DNSSEC

by

Steven Dormady

An Independent Study Project

Department of Computer Science

James Madison University

Department of Computer Science

April 2026

Contents

Abstract

Your abstract here...

Introduction

1.1 Motivation

DNS by itself is insecure and vulnerable to attack. With DNSSEC, these vulnerabilities are contained and mitigated.

Test below Why DNSSEC matters...[1]

1.2 Problem Statement

1.3 Contributions

What you contribute...

Background

2.1 DNS Research

Firstly, I think a bit of background about DNS helps to understand why it became what it is today. DNS started as ARPANET, which mapped names to addresses using a host file that was distributed to all entities whenever changes occurred. As it may seem, this system became rapidly unsustainable once there were over 100 networked entities, which led to DNS today.[2]

In my research about DNS, I learned a lot about what happens behind the scenes with DNS Servers. Cloudflare called it, ”DNS is like the phone book of the internet.” [1] All network systems operate with network addresses, such as IPv4 and IPv6. More or less, DNS translates domain names, like www.example.com, to IP addresses, like 127.0.0.1, so browsers can load internet resources.

DNS naming system is organized as a tree structure made up of multiple levels and naturally creates a distributed system. Each node in the tree is given a label which defines its Domain (its area or zone) of Authority. The topmost node in the tree is the Root Domain; it delegates to Domains at the next level which are generically known as the Top-Level Domains (TLDs). They in turn delegate to Second-Level Domains (SLDs), and so on. The Top-Level Domains (TLDs) include a special group of TLDs called the Country Code Top-Level Domains (ccTLDs), in which every country is assigned a unique two-character

country code. Below is a diagram demonstrating this hierarchyfigure 2.1.[2]

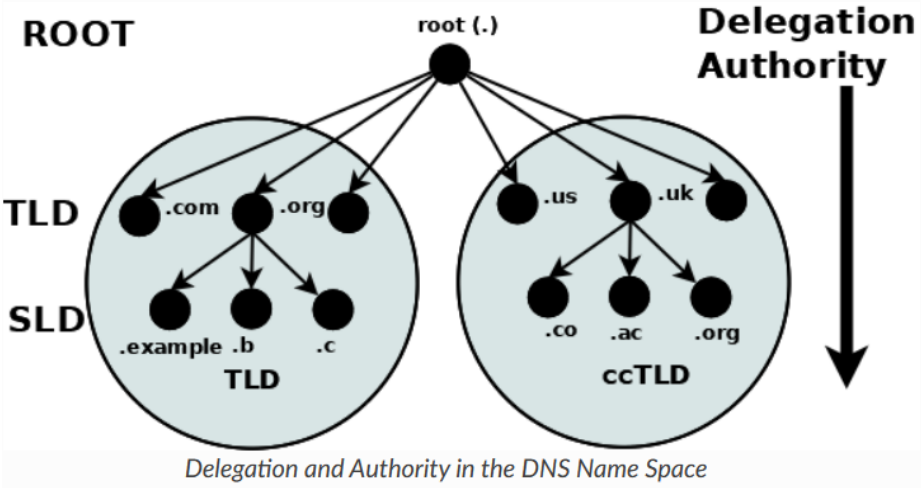


Figure 2.1: Test

A domain is the label of a node in the tree. A domain name uniquely identifies any node in the DNS tree and is written, left to right, by combining all the domain labels (each of which are unique within their parent's zone or domain of authority), with a dot separating each component, up to the root domain. In the above diagram the following are all domain names: example.com, b.com, ac.uk, us, and org. The root has a unique label of "." (dot), which is normally omitted when it is written as a domain name, but when it is written as a Fully Qualified Domain Name (FQDN) the dot must be present. As seen in figure 2.2:[2]

```
example.com      # domain name
example.com.     # FQDN
```

Figure 2.2: FDQN example, the dot on the far right making it FQDN

These root servers play a critical part of the DNS authoritative infrastructure. There are 13 root servers, which is historically tied with IPv4 data that could be packed into a 512-byte UDP message. This data limit is no longer an issue with all root servers supporting IPv4 and IPv6. In addition, almost all the root servers use anycast, with well over 300 instances

of the root servers now providing service worldwide. The root servers are the starting point for all name resolution within the DNS.[2]

The bridge between the user and DNS servers is the DNS resolver or name resolution.

2.1.1 DNS Records

These DNS records are the most common records used in building a DNS server. These records are all in db files, which are database files used for each zone defined. For example, the zone ".lab" will have a db.lab file. You can name it whatever, as long as it aligns with the named.conf.local zone path defined for the zone. It is best practice to have the domain name as the db files extension. For both the A and NS records, their line begins with an @ symbol.

SOA Records

SOA records, or "Start Of Authority" records, show the domain name that the specific db file has authority over. There are several sub records within. The MNAME is the domain name of the name server of the original or primary source of data for the zone. Like a .com, or for this study, .jmu and .lab. These are typically preceded with an ns1, which will come into play later with NS Records later. The next record is a RNAME, which is the Responsible Person record. For this project, I have root.jmu or root.lab for the root having responsibility. The next record is the serial number, which Bind uses to load different serial versions of the zone. This record has the space to be an unsigned 32 bit number. It should be updated every time there is a change to the db file. Some developers like to use the date it was edited on to help keep track of when it was last edited. Next, the refresh record is a 32 bit time interval before the zone should be refreshed. The next record is the retry record, whish is another 32 bit time interval that would run out before a failed refresh should give up. Lastly, the expire record defines another 32 bit time value that gives an upper bound on the time interval that can run before the zone is no longer authoritative. Below is an

example of an SOA record for BLANK zone.[3]

A Records

A records are address records, ”A” for short. These are the records you get when a domain maps to an IP address of a given domain. This is the most fundamental type of DNS record. There are also ”AAAA” records, which are for IPv6 traffic, but for this study we will only focus on IPv4 IPs. These records enable a user's device to connect with and load a website, or whatever is at the IP address.[4] As seen below when querying for an ip at one of my DNS Servers, I am returned with the IP address of that record.

NS Records

NS records stand for ”Nameserver” records, and indicate which DNS server is authoritative for that domain. Like in the SOA, ns1.lab shows which server is authoritative for the .lab domain. Domains can have multiple NS records, like .lab has, which indicate primary and secondary nameservers for domains. For instance, in db.lab, there are multiple NS records, one pointing the the server itself and another pointing to the secondary server for the .jmu.lab domain.[5] See below for an example.

2.1.2 DNS Vulnerabilities

DNS without DNSSEC is very insecure and vulnerable to a plethora of attacks.

DNS Spoofing/Cache Poisoning

DNS Tunneling

DNS Hijacking

NXDOMAIN Attack

2.2 VM Research

I have never dealt directly with Virtual Machines before, and building a network of them was something I had never really thought about. However, the thought excited me and it seemed like a great opportunity to learn a new skill.

The software I used to set up the testbed network for my DNSSEC program was VMWare Workstation. This software made it very easy to set up the virtual machines and network.

2.2.1 Bind9

I installed Bind9 on all of the VMs in the Virtual Network.

2.2.2 VM Network

On VM Workstation, it is very easy and simple to set up a Virtual Network. I made a custom network, isolated from the outside networks. The IP I used was 192.168.107.X, with a subnet mask of 255.255.255.0.

Bibliography

[1] Cloudflare Learning, "What is dns?." https://www.cloudflare.com/learning/dns/what-is-dns/, 2025. Accessed: 2025-12-22.

[2] BIND9, "Introduction to dns and bind 9." https://bind9.readthedocs.io/en/v9.20.17/chapter1.html#the-domain-name-system-dns, 2023. Accessed: 2026-1-1.

[3] P. Mockapetris, "Domain names - implementation and specification." https://www.rfc-editor.org/rfc/rfc1035, 1987. Accessed: 2026-1-1.

[4] Cloudflare Learning, "Dns a record." https://www.cloudflare.com/learning/dns/dns-records/dns-a-record/, 2025. Accessed: 2025-1-20.

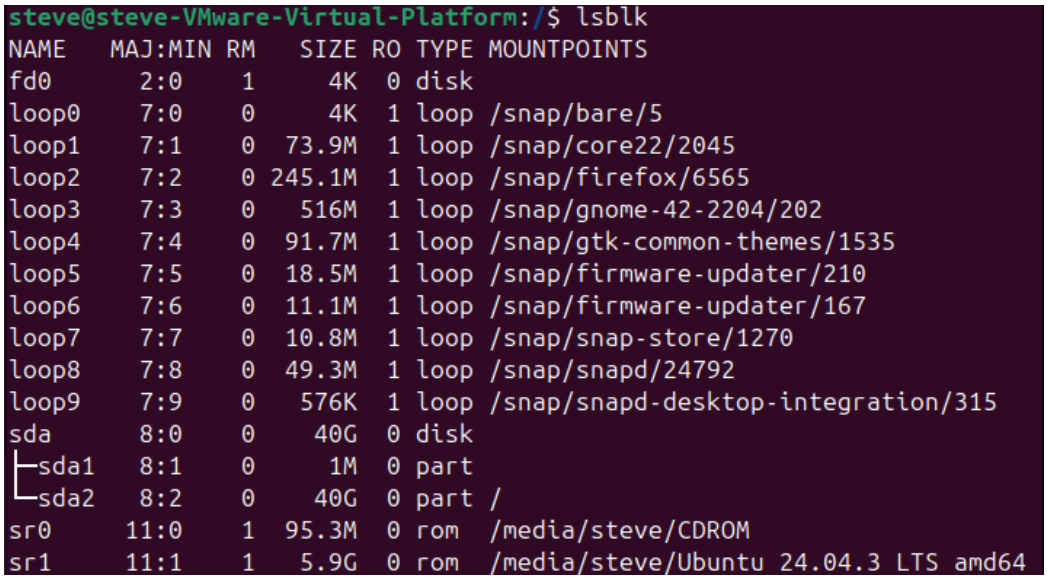
[5] Cloudflare Learning, "Dns ns record." https://www.cloudflare.com/learning/dns/dns-records/dns-ns-record/, 2025. Accessed: 2025-1-20.

AppendiX

A.1 Procedure Manual

Steps:

1. Creating a VM
    a. First, I created a VM on VMWorkstation. I selected custom setup and kept most of the settings default. My method for adding the OS to the machine was to download the LTS version of Ubuntu to my desktop and using an ISO image, I copied it onto the machine. You can create an account for your machine and name your machine whatever you may like. I gave my vm 2gb of ram and 20 gb of disk (disk space is a few steps later), as this was the recommended minimum for Ubuntu.
    b. FOR THE TIME BEING, use a NAT network condition so you can download necessary things from the internet. The default options till the end is fine, using the recommended options is something I would recommend too.
    c. Resizing VM Disk space: I started with 20gb of disk, enough to comfortably install ubuntu and have some room to work. I wanted the vm to have 40 gb of disk space to have room for software and programs I would need to implement DNSSEC later. It is probably best to backup or take a snapshot of your vm before doing this step.
        i. Turn off the machine. Expand the disk in the VM settings, with expand disk to the desired size. Then, restart the machine and follow the commands below.
        ii. For commands, I ran:
            1. *lsblk* - I ran this to see where my root partition was, like /dev/sda1 or whichever says 20G. In the picture below, it is sda2, originally 20G, now 40G after resizing.



2. Next, ensure that you have cloud guest tools installed: *sudo apt-get update* and *sudo apt-get install cloud-guest-utils*
3. Next, expand the partition with *sudo growpart /dev/sda 1*, change the number to the partion number shown by lsblk.
4. Then run *sudo resize2fs /dev/sda<previous partion number here>* .
5. The run df -h to verify the new size

d. After reportioning, we can download some software before switching to a virtual network. For ubuntu, I downloaded BIND9 to set up my DNS Servers
   i. Command: *sudo apt install bind9* and *sudo apt install dnsutils*
      1. DNSutils provides helpful troubleshooting and debugging tools, and may not be installed with bind9

e. Also, install gcc to run c programs you might need later. If you would like to as well, install Wireshark on a VM as well to monitor and see the traffic you are sending.

f. VMTOOLS
   i. I tried numerous trouble shooting techniques to install VM Tools on the vm, but I ended up doing it through the command line instead of VMWare. I did not have administrative privileges on the machine and I think that is what prevented me from doing it through VMWare.
   ii. To install them, make sure you have an internet connection and run :
      1. *sudo apt-get install open-vm-tools-desktop*. Says yes when prompted, and once complete check the status with: *systemctl status vmtoolsd*. You should see “active” in green text.

g. At this point, cloning once for another machine to treat as a dns server is not the worst idea, I made the mistake of cloning a bit too early so I just had to repeat steps on multiple vms. It wouldn’t hurt either to set up the virtual network and make sure your virtual network can be accessed by the virtual machine.

2. Setting Up the Virtual Network
   a. Under the “Edit” Tab in the menu bar, there should be a Virtual Network Editor Option. Select this to begin setting up the network.
   b. Select the “Add Network Option to create your new network.
   c. Make the vmnet# one that is not taken yet. For me, 0, 1, and 8 were in use, so I used vmnet5. Select Host-only for the network type.

d. After selecting add and creating the network, you will be able to change specifics of the network. Enter the virtual machine settings and add a network adapter setting. Deselect DHCP service to distribute IP addresses to VMs. The IP addresses will be statically assigned by you. Connecting a host virtual adapter can also be deselected, it should not be needed to connect with the host on this network.
e. You can manually input a private IP address, like 192.168.x.0, or if you save, the editor will give you a random 192.168.x.0 IP address.

3. Cloning the VM
    a. Ensure the VM you intend to clone is powered off.
        i. Before cloning, ensuring that you have a Network Adapter that is connected to the vmnet you created in the previous step. To do this, navigate to the vm settings by right-clicking on the vm itself, clicking on the vm menu tab, or while it is turned off it can be seen while viewing the vm.
        ii. Under “Hardware”, select add at the bottom of the screen. Select Network Adapter and finish.
        iii. Next, select the newly added Network Adapter, likely called Network Adapter 2 if you still have the original NAT connection.
        iv. Select Custom and your vmnet# should be available to choose from.
        v. At this stage, you shouldn’t really need the NAT connection anymore, so you can remove it or deselect both of its connection options, “connected” and “connect at power on” so that it is easily available and can be turned on whenever, versus having to add it whenever you may need to access the internet. I would say it is best practice to delete it so that it cannot be defaulted to and you can ensure that your local network is working correctly.

4. Configuring the Machines IP
    a. On your Client Machine:
        i. Open a terminal and run the command:
            1. *ip a*. If you get an outpout like the one below, then you are likely also getting “Connection issues” popups. This is because the vm is recognizing your vmnet, but doesn’t have an address yet. To set one, open up the network settings of your OS and navigate to IPv4 tab. Select manual for the method and then input an address. It’ll match you vmnet, like 192.168.x.y, x being what you set or assigned your vmnet, and y being

whatever you like, between 1 and 255. I chose 130 to start, but its personal preference.

```
d@d-VMware-Virtual-Platform:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP gro
up default qlen 1000
    link/ether 00:0c:29:b8:41:36 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet6 fe80::20c:29ff:feb8:4136/64 scope link tentative
       valid_lft forever preferred_lft forever
```

2. Then, for the subnet mask, input the same number that is listed on the vmnet.

ii. Also on the client machine, you should edit the netplan configuration. See the below photo to see the format, and replace the addresses field with your own client IP, name server field with your own name server ip, and search with your own domains. Ens36 is my leftover from when I added a NAT connection to download vmtools, and isn't necessary but it isn't hurting anything.

```
steve@steve-VMware-Virtual-Platform: /etc/netplan
-rw-r--r--   1 root root   255 Jan 27 23:50 01-network-manager-all.yaml
-rw-------   1 root root   505 Jan 29 16:36 90-NM-14f59568-5076-387a-aef6-10adfc
ca2e26.yaml
-rw-------   1 root root   583 Jan 27 23:36 90-NM-20befcd3-34f1-41a4-9523-18d8a5
89ff1a.yaml
steve@steve-VMware-Virtual-Platform:/etc/netplan$ cat 01-network-manager-all.yam
l
network:
  version: 2
  ethernets:
    ens33:
      dhcp4: no
      addresses:
        - 192.168.107.130/24
      nameservers:
        addresses:
          - 192.168.107.129
        search:
          - lab
          - jmu.lab

    ens36:
      dhcp4: yes
steve@steve-VMware-Virtual-Platform:/etc/netplan$
```

b. On the Primary Server:
    i. Open a terminal and run the command:
        1. *ip a*. If you get a similar output as before, then follow the same steps as above to set the IP and Subnet.
        2. However, for the DNS Server address, put the local 127.0.0.1, showing that this machine is a DNS Sever. Below should be the output after you add this to the settings:

```
steve@steve-VMware-Virtual-Platform:~$ resolvectl status
Global
         Protocols: -LLMNR -mDNS -DNSOverTLS DNSSEC=no/unsupported
  resolv.conf mode: stub
       DNS Servers: 127.0.0.1

Link 2 (ens33)
    Current Scopes: DNS
         Protocols: +DefaultRoute -LLMNR -mDNS -DNSOverTLS DNSSEC=no/unsupported
Current DNS Server: 127.0.0.1
       DNS Servers: 127.0.0.1
```

        3. The below netplan may or may not be necessary, use it as a last resort if you are still unable to connect at the end.

```
steve@steve-VMware-Virtual-Platform:~$ cat /etc/netplan/01-network-manager-all.y
aml
network:
  version: 2
  ethernets:
    ens33:
      dhcp4: no
      addresses:
        - 192.168.107.130/24
      nameservers:
        addresses:
          - 192.168.107.129
        search:
          - lab
          - jmu.lab

    ens36:
      dhcp4: yes
```

c. Back on the Client Machine:
    i. After setting the Primary Server IP, go back to the client and set the DNS in the same area of settings and put in the Primary Server address you just assigned.
    ii. Try pinging the client vm from another machine. The primary server with a command like ping 192.168.107.130 to ping the client machine to check for connectivity.

iii. In a terminal, run the command *resolvectl status,* and you should see something like this below

d. On the Delegated Server

i. Open a terminal and run:

1. *Ip a*. For this machine, all you need to do is set the IP address and subnet mask as you did before. Try pinging both machines like you did before, and if the output is receiving packets from other machines, they are connected.

5. Setting up DNS on the VM

a. Setting up a primary server.

i. This server will be, as the name says, the primary server that will be used when querying DNS. I will use example.jmu.lab for setting up the DNS. A good name for this vm would be VM1.

ii. First, you must add a zone. A zone in this instance would be “lab”. To add a zone, you must enter */etc/bind/named.conf.local* and edit it.

1. The syntax is seen below, and a db file is assigned to this zone.

iii. The db file does not automatically appear, hower making a copy of db.local into the name you gave it in the zone, mine being “db.lab”. Now, nano into the newly created db file.

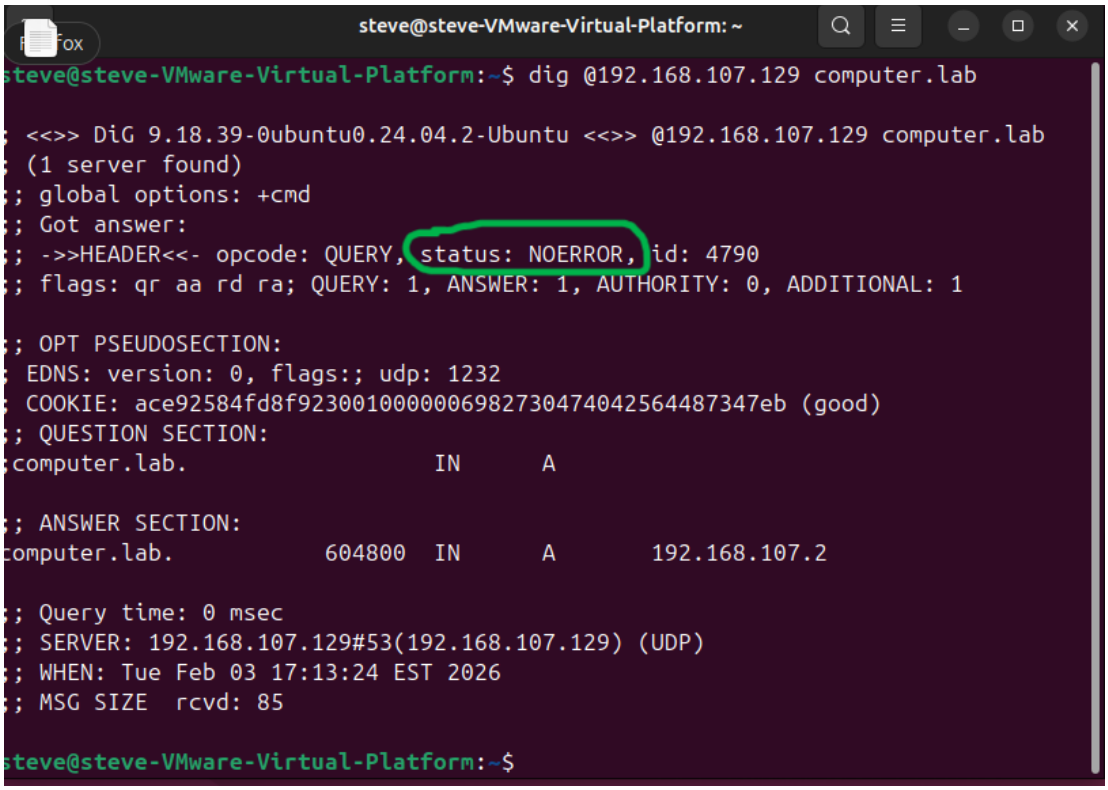
1. With the db file, you see lines that start with an @. The first line contains a lot of info, the line with SOA (Start of Authority). After the SOA, you need to edit the localhost part. For me, it would be lab. and root.lab.
2. Below that is the serial number, and this is an important part too. Every time you update this file, the serial number must be incremented too. It must only be incremented once if you make multiple changes before restarting bind9. The other numbers in the SOA aren’t too important right now.
3. Next, you must edit the lines at the bottom. At a minimum, you need an example IN A 192.168.x.y, x being the network octet and y being an octet of your choosing.
4. To make the primary server an authoritative server, @ IN NS ns.<domain>.lab. line is needed.
5. and a ns IN A 192.168.x.y line, the x being your network ip and the y octet being whatever you like, but stay away from the ips you have reserved for the Client and DNS servers.
6. To make this server authoritative, the jmu.lab. and ns1.jmu.lab. lines are required, with the ns1.jmu.lab. line ending with the delegated server’s IP address.
7. I haven’t edited the top comment, and it might be misleading but that is the comment from db.local, please disregard.

```
steve@steve-VMware-Virtual-Platform:/etc/bind$ sudo cat db.lab
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     ns1.lab. root.lab. (
                              5         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      ns1.lab.

ns1     IN      A       192.168.107.129

jmu.lab.        IN      NS      ns1.jmu.lab.
ns1.jmu.lab.    IN      A       192.168.107.128
bc      IN      A       192.168.107.1
computer        IN      A       192.168.107.2
steve@steve-VMware-Virtual-Platform:/etc/bind$
```

8. When adding to the domain server, you will likely run into errors. I these are after changing named.conf file, run a command to check for those errors: named-checkconf. I you are having issues with a zone or named-checkconf return ok, the try named-checkzone <zone-name> /etc/bind/db.<zone-name>. If it tells you an error, correct it and and keep trying till you get an OK.

```
steve@steve-VMware-Virtual-Platform:/etc/bind$ named-checkzone lab db.lab
zone lab/IN: loaded serial 5
OK
steve@steve-VMware-Virtual-Platform:/etc/bind$
```

9. At this point, running a dig command at the primary server is a good idea. Using the domain you made above, run a dig command like so:
    a. *dig @<primary server ip> example.jmu.lab* (my example domain). As you can see below, the status area shows whether or not querying the server was successful. You should see NOERROR, but you might see SERVFAIL or NXDOMAIN



10. SERVFAIL likely means there is an error in your network configuration, but running the check discussed above in 4.c.iv on the server could prove useful.
11. NXDOMAIN means that the was no domain found, maybe there was a typo, or once again run the check discussed above in 4.a.iii.8 on the server could prove useful.

b. Setting up the Secondary Server
    i. For setting up the secondary server, you follow the same steps.
    ii. First edit /etc/bind/named.conf.local to be like this for the delegated zone, .jmu in this case:

```
zone "jmu.lab" {
        type master;
        file "/etc/bind/db.jmu.lab";
};
```

    iii. In this db file, the name space record points towards the server itself, and the other records are viable domains for <name>.jmu.lab.

```
steve@steve-VMware-Virtual-Platform:~$ cat /etc/bind/db.jmu.lab
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     ns1.jmu root.jmu (
                              6         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;

@       IN      NS      ns1.jmu.

ns1     IN      A       192.168.107.128
example IN      A       192.168.107.50
shea    IN      A       192.168.107.69
john    IN      A       192.168.107.67
alley   IN      A       192.168.107.68
king    IN      A       192.168.107.66
main    IN      A       192.168.107.65
quad    IN      A       192.168.107.64
```

6. Creating a shared folder between the VM and Host machine
7. Capturing Traffic with Wireshark
    i. I kept getting ntp.ubuntu network traffic, caused by my isolated network wanting to access the outside. To stop this, I ran sudo timedatectl set-ntp off to stop this bloat from coming through. This was sending hundreds of pings every couple of seconds, making it hard to filter out so I disabled it.
8. Configuring DNSSEC

Source: Ubuntu Server Documentation - https://documentation.ubuntu.com/server/how-to/networking/install-dns/
Date accessed: 1/20/2026

Bind9 Man Pages, Date accessed 1/20/2026

https://bind9.readthedocs.io/en/v9.20.17/chapter3.html

Resolvctl Man Pages, Date accessed: 1/26/2026

https://www.freedesktop.org/software/systemd/man/latest/resolvectl.html

Resolved Man Pages, Date accessed: 1/27/2026

https://www.freedesktop.org/software/systemd/man/latest/systemd-resolved.service.html