
# DNS/DNSSEC Workshop in conjunction with Lanka Network Operators Group (LKNOG) Conference 12-16 August 2024

## Lab Exercises

**Champika Wijayatunga**
**Technical Engagement Sr. Manager - Asia Pacific**
**<champika.wijayatunga@icann.org>**

![icann](./icann.jpg)

# Configure a recursive resolver

During this practice we are only going to access the following equipment:

* **grpX-cli** : client
* **grpX-resolv1** & **grpX-resolv2** : recursive servers (resolvers)

## Setting up a BIND validating recursive server.

We use the container "Resolv 1" (recursive server) [**grpX-resolv1**].

This container already has the BIND9 packages downloaded and installed.

We switch to the root user:

```
$ sudo su -
```

We go to the /etc/bind directory:

```
# cd /etc/bind
```

At this point we must configure some BIND9 options.
To do this we edit the file `/etc/bind/named.conf.options`:

```
# nano named.conf.options
```

Now we add the options to indicate (when resolving) which are the IP addresses that will be able to send DNS queries and at the same time to which IP addresses our server will listen on port 53 (in this case both prefixes are identical). The file should be as follows:

```
options {
        directory "/var/cache/bind";

        // If there is a firewall between you and nameservers you want
        // to talk to, you may need to fix the firewall to allow
multiple
```

```
        // ports to talk. See http://www.kb.cert.org/vuls/id/800113

        // If your ISP provided one or more IP addresses for stable
        // nameservers, you probably want to use them as forwarders.
        // Uncomment the following block, and insert the addresses
replacing
        // the all-0's placeholder.

        // forwarders {
        //      0.0.0.0;
        // };

        //
============================================================================
==
        // If BIND logs error messages about the root key being
expired,
        // you will need to update your keys. See https://www.isc.org/
bind-keys
        //
============================================================================
==
        dnssec-validation auto;

        listen-on port 53 { localhost; 100.100.0.0/16; };
                                                <--- Add this
        listen-on-v6 port 53 { localhost; fd89:59e0::/32; };
                                <--- Add this

        allow-query { localhost; 100.100.0.0/16; fd89:59e0::/32; };
                <--- Add this

        recursion yes;


                <--- Add this
};
```

Once we finish editing the configuration file, we execute a command that allows us to quickly check if the configuration is semantically correct (if the command does not return anything, it means that it did not find errors in the configuration files):

```
# named-checkconf
```

Finally we restart the server so that it takes the configuration changes:

```
# systemctl restart bind9
```

And we check the status of the bind9 process:

```
# systemctl status bind9
```

We should obtain an output similar to the following:

```
● named.service - BIND Domain Name Server
   Loaded: loaded (/lib/systemd/system/named.service; enabled; vendor
preset: enabled)
  Drop-In: /etc/systemd/system/service.d
           └─lxc.conf
   Active: **active (running)** since Thu 2021-05-13 01:38:27 UTC; 4s
ago
    Docs: man:named(8)
  Main PID: 849 (named)
   Tasks: 50 (limit: 152822)
   Memory: 103.2M
   CGroup: /system.slice/named.service
           └─849 /usr/sbin/named -f -u bind

May 13 01:38:27 resolv1.grpX.<lab_domain>.te-labs.training named[849]:
**command channel listening on ::1#953**
May 13 01:38:27 resolv1.grpX.<lab_domain>.te-labs.training named[849]:
managed-keys-zone: loaded serial 6
May 13 01:38:27 resolv1.grpX.<lab_domain>.te-labs.training named[849]:
zone 0.in-addr.arpa/IN: loaded serial 1
May 13 01:38:27 resolv1.grpX.<lab_domain>.te-labs.training named[849]:
zone 127.in-addr.arpa/IN: loaded serial 1
May 13 01:38:27 resolv1.grpX.<lab_domain>.te-labs.training named[849]:
zone localhost/IN: loaded serial 2
May 13 01:38:27 resolv1.grpX.<lab_domain>.te-labs.training named[849]:
zone 255.in-addr.arpa/IN: loaded serial 1
May 13 01:38:27 resolv1.grpX.<lab_domain>.te-labs.training named[849]:
**all zones loaded**
May 13 01:38:27 resolv1.grpX.<lab_domain>.te-labs.training named[849]:
**running**
May 13 01:38:27 resolv1.grpX.<lab_domain>.te-labs.training named[849]:
managed-keys-zone: Key 20326 for zone . is now trusted (acceptance
timer>
May 13 01:38:27 resolv1.grpX.<lab_domain>.te-labs.training named[849]:
resolver priming query complete
```

### Test your new validating resolver

Run the following commands and confirm if you receive the "ad" flag:

1. dig SOA com. @100.100.X.67
2. dig A www.icann.org @100.100.X.67
3. dig NS icann.org @100.100.X.67
4. dig NS *grpX*.<*lab_domain*>.te-labs.training @100.100.X.67
5. dig SOA *grpX*.<*lab_domain*>.te-labs.training @100.100.X.67
6. dig SOA *grpX*.<*lab_domain*>.te-labs.training @100.100.X.130

Compare the flags on responses to Q5 and Q6. Why are they different ?

### Set your recursive resolver OS to use your local validating recursive resolver sofltware.
Still in **grpX-resolv1** server, edit the **/etc/resolv.conf** config file and replace whatever is there by the following only:

```
nameserver 100.100.X.67
```

Save and exit.

Then try the following queries:

1. dig SOA com.
2. dig NS com.
3. dig A www.icann.org
4. dig NS icann.org
5. dig NS *grpX*.<*lab_domain*>.te-labs.training
6. dig SOA *grpX*.<*lab_domain*>.te-labs.training

## Setting up an Unbound recursive server

We use the container "Resolv 2" (recursive server) [**grpX-resolv2**].

This container already has the UNBOUND packages downloaded and installed.

At this point we must configure some UNBOUND options.
To do this we switch to the root user and edit **/etc/unbound/unbound.conf** config file:

```
$ sudo su -
# nano /etc/unbound/unbound.conf
```

```

Now we add the options to indicate (when resolving) which are the interfaces on which it will listen for queries, the IP addresses that will be able to send it DNS queries, the port it will use (53), and some other parameters. The file should be as follows:

```
# Unbound configuration file for Debian.
#
# See the unbound.conf(5) man page.
#
# See /usr/share/doc/unbound/examples/unbound.conf for a commented
# reference config file.
#
# The following line includes additional configuration files from the
# /etc/unbound/unbound.conf.d directory.

server:
        interface: 0.0.0.0
        interface: ::0

        access-control: 127.0.0.0/8 allow
        access-control: 100.100.0.0/16 allow
        access-control: fd89:59e0::/32 allow

        port: 53

        do-udp: yes
        do-tcp: yes
        do-ip4: yes
        do-ip6: yes

include: "/etc/unbound/unbound.conf.d/*.conf"
```

Once we finish editing the configuration file, we execute a command that allows us to quickly check if the configuration is semantically correct:

```
# unbound-checkconf
```

If it is correct, it will return something similar to the following:

```
unbound-checkconf: no errors in /etc/unbound/unbound.conf
```
```

Finally we restart the server so that it takes the configuration changes:

```
# systemctl restart unbound
```

And we check the status of the UNBOUND process:

```
# systemctl status unbound
```

We should obtain an output similar to the following:

```
● unbound.service - Unbound DNS server
Loaded: loaded (/lib/systemd/system/unbound.service; enabled; vendor preset: enabled)
Drop-In: /etc/systemd/system/service.d
        └─lxc.conf     Active: active (running) since Thu 2021-05-13 03:49:11 UTC; 13s ago       Docs: man:unbound(8)
        Process: 571 ExecStartPre=/usr/lib/unbound/package-helper chroot_setup (code=exited, status=0/SUCCESS)
        Process: 574 ExecStartPre=/usr/lib/unbound/package-helper root_trust_anchor_update (code=exited, status=0/SUCCESS)   Main PID: 578 (unbound)      Tasks: 1 (limit: 152822)     Memory: 7.8M
        CGroup: /system.slice/unbound.service
        └─578 /usr/sbin/unbound -d
May 13 03:49:10 resolv2.grpX.<lab_domain>.te-labs.training unbound[178]: [178:0] info: [25%]=0 median[50%]=0 [75%]=0
May 13 03:49:10 resolv2.grpX.<lab_domain>.te-labs.training unbound[178]: [178:0] info: lower(secs) upper(secs) recursions
May 13 03:49:10 resolv2.grpX.<lab_domain>.te-labs.training unbound[178]: [178:0] info:    0.000000    0.000001 1
May 13 03:49:11 resolv2.grpX.<lab_domain>.te-labs.training package-helper[577]: /var/lib/unbound/root.key has content
May 13 03:49:11 resolv2.grpX.<lab_domain>.te-labs.training package-helper[577]: success: the anchor is ok
May 13 03:49:11 resolv2.grpX.<lab_domain>.te-labs.training unbound[578]: [578:0] notice: init module 0: subnet
May 13 03:49:11 resolv2.grpX.<lab_domain>.te-labs.training unbound[578]: [578:0] notice: init module 1: validator
May 13 03:49:11 resolv2.grpX.<lab_domain>.te-labs.training unbound[578]: [578:0] notice: init module 2: iterator
May 13 03:49:11 resolv2.grpX.<lab_domain>.te-labs.training unbound[578]: [578:0] info: start of service (unbound 1.9.4).
May 13 03:49:11 resolv2.grpX.<lab_domain>.te-labs.training systemd[1]: Started Unbound DNS server.
```

### Test your new recursive resolver

Run the following commands and confirm if you receive the "ad" flag:

1. dig SOA com. @100.100.X.68
2. dig NS com. @100.100.X.68
3. dig A www.icann.org @100.100.X.68
4. dig NS icann.org @100.100.X.68
5. dig NS *grpX*.<*lab_domain*>.te-labs.training @100.100.X.68
6. dig SOA *grpX*.<*lab_domain*>.te-labs.training @100.100.X.68
7. dig SOA *grpX*.<*lab_domain*>.te-labs.training @100.100.X.130

### Set your recursive resolver OS to use your local validating recursive resolver software.
Still in **grpX-resolv2** server, edit the **/etc/resolv.conf** config file and replace whatever is there by the following only:

```
nameserver 100.100.X.68
```

Save and exit. Then try the following queries:

1. dig SOA com.
2. dig NS com.
3. dig A www.icann.org
4. dig NS icann.org
5. dig NS *grpX*.<*lab_domain*>.te-labs.training
6. dig SOA *grpX*.<*lab_domain*>.te-labs.training

### Set your lab servers to use the new recursive resolvers.
Now that you have two recursive resolvers that are working well, you can configure all the machines in your group to use them for DNS resolution. Connect to each machine and update the **/etc/resolv.conf** file by the following:

```
nameserver 100.100.X.67
nameserver 100.100.X.68
nameserver 9.9.9.9      # a backup resolver from a different network
```