DigitalOcean | Community

Menu

By: Mitchell Anicas 36 18

How To Configure BIND as a Private Network DNS Server on Ubuntu 14.04

Aug 12, 2014 DNS, Networking, DigitalOcean Ubuntu

Tutorial Series

This tutorial is part 6 of 7 in the series: An Introduction to Managing DNS

Introduction

An important part of managing server configuration and infrastructure includes maintaining an easy way to look up network interfaces and IP addresses by name, by setting up a proper Domain Name System (DNS). Using fully qualified domain names (FQDNs), instead of IP addresses, to specify network addresses eases the configuration of services and applications, and increases the maintainability of configuration files. Setting

up your own DNS for your private network is a great way to improve the management of your servers.

In this tutorial, we will go over how to set up an internal DNS server, using the BIND name server software (BIND9) on Ubuntu 14.04, that can be used by your Virtual Private Servers (VPS) to resolve private host names and private IP addresses. This provides a central way to manage your internal hostnames and private IP addresses, which is indispensable when your environment expands to more than a few hosts.

The CentOS version of this tutorial can be found here.

Prerequisites

To complete this tutorial, you will need the following:

- Some servers that are running in the same datacenter and have private networking enabled
- A new VPS to serve as the Primary DNS server, *ns1*
- Optional: A new VPS to serve as a Secondary DNS server, *ns2*
- Root access to all of the above (steps 1-4 here)

If you are unfamiliar with DNS concepts, it is recommended that you read at least the first three parts of our Introduction to Managing DNS.

Example Hosts

For example purposes, we will assume the following:

- We have two existing VPS called "host1" and "host2"
- Both VPS exist in the nyc3 datacenter
- Both VPS have private networking enabled (and are on the 10.128.0.0/16 subnet)
- Both VPS are somehow related to our web application that runs on "example.com"

With these assumptions, we decide that it makes sense to use a naming scheme that uses "nyc3.example.com" to refer to our private subnet or zone. Therefore, *host1*'s private Fully-Qualified Domain Name (FQDN) will be "host1.nyc3.example.com". Refer to the following

table the relevant details:

Host	Role	Private FQDN	Private IP Address
host1	Generic Host 1	host1.nyc3.example.com	10.128.100.101
host2	Generic Host 2	host2.nyc3.example.com	10.128.200.102

Note: Your existing setup will be different, but the example names and IP addresses will be used to demonstrate how to configure a DNS server to provide a functioning internal DNS. You should be able to easily adapt this setup to your own environment by replacing the host names and private IP addresses with your own. It is not necessary to use the region name of the datacenter in your naming scheme, but we use it here to denote that these hosts belong to a particular datacenter's private network. If you utilize multiple datacenters, you can set up an internal DNS within each respective datacenter.

Our Goal

By the end of this tutorial, we will have a primary DNS server, *ns1*, and optionally a secondary DNS server, *ns2*, which will serve as a backup.

Here is a table with example names and IP addresses:

Host	Role	Private FQDN	Private IP Address
ns1	Primary DNS Server	ns1.nyc3.example.com	10.128.10.11
ns2	Secondary DNS Server	ns2.nyc3.example.com	10.128.20.12

Let's get started by installing our Primary DNS server, ns1.

Install BIND on DNS Servers

Note: Text that is highlighted in red is important! It will often be used to denote something that needs to be replaced with your own settings or that it should be modified or added to a configuration file. For example, if you see something like host1.nyc3.example.com, replace it with the FQDN of your own server. Likewise, if you see host1_private_IP, replace it with the private IP address of your own server.

On both DNS servers, *ns1* and *ns2*, update apt:

```
$ sudo apt-get update
```

Now install BIND:

```
$ sudo apt-get install bind9 bind9utils bind9-doc
```

IPv4 Mode

Before continuing, let's set BIND to IPv4 mode. On both servers, edit the `bind9` service parameters file:

```
$ sudo vi /etc/default/bind9
```

Add "-4" to the `OPTIONS` variable. It should look like the following:

/etc/default/bind9

```
OPTIONS="-4 -u bind"
```

Save and exit.

Now that BIND is installed, let's configure the primary DNS server.

Configure Primary DNS Server

BIND's configuration consists of multiple files, which are included from the main configuration file, `named.conf`. These filenames begin with "named" because that is the name of the process that BIND runs. We will start with configuring the options file.

Configure Options File

On *ns1*, open the `named.conf.options` file for editing:

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

Above the existing `options` block, create a new ACL block called "trusted". This is where we will define list of clients that we will allow recursive DNS queries from (i.e. your servers that are in the same datacenter as ns1). Using our example private IP addresses, we will add *ns1*, *ns2*, *host1*, and *host2* to our list of trusted clients:

/etc/bind/named.conf.options — 1 of 3

```
acl "trusted" {
        10.128.10.11;    # ns1 - can be set to localhost
        10.128.20.12;    # ns2
        10.128.100.101;  # host1
        10.128.200.102;  # host2
};
```

Now that we have our list of trusted DNS clients, we will want to edit the `options` block. Currently, the start of the block looks like the following:

/etc/bind/named.conf.options — 2 of 3

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

Below the `directory` directive, add the highlighted configuration lines (and substitute in the proper *ns1* IP address) so it looks something like this:

/etc/bind/named.conf.options — 3 of 3

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

        recursion yes;                 # enables resursive queries
        allow-recursion { trusted; };  # allows recursive queries from "trusted" clients
        listen-on { 10.128.10.11; };   # ns1 private IP address - listen on private netw
        allow-transfer { none; };      # disable zone transfers by default

        forwarders {
                8.8.8.8;
                8.8.4.4;
```

```
        };
...
};
```

Now save and exit `named.conf.options`. The above configuration specifies that only your own servers (the "trusted" ones) will be able to query your DNS server.

Next, we will configure the local file, to specify our DNS zones.

Configure Local File

On *ns1*, open the `named.conf.local` file for editing:

```
$ sudo vi /etc/bind/named.conf.local
```

Aside from a few comments, the file should be empty. Here, we will specify our forward and reverse zones.

Add the forward zone with the following lines (substitute the zone name with your own):

/etc/bind/named.conf.local — 1 of 2

```
zone "nyc3.example.com" {
    type master;
    file "/etc/bind/zones/db.nyc3.example.com"; # zone file path
    allow-transfer { 10.128.20.12; };         # ns2 private IP address - secondary
};
```

Assuming that our private subnet is *10.128.0.0/16*, add the reverse zone by with the following lines (note that our reverse zone name starts with "128.10" which is the octet reversal of "10.128"):

/etc/bind/named.conf.local — 2 of 2

```
zone "128.10.in-addr.arpa" {
    type master;
    file "/etc/bind/zones/db.10.128";  # 10.128.0.0/16 subnet
    allow-transfer { 10.128.20.12; };  # ns2 private IP address - secondary
```

```
};
```

If your servers span multiple private subnets but are in the same datacenter, be sure to specify an additional zone and zone file for each distinct subnet. When you are finished adding all of your desired zones, save and exit the `named.conf.local` file.

Now that our zones are specified in BIND, we need to create the corresponding forward and reverse zone files.

Create Forward Zone File

The forward zone file is where we define DNS records for forward DNS lookups. That is, when the DNS receives a name query, "host1.nyc3.example.com" for example, it will look in the forward zone file to resolve *host1*'s corresponding private IP address.

Let's create the directory where our zone files will reside. According to our *named.conf.local* configuration, that location should be `/etc/bind/zones`:

```
$ sudo mkdir /etc/bind/zones
```

We will base our forward zone file on the sample `db.local` zone file. Copy it to the proper location with the following commands:

```
$ cd /etc/bind/zones
$ sudo cp ../db.local ./db.nyc3.example.com
```

Now let's edit our forward zone file:

```
$ sudo vi /etc/bind/zones/db.nyc3.example.com
```

Initially, it will look something like the following:

/etc/bind/zones/db.nyc3.example.com — original

```
$TTL    604800
@       IN      SOA     localhost. root.localhost. (
                              2         ; Serial
```

```
                        604800          ; Refresh
                         86400          ; Retry
                       2419200          ; Expire
                        604800 )        ; Negative Cache TTL
;
@       IN      NS      localhost.      ; delete this line
@       IN      A       127.0.0.1       ; delete this line
@       IN      AAAA    ::1             ; delete this line
```

First, you will want to edit the SOA record. Replace the first "localhost" with *ns1*'s FQDN, then replace "root.localhost" with "admin.nyc3.example.com". Also, every time you edit a zone file, you should increment the *serial* value before you restart the `named` process--we will increment it to "3". It should look something like this:

/etc/bind/zones/db.nyc3.example.com — updated 1 of 3

```
@       IN      SOA     ns1.nyc3.example.com. admin.nyc3.example.com. (
                              3         ; Serial
```

Now delete the three records at the end of the file (after the SOA record). If you're not sure which lines to delete, they are marked with a "delete this line" comment above.

At the end of the file, add your nameserver records with the following lines (replace the names with your own). Note that the second column specifies that these are "NS" records:

/etc/bind/zones/db.nyc3.example.com — updated 2 of 3

```
; name servers - NS records
    IN      NS      ns1.nyc3.example.com.
    IN      NS      ns2.nyc3.example.com.
```

Then add the A records for your hosts that belong in this zone. This includes any server whose name we want to end with ".nyc3.example.com" (substitute the names and private IP addresses). Using our example names and private IP addresses, we will add A records for *ns1*, *ns2*, *host1*, and *host2* like so:

/etc/bind/zones/db.nyc3.example.com — updated 3 of 3

```
; name servers - A records
```

```
ns1.nyc3.example.com.          IN      A       10.128.10.11
ns2.nyc3.example.com.          IN      A       10.128.20.12

; 10.128.0.0/16 - A records
host1.nyc3.example.com.        IN      A      10.128.100.101
host2.nyc3.example.com.        IN      A      10.128.200.102
```

Save and exit the `db.nyc3.example.com` file.

Our final example forward zone file looks like the following:

/etc/bind/zones/db.nyc3.example.com — updated

```
$TTL    604800
@       IN      SOA     ns1.nyc3.example.com. admin.nyc3.example.com. (
                  3       ; Serial
             604800     ; Refresh
              86400     ; Retry
            2419200     ; Expire
             604800 )   ; Negative Cache TTL
;
; name servers - NS records
     IN      NS      ns1.nyc3.example.com.
     IN      NS      ns2.nyc3.example.com.

; name servers - A records
ns1.nyc3.example.com.          IN      A       10.128.10.11
ns2.nyc3.example.com.          IN      A       10.128.20.12

; 10.128.0.0/16 - A records
host1.nyc3.example.com.        IN      A      10.128.100.101
host2.nyc3.example.com.        IN      A      10.128.200.102
```

Now let's move onto the reverse zone file(s).

Create Reverse Zone File(s)

Reverse zone file are where we define DNS PTR records for reverse DNS lookups. That is, when the DNS receives a query by IP address, "10.128.100.101" for example, it will look in the reverse zone file(s) to resolve the corresponding FQDN, "host1.nyc3.example.com" in this case.

On *ns1*, for each reverse zone specified in the `named.conf.local` file, create a reverse zone file. We will base our reverse zone file(s) on the sample `db.127` zone file. Copy it to the proper location with the following commands (substituting the destination filename so it matches your reverse zone definition):

```
$ cd /etc/bind/zones
$ sudo cp ../db.127 ./db.10.128
```

Edit the reverse zone file that corresponds to the reverse zone(s) defined in `named.conf.local`:

```
$ sudo vi /etc/bind/zones/db.10.128
```

Initially, it will look something like the following:

/etc/bind/zones/db.10.128 — original

```
$TTL    604800
@       IN      SOA     localhost. root.localhost. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      localhost.      ; delete this line
1.0.0   IN      PTR     localhost.      ; delete this line
```

In the same manner as the forward zone file, you will want to edit the SOA record and increment the *serial* value. It should look something like this:

/etc/bind/zones/db.10.128 — updated 1 of 3

```
@       IN      SOA     ns1.nyc3.example.com. admin.nyc3.example.com. (
                              3         ; Serial
```

Now delete the two records at the end of the file (after the SOA record). If you're not sure which lines to delete, they are marked with a "delete this line" comment above.

At the end of the file, add your nameserver records with the following lines (replace the names with your own). Note that the second column specifies that these are "NS" records:

/etc/bind/zones/db.10.128 — updated 2 of 3

```
; name servers - NS records
      IN      NS      ns1.nyc3.example.com.
      IN      NS      ns2.nyc3.example.com.
```

Then add `PTR` records for all of your servers whose IP addresses are on the subnet of the zone file that you are editing. In our example, this includes all of our hosts because they are all on the 10.128.0.0/16 subnet. Note that the first column consists of the last two octets of your servers' private IP addresses in reversed order. Be sure to substitute names and private IP addresses to match your servers:

/etc/bind/zones/db.10.128 — updated 3 of 3

```
; PTR Records
11.10   IN      PTR     ns1.nyc3.example.com.    ; 10.128.10.11
12.20   IN      PTR     ns2.nyc3.example.com.    ; 10.128.20.12
101.100 IN      PTR     host1.nyc3.example.com.  ; 10.128.100.101
102.200 IN      PTR     host2.nyc3.example.com.  ; 10.128.200.102
```

Save and exit the reverse zone file (repeat this section if you need to add more reverse zone files).

Our final example reverse zone file looks like the following:

/etc/bind/zones/db.10.128 — updated

```
$TTL    604800
@       IN      SOA     nyc3.example.com. admin.nyc3.example.com. (
                              3         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
; name servers
      IN      NS      ns1.nyc3.example.com.
      IN      NS      ns2.nyc3.example.com.
```

```
; PTR Records
11.10   IN      PTR     ns1.nyc3.example.com.    ; 10.128.10.11
12.20   IN      PTR     ns2.nyc3.example.com.    ; 10.128.20.12
101.100 IN      PTR     host1.nyc3.example.com.  ; 10.128.100.101
102.200 IN      PTR     host2.nyc3.example.com.  ; 10.128.200.102
```

Check BIND Configuration Syntax

Run the following command to check the syntax of the `named.conf*` files:

```
$ sudo named-checkconf
```

If your named configuration files have no syntax errors, you will return to your shell prompt and see no error messages. If there are problems with your configuration files, review the error message and the Configure Primary DNS Server section, then try `named-checkconf` again.

The `named-checkzone` command can be used to check the correctness of your zone files. Its first argument specifies a zone name, and the second argument specifies the corresponding zone file, which are both defined in `named.conf.local`.

For example, to check the "nyc3.example.com" forward zone configuration, run the following command (change the names to match your forward zone and file):

```
$ sudo named-checkzone nyc3.example.com db.nyc3.example.com
```

And to check the "128.10.in-addr.arpa" reverse zone configuration, run the following command (change the numbers to match your reverse zone and file):

```
$ sudo named-checkzone 128.10.in-addr.arpa /etc/bind/zones/db.10.128
```

When all of your configuration and zone files have no errors in them, you should be ready to restart the BIND service.

Restart BIND

Restart BIND:

```
$ sudo service bind9 restart
```

Your primary DNS server is now setup and ready to respond to DNS queries. Let's move on to creating the secondary DNS server.

Configure Secondary DNS Server

In most environments, it is a good idea to set up a secondary DNS server that will respond to requests if the primary becomes unavailable. Luckily, the secondary DNS server is much easier to configure.

On *ns2*, edit the `named.conf.options` file:

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

At the top of the file, add the ACL with the private IP addresses of all of your trusted servers:

/etc/bind/named.conf.options — updated 1 of 2 (secondary)

```
acl "trusted" {
        10.128.10.11;   # ns1
        10.128.20.12;   # ns2 - can be set to localhost
        10.128.100.101;  # host1
        10.128.200.102;  # host2
};
```

Below the `directory` directive, add the following lines:

/etc/bind/named.conf.options — updated 2 of 2 (secondary)

```
        recursion yes;
        allow-recursion { trusted; };
        listen-on { 10.128.20.12; };      # ns2 private IP address
        allow-transfer { none; };          # disable zone transfers by default

        forwarders {
```

```
                8.8.8.8;
                8.8.4.4;
        };
```

Save and exit `named.conf.options`. This file should look exactly like *ns1*'s `named.conf.options` file except it should be configured to listen on *ns2*'s private IP address.

Now edit the `named.conf.local` file:

```
$ sudo vi /etc/bind/named.conf.local
```

Define slave zones that correspond to the master zones on the primary DNS server. Note that the type is "slave", the file does not contain a path, and there is a `masters` directive which should be set to the primary DNS server's private IP. If you defined multiple reverse zones in the primary DNS server, make sure to add them all here:

/etc/bind/named.conf.local — updated (secondary)

```
zone "nyc3.example.com" {
    type slave;
    file "slaves/db.nyc3.example.com";
    masters { 10.128.10.11; };  # ns1 private IP
};

zone "128.10.in-addr.arpa" {
    type slave;
    file "slaves/db.10.128";
    masters { 10.128.10.11; };  # ns1 private IP
};
```

Now save and exit `named.conf.local`.

Run the following command to check the validity of your configuration files:

```
$ sudo named-checkconf
```

Once that checks out, restart bind