Recursive Bind DNS server installation step by step

Step 1: Package updating

First, we will log in to the relevant servers and ensure all packages are up to date on Debian. Follow these steps to update the software packages:

Download the latest package information from the sources:

$ sudo apt-get update -y

Upgrade the outdated packages:

$ sudo apt-get upgrade -y

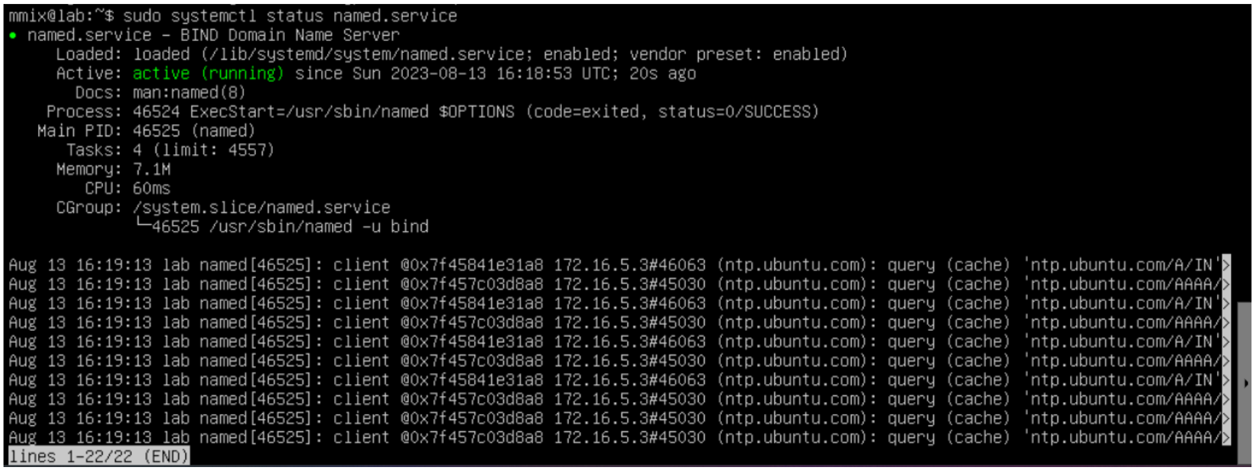
Step 2: installation Bind Daemon.

Next, we will install three Bind9 daemon packages on the DNS server. Run the following command:

$ sudo apt install bind9 bind9-dnsutils bind9-doc -y

Once the installation is complete, check the service status with this command:

$ sudo systemctl status named.service



```
mmix@lab:~$ sudo systemctl status named.service
● named.service - BIND Domain Name Server
     Loaded: loaded (/lib/systemd/system/named.service; enabled; vendor preset: enabled)
     Active: active (running) since Sun 2023-08-13 16:18:53 UTC; 20s ago
       Docs: man:named(8)
    Process: 46524 ExecStart=/usr/sbin/named $OPTIONS (code=exited, status=0/SUCCESS)
   Main PID: 46525 (named)
      Tasks: 4 (limit: 4557)
     Memory: 7.1M
        CPU: 60ms
     CGroup: /system.slice/named.service
             └─46525 /usr/sbin/named -u bind

Aug 13 16:19:13 lab named[46525]: client @0x7f45841e31a8 172.16.5.3#46063 (ntp.ubuntu.com): query (cache) 'ntp.ubuntu.com/A/IN'>
Aug 13 16:19:13 lab named[46525]: client @0x7f457c03d8a8 172.16.5.3#45030 (ntp.ubuntu.com): query (cache) 'ntp.ubuntu.com/AAAA/>
Aug 13 16:19:13 lab named[46525]: client @0x7f45841e31a8 172.16.5.3#46063 (ntp.ubuntu.com): query (cache) 'ntp.ubuntu.com/A/IN'>
Aug 13 16:19:13 lab named[46525]: client @0x7f457c03d8a8 172.16.5.3#45030 (ntp.ubuntu.com): query (cache) 'ntp.ubuntu.com/AAAA/>
Aug 13 16:19:13 lab named[46525]: client @0x7f45841e31a8 172.16.5.3#46063 (ntp.ubuntu.com): query (cache) 'ntp.ubuntu.com/A/IN'>
Aug 13 16:19:13 lab named[46525]: client @0x7f457c03d8a8 172.16.5.3#45030 (ntp.ubuntu.com): query (cache) 'ntp.ubuntu.com/AAAA/>
Aug 13 16:19:13 lab named[46525]: client @0x7f45841e31a8 172.16.5.3#46063 (ntp.ubuntu.com): query (cache) 'ntp.ubuntu.com/A/IN'>
Aug 13 16:19:13 lab named[46525]: client @0x7f457c03d8a8 172.16.5.3#45030 (ntp.ubuntu.com): query (cache) 'ntp.ubuntu.com/AAAA/>
Aug 13 16:19:13 lab named[46525]: client @0x7f457c03d8a8 172.16.5.3#45030 (ntp.ubuntu.com): query (cache) 'ntp.ubuntu.com/AAAA/>
Aug 13 16:19:13 lab named[46525]: client @0x7f457c03d8a8 172.16.5.3#45030 (ntp.ubuntu.com): query (cache) 'ntp.ubuntu.com/AAAA/>
lines 1-22/22 (END)
```

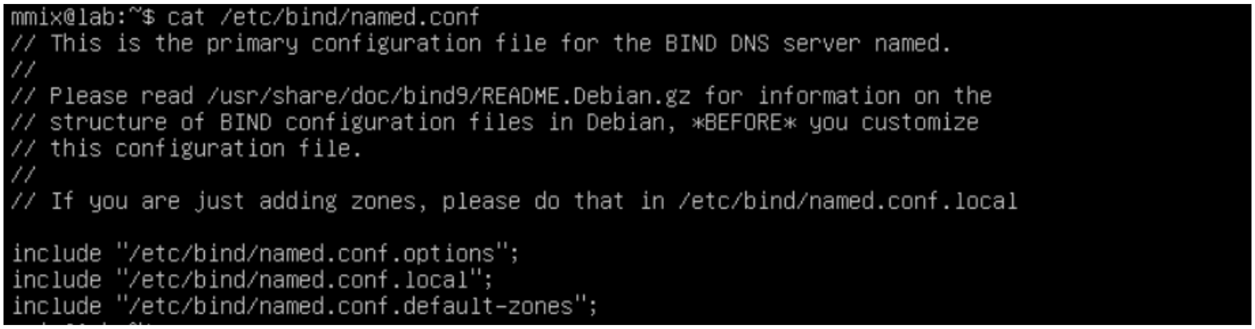
Optionally, you can enable or disable the Bind service with the following commands:

$ sudo systemctl enable/disable named.service

Step 3: Begin configuration.

To start configuring Bind, view the configuration file using:

$ cat /etc/bind/named.conf



```
mmix@lab:~$ cat /etc/bind/named.conf
// This is the primary configuration file for the BIND DNS server named.
//
// Please read /usr/share/doc/bind9/README.Debian.gz for information on the
// structure of BIND configuration files in Debian, *BEFORE* you customize
// this configuration file.
//
// If you are just adding zones, please do that in /etc/bind/named.conf.local

include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
```

Configure the /etc/bind/named.config.options file.

To configure the `/etc/bind/named.conf.options` file for your DNS setup using the `172.16.255.0/24` network as your client network and `172.16.1.2` as your server's listening IP, follow these steps:

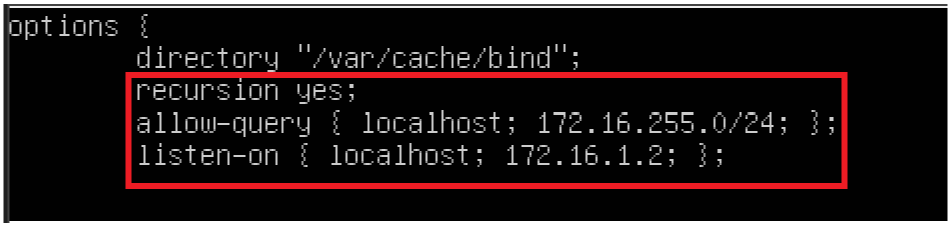
Open the `/etc/bind/named.conf.options` file in a text editor:

$ sudo nano /etc/bind/named.conf.options

Add or modify the following lines within the **`options`** block:

recursion yes;
allow-query { localhost; 172.16.255.0/24; };
listen-on { localhost; 172.16.1.2; };

Save and close the file.



```
options {
        directory "/var/cache/bind";
        recursion yes;
        allow-query { localhost; 172.16.255.0/24; };
        listen-on { localhost; 172.16.1.2; };
```

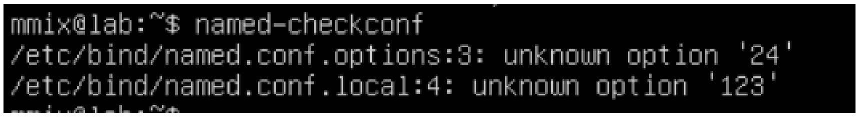
After configuring, you can save your changes by pressing `Ctrl + O`, then exit the text editor by pressing `Ctrl + X`. Once done, verify if the configuration file is working correctly by running the following command:

This configuration enables recursion, allows DNS queries from `localhost` and the `172.16.255.0/24` client network, and configures the server to listen on `localhost` and `172.16.1.2`.

$ named-checkconf

If your configuration file is correct, the `named-checkconf` command will not display any output, indicating that there are no errors. However, if there are issues, you'll see an error message.

It may look something like this:



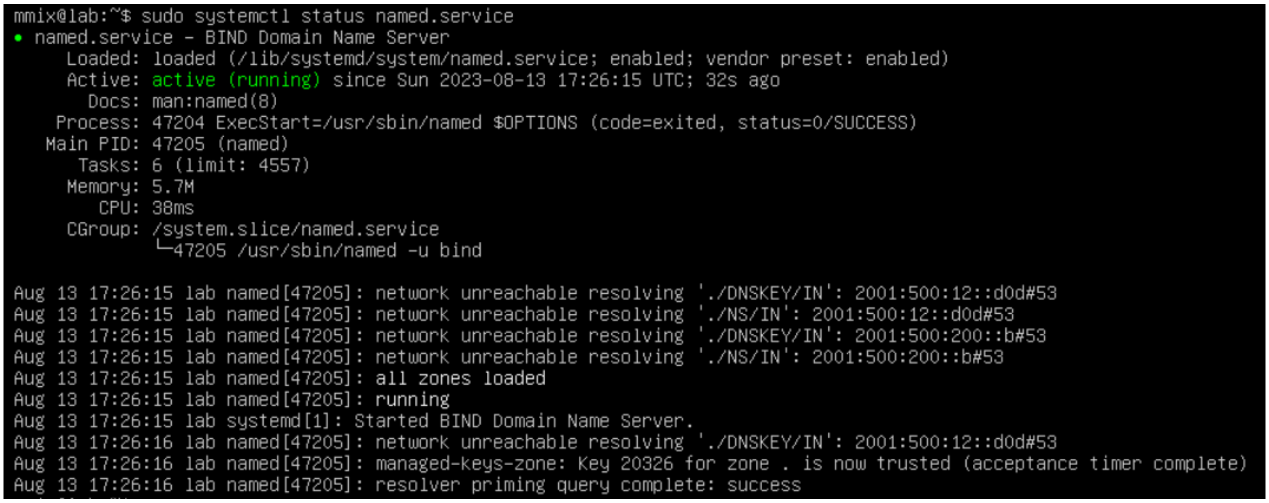
```
mmix@lab:~$ named-checkconf
/etc/bind/named.conf.options:3: unknown option '24'
/etc/bind/named.conf.local:4: unknown option '123'
```

(Error messages can vary depending on the issue.)

In case of errors, check the specific line mentioned and correct the configuration accordingly. After resolving the issue, re-run the check and restart the Bind service.

If everything works correctly, restart the Bind daemon service and check its status with the following commands:

$ sudo systemctl restart named.service
$ sudo systemctl status named.service



```
mmix@lab:~$ sudo systemctl status named.service
● named.service - BIND Domain Name Server
     Loaded: loaded (/lib/systemd/system/named.service; enabled; vendor preset: enabled)
     Active: active (running) since Sun 2023-08-13 17:26:15 UTC; 32s ago
       Docs: man:named(8)
    Process: 47204 ExecStart=/usr/sbin/named $OPTIONS (code=exited, status=0/SUCCESS)
   Main PID: 47205 (named)
      Tasks: 6 (limit: 4557)
     Memory: 5.7M
        CPU: 38ms
     CGroup: /system.slice/named.service
             └─47205 /usr/sbin/named -u bind

Aug 13 17:26:15 lab named[47205]: network unreachable resolving './DNSKEY/IN': 2001:500:12::d0d#53
Aug 13 17:26:15 lab named[47205]: network unreachable resolving './NS/IN': 2001:500:12::d0d#53
Aug 13 17:26:15 lab named[47205]: network unreachable resolving './DNSKEY/IN': 2001:500:200::b#53
Aug 13 17:26:15 lab named[47205]: network unreachable resolving './NS/IN': 2001:500:200::b#53
Aug 13 17:26:15 lab named[47205]: all zones loaded
Aug 13 17:26:15 lab named[47205]: running
Aug 13 17:26:15 lab systemd[1]: Started BIND Domain Name Server.
Aug 13 17:26:16 lab named[47205]: network unreachable resolving './DNSKEY/IN': 2001:500:12::d0d#53
Aug 13 17:26:16 lab named[47205]: managed-keys-zone: Key 20326 for zone . is now trusted (acceptance timer complete)
Aug 13 17:26:16 lab named[47205]: resolver priming query complete: success
```

If the service is active and running, the configuration has been applied successfully.

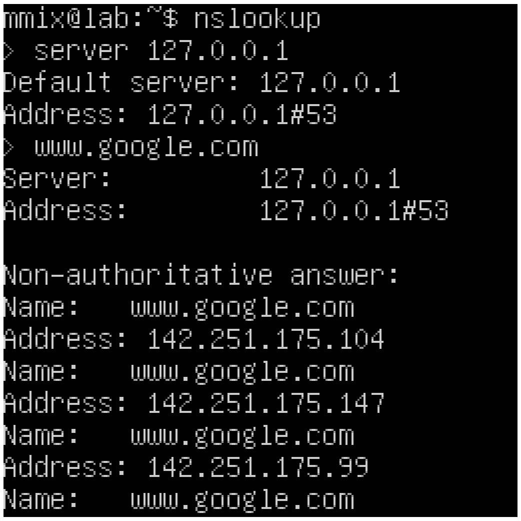
If everything is set up correctly, you can now assign the DNS server's IP address and begin resolving queries.

Since you are using a recursive DNS server, you can perform DNS resolution using the server's localhost IP, `127.0.0.1`.

Here’s an example of how to resolve a query using the DNS server:

$ nslookup

- **Server 127.0.0.1**
- **www.google.com**



```
mmix@lab:~$ nslookup
> server 127.0.0.1
Default server: 127.0.0.1
Address: 127.0.0.1#53
> www.google.com
Server:         127.0.0.1
Address:        127.0.0.1#53

Non-authoritative answer:
Name:   www.google.com
Address: 142.251.175.104
Name:   www.google.com
Address: 142.251.175.147
Name:   www.google.com
Address: 142.251.175.99
Name:   www.google.com
```

The output will show details like the resolved IP address, query time, and other DNS-related information, confirming that your recursive DNS server is working as expected.

Now, you've successfully finished the recursive DNS server installation!

Step 4: Enable Logging

Whenever you modify any configuration file, it's a good practice to check the system logs to monitor the current state and catch any potential issues. Generally, you can view the logs in the syslog to see what the system is reporting.

To check the syslog, use the following command:

$ tail -f /var/log/syslog

This command will display the latest system events and continuously update in real-time, allowing you to monitor the system as changes are made.

You can also filter for specific messages related to Bind by using `grep`:

$ sudo grep named /var/log/syslog

This will show you all the log entries related to the Bind service. Monitoring logs ensures that your configuration changes are applied correctly and helps you troubleshoot any issues that may arise.

To define a specific log for the BIND DNS server, follow these steps:

1. Create new directly/file for bind DNS query log.
   ```
   $ sudo mkdir /var/log/named
   ```
2. Change ownership of the directory to the BIND user:
   ```
   $ sudo chown bind:bind /var/log/named
   ```

Add logging configuration to the `named.conf.options` file:

Go to the `/etc/bind/` directory and open the `named.conf.options` file in a text editor:

```
        $ sudo nano /etc/bind/named.conf.options
```

```
logging{
     channel query_logging {
          file "/var/log/named/query.log" versions 3 size 10m;
          severity debug 3;
          print-time yes;
          print-severity yes;
          print-category yes; };
     category queries {
          query_logging; };
};
```

Save the file and exit the text editor.

```
      $ cat named.conf.options
```

After configuring the logging for BIND, follow these steps to ensure everything is working correctly:

```
     $ named-checkconf
```

If there are no errors, proceed to the next steps.

Restart the BIND service to apply the changes:

```
     $ sudo systemctl restart named.service
     $ sudo systemctl status named.service
```

Monitor the BIND query log to verify logging is working as expected:

```
$ tail -f /var/log/named/query.log
```

This will display the latest entries in the query log and update in real-time, allowing you to verify that the logging is functioning properly.

Step 5: Anycast Routing

In this Lab, we will use **10.10.10.10 & 10.10.10.11** IP as an anycast IP addresses.

Configuration Virtual Interface

To configure a virtual network interface for anycast IP addresses using a dummy interface, follow these steps:

Log in to your server and navigate to the systemd network directory:

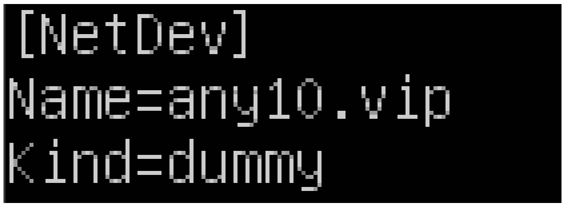
```
$ cd /etc/systemd/network/
```

Create and edit a new configuration file for the dummy interface:

```
$ sudo nano any10.netdev
```

Add the following configuration to the file:

```
[NetDev]
Name=any10.vip
Kind=dummy
```



```
[NetDev]
Name=any10.vip
Kind=dummy
```

This defines a virtual network interface named **`any10.vip`** of type **`dummy`**.

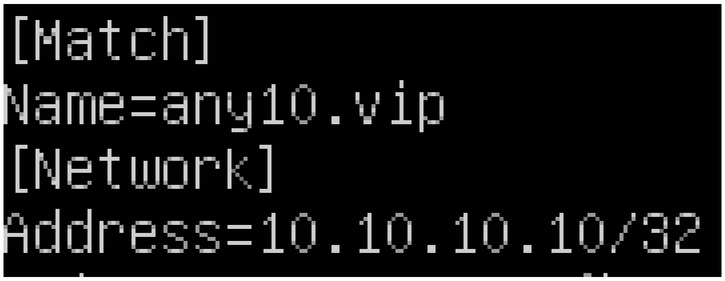
Save and exit the text editor **(**`Ctrl + O`**,** `Enter`**,** `Ctrl + X` **for nano)**

Next, configure the virtual interface with an IP address. Create or edit the corresponding network configuration file:

```
$ sudo nano any10.network
```

Add the following configuration to the file:

```
[Match]
Name=any10.vip
[Network]
Address=10.10.10.10/32
```



```
[Match]
Name=any10.vip
[Network]
Address=10.10.10.10/32
```

This assigns the anycast IP addresses to the `any10.vip` virtual interface.

Restart the **`systemd-networkd`** service to apply the changes:

```
$ sudo systemctl restart systemd-networkd
```

Verify the configuration and the virtual interface:

```
$ ip a show any10.vip
```

This will display the details of the **`any10.vip`** virtual interface and confirm that the IP addresses have been assigned correctly.

After, we need to configure another virtual interface for 10.10.10.11/32 network. Please repeat the above step.

To verify that your virtual network interface is configured correctly and the IP addresses are assigned, use the following command:

```
$ ip addr
```

To configure the BIND DNS server to listen for DNS queries on the anycast IP addresses, follow these steps:

Navigate to the BIND configuration directory:

```
$ cd /etc/bind/
```

Edit the `named.conf.options` file:

```
$ sudo nano named.conf.options
```

Modify the `listen-on` directive to include your anycast IP addresses:

```
listen-on { localhost; 10.10.10.10; 10.10.10.11; };
```

Save the file and exit the text editor.

Check the configuration for syntax errors:

```
$ named-checkconf
```

If there are no errors, restart the BIND service:

```
$ sudo systemctl restart named.service
```

Check the status of the BIND service to ensure it is running correctly:

```
$ sudo systemctl status named.service
```

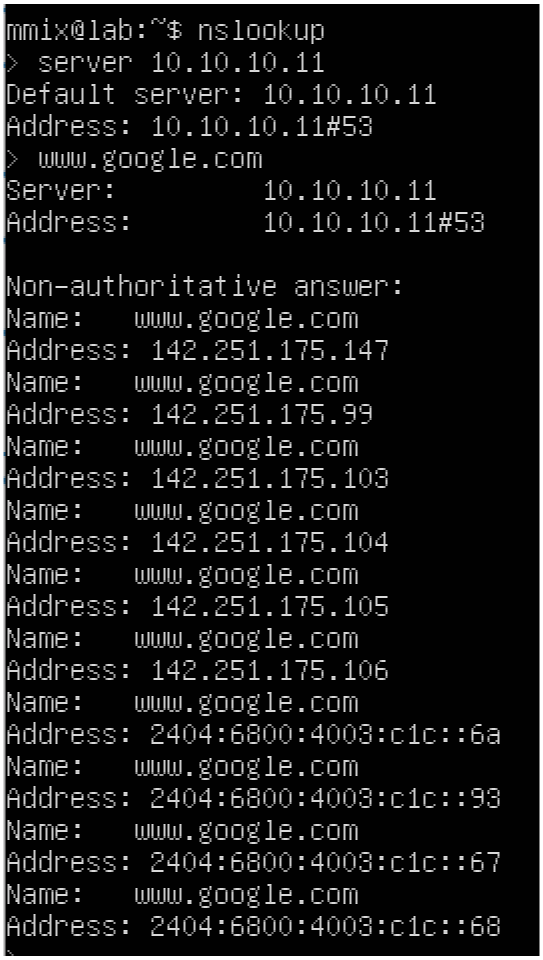
Test DNS query resolution using the anycast IP addresses:

Check the figure below for an example query resolve using anycast IP 10.10.10.11.

```
$ nslookup
```

▯ **Server 10.10.10.11**

▯ **www.google.com**



```
mmix@lab:~$ nslookup
> server 10.10.10.11
Default server: 10.10.10.11
Address: 10.10.10.11#53
> www.google.com
Server:         10.10.10.11
Address:        10.10.10.11#53

Non-authoritative answer:
Name:   www.google.com
Address: 142.251.175.147
Name:   www.google.com
Address: 142.251.175.99
Name:   www.google.com
Address: 142.251.175.103
Name:   www.google.com
Address: 142.251.175.104
Name:   www.google.com
Address: 142.251.175.105
Name:   www.google.com
Address: 142.251.175.106
Name:   www.google.com
Address: 2404:6800:4003:c1c::6a
Name:   www.google.com
Address: 2404:6800:4003:c1c::93
Name:   www.google.com
Address: 2404:6800:4003:c1c::67
Name:   www.google.com
Address: 2404:6800:4003:c1c::68
```

When you finish configuration on Server 1, Please repeat the above steps on Server 2.

Configuration Anycast IP reachable

To configure your router to ensure that the anycast IP addresses are reachable, follow these general steps. The exact commands may vary depending on your router's brand and model. Here’s a broad outline for configuring routes to reach the anycast IP addresses:

Log in to your router:

Access your router’s command line interface (CLI) or web interface. If using CLI, you may need to use SSH or a console connection.

Add static routes (if needed):

If your router requires specific static routes to reach the anycast IP addresses, configure them as follows:

On Cisco Routers (Example):

Router# configure terminal

Router(config)# ip route 10.10.10.10 255.255.255.255 <next-hop-IP>

Router(config)# ip route 10.10.10.11 255.255.255.255 <next-hop-IP>

Router(config)# end

Router# write memory

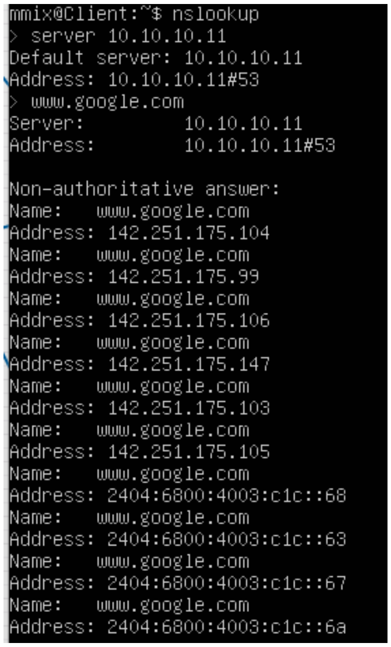
Now, both servers anycast IP is reachable. You can query from your client network.

To query from client machine as below

$ nslookup

- **Server 10.10.10.11**
- **www.google.com**

The output should appear as follows:



```
mmix@Client:~$ nslookup
> server 10.10.10.11
Default server: 10.10.10.11
Address: 10.10.10.11#53
> www.google.com
Server:         10.10.10.11
Address:        10.10.10.11#53

Non-authoritative answer:
Name:   www.google.com
Address: 142.251.175.104
Name:   www.google.com
Address: 142.251.175.99
Name:   www.google.com
Address: 142.251.175.106
Name:   www.google.com
Address: 142.251.175.147
Name:   www.google.com
Address: 142.251.175.103
Name:   www.google.com
Address: 142.251.175.105
Name:   www.google.com
Address: 2404:6800:4003:c1c::68
Name:   www.google.com
Address: 2404:6800:4003:c1c::63
Name:   www.google.com
Address: 2404:6800:4003:c1c::67
Name:   www.google.com
Address: 2404:6800:4003:c1c::6a
```

Step 6. DNS Server Outbound Query

To configure a specific IP address for DNS server outbound queries, follow these steps:

Create a Virtual Interface for the Public IP:

1. First, we need to set up a virtual network interface for the public IP address.

   "Please repeat the steps outlined above to create the virtual dummy interface."

To configure the DNS server for outbound queries, follow these steps:

1. Navigate to the BIND9 configuration directory:

   $ cd /etc/bind/

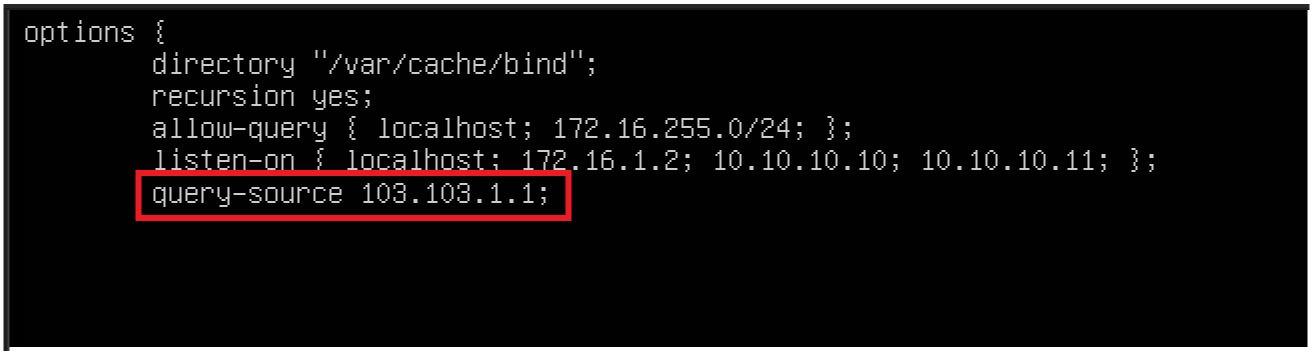
2. Edit the BIND9 options file:

   $ nano named.conf.options

   Add or modify the `query-source` directive to specify the desired outbound IP address:

   query-source 103.103.1.1;

The output should resemble the following:



```
options {
        directory "/var/cache/bind";
        recursion yes;
        allow-query { localhost; 172.16.255.0/24; };
        listen-on { localhost; 172.16.1.2; 10.10.10.10; 10.10.10.11; };
        query-source 103.103.1.1;
```

3. After configuring, verify that everything is functioning correctly:

   Check the BIND configuration for errors:

   $ named-checkconf

If no errors are found, restart the BIND service and check its status:

$ sudo systemctl restart named.service
$ sudo systemctl status named.service

Verify Operation

To verify the setup, we will send query messages from the client to the DNS server. First, monitor the BIND log file on the server to track these queries. Use the following command to view the log file;

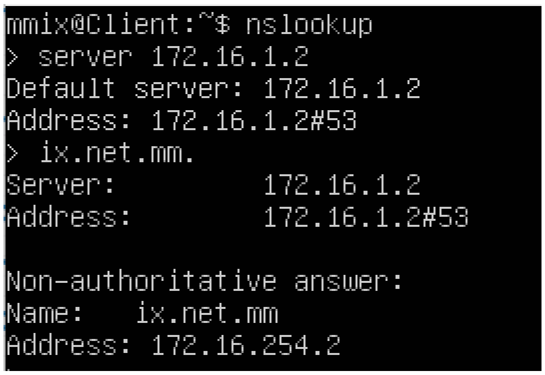
$ tail -f /var/log/named/query.log

Next, log in to your client machine and send query messages using the following command:

$ nslookup

- **Server 172.16.1.2**
- **Ix.net.mm.**

The output should look like this:

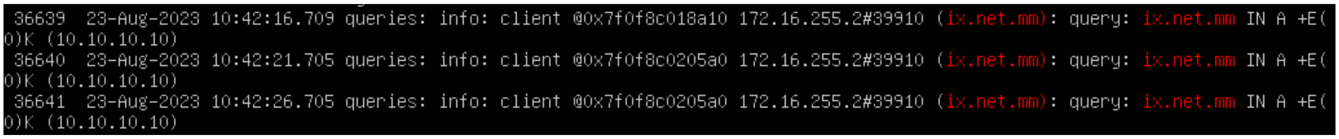


```
mmix@Client:~$ nslookup
> server 172.16.1.2
Default server: 172.16.1.2
Address: 172.16.1.2#53
> ix.net.mm.
Server:         172.16.1.2
Address:        172.16.1.2#53

Non-authoritative answer:
Name:   ix.net.mm
Address: 172.16.254.2
```

At the designated time, check the log messages on Server 1.

The output should appear as follows:



```
 36639  23-Aug-2023 10:42:16.709 queries: info: client @0x7f0f8c018a10 172.16.255.2#39910 (ix.net.mm): query: ix.net.mm IN A +E(
0)K (10.10.10.10)
 36640  23-Aug-2023 10:42:21.705 queries: info: client @0x7f0f8c0205a0 172.16.255.2#39910 (ix.net.mm): query: ix.net.mm IN A +E(
0)K (10.10.10.10)
 36641  23-Aug-2023 10:42:26.705 queries: info: client @0x7f0f8c0205a0 172.16.255.2#39910 (ix.net.mm): query: ix.net.mm IN A +E(
0)K (10.10.10.10)
```

If your configuration is correct, the `ix.net.mm` record should resolve to `172.16.254.2`.

That's it! You've successfully completed the lab. Congratulations! If you have any more questions or need further assistance, feel free to ask.