DNS Session 3: Configuration of Authoritative Nameservice

Brian Candler
NSRC

1

Recap

➤ DNS is a distributed database
➤ Resolver asks Cache for information
➤ Cache traverses the DNS delegation tree to find Authoritative nameserver which has the information requested
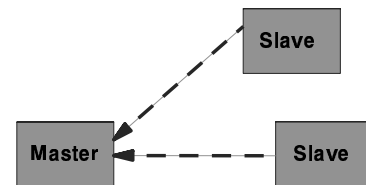➤ Bad configuration of authoritative servers can result in broken domains

2

DNS Replication

➤ For every domain, we need more than one authoritative nameserver with the same information (RFC 2182)
➤ Data is entered in one server (Master) and replicated to the others (Slave(s))
➤ Outside world cannot tell the difference between master and slave
  → NS records are returned in random order for equal load sharing
➤ Was called "primary" and "secondary"

3

Slaves connect to Master to retrieve copy of zone data

The master does not "push" data to the slaves

4

When does replication take place?

➤ Slaves poll the master periodically - called the "Refresh Interval" - to check for new data
  → Originally this was the only mechanism
➤ With new software, master can also notify the slaves when the data changes
  → Results in quicker updates
➤ The notification is unreliable (e.g. network might lose a packet) so we still need checks at the Refresh Interval

5

Serial Numbers

➤ Every zone file has a Serial Number
➤ Slave will only copy data when this number INCREASES
  → Periodic UDP query to check Serial Number
  → If increased, TCP transfer of zone data
➤ It is your responsibility to increase the serial number after every change, otherwise slaves and master will be inconsistent

6

Recommended serial number format: YYYYMMDDNN

➤ YYYY = year
➤ MM = month (01-12)
➤ DD = day (01-31)
➤ NN = number of changes today (00-99)
  → e.g. if you change the file on 3rd March 2004, the serial number will be 2004030300. If you change it again on the same day, it will be 2004030301

Serial Numbers: Danger 1

➤ If you ever *decrease* the serial number, the slaves will *never update again* until the serial number goes above its previous value
➤ RFC1912 section 3.1 explains how to fix this problem
➤ At worst, you have to contact all your slaves and get them to delete their copy of the zone data

Serial Numbers: Danger 2

➤ Serial no. is a 32-bit unsigned number
➤ Range: 0 to 4,294,967,295
➤ Any value larger than this is silently truncated
➤ e.g. 20040303000 (note extra digit)
  → = 4AA7EC198 (hex)
  → = AA7EC198 (32 bits)
  → = 2860433816
➤ If you make this mistake, then correct it, the serial number will have decreased

Configuration of Master

➤ /etc/named.conf points to zone file (manually created)
➤ Choose a logical place to keep them
  → e.g. /var/named/m/tiscali.co.uk
  → or /var/named/m/uk.co.tiscali

```
zone "example.com" {
    type master;
    file "m/example.com";
    allow-transfer { 192.188.58.126;
                     192.188.58.2; };
};
```

Configuration of Slave

➤ /etc/named.conf points to IP address of master and location of zone file
➤ Zone files are transferred automatically
  → Don't touch them

```
zone "example.com" {
    type slave;
    masters { 192.188.58.126; };
    file "s/example.com";
    allow-transfer { none; };
};
```

Master and Slave

➤ It's perfectly OK for one server to be Master for some zones and Slave for others
➤ That's why we recommend keeping the files in different directories
  → /var/named/m/
  → /var/named/s/

allow-transfer { ... }

➤ Remote machines can request a transfer of the entire zone contents
➤ By default, this is permitted to anyone
➤ Better to restrict this
➤ You can set a global default, and override this for each zone if required

```
options {
    allow-transfer { 127.0.0.1; };
};
```

13

Structure of a zone file

➤ Global options
 → $TTL 1d
 → Sets the default TTL for all other records
➤ SOA RR
 → "Start Of Authority"
 → Housekeeping information for the zone
➤ NS RRs
 → List all the nameservers for the zone, master and slaves
➤ Other RRs
 → The actual data you wish to publish

1

Format of a Resource Record

```
www      3600     IN     A     212.74.112.80
Domain   TTL      Class  Type  Data
```

➤ One per line (except SOA can extend over several lines)
➤ If you omit the Domain Name, it is the same as the previous line
➤ TTL shortcuts: eg. 60s, 30m, 4h, 1w2d
➤ If you omit the TTL, it takes the $TTL default value
➤ If you omit the Class, it defaults to IN
➤ Type and Data cannot be omitted
➤ Comments start with SEMICOLON (;)

15

Shortcuts

➤ If the Domain Name does not end in a dot, the zone's own domain ("origin") is appended
➤ A Domain Name of "@" means the origin itself
➤ e.g. in zone file for example.com:
 → @ means **example.com.**
 → www means **www.example.com.**

1

If you write this...

```
$TTL 1d
@                       SOA ( ... )
                        NS  ns0
                        NS  ns0.as9105.net.
; Main webserver
www                     A   212.74.112.80
                        MX 10 mail
```

... it becomes this

```
example.com.     86400  IN  SOA ( ... )
example.com.     86400  IN  NS  ns0.example.com.
example.com.     86400  IN  NS  ns0.as9105.net.
www.example.com. 86400  IN  A   212.74.112.80
www.example.com. 86400  IN  MX 10 mail.example.com.
```

17

Format of the SOA record

```
$TTL 1d

@  1h  IN  SOA  ns1.example.net. brian.nsrc.org. (
                2004030300      ; Serial
                8h              ; Refresh
                1h              ; Retry
                4w              ; Expire
                1h )            ; Negative

        IN  NS  ns1.example.net.
        IN  NS  ns2.example.net.
        IN  NS  ns1.othernetwork.com.
```

1

Format of SOA record

➤ ns1.example.net
  → hostname of master nameserver
➤ brian.nsrc.org.
  → E-mail address of responsible person, with "@" changed to dot
➤ Serial number
➤ Refresh interval
  → How often Slave checks serial number on Master
➤ Retry interval
  → How often Slave checks serial number if the master did not respond

Format of SOA record (cont)

➤ Expiry time
  → If the slave is unable to contact the master for this period of time, it will delete its copy of the zone data
➤ Negative / Minimum
  → Old software used this as a minimum value of the TTL
  → Now it is used for negative caching: indicates how long a cache may store the non-existence of a RR
➤ RIPE-203 has recommended values
  → http://www.ripe.net/ripe/docs/dns-soa.html

Format of NS records

```
$TTL 1d

@  1h  IN  SOA  ns1.example.net. brian.nsrc.org. (
            2004030300      ; Serial
            8h              ; Refresh
            1h              ; Retry
            4w              ; Expire
            1h )            ; Negative

       IN  NS   ns1.example.net.
       IN  NS   ns2.example.net.
       IN  NS   ns1.othernetwork.com.
```

➤ List all authoritative nameservers for the zone - master and slave(s)
➤ Must point to HOSTNAME not IP address

Format of other RRs

➤ IN A 1.2.3.4
➤ IN MX 10 mailhost.example.com.
  → The number is a "preference value". Mail is delivered to the lowest-number MX first
  → Must point to HOSTNAME not IP address
➤ IN CNAME host.example.com.
➤ IN PTR   host.example.com.
➤ IN TXT   "any text you like"

When you have added or changed a zone file:

➤ Check the serial number!
➤ **named-checkzone example.com**
  → bind 9 feature
  → reports syntax errors; correct them!
➤ **rndc reload**
  → or: rndc reload example.com
➤ tail /var/log/messages

These checks are ESSENTIAL

➤ If you have an error in named.conf or a zone file, named will continue to run but not authoritative for the bad zone(s)
➤ You will be lame for the zone without realising it
➤ Slaves will not be able to contact the master
➤ Eventually (e.g. 4 weeks later) the slaves will expire the zone
➤ Your domain will stop working

Other checks you can do

➤ dig +norec @x.x.x.x example.com. soa
  → Check the AA flag
  → Check the master and all the slaves
  → Check the serial numbers match
➤ dig @x.x.x.x example.com. axfr
  → "Authority Transfer"
  → Requests a full copy of the zone contents over TCP, as slaves do to master
  → This will only work from IP addresses listed in the allow-transfer {...} section

So now you have working authoritative nameservers!

➤ But remember that none of this will work until you have *delegation* from the domain above
➤ That is, they put in NS records for your domain, pointing at your nameservers
➤ You have also put NS records within the zone file
➤ The two sets should match

TOP TEN ERRORS
in authoritative nameservers

➤ All operators of auth nameservers should read RFC 1912
  → Common DNS Operational and Configuration Errors
➤ See also RFC 2182
  → Selection and Operation of Secondary DNS Servers

1. Serial number errors

➤ Forgot to increment serial number
➤ Incremented serial number, and then decremented it
➤ Used serial number greater than 2^32
➤ Impact:
  → Slaves do not update
  → Master and slaves have inconsistent data
  → Caches will sometimes get the new data and sometimes old - intermittent problem

2. Comments in zone files starting '#' instead of ';'

➤ Syntax error in zone file
➤ Master is no longer authoritative for the zone
➤ Slaves cannot check SOA
➤ Slaves eventually expire the zone, and your domain stops working entirely
➤ Use 'named-checkzone'
➤ Use 'tail /var/log/messages'

3. Other syntax errors in zone files

➤ e.g. omitting the preference value from MX records
➤ Same impact

4. Missing the trailing dot

```
; zone example.com.
@  IN  MX 10  mailhost.example.com

becomes

@  IN  MX 10  mailhost.example.com.example.com.
```

```
; zone 2.0.192.in-addr.arpa.
1  IN  PTR  host.example.com

becomes

1  IN  PTR  host.example.com.2.0.192.in-addr.arpa.
```

31

5. NS or MX records pointing to IP address

➤ They must point to hostnames, not IP addresses
➤ Unfortunately a few mail servers *do* accept IP addresses in MX records, so you may not see a problem with all remote sites

3

6. Slave cannot transfer zone from master

➤ Access restricted by allow-transfer { ... } and slave not listed
➤ Or IP filters not configured correctly
➤ Slave will be lame (non-authoritative)

33

7. Lame delegation

➤ You cannot just list any nameserver in NS records for your domain
➤ You must get agreement from the nameserver operator and they must configure it as a slave for your zone
➤ At best: slower DNS resolution and lack of resilience
➤ At worst: intermittent failures to resolve your domain

3

8. No delegation at all

➤ You can configure "example.com" on your nameservers but the outside world will not send requests to them until you have delegation
➤ The problem is hidden if your nameserver is acting both as your cache and as authoritative nameserver
➤ Your own clients can resolve www.example.com, but the rest of the world cannot

35

9. Out-of-date glue records

➤ See later

3

10. Not managing TTL correctly during changes

➤ e.g. if you have a 24 hour TTL, and you swing www.example.com to point to a new server, then there will be an extended period when some users hit one machine and some hit the other
➤ Follow the procedure:
  → Reduce TTL to 10 minutes
  → Wait at least 24 hours
  → Make the change
  → Put the TTL back to 24 hours

?

Final topics

➤ Reverse DNS
➤ How to delegate a subdomain

How to manage reverse DNS

➤ If you have at least a /24 of address space then your provider will arrange delegation to your nameservers
➤ e.g. your netblock is 192.0.2.0/24
➤ Set up zone 2.0.192.in-addr.arpa.
➤ If you have more than a /24 then each /24 will be a separate zone
➤ If you are lucky enough to have a /16 then it will be a single zone
  → 172.16.0.0/16 is 16.172.in-addr.arpa.

Example: 192.0.2.0/24

```
zone "2.0.192.in-addr.arpa" {
    type master;
    file "m/192.0.2";
    allow-transfer { ... };
};
```

```
            /var/named/m/192.0.2

@  IN  SOA   ....
   IN  NS    ns0.example.com.
   IN  NS    ns0.othernetwork.com.

1  IN  PTR   router-e0.example.com.
2  IN  PTR   ns0.example.com.
3  IN  PTR   mailhost.example.com.
4  IN  PTR   www.example.com.
; etc
```

How it works

➤ e.g. for 192.0.2.4, the remote host will lookup 4.2.0.192.in-addr.arpa. (PTR)
➤ The query follows the delegation tree as normal. If all is correct, it will reach your nameservers and you will reply
➤ Now you can see why the octets are reversed
  → The owner of a large netblock (192/8) can delegate reverse DNS in chunks of /16. The owner of a /16 can delegate chunks of /24

There is nothing special about reverse DNS

➤ You still need master and slave(s)
➤ It won't work unless you get delegation from above
➤ DO make sure that if you have PTR records for an IP address, that the hostname resolves back to the same IP address
  → Otherwise many sites on the Internet will believe you are spoofing reverse DNS and will refuse to let you connect

What if you have less than /24?

➤ Reverse DNS for the /24 has been delegated to your upstream provider
➤ Option 1: ask your provider to insert PTR records into their DNS servers
  → Problem: you have to ask them every time you want to make a change
➤ Option 2: follow the procedure in RFC2317
  → Uses a trick with CNAME to redirect PTR requests for your IPs to your nameservers

e.g. You own 192.0.2.64/29

```
; In the provider's 2.0.192.in-addr.arpa zone file
64   IN  CNAME  64.64/29.2.0.192.in-addr.arpa.
65   IN  CNAME  65.64/29.2.0.192.in-addr.arpa.
66   IN  CNAME  66.64/29.2.0.192.in-addr.arpa.
67   IN  CNAME  67.64/29.2.0.192.in-addr.arpa.
68   IN  CNAME  68.64/29.2.0.192.in-addr.arpa.
69   IN  CNAME  69.64/29.2.0.192.in-addr.arpa.
70   IN  CNAME  70.64/29.2.0.192.in-addr.arpa.
71   IN  CNAME  71.64/29.2.0.192.in-addr.arpa.
64/29 IN  NS  ns0.customer.com.
64/29 IN  NS  ns1.customer.com.
```

Set up zone "64/29.2.0.192.in-addr.arpa" on your nameservers

```
65   IN  PTR    www.customer.com.
66   IN  PTR    mailhost.customer.com.
; etc
```

How do you delegate a subdomain?

➤ In principle straightforward: just insert NS records for the subdomain, pointing at someone else's servers
➤ If you are being careful, you should first *check* that those servers are authoritative for the subdomain
  → using "dig" on all the servers
➤ If the subdomain is managed badly, it reflects badly on you!

Zone file for "example.com"

```
$TTL 1d
@  1h  IN  SOA  ns1.example.net. brian.nsrc.org. (
                2004030300    ; Serial
                8h            ; Refresh
                1h            ; Retry
                4w            ; Expire
                1h )          ; Negative

        IN  NS  ns1.example.net.
        IN  NS  ns2.example.net.
        IN  NS  ns1.othernetwork.com.

; My own zone data
        IN  MX  10  mailhost.example.net.
www     IN  A   212.74.112.80

; A delegated subdomain
subdom IN  NS  ns1.othernet.net.
        IN  NS  ns2.othernet.net.
```

There is one problem here:

➤ NS records point to names, not IPs
➤ What if "example.com" is delegated to "ns.example.com"?
➤ Someone who is in the process of resolving (say) www.example.com has to first resolve ns.example.com
➤ But they cannot resolve ns.example.com without first resolving ns.example.com !!

In this case you need "glue"

➤ A "glue record" is an A record for the nameserver
➤ Example: consider the .com nameservers

```
; this is the com. zone

example        NS  ns.example.com.
               NS  ns.othernet.net.

ns.example.com.  A 192.0.2.1    ; GLUE RECORD
```

Don't put in glue records except where necessary

➤ In the previous example, "ns.othernet.net" is not a subdomain of "example.com". Therefore no glue is needed.
➤ Out-of-date glue records are a big source of problems
  → e.g. after you have renumbered your nameserver to another network
➤ Difficult to debug, requires "dig +norec"

Example where a glue record IS needed

```
; My own zone data
       IN  MX  10  mailhost.example.net.
www    IN  A   212.74.112.80

; A delegated subdomain
subdom    IN  NS  ns1.subdom        ; needs it
          IN  NS  ns2.othernet.net. ; doesn't
ns1.subdom  IN  A  192.0.2.4
```

Checking for glue records

➤ dig +norec @a.gtld-servers.net. www.as9105.net. a
➤ Look for A records in the "Additional" section whose TTL does not count down

```
$ dig +norec @a.gtld-servers.net. www.as9105.net. a
...
;; flags: qr; QUERY: 1, ANSWER: 0, AUTHORITY: 2, ADD'L: 1
;; QUERY SECTION:
;;       www.as9105.net, type = A, class = IN

;; AUTHORITY SECTION:
as9105.net.            2D IN NS      ns0.as9105.com.
as9105.net.            2D IN NS      ns0.tiscali.co.uk.

;; ADDITIONAL SECTION:
ns0.as9105.com.        2D IN A       212.139.129.130
```

DNS: overall summary

➤ Distributed database of RRs
➤ Three roles: resolver, cache, authoritative
➤ Resolver statically configured with the nearest cache(s)
  → e.g. /etc/resolv.conf
➤ Caches statically configured with a list of root nameservers
  → zone type "hint", /var/named/named.ca

DNS: overall summary (cont)

➤ Root nameservers contain delegations (NS records) to gtld or country-level servers (com, uk etc)
➤ Further delegations to subdomains
➤ Cache finally locates an authoritative server containing the RRs we require
➤ Errors in delegation or in configuration of authoritative servers result in no answer or inconsistent answers