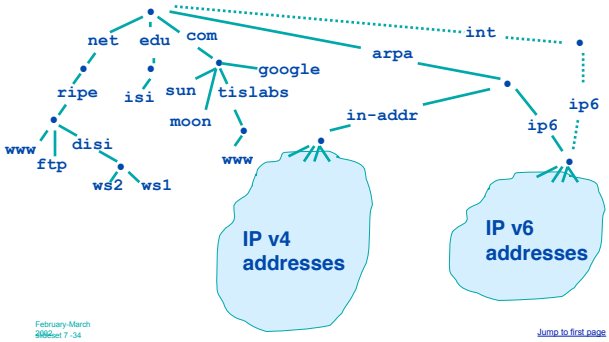
- Just as with IPv4 the responsibility for maintaining the reverse map can be delegated through the address hierarchy
- Number is translated into 4 bit nibbles under the ip6.arpa (ip6.int) TLD.

```
2001:0238::a00:46ff:fe06:1460
```

maps to:

```
0.6.4.1.6.0.e.f.f.f.6.4.0.0.a.0.0.0.0.0.0.0.0.0.8.3.2.0.1.0.0.2.ip6.arpa.
```


The reverse tree



Setting up reverse for SUB TLA

- Remember initial allocation is /32

```
FP |                 /35 allocations|
|3 |                             /32 |
|--|-----------------------------|--|----.....
00100000000000010000001000111000000?
------------ 35 bits --------------
   2   0   0   1:  0   2   3   8   0/35
```

- 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
```

Can be 1 or 0

DNS data and the transport layer

- In principle the transport layer does not have influence on DNS data;
  - Data can be published by servers running on IPv4 or IPv6, content should not differ
  - Transition problem: IPv6 client might not be able to see IPv6 servers and vice verse
  - Transition problems are by far not solved
- Exception to above: IPv4 mapped addresses
  - Mapping is depended on OS libraries

Questions

- Let’s do it…..