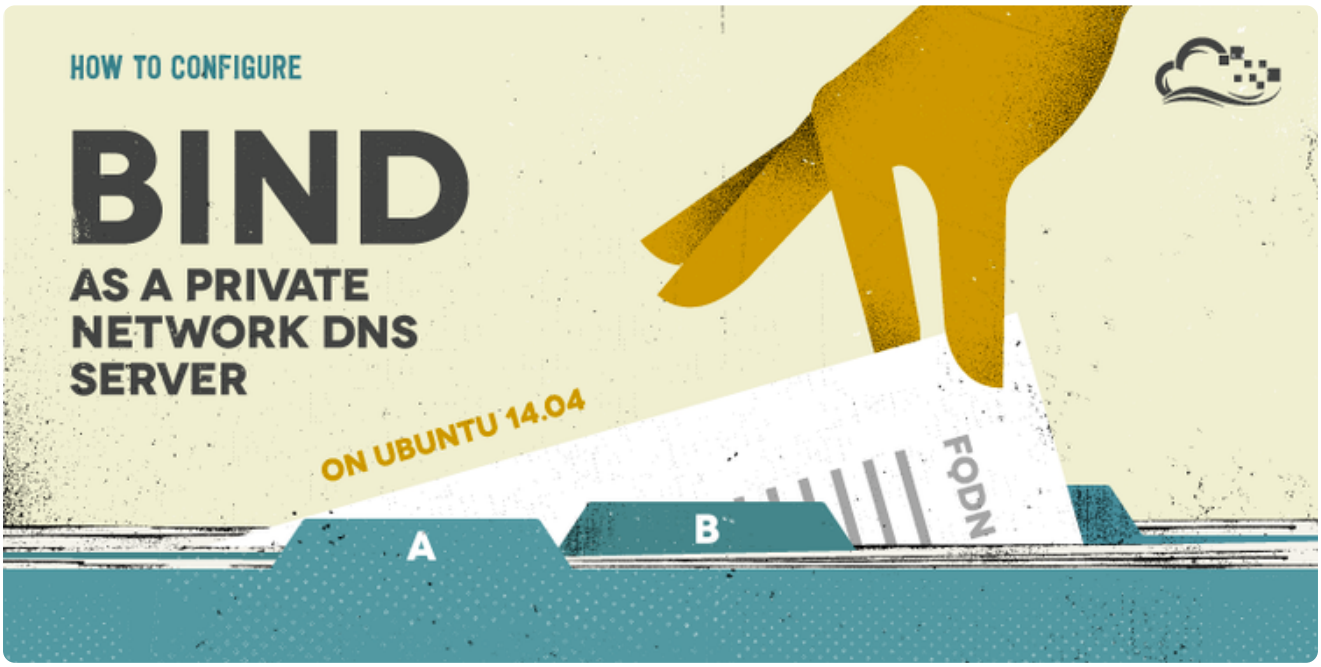

TUTORIAL

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

Ubuntu Networking DNS

By Mitchell Anicas

Posted August 12, 2014 1.1m

This article uses Ubuntu 14.04

This distro reached end of life (EOL) on Apr 2019.

Ubuntu 14.04

(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:

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.

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

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

```
                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

```
};
```

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
```

```
                        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

```
; 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

```
$ 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.

```
; 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.
```

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:

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;
};
```

```
$ 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

```
$ sudo service bind9 restart
```

Now you have primary and secondary DNS servers for private network name and IP

depending on OS, but for most Linux distributions it involves adding your name servers to the `/etc/resolv.conf` file.

Ubuntu Clients

On Ubuntu and Debian Linux VPS, you can edit the `head` file, which is prepended to `resolv.conf` on boot:

```
$ sudo vi /etc/resolvconf/resolv.conf.d/head
```

Add the following lines to the file (substitute your private domain, and *ns1* and *ns2* private IP addresses):

/etc/resolvconf/resolv.conf.d/head

```
search nyc3.example.com  # your private domain
nameserver 10.128.10.11  # ns1 private IP address
nameserver 10.128.20.12  # ns2 private IP address
```

Now run `resolvconf` to generate a new `resolv.conf` file:

```
$ sudo resolvconf -u
```

Your client is now configured to use your DNS servers.

CentOS Clients

On CentOS, RedHat, and Fedora Linux VPS, simply edit the `resolv.conf` file:

```
$ sudo vi /etc/resolv.conf
```

Now save and exit. Your client is now configured to use your DNS servers.

Test Clients

Use `nslookup` to test if your clients can query your name servers. You should be able to do this on all of the clients that you have configured and are in the "trusted" ACL.

Forward Lookup

For example, we can perform a forward lookup to retrieve the IP address of *host1.nyc3.example.com* by running the following command:

```
$ nslookup host1
```

Querying "host1" expands to "host1.nyc3.example.com because of the `search` option is set to your private subdomain, and DNS queries will attempt to look on that subdomain before looking for the host elsewhere. The output of the command above would look like the following:

```
Output:
Server:     10.128.10.11
Address:    10.128.10.11#53

Name:   host1.nyc3.example.com
Address: 10.128.100.101
```

Reverse Lookup

To test the reverse lookup, query the DNS server with *host1*s private IP address:

```
Server:     10.128.10.11
Address:    10.128.10.11#53

11.10.128.10.in-addr.arpa   name = host1.nyc3.example.com.
```

If all of the names and IP addresses resolve to the correct values, that means that your zone files are configured properly. If you receive unexpected values, be sure to review the zone files on your primary DNS server (e.g. `db.nyc3.example.com` and `db.10.128`).

Congratulations! Your internal DNS servers are now set up properly! Now we will cover maintaining your zone records.

Maintaining DNS Records

Now that you have a working internal DNS, you need to maintain your DNS records so they accurately reflect your server environment.

Adding Host to DNS

Whenever you add a host to your environment (in the same datacenter), you will want to add it to DNS. Here is a list of steps that you need to take:

Primary Nameserver

- Forward zone file: Add an "A" record for the new host, increment the value of "Serial"
- Reverse zone file: Add a "PTR" record for the new host, increment the value of "Serial"
- Add your new host's private IP address to the "trusted" ACL (`named.conf.options`)

Then reload BIND:

```
$ sudo service bind9 reload
```

An Introduction to Managing DNS **How To Configure BIND as a Pri...**

Configure New Host to Use Your DNS

- Configure resolv.conf to use your DNS servers
- Test using `nslookup`

Removing Host from DNS

If you remove a host from your environment or want to just take it out of DNS, just remove all the things that were added when you added the server to DNS (i.e. the reverse of the steps above).

Conclusion

Now you may refer to your servers' private network interfaces by name, rather than by IP address. This makes configuration of services and applications easier because you no longer have to remember the private IP addresses, and the files will be easier to read and understand. Also, now you can change your configurations to point to a new servers in a single place, your primary DNS server, instead of having to edit a variety of distributed configuration files, which eases maintenance.

Once you have your internal DNS set up, and your configuration files are using private FQDNs to specify network connections, it is **critical** that your DNS servers are properly maintained. If they both become unavailable, your services and applications that rely on them will cease to function properly. This is why it is recommended to set up your DNS with at least one secondary server, and to maintain working backups of all of them.

Next in series: How To Use NSD, an Authoritative-Only DNS Server, on Ubuntu 14.04 →

About the authors

Mitchell Anicas

Software Engineer @ DigitalOcean. Former Señor Technical Writer (I no longer update articles or respond to comments).

Tutorial Series

An Introduction to Managing DNS

DNS, or the domain name system, is an essential component of modern internet communication. It allows us to reference computers by names instead of IP addresses. In this series, we will cover the basic ideas behind DNS so that you feel comfortable working with it. Afterwards, we will walk through various ways that you can gain greater control over your domains and DNS resolution.

Next in series: How To Use NSD, an Authoritative-Only DNS Server, on Ubuntu 14.04 →

Still looking for an answer?

33 Comments

tomcampbell August 29, 2014

1 N00b here. It would be helpful if the intro paragraph told me what BIND is and why I need it. If I understand correctly, How to Set up a Host Name with DigitalOcean covers much of the same ground, so I would be delighted if the article told me when to use which.

Thanks for all these excellent docs, BTW.

Reply Report

> **manicas** August 29, 2014
>
> 0 This tutorial is part of a DNS tutorial series, which explains the basics of DNS and shows a few ways to implement it. BIND is an implementation of the DNS protocols.
>
> This particular tutorial focuses on setting up a DNS for your private network (so your servers can refer to each others' private network interfaces by name instead of IP address). If you are hosting a fairly simple infrastructure, and only require your public users to be able to access your website by name, use DigitalOcean's name servers (the method in the tutorial you linked).
>
> I would recommend reading the first two parts of the series to get an idea of why you would use BIND.
>
> Reply Report

Digidomain October 21, 2014

0 Thank you very much for an excellent tutorial! Helped me to get bind9 configured and working properly. Keep up the great work!

Reply Report

andre8525 November 26, 2014

0 Hi Mitchell,

Thank you for the detailed instructions.
If i have about 20 servers in 2 racks in the same datacenter but are in different Vlans, what

SOA

Could not find reverse address for 2400:cb00:2049:1:0:0:adf5:3a33 (3.3.a.3.5.f.d.a.0.0.0.0.0.0.0.0.1.0.0.0.9.4.0.2.0.0.b.c.0.0.4.2.ip6.arpa.).

PTR record(s) for the address could not be found in the .arpa-zone. (ip6.arpa. for IPv6 addresses and in-addr.arpa. for IPv4).

Thank you

Reply Report

chrisyeun January 11, 2015

0 Nice tutorial. One suggestion...I would define the name servers with an '@' instead of whitespace for both the forward and reverse zone files. named-checkzone will fail.

Reply Report

alexshemesh February 26, 2015

0 Hi.
I cannot make it work somehow.
I did everything like in tutorial but hostname resolution wont work.
How can i debug the problem? Logs on BIND 9? System Logs?

Reply Report

naftilos76 May 4, 2015

0 Hi, i have setup my own dns private server with Bind BUT instead of using separate vps's i used one single vps that has all components of a server like apache, postfix/dovecot and off course bind. However, i cannot understand what the purpose of the forwarders block is in the file named.conf.options (included by named.conf). I read a few articles and i concluded that the name servers in that block are used to do some of the recursive work that otherwise would be done by bind itself. However, in my case a private dns server that does not allow recursive queries for any domain other than the ones hosted in my server would NOT need the forwarders block because it only serves from its' cache. Is my thinking correct?

For exemple: I Have a Mikrotik with LoadBalance in my school with 2 links 10mbps = 20mbps
and like client a Mikrotik 1100 with raptorCache installed the machine with bind9 and I´d like to configure the bind9 correnctly ...

this my Cache with bind9
loadbalance links 192.168.168.1
Client = 192.168.50.1

> cache= 192.168.10.2:82 with raptor and bind9 installed <<<<
> DNS
> domain localdomain
> search proxy.os.com
> nameserver 127.0.0.1
> nameserver 200.165.132.154
> nameserver 8.8.4.4
> nameserver 8.8.8.8

Reply Report

andy.pixiesky June 19, 2015

0 As always these tutorials are spot on you've saved my I.T. life!

Reply Report

jmorda July 18, 2015

0 Thank you for this tutorial. It helps me a lot because i do not setup a DNS server every day...

Reply Report

jatsrt August 10, 2015

0 So, one note on the slave server. If your primary went down and you restarted the secondary, you would lose your ability to resolve.

You have "slaves/" prefixed to your sone files on the slave, which is fine, but I believe it

radhe2111 September 24, 2015

0 Hi everyone,
I have done exactly like this tutorial. But when I check zone configurations at the end by typing “sudo named-checkzone nyc3.example.com db.nyc3.example.com” & “sudo named-checkzone 128.10.in-addr.arpa /etc/bind/zones/db.10.128”,
I get an error : *zone nyc3.example.com has no NS records. Hence not loaded due to errors. *

I have checked the zone files created in the “Zone” directory in “etc” (according to this tutorial), and there are NS records in both the zone files. (Inserted exactly like in this tutorial). Still, I am getting the same error.
Can anyone please help in this problem?

Thank you for your time.

Reply Report

> BrookDO September 24, 2015
>
> 0 What does your zone file look like - could you potentially share your reverse-zone file for /etc/bind/zones/d/nyc3.example.com ?
>
> Reply Report

meliboo March 23, 2016

0 Hi, I am not sure what is wrong with the below, but I get an error when I run named-checkconf

/etc/bind/named.conf.local:11: unknown option ‘zone’
/etc/bind/named.conf.local:17: unknown option 'zone’

11 zone “lab.melina.com” {
12 type master;
13 file “/etc/bind/zones/db.lab.melina.com”;
14 allow-transfer { 10.128.20.12; };
15 };

mike101 March 26, 2016

0 I finished all steps as shown in the tutorial but it seems i do not have the DNS set up correctly. When i am trying to resolve IP's from h3 (the future DB Server) with nslookup, i get the following output:

root@h3:~# nslookup h1
Server: 2001:4860:4860::8844
Address: 2001:4860:4860::8844#53

** server can't find h1: NXDOMAIN

now here is my specific question:
Do I have to create A-Records in the Digital Ocean Nameserver (via https://cloud.digitalocean.com/networking#actions-domains), which resolve to the (public/private?!) IP of the hosts, before starting the tutorial? Meaning the hosts h1 and h2.

I am asking this question, because it seems to me, that this is a prerequisity in the tutorial, but it is not named explictly. There is shown a small table in the "Example Hosts"-part where the Private FQDN of host1 is shown as host1.nyc3.example.com. Do i have to define this private FQDN via an A-Record or am i getting it wrong?

Reply Report

> mike101 March 26, 2016
>
> 0 got the right reply from the great DigitalOcean support. If anyone else got the problem:
>
> *As for the problem, the reason you are having this issue is because currently your droplet is configured to use Google for DNS, which can be seen from the nslookup command you ran:
> Server: 2001:4860:4860::8844
>
> To use your new private DNS servers, you will want to make sure that your droplet's are configured to use the Private IP's of your DNS servers (h1 and h2) to do DNS lookups to fix this issue. To do this, you will want to edit /etc/network/interfaces and

0 I have a question. I've setup the my server based on your instructions and it works with the servers that I've added which is awesome. I also want the dns server to automatically update its records for all the different hosts on my network. I have roughly 80 different machines, laptops desktops, servers, phones that are pulling there IP's through dhcp. How do i configure the dns server to cache all of my network devices FQDN so I can access each machine via its FQDN. Windows Server DNS some how does this. Do you have any input on this process?

Thanks,

Jon

Reply Report

thinkingmedia June 13, 2016

0 This tutorial is missing firewall rules for Bind9. Make sure to allow DNS otherwise the clients will not resolve.

sudo ufw allow Bind9

Reply Report

nfaditya July 5, 2016

0 Create Reverse Zone File(s)

4th code block mentions – ns1.nyc3.example.com. after SOA

and after that in 7th code block (which is the review screen over there..) after SOA it mentions nyc3.example.com.

creating confusion, not sure what I am suppose to do

and the counter was too incremented to 2 value from 1 to 3.

please clarify...

I went through the whole process, and the server is up and running

> **nfaditya** July 8, 2016
>
> 0 Please ignore, got it working, I missed a “/” in the reverse zone file path! Sorry about that!
>
> Reply Report

fandi July 18, 2016

0 Hi
I have problem when i check nslookup on CMD Laptop.
Previously, I had checked in Putty and the results are successful:

```
#nslookup ns1
Server :          <my ip>
Address :          <my ip>#53

Name :          <my nameserver>
Address :          <my ip>
```

But, when i try on CMD Laptop, showing:

```
C:\WINDOWS\system32>nslookup <nameserver>
DNS rewuest time out.

Server :          UnKnown
Address :          <other ip>
```

Thanks before.

Reply Report

phogan September 5, 2016

0 When I do nslookup app2 (or app1, ns1, ns2 depnding on which droplet i am on) I always get the external IP address of my ns1 - any ideas where I have gone wrong? Anything in

An Introduction to Managing DNS **How To Configure BIND as a Pri...**

Reply Report

Load More Comments

GET OUR BIWEEKLY NEWSLETTER

Sign up for Infrastructure as a Newsletter.

Featured on Community Kubernetes Course Learn Python 3 Machine Learning in Python Getting started with Go Intro to Kubernetes

DigitalOcean Products Virtual Machines Managed Databases Managed Kubernetes Block Storage Object Storage Marketplace VPC Load Balancers

Load Balancers Hatch Startup Program
Block Storage Shop Swag
API Documentation Research Program
Documentation Open Source
Release Notes Code of Conduct