RHCE
BOOT CAMP

BIND

CONFIG FILES

- BIND basically has two types of configuration files:
  - BIND configuration file, specific to BIND and it's features
  - Database files, or zone files, which contain DNS resource records used to describe all of the DNS information needed in a domain

RESOURCE RECORDS

- A resource record contains the DNS *information* about a domain.

- There are several types of resource records, including address records (A), mail exchangers (MX) and name servers (NS).

- Every domain has at least 2 resource records, an SOA and an NS. But that wouldn't be a very useful domain, so there are usually quite a few more records, defining addresses, mail exchangers, canonical names and more.

SOA

- Start Of Authority: This resource record defines authority for a zone.

```
domain IN SOA nameserver adminemail (
        serial
        refresh
        retry
        expire
        negativettl
)
```

We'll discuss these later!

SOA EXAMPLE

```
rackspace.com. IN SOA ns1.rackspace.com. dnsadmin.rackspace.com. (
   2009123004   ; Serial number
   3h           ; Refresh interval
   1h           ; Retry interval
   1w           ; Expires
   1h           ; Negative TTL
)
```

NS

- Name Server: Defines authoritative nameservers for the zone.

  zone **IN NS** *nameserver*

- Example:

  ```
  rackspace.com. IN NS ns1.rackspace.com.
  ```

A

- Address: Maps hostnames to IP addresses

  hostname **IN A** *ipaddress*

- Example:

```
ns1.rackspace.com. IN A 192.168.1.5
```

CNAME

- Canonical Name: Maps *alias* hostnames to their *canonical* counterparts.

  aliashostname **IN CNAME** *canonicalhostname*

- Example:

  ```
  ns.rackspace.com. IN CNAME ns1.rackspace.com.
  ```

CANONICAL?

- In layman terms, canonical is another way of saying “real”, “absolute” or “official”.
- So a canonical name refers to the “official” name for a host.
- Creating an alias for a host means that you have to decide on the canonical name, which would be some A record.
- When a resolver performs an A query on a CNAME, the nameserver looks up the canonical name to find out the address to return.

WHY NOT JUST USE A?

- There are basically two reasons to use CNAME records instead of just lots of A records.
  - First, ease of maintenance. If you need 10 names for one machine, defining them with CNAME is easiest if you then need to change the IP address of the machine. Only one change instead of 11.
- Second, *canonicalization*. Some services, most notably sendmail, will convert all aliases into their canonical names. This simplifies mail configuration.

PTR

- Pointer: Maps an IP address back to a name, specifically the canonical name.

  ipaddressdomain **IN PTR** *canonicalhostname*

- Example:

  ```
  5.1.168.192.in-addr.arpa. IN PTR ns1.rackspace.com.
  ```

PTR

- Remember, there is only one PTR record for a given IP address, and it should always point to the *canonical hostname*.

- Also, as a side note, make sure your mail servers map both directions exactly. This is important for proper authentication:

  - `mailer.mydomain.com. -> 192.168.1.50`

  - `50.1.168.192.in-addr.arpa. -> mailer.mydomain.com`

MX

- Mail Exchanger: Defines hosts responsible for incoming email for the named zones.

```
zone IN MX preference mailhandler
```

- Example:

```
rackspace.com. IN MX 10 mail1.rackspace.com.
```

MX RECORDS

- MX records allow for enhanced mail routing functionality.
- When an email is shipped out, the server canonicalizes the delivery address. So, for example, bob@ns.rackspace.com becomes bob@ns1.rackspace.com.
- Then the server looks up the MX records for ns1.rackspace.com, choosing the record with the **lowest** preference and attempting delivery there. If delivery fails, the next lowest is attempted.
- This allows for backup email servers!

PUTTING IT ALL TOGETHER

db.rackspace.com:

```
$TTL 1h
rackspace.com. IN SOA ns1.rackspace.com. dnsadmin.rackspace.com. (
   2009123004   ; Serial number
   3h           ; Refresh interval
   1h           ; Retry interval
   1w           ; Expires
   1h           ; Negative TTL
)

rackspace.com. IN NS ns1.rackspace.com.
ns1.rackspace.com. IN A 192.168.1.5
mail1.rackspace.com. IN A 192.168.1.20
ns.rackspace.com. IN CNAME ns1.rackspace.com.
rackspace.com. IN MX 10 mail1.rackspace.com.
```

TOGETHER...

db.192.168.1:

```
$TTL 1h
1.168.192.in-addr.arpa. IN SOA ns1.rackspace.com. dnsadmin.rackspace.com. (
   2009123004   ; Serial number
   3h           ; Refresh interval
   1h           ; Retry interval
   1w           ; Expires
   1h           ; Negative TTL
)

1.168.192.in-addr.arpa. IN NS ns1.rackspace.com.
5.1.168.192.in-addr.arpa. IN PTR ns1.rackspace.com.
20.1.168.192.in-addr.arpa. IN PTR mail1.rackspace.com.
```

SO WHAT ELSE?

- In addition to the zone files for your domains, you need a couple more zone files to get BIND up and running.
  - Loopback address
  - Root hints

LOOPBACK ADDRESS?

- Someone has to take responsibility for loopback address requests! It's simple enough. `db.127.0.0`:

```
$TTL 1w
0.0.127.in-addr.arpa. IN SOA ns1.rackspace.com. dnsadmin.rackspace.com. (
   2009123004   ; Serial number
   3h           ; Refresh interval
   1h           ; Retry interval
   1w           ; Expires
   1h           ; Negative TTL
)

0.0.127.in-addr.arpa. IN NS ns1.rackspace.com.
1.0.0.127.in-addr.arpa. IN PTR localhost.
```

ROOT HINTS!

- The root hints tell the nameserver where those DNS Root Servers are located, so that requests for hosts outside of your authoritative zones can be resolved.
- This one is the simplest to put together. You don't even have to write it!
- Simply ftp the db.cache file from ftp.rs.internic.net/domain

WHEW

- Finally. All of the zone files are put together and ready. Final step? Configuring BIND.

- The config file is generally `/var/named/chroot/etc/named.conf`

NAMED.CONF GENERAL

```
acl “clients” { 192.168.0.0/24;};

acl “servers” { 192.168.1.2; 192.168.1.3;};

options {

    directory “/var/named”;

    forwarders { 192.168.0.254; };

    allow-query { clients; };

    allow-transfer { servers; };

};
```

NAMED.CONF ZONES

```
zone “example.com” IN {

     type master;

     file “example.com.zone”;

};

zone “example.com” IN {

     type slave;

     file “slave.example.com.zone”;

     masters { 192.168.2.254; };

};
```

LAB

1. Configure your machine to act as the authoritative nameserver for a "`demoX.example.com`" domain and a "`rhceX.example.com`" domain, where X is your station number.

2. For both domains, configure these records:

   - "`www`", "`mail`", and "`ns`" should all resolve to your IP address
   - "`web`" should resolve to "`www`"
   - "`mail`" should be listed as the primary MTA for the domain
   - "`ns`" should be listed as the DNS server for the domain

3. Also configure your machine to respond to reverse DNS lookups, such that your own IP address will resolve to "`www.rhceX.example.com`"

4. Set up your machine to forward any other DNS requests to server1.

```
slideshow.end();
```