DNS server configuration in Linux

Introduction

An important part of managing server configuration and infrastructure is keeping network interfaces and IP addresses lookup by name in an easy way by configuring the appropriate Domain Name System (DNS) file. Using fully qualified domain names (FQDNs) instead of private network IPs is a great way to streamline server management. The steps below will show you how to set up an internal DNS server using the BIND server software (BIND9) on Ubuntu 16.04 (or 14.04), which Virtual Private Servers (VPS) can use to resolve private hostnames and private IP addresses. This provides a central way to manage your internal hostnames and private IP addresses, which is highly desirable when your network environment expands to more than a few hosts.

Prerequisites

- several machines that operate in the same data center and on the same private network,
- VPS (Virtual Private Server) serving as the primary DNS server, ns1,
- optional: VPS server serving as secondary DNS server, ns2,
- root access
-

Sample host configuration
For testing purposes, we assume the following configuration:

- We have two existing VPS named "host1" and "host2"
- Both VPS exist in "my" datacenter
- Both VPS have private network enabled (and are in subnet 10.128.0.0/16)
- Both VPS servers are somehow related to the web application running at "example.com"

With these assumptions, we decided it made sense to use a naming scheme where "my.example.com" refers to our private subnet or zone. Therefore, host1's private fully qualified domain name (FQDN) will be "host1.my.example.com". The table below shows the details of this configuration:

Host	Rola	FQDN	Adres prywatny IP
host1	ogólny host 1	host1.my.example.com	10.128.100.101
host2	ogólny host 2	host2.my.example.com	10.128.200.102

Note: The target user configuration will be different, but the sample names and IP addresses will be used to demonstrate how to configure the DNS server to provide working internal DNS. Customize

the configuration shown for your environment by replacing the hostnames and private IP addresses with your own. It is not necessary to use the name of the datacenter region in the naming scheme, but we use it here to indicate that these hosts belong to the private network of a specific datacenter. If you are using multiple datacenters, you can configure internal DNS in each relevant datacenter.

Our goal

We want to set up two DNS servers, a primary ns1 server and an optional secondary ns2 server to serve as a backup. Here is a table with sample names and IP addresses:

Installing BIND on DNS servers

Note: the text highlighted in red is important! It will often be used to mean something that needs to be replaced with your own settings or modified or added to the configuration file. For example, if you see something like host1.my.example.com, replace that with the FQDN of your own server. Similarly, if we have host1_private_IP, replace it with the private IP address of your server.

On both DNS servers, ns1 and ns2, update information about available packages:

sudo apt-get update

and then install bind:

sudo apt-get install bind9 bind9utils bind9-doc

IPv4 mode

Before we proceed, let's set BIND to IPv4 mode. On both servers, edit the bind9 service parameters file "etc/default/bind9" sudo vi /etc/default/bind9 Add "-4 -u bind" to the OPTIONS variable. The line should look like this:

OPTIONS="-4 -u bind"

Save and exit. After installing BIND, let's configure the primary DNS server.

Configuration of the primary DNS server

The BIND configuration consists of a number of files that are contained in the main configuration file named.conf. These filenames start with "named" because that is the name of the process that BIND starts. We'll start by configuring the options file. Options file configuration On ns1, let's open the named.conf.options file for editing: sudo vi /etc/bind/named.conf.options

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

Over the existing "options" block let's create a new ACL block named "trusted ". Here we will define a list of clients from which we will allow recursive DNS queries (i.e. our servers which are in the same datacenter as ns1). Using our example private IP address, we will add ns1, ns2, host1, and host2 to our list of trusted clients:

```
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'll want to edit the options block. The beginning of the block now looks like this:

```
options {

directory "/var/cache/bind"; ...

}
```

Under the "directory" directive, add the highlighted configuration lines (and substitute them in the appropriate ns1 IP) so that it looks something like this:

```
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 network only

allow-transfer { none; }; # disable zone transfers by default

forwarders { 8.8.8.8; 8.8.4.4; }; ...

};
```

Now save and close named.conf.options. The above configuration specifies that only your own servers (those "trusted") will be able to query the DNS server. Next, we'll set up a local file to specify our DNS zones.

Configuration of local files

On ns1, open the named.conf.local file for editing: sudo vi /etc/bind/named.conf.local Apart from a few comments, the file should be empty. Here we will define our forward and reverse zones. Add a forward zone with the following lines (replace the zone name with your own):

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

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

```
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 separate subnet. When you have finished adding all the desired zones, save and close the named.conf.local file. Now that our zones are specified in BIND, we need to create the appropriate forward and reverse zone files.

Creating a forward zone file

The forward zone file is where we define DNS records for forward DNS lookups. This means that when the DNS server receives a name query, for example "host1.my.example.com", it will look forward in the zone file to find the corresponding private IP address of host1. Let's create a directory for our zone files. According to our named.conf.local configuration, this location should be /etc/bind/zones:

sudo mkdir /etc/bind/zones

Our forward zone file is based on the sample db.local zone file. Let's copy it to the appropriate location with the following commands:

cd /etc/bind/zones

sudo cp ../db.local ./db.my.example.com

Now let's edit our forward zone file:

sudo vi /etc/bind/zones /db.my.example.com

Initially it will look something like this:

```
$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, we'll want to edit the SOA record. Let's replace the first "localhost" with the FQDN ns1, then "root.localhost" replace "admin.my.example.com". Also, whenever we edit a zone file, before restarting the named process, the serial value must be incremented - we will increment it to "3". The entry should look something like this:

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

Now let's delete the three records at the end of the file (after the SOA record). If you are unsure which lines to delete, they are marked with the "delete this line" comment above.

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

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

Then add A records for your hosts in that zone. This includes any server whose name ends in ".my.example.com" (replace names and private IP addresses). Using our sample names and private IP addresses, we'll add A records for ns1, ns2, host1, and host2 as follows:

; name servers - A records ns1.my.example.com.

IN A 10.128.10.11 ns2.my.example.com.

IN A 10.128.20.12 ; 10.128.0.0/16 -

A records host1.my.example.com.

IN A 10.128.100.101 host2.my.example.com. IN A 10.128.200.102

Save and close the db.my.example.com file. Our last sample forward zone file looks like this:

$TTL 604800 @ IN SOA ns1.my.example.com. admin.my.example.com.

(3 ; Serial 604800 ; Refresh 86400 ; Retry 2419200 ; Expire 604800) ; Negative Cache TTL ;

; name servers - NS records

IN NS ns1.my.example.com.

IN NS ns2.my.example.com.

; name servers - A records

ns1.my.example.com.

IN A 10.128.10.11 ns2.my.example.com.

IN A 10.128.20.12 ; 10.128.0.0/16 - A records host1.my.example.com. IN A 10.128.100.101 host2.my.example.com. IN A 10.128.200.102

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

Inverted zone file

The reverse zone file is where we define the DNS PTR records for reverse DNS lookup. This means that when the DNS server receives a query based on an IP address, for example "10.128.100.101", it will look in the reverse zone files to resolve the appropriate FQDN, in this case "host1.my.example.com" . On ns1, for each reverse zone specified in the named.conf.local file, let's create a reverse zone file. We base our inverse zone file(s) on the example db.127 zone file. Let's copy it to the appropriate location with the following commands (substituting the name of the destination file to match the definition of the reverse zone):

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

We edit the inverted zone file that corresponds to the inverted zone(s) defined in named.conf.local:

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

Initially it will look something like this:

$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 way as for the forward zone file, we will edit the SOA record and increment the serial value. It should look something like this:

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

Now let's delete the two records at the end of the file (after the SOA record). If we are unsure which lines to remove, they are marked with the "delete this line" comment above.

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

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

We then add PTR records for all servers whose IP addresses are on the subnet of the zone file you are editing. In our example, this includes all of our hosts as they are all on subnet 10.128.0.0/16. Note that the first column consists of the last two octets of the private IP addresses of the servers in reverse order. Remember to replace the names and private IP addresses to match our servers:

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

Save and close the inverted zone file (repeat this section if you want to add more inverted zone files).

Our last sample reverse zone file looks like this:

$TTL 604800 @ IN SOA my.example.com. admin.my.example.com. (3 ; Serial 604800 ; Refresh 86400 ; Retry 2419200 ; Expire 604800) ; Negative Cache TTL ; name servers IN NS ns1.my.example.com. IN NS ns2.my.example.com. ; PTR Records 11.10 IN PTR ns1.my.example.com. ; 10.128.10.11 12.20 IN PTR ns2.my.example.com. ; 10.128.20.12 101.100 IN PTR host1.my.example.com. ; 10.128.100.101 102.200 IN PTR host2.my.example.com. ; 10.128.200.102

We check the BIND configuration syntax Run the following command to check the syntax of the named.conf * files: sudo named-checkconf If the named configuration files do not contain syntax errors, the system will return to the shell prompt and we will not see any error messages. If there are problems with the configuration files, let's see the error message. After correction, try named-checkconf again. The named-checkzone command can be used to validate zone files. Its first argument specifies the name of the zone, and the second argument specifies the corresponding zone file, which are defined in named.conf.local. For example, to check the configuration of the "my.example.com" forwarding zone, run the following command (rename to match the forwarding zone and file):

sudo named-checkzone my.example.com db.my.example.com

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

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

Once all configuration and zone files are free of errors, we can restart the BIND service.

Once all configuration and zone files are free of errors, we can restart the BIND service. Restarting the BIND service sudo service bind9 restart Our primary DNS server is now set up and ready to respond to DNS queries. Let's move on to creating a secondary DNS server.

Secondary DNS server configuration

In most environments, it's a good idea to set up a secondary DNS server to respond to requests if the primary is unavailable. Fortunately, a secondary DNS server is much easier to set up.

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

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

At the top of the file we add an ACL list with the private IP addresses of all trusted servers:

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 };

Under the directory directive, let's add the following lines:

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 close named.conf.options. This file should look exactly like the named.conf.options file from ns1, except that it should be configured to listen on the private IP address of ns2. Now we edit the named.conf.local file:

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

Let's define subzones that correspond to the root zones on the primary DNS server. Note that the type is "slave", the file does not contain a path, and there is a master directive that should be set to the private IP address of the primary DNS server. If we have defined multiple reverse zones on the primary DNS server, remember to add them all here:

zone "my.example.com" { type slave; file "db.my.example.com"; masters { 10.128.10.11; }; # ns1 private IP }; zone "128.10.in-addr.arpa" { type slave; file "db.10.128";

masters { 10.128.10.11; }; # ns1 private IP };

Now save and close the named.conf.local file.

Configuration of DNS on clients machine

Before all of our servers on the "trusted" ACL can query our DNS servers, we need to configure each of them to use ns1 and ns2 as nameservers. This process varies by operating system, but for most Linux distributions, it requires adding nameservers to the /etc/resolv.conf file.

Ubuntu clients On Ubuntu and Debian Linux VPS, you can edit the head file that is included in resolv.conf on boot:

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

Let's add the following lines to the file (replace your private domain and private IP addresses ns1 and ns2):

search my.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

We now have hosts configured to use local DNS servers.