2014

Linux Academy.com

Hands On Lab

Ubuntu and DNS/Bind Server

Table of Contents

Introduction

The Domain Name Service (DNS) is a public Internet service that maps one or more IP addresses and Fully Qualified Domain Names (FQDN) to each other (both forward – name to IP, and reverse – IP to name).

Servers that run the BIND (Berkley Internet Naming Daemon) service are commonly referred to as name servers.

Goals

This lab will introduce you to the concepts of DNS/BIND services from a client and server perspective. By the end of this document, you will have built an Ubuntu 14.04 LTS BIND Server and created a client configuration to test any created DNS entries for the domain you choose to set up.

Packages, Resources and Prerequisites

The packages involved in our lab are:

- bind9
- bind9-doc
- bind9utils
- apache2
- lynx-cur

The resources you will be accessing during the course of this lab are:

- Ubuntu 14.04 LTS Server: Apache and Test Client
    - You will use this to test your server and domain additions once completed
- Ubuntu 14.04 LTS Server: BIND9 Server
    - This is the server that you will deploy, install and configure BIND on, create your sample domain and add DNS entries to

Prerequisites to this lab:

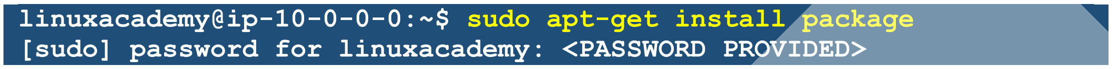
- A LinuxAcademy.com Lab+ Subscription
- Internet Access and SSH Client
    - You will need to connect to the public IP of the server in order to configure BIND, the only method of connectivity is over SSH
        - SSH client can be Windows (i.e. Putty) or from another Linux system shell

- Login Information (Provided When The Server Starts Up)

Document Conventions

Just a couple of housekeeping items to go over so you can get the most out of this Hands On Lab without worrying about how to interpret the contents of the document.

When we are demonstrating a command that you are going to type in your shell while connected to a server, you will see a box with a dark blue background and some text, like this:



```
linuxacademy@ip-10-0-0-0:~$ sudo apt-get install package
[sudo] password for linuxacademy: <PASSWORD PROVIDED>
```

That breaks down as follows:

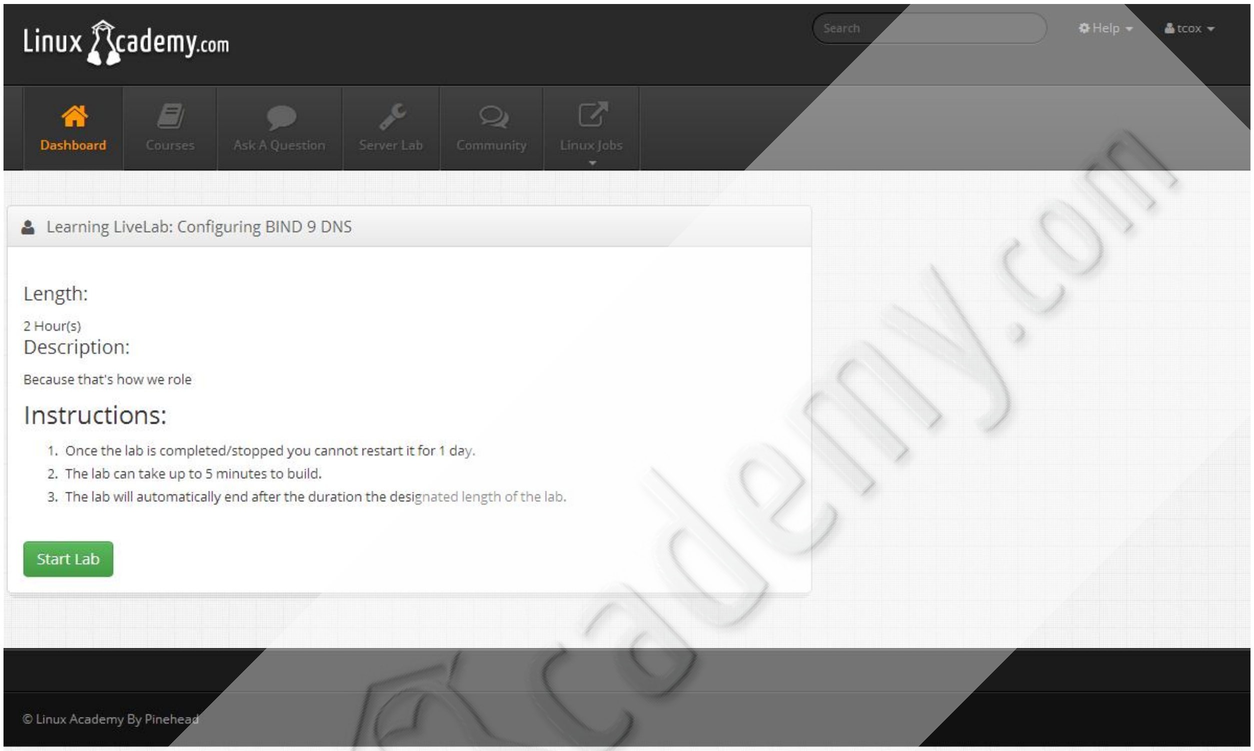
- The white text that looks something like "linuxacademy@ip-10-0-0-0:~$: "should be interpreted as the console prompt your cursor will be sitting at when you are logged into the server. You are not typing that into the console, it is just there. Note that the portion that says "ip-10-0-0-0" will be different for you based on the IP address of your system.
- The bold yellow text in any command example is the actual command you will type in your shell.
- Any lines subsequent to the bold yellow command that you type in is to be considered as the response you can expect from your command entry.

If you do not see the command prompt as the first line in the example, then all the white text is an example of a text, script or configuration file and is intended to be typed in its entirety (in the event you are instructed to create a previously non-existent file) or to compare against the contents of the file on your system.

One final convention, if you see a "~" at the end of a line in a command or text example, that will indicate that the line overflowed the end of line. Meaning you should just keep typing without hitting enter to start a new line until the natural end of the command or entry.

General Process

When you are ready to begin the Hands On Lab, log into your Linux Academy Lab+ subscription and navigate to the "Live Labs" section on the Dashboard. Once you choose the "Configuring BIND 9 DNS" Lab from the list, you will see the screen below:



Learning LiveLab: Configuring BIND 9 DNS

Length:

2 Hour(s)

Description:

Because that's how we role

Instructions:

1. Once the lab is completed/stopped you cannot restart it for 1 day.
2. The lab can take up to 5 minutes to build.
3. The lab will automatically end after the duration the designated length of the lab.

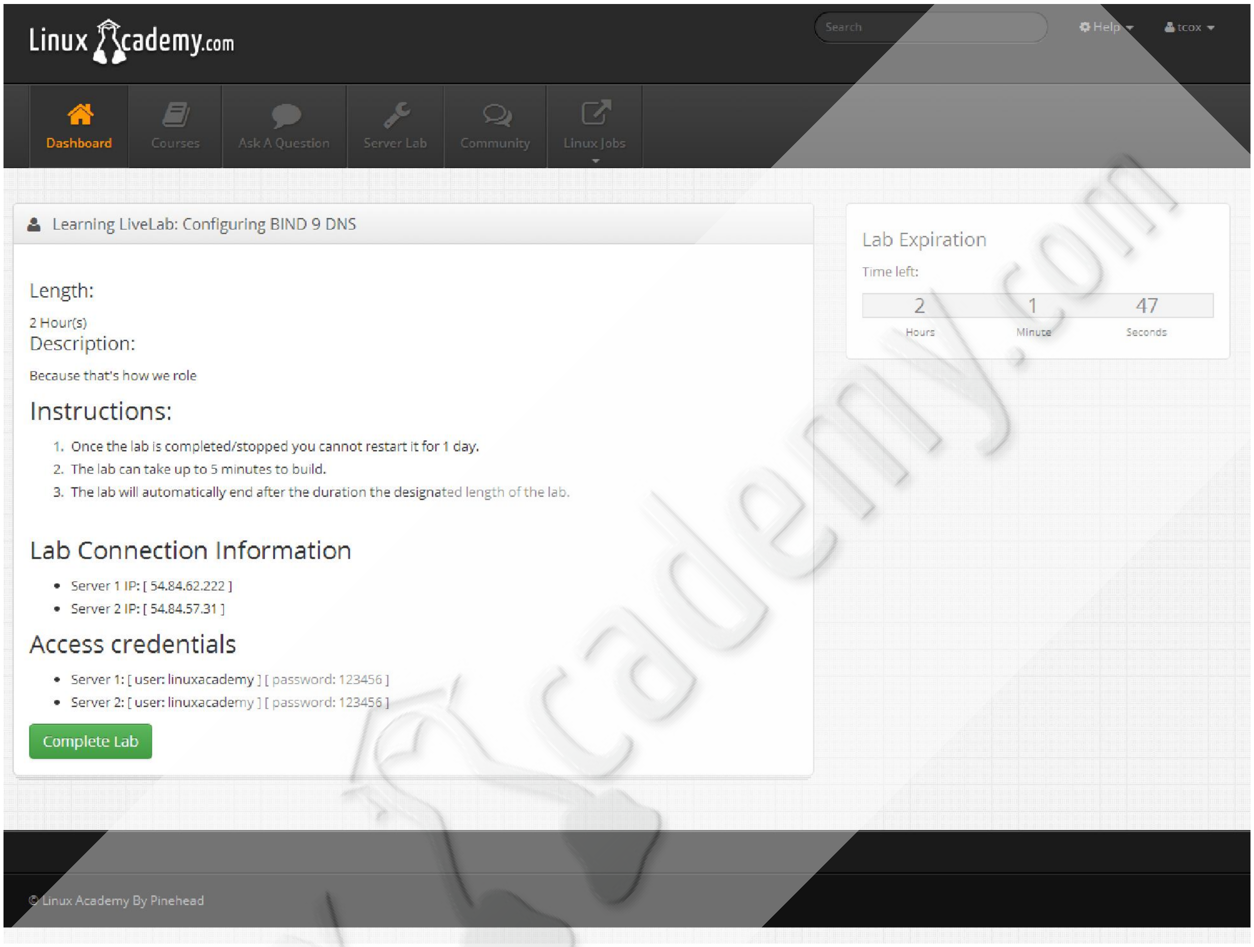
[Linux Academy Live Labs+ - Start Screen]

A few things to note before you start this process:

- When you launch the lab from this screen, it may take up to FIVE MINUTES for your servers to be deployed and be available for your use.
- Do not leave your desktop and come back, once the servers are launched, you will only have TWO HOURS to complete this lab from start to finish. After that point, the servers time out and are deleted permanently. Any and all work that you have done will then be lost.
- You can only use this lab ONCE PER DAY. If you try to use it more than that after completing the lab or the servers timing out, the screen will tell you when it will be available to you again.

- Other than those descriptions, you may retry any of the Labs+ labs as many times as you wish as long as you are a subscriber.

Once you have clicked on the 'Start Lab' button that you see above, a process will launch on our servers that will deploy the two servers we will use in our lab for testing. After a few minutes of processing (and you will see a status message that says "Creating Lab… Please Wait"), you should see a screen that looks like this one:



Learning LiveLab: Configuring BIND 9 DNS

Length:

2 Hour(s)

Description:

Because that's how we role

Instructions:

1. Once the lab is completed/stopped you cannot restart it for 1 day.
2. The lab can take up to 5 minutes to build.
3. The lab will automatically end after the duration the designated length of the lab.

Lab Connection Information

- Server 1 IP: [54.84.62.222]
- Server 2 IP: [54.84.57.31]

Access credentials

- Server 1: [user: linuxacademy] [password: 123456]
- Server 2: [user: linuxacademy] [password: 123456]

Complete Lab

Lab Expiration

Time left:

2	1	47
Hours	Minute	Seconds

© Linux Academy By Pinehead

[Linux Academy Live Labs+ - Start Screen]

You will see all the information you need to access your servers from another system. Specifically, you need:

- The two server public IP addresses
- Access credentials for both

One thing to note is that, in addition to the two IPs that you see above, each server will have another IP assigned to it in the 10.0.0.x subnet. This is a private IP address and will not route outside of your private server pool. Server 1 will have a dynamic private address in the 10.0.0.x subnet while Server 2 will have a static private address of 10.0.0.100.

In our setup, Server 1 will function as an Apache server that we will use to test our DNS server. Server 2 will function as our primary BIND server for the purposes of this lab. Let's spend a quick couple of minutes setting up our Apache server.

Apache Server Setup

We are going to configure a VERY basic Apache web server, create a basic HTML file that we can use to verify that it is running and install a text based browser that we can use for our testing later.

Start by installing the Apache 2.x web server (and subsequently starting it) by performing the following commands:

```
linuxacademy@ip-10-0-0-162:~$ sudo apt-get install apache2
[sudo] password for linuxacademy: <PASSWORD PROVIDED>
```

This will install a number of packages and modules in addition to the Apache server. It will only take a moment or two to complete and once it is done, Ubuntu will automatically launch the Apache server on the default port 80. You can verify that it is listening by executing a local telnet connection on port 80 and you should see the following:

```
linuxacademy@ip-10-0-0-162:~$ telnet localhost 80
Trying 127.0.0.1…
Connected to localhost.
Escape character is `^]`.
```

Pressing the escape character of ']' will return an HTML message to you and close the connection, but that verifies that our server is answering on port 80. We are now going to create the most basic of HTML pages so that we can test our web server by name once the DNS server is up and running. Change to the /var/www/ directory and create the following file called "test.txt" in the /var/www directory (you can copy and paste):

```
<HTML>

<BODY>
   www.linuxacademy.lab
</BODY>

</HTML>
```

This file will simply display in our text based browser the name we will be assigning to this Apache server in DNS. We will use the Lynx text browser in our console to do so after we set up our BIND server.

Our local Apache server is now completed for now. We will be coming back to reconfigure the DNS client once we have our BIND server ready so that we get resolution for the IPs that we created when adding to our customized, but private, domain name.

BIND9 Configuration Types

Although this lab focuses on the most common BIND server configuration (primary master DNS server), there are actually several different types of configurations available. They are:

- Primary Master Server
    - This is the server we will be setting up today. This will serve one or more DNS records (as a group, generally referred to as a Zone), for a registered domain or a private domain (in our case, it will be a private domain setup for the sake of our examples).
- Secondary Master Server
    - This DNS server is used in case the primary server is unavailable. It contains a replicated copy of the Primary Server's zone records and is used generally in larger setups where redundancy is key for multiple domains.
- Caching Name Server
    - This configuration simply looks up requests for names and remembers (caches) the answer for the next query. If you have a very slow internet connection, this can cut down on the latency and reducing bandwidth requirements for your network.
- Hybrid
    - You can combine Caching Name Server and Primary or Secondary Master Servers together to create a hybrid environment for your needs.

General BIND9 Notes

Keep in mind that there are certain things you will want to have set up or known in advance of configuring your BIND9 server:

- Static IP Address
    - You will want to be sure you have a static IP address on the server to bind your DNS service to. This address can be a private address (as in our case) if you are only going to be serving private domain records and addresses.
- Forwarding
    - If your server is going to be the only DNS server your clients have direct access to, be sure you have configured the server itself as a client to an acceptable public DNS server so that forwarding will work. That means if a client asks for the

domain “google.com” which your DNS server does not own the zone for, it will forward that request to the public DNS server configured and respond to the client with the appropriate information (this is how we will be doing our lab as you will see).

- Local Host Entry
  - Be sure your DNS server has a local host entry for the real hostname (example in the next section).

BIND9 Server – Preparation

The most important thing to remember about your BIND9 server to be is that it MUST have at least one static IP address that we can bind the zones we will host to. Since those configuration files are tied by name to one of the interfaces and the domain itself must be assigned an IP within them, we have to have an unchangeable IP we can use. You may have any number of network interfaces and associated IP addresses as long as at least one is static.

Next, be sure that the hostname (short and fully qualified) exist as an entry in your local hosts file. Using our setup, the private server IP of 10.0.0.100 is going to be our BIND9 server. On that server, make sure your local /etc/hosts file looks something like this:

```
127.0.0.1            localhost

# The following lines are desirable for IPv6 capable hosts
::1                  ip6-localhost ip6-loopback
fe00::0              ip6-localnet

10.0.0.100           ip-10-0-0-100.linuxacademy.lab ip-10-0-0-100
```

The IPv6 information does not have to be there for our purposes since we will not be binding or using IPv6. The line with our local server IP is setting our host file appropriately using the short name and a fully qualified domain name. Since we will be using the private domain ‘linuxacademy.lab’ for the purposes of our exercises, that is what we list for the FQDN.

Once that is complete, you can test by pinging the short and FQDN of your local host and be sure you get a response with the appropriate IP. If you get any errors regarding “Unknown Host”, be sure your IP, hostname and FQDN are saved properly in the /etc/hosts file as listed above.

BIND9 Server – Installation

Strictly in terms of pure installation steps, there really is only one. You can have everything you need pulled down and installed from the standard Ubuntu repositories by executing the following command:

```
linuxacademy@ip-10-0-0-100:~$ sudo apt-get install bind9 bind9utils
```

We you can also install the BIND9 documentation by including the “bind9-doc” package, but we are trying to get this done within our two hour window so I am skipping it here. These packages may also install some dependencies depending on the state of your system when you run the command but when you are done, you will have a skeleton BIND9 server setup with the BIND9 service running once it is complete. Now, we need to set up our configuration.

BIND9 Server – Service Configuration

Our first configuration task will be to make some changes to our name service configuration files. These files can be found in the /etc/bind directory and they are empty templates you can use to set up your name server.

Let’s edit our local configuration file called “/etc/bind/named.conf.options”. This file will allow us to configure a forwarding DNS server so that requests that come in for names we do not host the zone for, we can forward the appropriate response back to the requesting client. For the purposes of our lab, the file should look like this:

```
forwarders {
# Your ISP DNS IP(s) Here
10.0.0.2;  # Amazon EC2 DNS Server for our network
8.8.8.8;   # Google Public DNS
};
```

One of the first things you will notice you need to be careful of is punctuation in all of our configuration files. Be sure that if you see a semicolon (;) or a period (.) at the end of any line or name, you duplicate that. These are required for the BIND9 service to work correctly.

Next up, we need to edit another configuration file called “/etc/bind/named.conf.local”. This particular configuration file will contain the base zone or zones that your system will be responsible for replying for. You may have one or more zones, and we add them forward and reverse like so:

```
# our forward zone and domain name
zone “linuxacademy.lab” {
   type master;
   file “/etc/bind/zones/db.linuxacademy.lab”;
};

# our reverse zone and server info
# DNS Server IP: 10.0.0.100 (Just for easy reference)
zone “0.0.10.in-addr.arpa” {
   type master;
   file “/etc/bind/zones/db.10”;
};
```

Once these files are saved, our BIND9 service is configured but we do not have the forward and reverse zone files referred to within the configuration set up. As an aside, remember that forward and reverse lookups are name to IP and IP to name. Many times they are required when accessing a service, sending an email from one server to another, etc. in order for the receiving server to be sure that the traffic is coming from the expected party rather than from a spoofed connection or “man in the middle” attack.

Let’s take a look at our zone configuration files next.

BIND9 Server – Zone Configuration

We have installed our server applications, configured the local host entries and picked a domain name/zone to host with our server and added the appropriate references in the BIND9 configuration files.

In the previous section, we referenced two files that we need to now create, our forward and reverse zone files. Fortunately, BIND9 services install some templates for us to use. Create a subdirectory in /etc/bind called “zones” for our use:

```
linuxacademy@ip-10-0-0-100:~$ cd /etc/bind
linuxacademy@ip-10-0-0-100:/etc/bind$ sudo mkdir zones
linuxacademy@ip-10-0-0-100:/etc/bind$ cd zones
```

Now we need to copy a template file for our use, first up, we will create the forward lookup zone as follows:

```
linuxacademy@ip-10-0-0-100:/etc/bind/zones$ sudo cp ../db.local   ~
db.linuxacademy.lab
```

Here is a sample of what that file should be edited to look like once complete (again, this can be repeated exactly for the purposes of this lab, however, be sure to substitute the IP address of your Apache server for the one listed below used as an example):

```
;
; BIND data file for local loopback interface
;
$TTL   604800
@      IN     SOA    ip-10-0-0-100.linuxacademy.lab. ~
admin.linuxacademy.lab. (
                        3            ; Serial
                   604800            ; Refresh
                    86400            ; Retry
                  2419200            ; Expire
                   604800 )    ; Negative Cache TTL
;
linuxacademy.lab. IN     NS     ip-10-0-0-100.linuxacademy.lab.
linuxacademy.lab. IN     A      10.0.0.100
ip-10-0-0-100     IN     A      10.0.0.100
ip-10-0-0-162     IN     A      10.0.0.162
```

```
www               IN    A      10.0.0.162
```

This file does a number of things. The most important one is that it sets up the entire zone and binds it to the host name and IP address of the server. The zone is defined to be bound SOA (Start of Authority) to the hostname "ip-10-0-0-100.linuxacademy.lab.", please note the trailing period as we warned of previously. This is one of the most common configuration errors when setting up DNS. The second value of "admin.linuxacademy.lab." on the same line is simply the user on the domain that would receive email from the BIND9 service, you can define this to be any user on the system.

Inside the configuration brackets are some values, only one of which will you normally change. The Serial number ('3' in this case) is a number that must change to a different number every time you add or remove a DNS entry from this file or the reverse file. Additionally, the Serial number in each of the forward and reverse zone files must match exactly or the BIND9 service will fail to start.

The other values are various timeouts, in milliseconds, for your server to take and respond to requests from a client. You can change them as needed (be careful you are not too aggressive or you could cause your DNS request to fail to secondary DNS servers too quickly).

Underneath that configuration are the names and IPs we have to worry about. Our first entry of "linuxacademy.lab. IN NS ip-10-0-0-100.linuxacademy.lab." binds the "linuxacademy.lab." zone to the FQDN of our server (the reason we added that FQDN to the /etc/hosts file earlier). The second line adds a binding for the domain name to our BIND9 server IP address. Keep in mind we are using the 10.0.0.x network as a private network hosting a private domain. If this were a public server and a public domain you would see appropriate valid IPs and domains.

We then are adding the host name of our two lab servers to the forward zone and a second host name called "www" to our Apache server from earlier. Although we are adding this as a full DNS record, we could just as easily added "www" as a "CNAME" (a reference) to the host name of the Apache server itself rather than the IP.

Let's create our reverse zone file like so:

```
linuxacademy@ip-10-0-0-100:/etc/bind/zones$ sudo cp ../db.127 db.10
```

Here is the sample reverse zone configuration that matches our forward zone file:

```
;
; BIND reverse data file for local loopback interface
;
$TTL  604800
@     IN    SOA   ip-10-0-0-100.linuxacademy.lab.
admin.linuxacademy.lab. (
                        3           ; Serial
                   604800           ; Refresh
```

```
                      86400             ; Retry
                    2419200             ; Expire
                     604800 )     ; Negative Cache TTL
;
            IN     NS     ip-10-0-0-100.

100         IN     PTR    ip-10-0-0-100.linuxacademy.lab.
162         IN     PTR    ip-10-0-0-162.linuxacademy.lab.
162         IN     PTR    www.linuxacademy.lab.
```

You will notice that the values here match the values in the forward zone configuration (binding host name, email of the user defined, Serial, timeouts, etc). The difference being that this zone is configured, strangely enough, in the reverse order of the forward zone. Whereas the forward zone defines names and assigns them IP values, the reverse zone defines addresses in our private network and assigned them names.

This is needed for all kinds of network services and security utilities to work and communicate correctly.

Configuration Check

At this point, our installation and configuration of BIND9 should be done and all we need to do is restart the service. However, let's take a moment and check our zone files before we restart the service.

In order to check our forward lookup zone, you can execute the following command:

```
linuxacademy@ip-10-0-0-100:~$ sudo named-checkzone linuxacademy.lab
/etc/bind/zones/db.linuxacademy.lab
zone linuxacademy.lab /IN: loaded serial 3
Ok
```

This verifies that all of the configuration files are formatted correctly for completing a successful DNS lookup and returning the IP of a server name defined in our zone called "linuxacademy.lab".

Next, we will do the same thing to check our reverse lookup zone:

```
linuxacademy@ip-10-0-0-100:~$ sudo named-checkzone linuxacademy.lab
/etc/bind/zones/db.10
zone linuxacademy.lab /IN: loaded serial 3
Ok
```

Keep in mind that the line in both checks that ends with "serial 3" could be different in your scenario depending on the serial number you entered when editing your configuration files earlier. If you have made any mistakes in the configuration file, this utility is kind enough to tell you that there is an error on line number XXX. Compare your configuration file with our examples to resolve.

Now we need to edit our DNS resolution configuration file on the server. Your /etc/resolv.conf file should look something like this:

```
nameserver 10.0.0.100    # our nameserver
nameserver 10.0.0.2      # AWS nameserver
search     linuxacademy.lab
domain     linuxacademy.lab
```

Finally, we just need to restart our BIND9 service, which can be done like so:

```
linuxacademy@ip-10-0-0-100:~$ sudo /etc/init.d/bind9 restart
```

You can then check your log file for any errors related to the BIND9 service (ignoring any IPv6 errors that can appear). Check the log as follows:

```
linuxacademy@ip-10-0-0-100:~$ sudo tail -f /var/log/syslog
```

At this point, we can conduct some further local testing that will actually query the BIND9 service locally to be sure everything is responding as expected.

Check the Forward Zone

We can check our forward zone by querying the DNS service for a response by doing the following:

```
linuxacademy@ip-10-0-0-100:~$ host -l linuxacademy.lab
linuxacademy.lab name server ip-10-0-0-100.linuxacademy.lab.
linuxacademy.lab has address 10.0.0.100
ip-10-0-0-16.linuxacademy.lab has address 10.0.0.162
www.linuxacademy.lab has address 10.0.0.162
```

Now the output after the command will vary depending on the hosts that you added to your DNS server in the forward zone configured in the earlier section. In our examples earlier, we added our two server host names and a third entry called www (whose FQDN is www.linuxacademy.lab).

We can also execute the following command:

```
linuxacademy@ip-10-0-0-100:~$ nslookup linuxacademy.lab
Server: 10.0.0.100
Address: 10.0.0.100#53

Name: linuxacademy.lab
Address: 10.0.0.100
```

You can see from these results that the DNS service is serving the appropriate results to our forward zone lookups. If you get any error regarding “host unknown”, be sure that your local /etc/resolv.conf shows the appropriate entries from our example earlier (the first nameserver has to be our own IP address to have an opportunity for the service to respond).

Check the Reverse Zone

Similar to our forward zone lookup, we are going to perform the same queries, but using the IP address rather than the name and making sure we get the appropriate name back:

```
linuxacademy@ip-10-0-0-100:~$ host 10.0.0.100
100.0.0.10.in-addr.arpa domain name pointer ~
ip-10-0-0-100.linuxacademy.lab
```

Ok so we got the expected hostname back for our IP address, let’s try:

```
linuxacademy@ip-10-0-0-100:~$ nslookup 10.0.0.100
Server: 10.0.0.100
Address: 10.0.0.100#53

100.0.0.10.in-addr.arpa    name=ip-10-0-0-100.linuxacademy.lab
```

Errors that indicate “NXDOMAN” or “SERVFAIL” means that you have a problem with your zone file configurations. Again, compare them against the examples in this lab to resolve.

Configure Client and Verify

At this point we are done. We are going to quickly go back to our Apache server (in our case the IP of 10.0.0.162) and change the /etc/resolv.conf file so that it looks like this:

```
nameserver 10.0.0.100    # our nameserver
nameserver 10.0.0.2      # AWS nameserver
search     linuxacademy.lab
domain     linuxacademy.lab
```

This will make sure that our DNS queries go first to our configured BIND9 server that we just configured but then can failover to a secondary domain server if we get no response. We should then be able to do the following:

```
linuxacademy@ip-10-0-0-162:~$ nslookup www.linuxacademy.lab
Server: 10.0.0.100
Address: 10.0.0.100#53

Name: www.linuxacademy.lab
Address: 10.0.0.162
```

Voila! We got back the DNS entry for the Apache server as configured in the private DNS server we just set up. Congratulations! You can install the Lynx text browser (lynx-cur) and navigate to the web server to verify our test page from earlier:

```
linuxacademy@ip-10-0-0-162:~$ lynx
http://www.linuxacademy.lab/test.txt
```

You should now see the page we created that contains the simple text “www.linuxacademy.lab”. At this point, we have completed and validated our setup.

In the following Appendices, you will find the full sample configuration and zone files that we used during the course of our lab. If you copy them EXACTLY and simply substitute the private IP address of your Apache server for the address 10.0.0.162 as listed, your server will work flawlessly. However, don’t let our configuration be the end of your experimenting. Feel free to experiment with additional entries and then validate your zones. Make sure you update the serial number entry in both zone files and then restart the BIND9 service for your changes to take effect.

Thanks for joining us on this ride and feel free to ask any questions you may have. Good luck!

Appendix A – named.conf.local Example

```
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in
// your organization
// include "/etc/bind/zones.rfc1918";

// our created zone for the linuxacademy lab environment
zone "linuxacademy.lab" {
     type master;
     file "/etc/bind/zones/db.linuxacademy.lab";
};

// our reverse zone
// this server IP 10.0.0.100
zone "0.0.10.in-addr.arpa" {
     type master;
     file "/etc/bind/zones/db.10";
};
```

Appendix B – named.conf.options Example

```
// Local options
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 {
          10.0.0.2;
     };

     dnssec-validation auto;

     auth-nxdomain no;    # conform to RFC1035
     listen-on-v6 { any; };
};
```

Appendix C – db.linuxacademy.lab Forward Zone Example

```
;

; BIND data file for local private domain

;

$TTL 604800

@     IN    SOA  ip-10-0-0-100.linuxacademy.lab.
admin.linuxacademy.lab. (

                     3        ; Serial

                604800        ; Refresh

                 86400        ; Retry

               2419200        ; Expire

                604800 ) ; Negative Cache TTL

;

linuxacademy.lab.   IN   NS   ip-10-0-0-100.linuxacademy.lab.

linuxacademy.lab.   IN   A    10.0.0.100

ip-10-0-0-100       IN   A    10.0.0.100

ip-10-0-0-162       IN   A    10.0.0.162

www            IN   A    10.0.0.162
```

Appendix D – db.10 Reverse Zone Example

```
;
; BIND reverse data file for local private domain
;
$TTL 604800
@    IN   SOA  ip-10-0-0-100.linuxacademy.lab.
admin.linuxacademy.lab. (
                     3        ; Serial
                604800        ; Refresh
                 86400        ; Retry
               2419200        ; Expire
                604800 ) ; Negative Cache TTL
;
          IN   NS   ip-10-0-0-100.
162       IN   PTR  ip-10-0-0-162.linuxacademy.lab.
162       IN   PTR  www.linuxacademy.lab.
100       IN   PTR  ip-10-0-0-100.linuxacademy.lab.
```