IPV4 IN-ADDR.ARPA REVERSE MAPPING DOMAIN

Reverse Mapping looks horribly complicated. It is not. As with all things when we understand what is being done and why - all becomes as clear as mud.

We defined the normal domain name structure as a tree starting from the root. We write a normal domain name LEFT to RIGHT but the hierarchical structure is RIGHT to LEFT.

```
domain name = www.example.com

highest node in tree is = .com (technically the normally silent . for root is the

highest but...)

next (lower) = .example

next (lower) = www
```

An IPv4 address is written as:

```
192.168.23.17
```

This IPv4 address defines a host (17) which happens to be in a Class C address range (192.168.23.x). In this case the most important part (the highest node in the address hierarchy) is on the LEFT (192) not the RIGHT. This is a tad awkward and would make it impossible to construct a sensible tree structure that could be searched in a single lifetime.

The solution is to reverse the order of the address and place the result under the special domain IN-ADDR.ARPA (you will see this also written as in-addr.arpa which is perfectly legitimate since domain names are case insensitive but the case should be preserved between query and response. You may elect to use whatever you wish including IN-addr.Arpa if that is your preference).

The last part of the IPv4 Address (17) is the host address and hosts, from our previous reading, are typically defined inside a zone file so we will ignore it and only use the Class C address base. The result of our manipulations are:

IP address =192.168.23.17

Class C base = 192.168.23 ; omits the host address = 17

Reversed Class C base = 23.168.192

Added to IN-ADDR.ARPA domain = 23.168.192.IN-ADDR.ARPA

This is shown in figure 3.0 below.

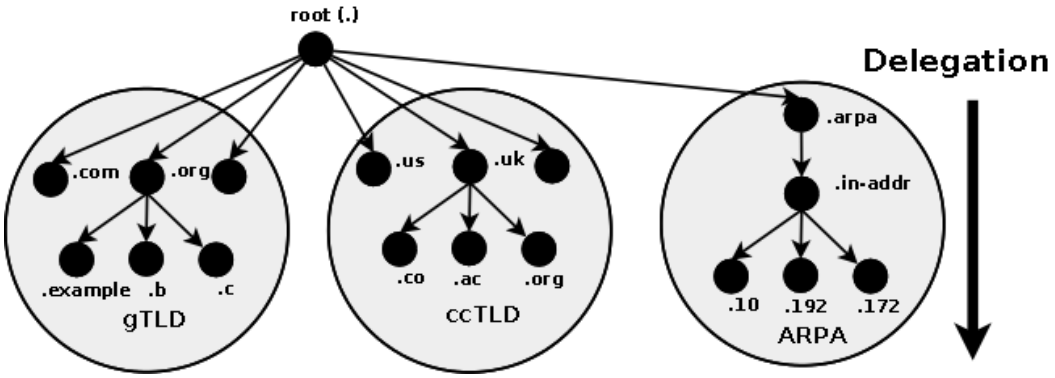

Figure 3.0 IN-ADDR.ARPA Reverse Mapping

Finally, we construct a zone file to describe all the hosts (nodes) in the Reverse Mapped zone using PTR Records. The resulting file will look something like this:

```
$TTL 2d  ; 172800 seconds

$ORIGIN 23.168.192.IN-ADDR.ARPA.

@          IN     SOA   ns1.example.com. hostmaster.example.com. (

                        2003080800 ; serial number

                        3h         ; refresh

                        15m        ; update retry

                        3w         ; expiry

                        3h         ; nx = nxdomain ttl

                        )
```

```
        IN    NS    ns1.example.com.
        IN    NS    ns2.example.com.
1       IN    PTR   www.example.com. ; qualified name
2       IN    PTR   joe.example.com.
.....
17      IN    PTR   bill.example.com.
.....
74      IN    PTR   fred.example.com.
....
```

Notes:

1. We must use qualified names ending with a dot (in fact they are Fully Qualified Domain Names - FQDNs) with the PTR target (left-hand) name in reverse mapped zone files because if we did not our $ORIGIN directive would lead to some strange results. For example, if we wrote an unqualified name such as:

```
2. 74       IN    PTR fred
```

Using the $ORIGIN substitution rule the above would expand to fred.23.168.192.IN-ADDR.ARPA. which is probably not what we intended.

3. If a reverse-map file is not included in our DNS configuration then, as normal, the query will pass to the DNS hierarchy. In the case where local IP addreses are globally routable (increasingly rare) an ISP or service provider may be responsible for maintaining the reverse map in which case such reverse-map queries must access the DNS hierarchy. Most frequently, private IP addresses are used, such as 192.168.x.x, 10.x.x.x or some IP ranges in 172.x.x.x, in local networks (and exclusively with home networks) with a NAT configuration translating to a global address before being routed to the public network. In such configurations it is imperative that a reverse-map covering the range of private IP addresses be included in the local DNS configuration. Failure to do so may cause local applications to fail or give strange results (none of them good) because the reverse-map request will be responded to by the DNS hierarchy where such private addresses are meaningless. In order to limit the negative effect of such mis-configuration BIND introduced the empty-zones-enable statement which defaults to a state that will minimise damage to the DNS hierarchy. However, BIND's default option does not

solve the problem for local applications which may depend upon correct local results from a reverse-map query. It is, and remains, a serious mis-configuration of DNS to not include a reverse-may for all local (RFC 1918) addresses.

4. There are no A RRs for the defined NS names (respectively ns1.example.com and ns2.example.com) since both are out-of-zone names. Any lookup is done via the forward zone file for example.com in which suitable A RRs for these names must exist.

Skeleton files may be generated using the IPv4 reverse map zone file tool.

Source: http://www.zytrax.com/books/dns/ch3/index.html#overview