Setting up a local DNS server in Ubuntu

Prerequisites

1. Make sure the networking for your VM is set to bridged.
2. Reboot your VM.
3. Install `net-tools` so that the `netstat` command will be available.

```
$ sudo apt install net-tools
```

4. Make sure your VM has an IP address in the form `192.168.x.y`, where `x` is at least 128. `y` can be any number (only necessary when we are working together on the MARS network).

```
# example of checking ip address.
# ensure that one of the output lines has an address
# that matches the pattern described above.
$ ip addr | grep inet

inet 127.0.0.1/8 scope host lo
inet6 ::1/128 scope host
inet 192.168.128.11/17 brd 192.168.255.255 scope global dynamic noprefixroute
enp0s3
inet6 fe80::b4c9:b0c5:932:6e18/64 scope link noprefixroute
```

> Write down the IP address for your VM. You will use it later when configuring a new DNS zone.

Before we start: examine your DNS configuration

DNS clients are called *resolvers*. Before we start, let's run the following commands, which can tell us the state of the default resolver on our VM.

```
$ cat /etc/resolv.conf

...
nameserver 127.0.0.53
options edns0 trust-ad
search .
```

This shows that we are using a local address, 127.0.0.53, as our nameserver.

```
$ resolvectl status

Global
```

```
        Protocols: -LLMNR -mDNS -DNSOverTLS DNSSEC=no/unsupported
resolv.conf mode: stub

Link 2 (enp0s3)
    Current Scopes: DNS
         Protocols: +DefaultRoute +LLMNR -mDNS -DNSOverTLS DNSSEC=no/unsupported
Current DNS Server: 192.168.128.1
       DNS Servers: 192.168.128.1
```

The **Global** section shows that the nameserver we get from `/etc/resolv.conf` (127.0.0.53) is operating in "stub mode." This means that the DNS resolver is not a DNS server but a caching resolver. If it has the results of a DNS query cached, it will return the answer to you immediately. If not, it will ask its nameserver for that information, cache the results, and then return the answer to you.

The section for **Link 2 (enp0s3)** is specific to the network adapter called *enp0s3* on my VM. For that adapter, the upstream DNS server is 192.168.128.1. This just means that if the "stub" resolver at 127.0.0.53 doesn't have the answer to a DNS query cached and has to ask another DNS resolver for help, the resolver it will forward the query to is 192.168.128.1.

> Note: the following `grep` pattern is more complicated than the one I used in class. It limits the list to exactly what I want to see. It also prints the column headers so that it's easier to know what the data means.
>
> If you are just doing this during an ad-hoc terminal session, saying `grep ':53'` or `grep 'LISTEN'` is probably fine until you learn more tricks. You'll just have to rely on your eyeballs a little bit.
>
> To see how the regular expression `(Recv-Q|:53\b.*\bLISTEN\b)` works, you can visit an online regex parsing site like regular expressions 101.
>
> If you want to learn regular expressions, I recommend this tutorial. It is the site I used to teach myself regular expressions.

```
$ sudo netstat -anp | grep -E --color=no '(Recv-Q|:53\b.*\bLISTEN\b)'

Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN
481/systemd-resolve
```

This shows that currently, the only service listening on port 53, on either TCP or UDP, is `systemd-resolve`, which has a process ID of 481.

Installing BIND: the Berkeley Internet Name Domain package

> One website that has a good tutorial on configuring BIND is Linux DNS server BIND configuration, hosted by LinuxConfig.org.

BIND is a common domain name server, and typically the first one that people learn to run. We will do the following things with BIND.

- install it
- inspect DNS on our VM immediately after installation
- configure a new Internet domain name on it
- see that it resolves names for that new domain properly
- if there is time, we will use Chris's laptop as a parent server so that all of the new nameservers can work together

Installing BIND

```
sudo apt install bind9 bind9-utils bind9-doc dnsutils; echo -e "\nReturn code: $?"

# lots of output omited, but then...

Return code: 0
```

This will output lots of spiffy, mysterious things. We don't need to be concerned with what they all mean, as long as the process succeeds. How can you know if it succeeded? The `echo` statement I added after the `apt install` statement will show you a return code. If it 0, that means success.

Examining your DNS configuration after installing BIND

Let's take a look at your DNS configuration after installing BIND. Running the same two commands, did anything change?

Short answer:

- `/etc/resolv.conf` did not change
- output from `resolvectl` did not change
- however the `netstat` output changed quite a bit:

```
$ sudo netstat -anp | grep -E --color=no '(Recv-Q|:53\b.*\bLISTEN\b)' | sort

Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN
5415/named
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN
5415/named
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN
5415/named
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN
481/systemd-resolve
tcp        0      0 192.168.128.11:53       0.0.0.0:*               LISTEN
5415/named
tcp        0      0 192.168.128.11:53       0.0.0.0:*               LISTEN
5415/named
tcp        0      0 192.168.128.11:53       0.0.0.0:*               LISTEN
5415/named
tcp6       0      0 ::1:53                  :::*                    LISTEN
```

```
5415/named
tcp6       0      0 ::1:53                  :::*                    LISTEN
5415/named
tcp6       0      0 ::1:53                  :::*                    LISTEN
5415/named
tcp6       0      0 fe80::b4c9:b0c5:932::53 :::*                    LISTEN
5415/named
tcp6       0      0 fe80::b4c9:b0c5:932::53 :::*                    LISTEN
5415/named
tcp6       0      0 fe80::b4c9:b0c5:932::53 :::*                    LISTEN
5415/named
```

There are so many rows this time that I sorted them so that I could understand them more easily. Almost all of these listeners are owned by `named`, which is the BIND DNS Server. However, the `systemd-resolve` service is still listening on port 53 on address 127.0.0.53, and it can still work there, and moreover, we know from `/etc/resolv.conf` that it is still our default nameserver. We'll come back to that later.

Configuring a new Internet Domain Name (Zone)

Your own domain to play with

1. I have registered one domain for each of you to use. I hope I spelled your names correctly.
    - Anna, yours is **anna-kuefler.com**
    - Michael, yours is **michael-tice.com**
    - Tristan, yours is **tristan-ribble.com**
    - chris-leonard.com is not available, so I will use another domain I own, **cyber-pro.be**.

Generate a basic zone file for your domain

1. Use this site to generate a new zone file using the following values.
    - **Domain**: Use the name of your domain, as given above.
    - **IP Address**: Use your VM's IP address. *I told you to write that down, remember?*
    - **Admin email**: This can be any email.
    - **Nameserver 1**: Use your `ns1` followed with your domain name.
        - For example, if your domain name is **pat-smith.com**, then your **Nameserver 1** field should be `ns1.pat-smith.com`.
    - **Optional**: Select the NS option.
    - Leave everything else the same, and click **Generate Zone File**.

    In my case, my domain is `cyber-pro.be`, my hostname is `detroit`, and my IP address is 192.168.28.11. Using my real email address, chris@databaseguy.com, generated this zone file:

```
; BIND zonefile for cyber-pro.be

$TTL    7200
cyber-pro.be.           IN      SOA     ns1.cyber-pro.be.
chris.databaseguy.com. (
```

```
                                        2023021901      ; Serial
                                        7200            ; Refresh
                                        3600            ; Retry
                                        604800          ; Expire
                                        7200)           ; NegativeCacheTTL

                        IN      NS      ns1.cyber-pro.be.

cyber-pro.be.           IN      A       192.168.128.11
www                     IN      CNAME   cyber-pro.be.
ns1                     IN      A       192.168.128.11
ns2                     IN      A       192.168.128.11
```

Save your zone file on your VM

1. As root, create the directory `/etc/bind/zones/master/`.

```
$ sudo mkdir -p /etc/bind/zones/master/
```

2. As root, you need to create the file that will store the zone information you just generated. Each of your files will have a different **filename**. This **filename** is basically your domain name, but with the "labels" reversed.
    - For Anna, the **filename** is com.anna-kuefler
    - For Michael, the **filename** is com.michael-tice
    - For Tristan, the **filename** is com.tristan-ribble
    - For me, the **filename** is be.cyber-pro
3. Here's how to create the file. Make sure to use your **filename** instead of `<filename>` in the command below.
    - First, launch `nano` using the correct filename listed in the previous step.

```
$ sudo nano /etc/bind/zones/master/<filename>
```

    - Next, paste the content of the zone file you generated into this file.
    - Press Ctrl-O to save (write Out) the file. Press **enter** after confirming the filename.
    - Press Ctrl-X to exit the program.

Make changes to your DNS configuration files

When `named` starts, it reads several configuration files. We need to put information into the file `/etc/bind/named.conf.local` to tell it where to find the file we just created so that it can load information about our domain.

1. Run `sudo nano /etc/bind/named.conf.local` to edit this file. All of the lines that begin `//` are comment lines that `named` will ignore. At the bottom of the file, after all of the comments, add the

following information. Where I have entered `cyber-pro.be`, you should use your domain name instead. Where I have the file name `be.cyber-pro`, you should use your filename instead.

```
zone "cyber-pro.be" {
   type master;
   file "/etc/bind/zones/master/be.cyber-pro";
};
```

2. We also want to edit the file `/etc/bind/named.conf.options`, so that our DNS server knows where to forward DNS queries for which it does not know the answers. Open this file by running `sudo nano /etc/bind/named.conf.options`. Find the `forwarders` section. It should look like this.

```
    // forwarders {
    //     0.0.0.0;
    // };
```

This section is a comment because of the slashes in front of it. Because there are no un-commented `forwarders` sections in this file, our DNS server cannot resolve queries for zones that are not defined by zone files on our VM. For example, our DNS server cannot resolve godaddy.com.

To fix this, remove the slashes to un-comment this section (so that `named` will read it now). Change 0.0.0.0 to 8.8.8.8. This is one of Google's public nameservers. Usually more than one nameserver is specified, but this is fine for our purposes.

When you're done, this part of the file should look like this. *There are two semicolons. TWO.*

```
    forwarders {
        8.8.8.8;
    };
```

See If It Works!

To see if this all works, run the following commands.

1. Restart your DNS server

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

2. Use `nslookup` to test your DNS server.

I'm going to take advantage of the fact that all of our DNS servers will be listening on 127.0.0.1 here. You should also be able to use your "main" IP address here.

- Start `nslookup`. Once you are inside `nslookup`, enter these commands, using your own domain name instead of `<domain-name.com>`.
    - The first command is `server 127.0.0.1`. This tells `nslookup` to use the DNS server that is listening at that IP address. The default port for DNS is 53. If you are not sure what is listening on this address at port 53, you can use our earlier `netstat` commands to find out.
    - The second command is just the name `google.com.`. Notice the second `.`. This tells your DNS server to try to get an address for `google.com`. If this succeeds, this means that your DNS server is able to forward queries to 8.8.8.8, Google's public DNS server, which in turn (and not surprisingly) is able to tell you an address for `google.com`.
    - The third command is just your own domain name (with a `.` at the end). If this succeeds, it means that the resolver is communicating with your local DNS server, and that your local DNS server was able to read the zone file you created.
        - It also means that the `zone` record you put into `named.conf.local` was correct.
        - It means a lot of things are correct, to be hones.
        - It's hard to get here the first time you try. Really. If you did, show me. I'll be psyched for you.

Here is an example from my VM.

```
$ nslookup
> server 127.0.0.1
Default server: 127.0.0.1
Address: 127.0.0.1#53
> google.com.
Server:         127.0.0.1
Address:        127.0.0.1#53

Non-authoritative answer:
Name:   google.com
Address: 172.217.2.46
Name:   google.com
Address: 2607:f8b0:4009:802::200e
> cyber-pro.be.
Server:         127.0.0.1
Address:        127.0.0.1#53

Name:   cyber-pro.be
Address: 192.168.128.11
> exit
```

It Didn't Work!!!

Yeah, I told you that might happen. If it did, remember that we were able to get some clues about what went wrong by looking at the output from this command:

Just show me the last 1000 records in the `syslog` and I'll scroll up and down:

```
$ tail -1000 /var/log/syslog
```

Search the last 1000 records of the `syslog` for `named` (of course you could also search for other strings):

```
$ grep named /var/log/syslog
```

Show me the tail of the `syslog` and keep showing additional records as they arrive:

```
$ tail -f /var/log/syslog
```

Ask Me Next Week

1. How do I get my network card to use my nameserver automatically?
2. How do I make that keep working after I reboot?
3. Can I use names other than NS1, NS2, etc., for my nameservers?
    - Wait, you already told us we could do that. So, *how* do we do that?