CIS 444 - DNS Report

Lucas Hasting

October 20, 2024

Contents

1 What is DNS? 3

1.1 DNS Record Types . . . 3

2 How DNS Works 4

2.1 DNS Structure . . . 5

2.2 Zone Records . . . 7

2.3 Address Resolution . . . 7

3 DNSSEC 8

3.1 Security of DNS . . . 8

3.2 What is DNSSEC? . . . 8

3.3 NSEC Records . . . 10

4 DNS CMD Tools/File Locations 11

4.1 Tools . . . 11

4.2 File Locations . . . 11

5 DNS Process 12

5.1 Setting up the environment . . . 12

5.2 More details about the lab . . . 17

5.3 Making Installations . . . 18

5.4 Configuring DNS Options . . . 21

5.5 Configuring DNS Zones . . . 23

5.5.1 Root Server Zone Declaration . . . 23

5.5.2 TLD Zone Declaration . . . 23

5.5.3 Sub Zone Declaration . . . 24

5.6 Configuring DNS Zone Files . . . 25

5.6.1 Root Server . . . 25

5.6.2 TLD Server . . . 26

5.6.3 Sub Zone Server . . . 27

5.7 Adding Root Hints . . . 27

5.8 Activating BIND and /etc/resolv.conf . . . 29

5.8.1 Activating BIND (exclude client) . . . 29

5.8.2 configure /etc/resolv.conf . . . 29

5.9 Testing and Debugging DNS . . . 30

5.9.1 Debugging . . . 30

5.9.2 Testing . . . 32

5.10 More Reading . . . 32

1 What is DNS?

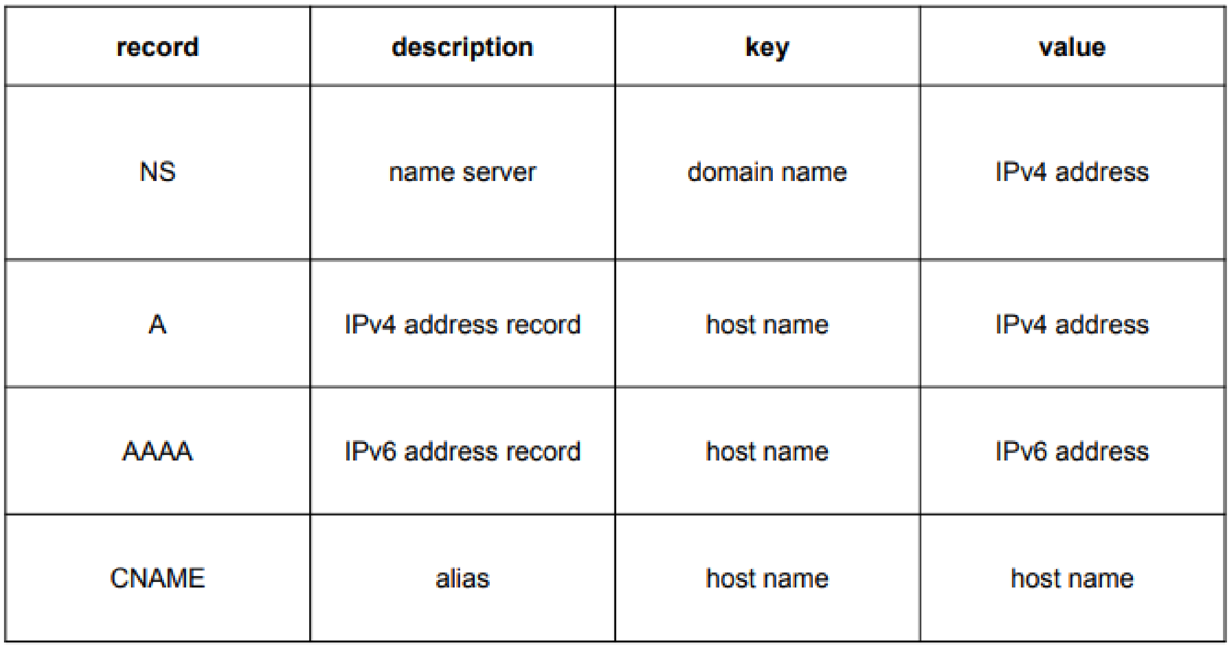
DNS (Domain Name System) is a system for mapping domain names to IP addresses. Before DNS, a file called "HOSTS.txt" was used to make the mapping. It only supported Domain Name → IP Address. The file was stored on Stanford Research Institute's FTP server, and it had to be downloaded regularly by every user (a bad system). DNS is a solution that offers more scalability.

A client program may use the HOST file before it resolves using DNS, it depends on the implementation. However, if HOST is used before DNS, the file can be used for an attack.

DNS was written in 1983, and the server was rewritten as BIND in 1985. It has operated flawlessly since it was created, but it is not a flawless system. It makes use of a distributed database of mappings, and it supports more record types than just domain name → IP address, which is listed in the next sub-section.

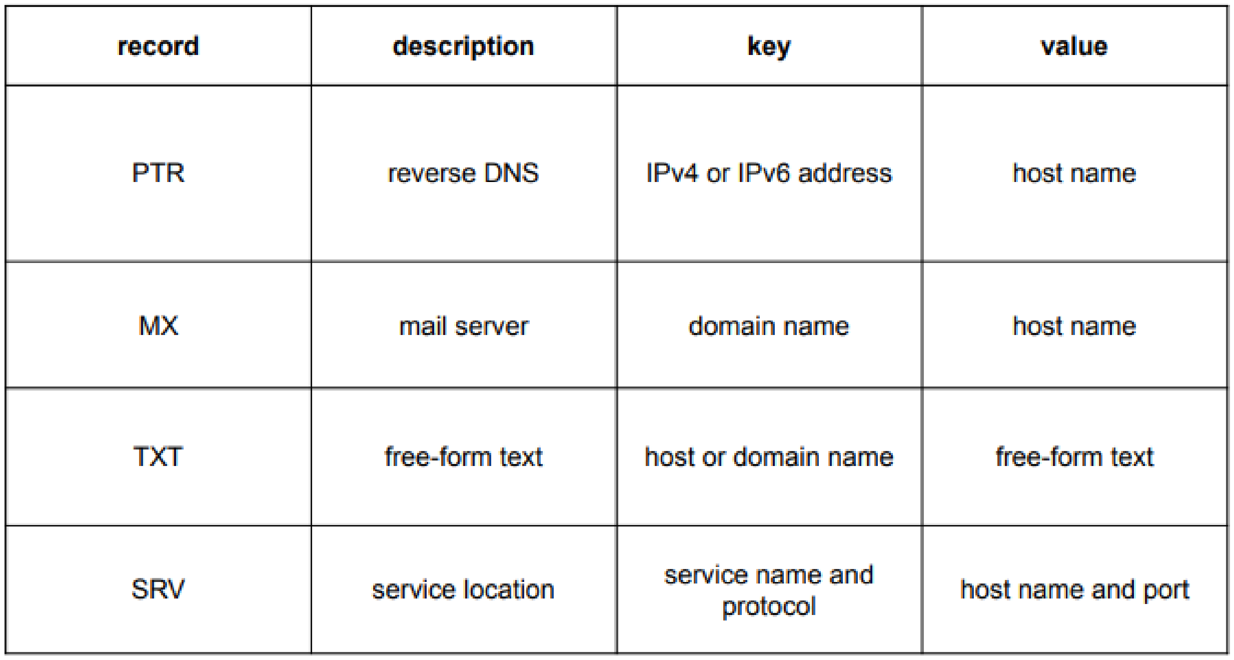
DNS info can be put onto a system when using DHCP (Dynamic Host Configuration Protocol).

1.1 DNS Record Types



record	description	key	value
NS	name server	domain name	IPv4 address
A	IPv4 address record	host name	IPv4 address
AAAA	IPv6 address record	host name	IPv6 address
CNAME	alias	host name	host name

Figure 1:



record	description	key	value
PTR	reverse DNS	IPv4 or IPv6 address	host name
MX	mail server	domain name	host name
TXT	free-form text	host or domain name	free-form text
SRV	service location	service name and protocol	host name and port

Figure 2:

2 How DNS Works

DNS is split into many zones, and those zones can be split into sub-zones. A zone can delegate control of its sub-zone to another server, and the sub-zone may be under the control of a different organization.

Example Domain Name: www.google.com

www is the host, google is the domain/sub-domain (or sub-zone), and com is the top-level domain. The root zone is implied, but it is the final dot in "www.google.com.". Domain names are processed from right to left.

The host contains the IP address, sometimes the host can resolve to another domain.

Read-only copies of entire zones can be sent to other servers, this can be used for load-balancing or failure mitigation.

Query responses can be cached to speed subsequent queries, every response has an associated lifetime that it will be cached for.

If a new record is added, all the records need to balance themselves out, this is called propagation.

2.1 DNS Structure

The structure of DNS is tree-like. A query starts at the root (”.”) zone, servers for the root zone are all over the world. All records in a zone are maintained by the same entity, but a portion of a zone can be delegated to another entity.

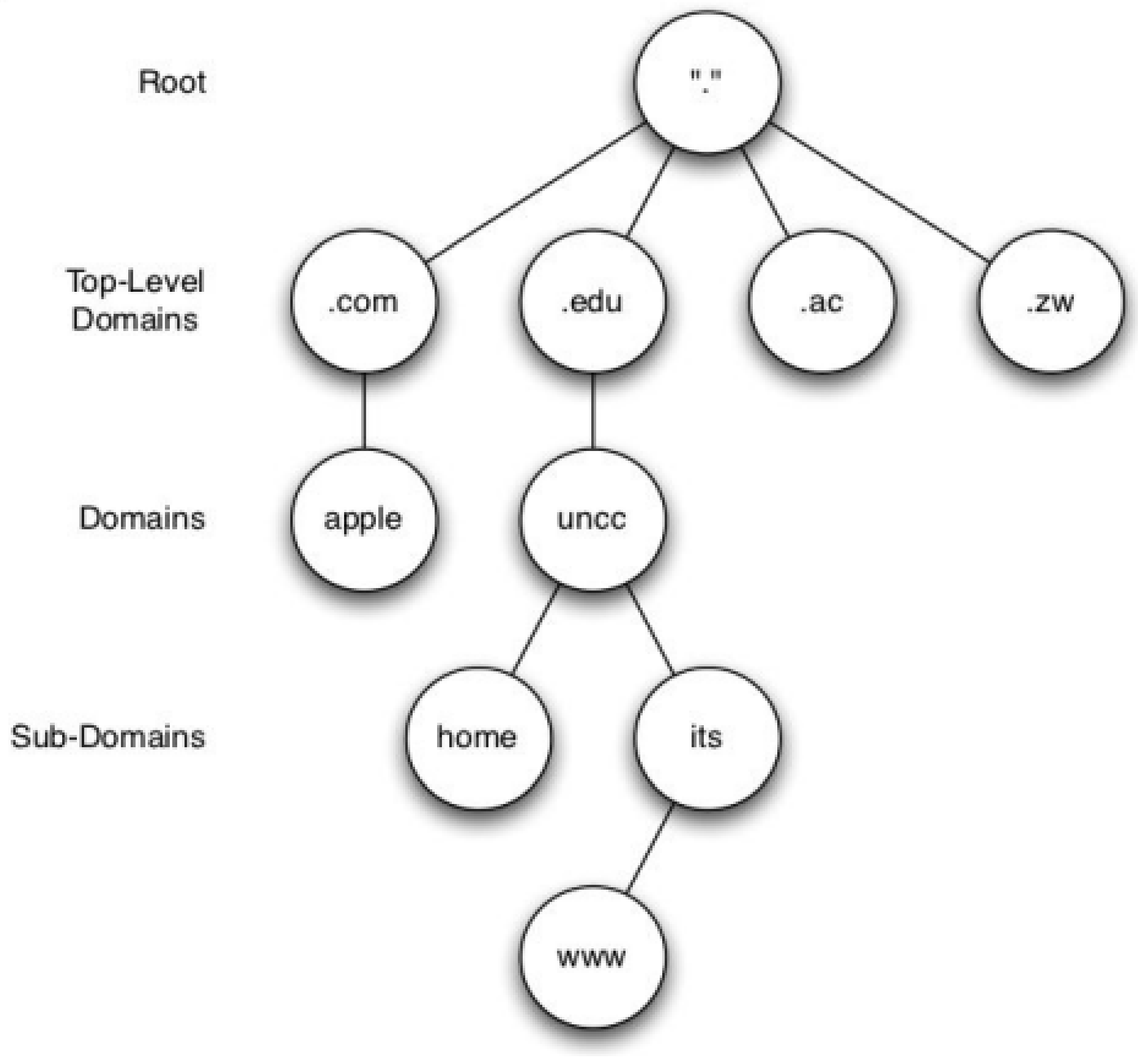


Figure 3:

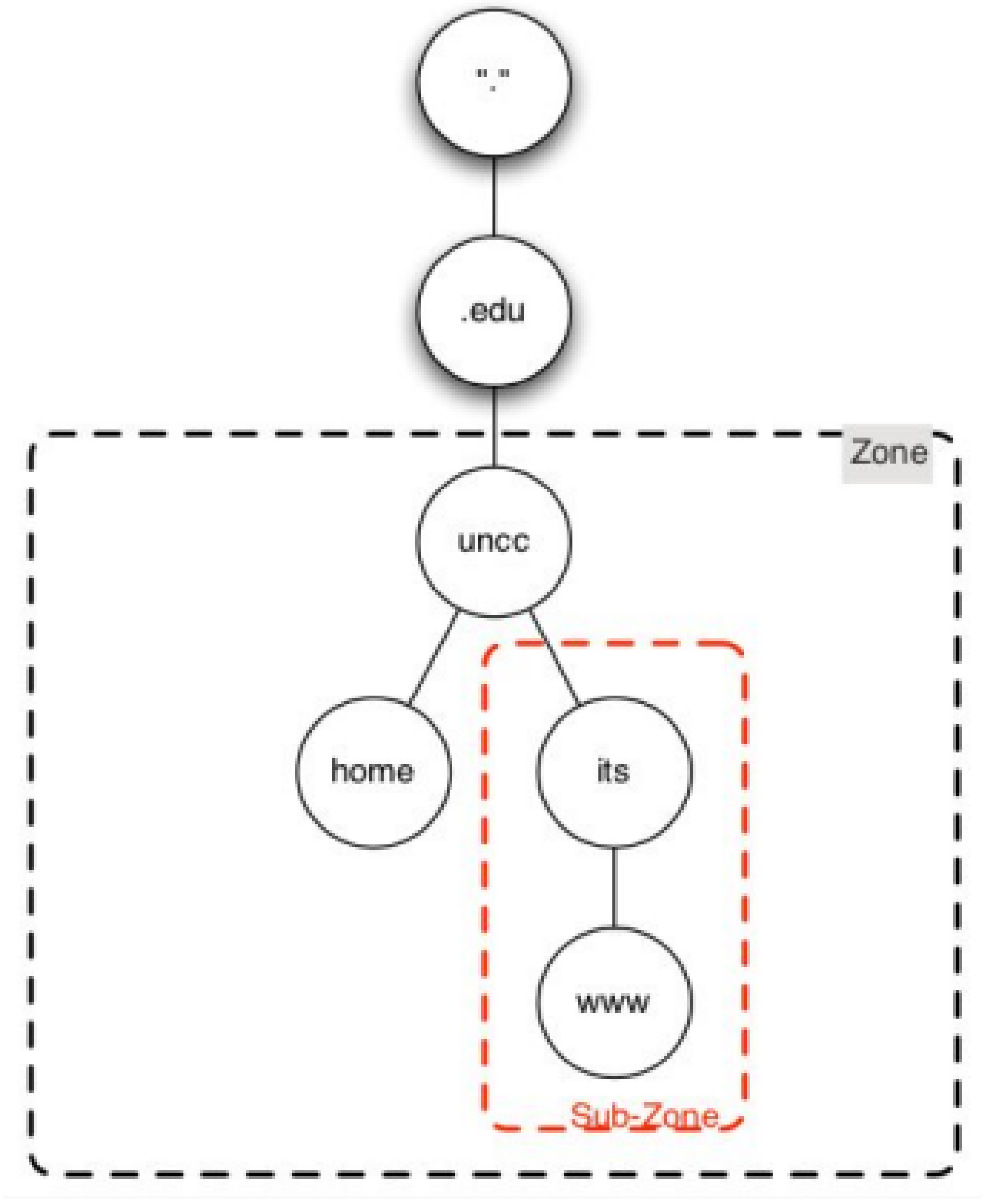


Figure 4:

2.2 Zone Records

Records are stored in a zone file that the administrator creates, this is done in section 5.6.

Every zone is required to have an SOA (start of authority) record. It contains the authoritative name server, email contact, serial number of zone, refresh time, retry time, expire time (for zone replication), and cache time-to-live for negative responses (an example with comments is located in section 5.6). The serial number used can be any number, it is usually the date, and it represents the version of the configuration.

Next, glue records can be used to delegate a sub-zone to another server, these records are hard-code A/AAA records of the sub-zone's DNS servers and prevent circular dependency.

A name server is used to translate domain names $\rightarrow$ IP address. It can have authority over 0 or more zones, in the case it has authority over 0 zones, it is a caching name server, the BIND implementation is used in the lab.

Root Hints are used in order to redirect a query to the root if the name server does not have an answer. An example of this is shown in section 5.7.

2.3 Address Resolution

DNS lookups on a host are handled by a resolver library, a client can write their own resolver implementation using these libraries.

Addresses can be resolved iteratively, meaning it iterates through multiple servers until an answer is found. This approach returns partial responses/errors. It is used by servers.

Addresses can be resolved recursively, meaning a server makes the same query to multiple servers until it receives an answer. This approach returns complete responses/errors. It is used by clients.

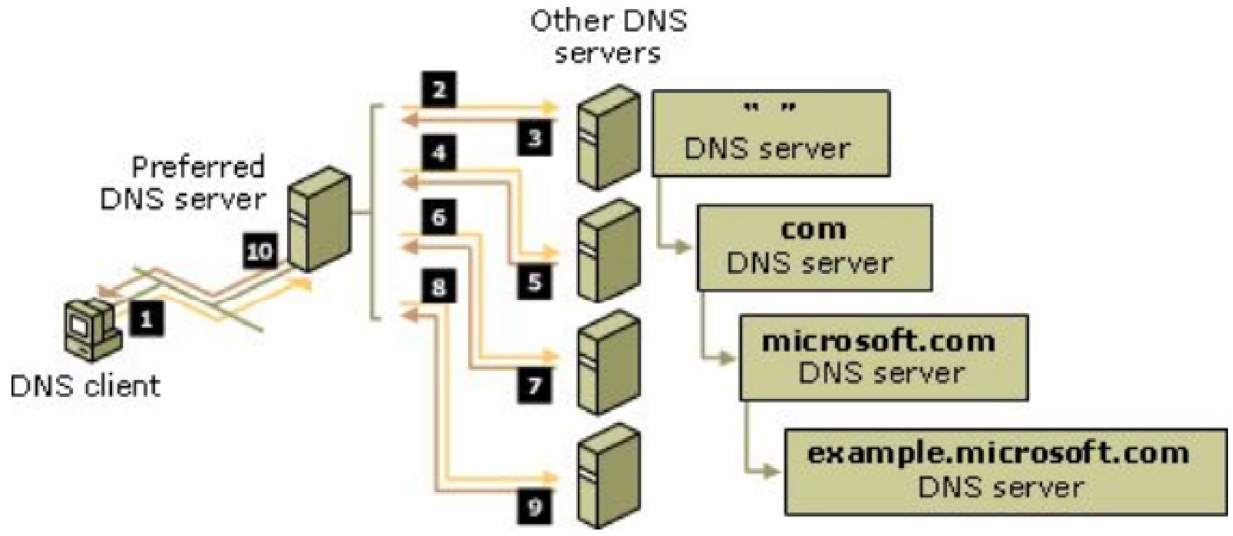


Figure 5:

3 DNSSEC

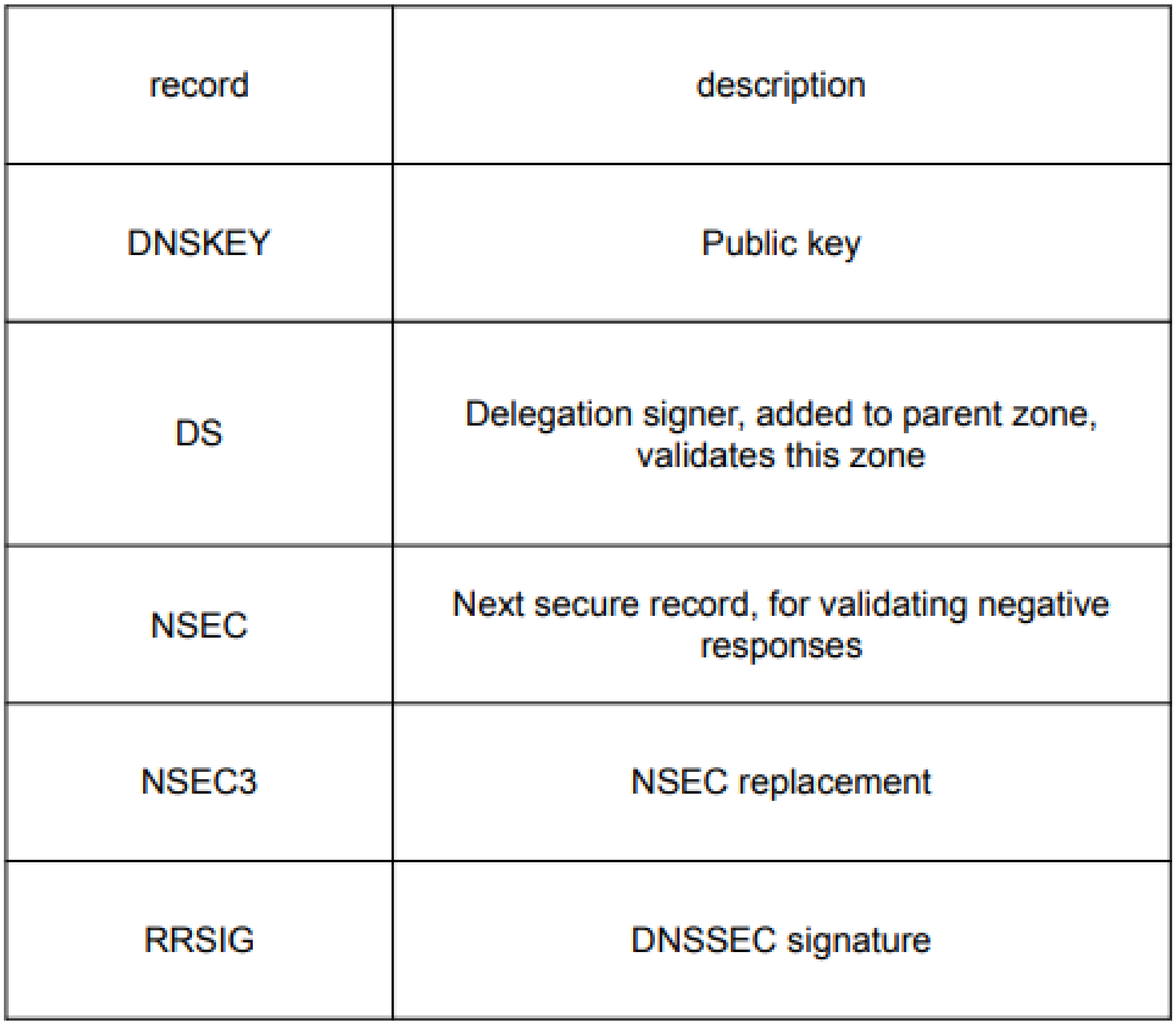
Keep in mind, DNSSEC is not used in the lab (section 5).

3.1 Security of DNS

Implementations of DNS have a history of security flaws, they include the following: Any server in your path can modify responses, any server in your path can see requests, and zone transfers are a security hole. DNSSEC was made to improve the security of DNS, however, it only works when all servers are using it.

3.2 What is DNSSEC?

Extends DNS by cryptographically sign responses. It guarantees resource records have not been tampered with, and ensures NXDOMAIN responses are genuine. It is implemented using resource records as shown below:



record	description
DNSKEY	Public key
DS	Delegation signer, added to parent zone, validates this zone
NSEC	Next secure record, for validating negative responses
NSEC3	NSEC replacement
RRSIG	DNSSEC signature

Figure 6:

DNSSEC uses public-private key cryptography (asymmetric encryption). The two key sets are: a zone-signing key and key-signing key.

The zone-signing key is used to sign all records in a zone, and stored in a DNSKEY resource record. It should be switched out often since it will be used often.

The key-signing key is used to sign a zone-signing key, and stored in a DNSKEY resource record. A pointer to key-signing key's resource record and its digest are stored in a DS record in the parent zone which create a chain of trust.

3.3 NSEC Records

NSEC records create a linked-list of all records in a zone. NXDOMAIN responses can reference the NSEC records that would come before and after the query and proves that there is no record that exists.

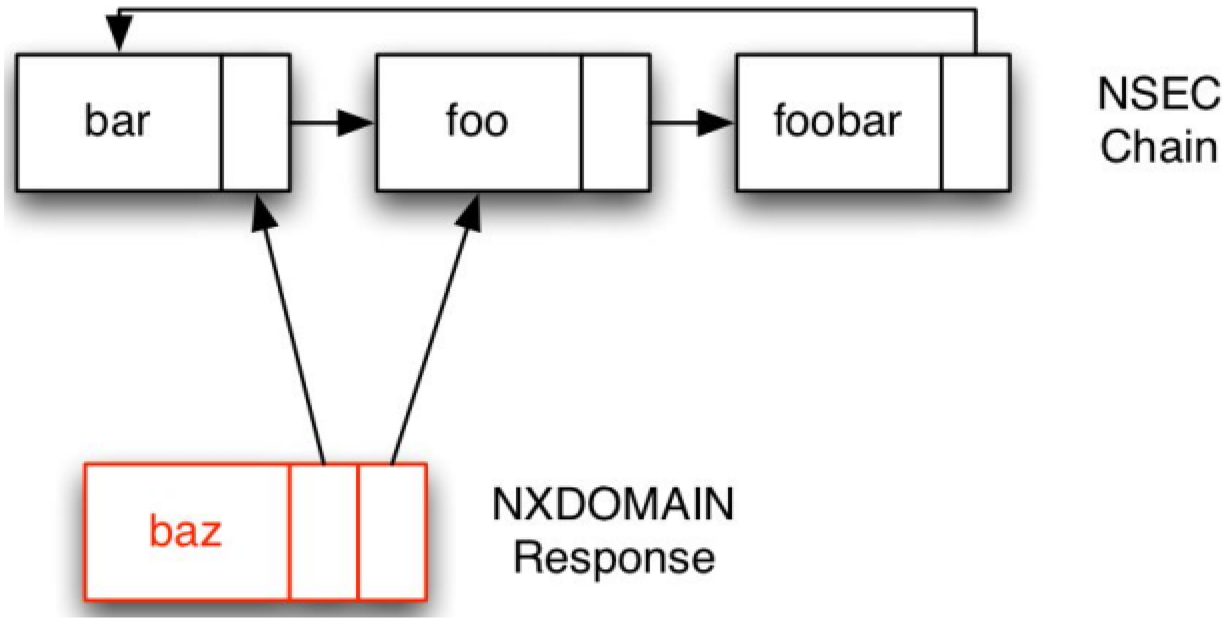


Figure 7:

DNSSEC allows walking of a domain via NSEC records, this was fixed in RFC5155 with the introduction to NSEC3.

NSEC3 replaced NSEC, it used a linked list of the hash of each record in a zone, and NXDOMAIN responses can reference the two NSEC records that would come before and after the query.

4 DNS CMD Tools/File Locations

4.1 Tools

dig (domain information groper), nslookup (name server lookup), and host are all DNS clients. They talk directly to the DNS server, bypassing hosts resolver library.

The syntax for the commands is listed below:

```
<command> <domain>
```

dig can also use the +trace argument in order to provide more troubleshooting information, the syntax is listed below:

```
dig +trace <domain>
```

While debugging, response codes may be encountered, below are a few of them:

```
NOERROR: Query completed successfully
NXDOMAIN: Query returned with a  no  such domain  error
SERVFAIL: Unable to contact the server
```

getent (get entry) can get database info, and can be used with DNS, syntax for how to do this is shown below:

```
getent <database> <key>
getent <hosts> <domain>
```

systemctl can be used to control daemons (services), some commands are listed below, a description for the first 3 are not provided as they are self-explanatory.

```
systemctl start <service>
systemctl restart <service>
systemctl stop <service>
systemctl enable <service>
```

the enable option sets the service to start when the server boots.

4.2 File Locations

Below are some important files to remember:

```
/etc/resolv.conf: specifies DNS servers, it is the starting point for
    resolving DNS
/etc/nsswitch.conf: specifies how addresses lookups are performed
```

Debian BIND files to remember:

```
/etc/bind/named.conf: contains bind configurations (includes multiple
    files which are listed below)
/etc/bind/named.conf.options: contains named options
/etc/bind/named.conf.local: contains server zone declarations
/etc/bind/named.conf.default-zones: contains default server zone
    declarations (contains the root hints)
```

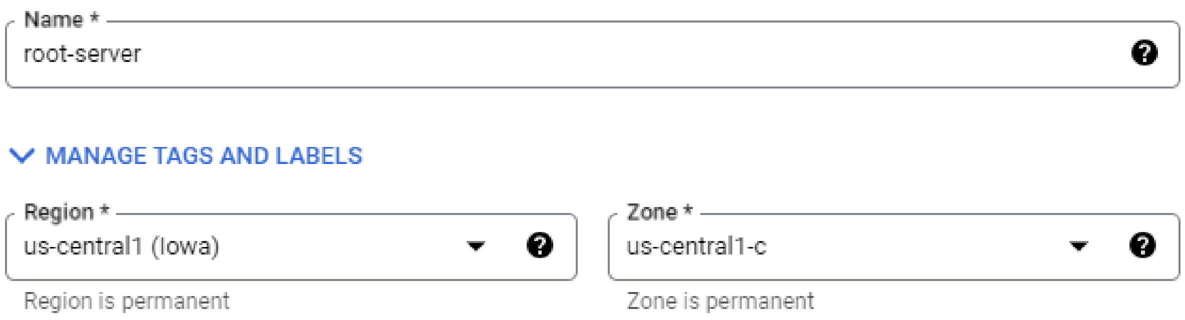
5 DNS Process

5.1 Setting up the environment

For this section, the purpose is to create the virtual machines in google cloud to be used for the lab.

Process:

```
Repeat 5 times:
    create a new VM instance
    select a name (Root, TLD, Sub Zone, Host, Client)
    change the zone to us-central-c
    choose the N1 machine configuration
    select the shared core, f1-micro machine type
    (exclude the client) go to advanced options -> Networking:
        click on the (first) default IP interface
        click ephemeral on the Primary Internal IPv4 Address
        click Reserve Static Internal IPv4 Address and then give it a name
    and reserve it
    (optional) in advanced options, select security -> managae access, and
    then add ssh keys for authentication
```


Name * root-server

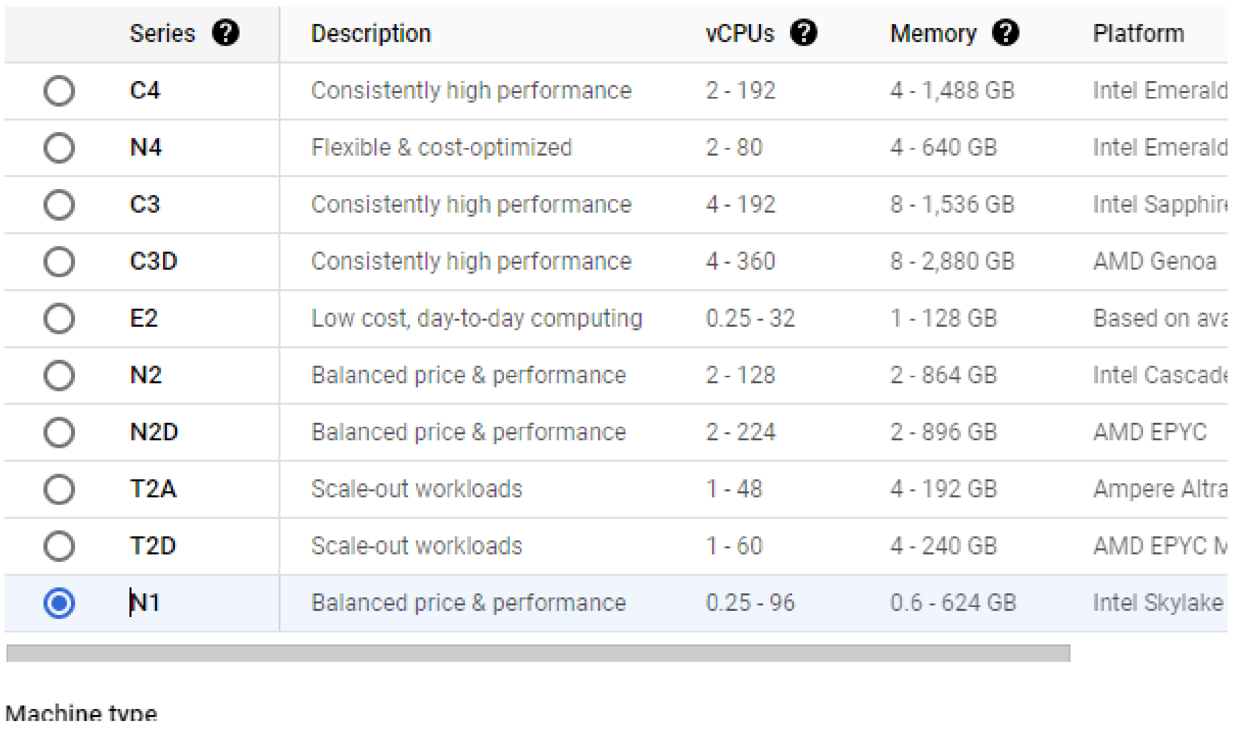
Region * us-central1 (Iowa)

Region is permanent

Zone * us-central1-c

Zone is permanent

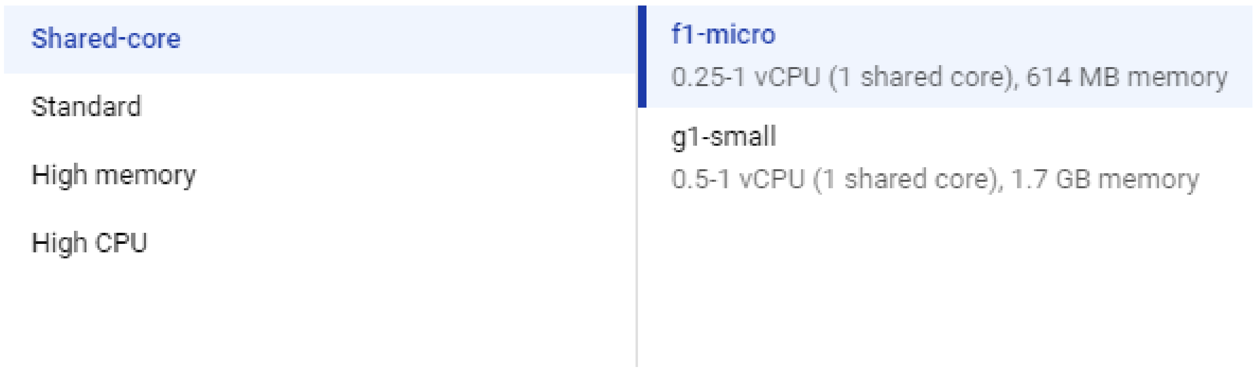
Figure 8:



	Series	Description	vCPUs	Memory	Platform
○	C4	Consistently high performance	2 - 192	4 - 1,488 GB	Intel Emerald
○	N4	Flexible & cost-optimized	2 - 80	4 - 640 GB	Intel Emerald
○	C3	Consistently high performance	4 - 192	8 - 1,536 GB	Intel Sapphir
○	C3D	Consistently high performance	4 - 360	8 - 2,880 GB	AMD Genoa
○	E2	Low cost, day-to-day computing	0.25 - 32	1 - 128 GB	Based on ava
○	N2	Balanced price & performance	2 - 128	2 - 864 GB	Intel Cascade
○	N2D	Balanced price & performance	2 - 224	2 - 896 GB	AMD EPYC
○	T2A	Scale-out workloads	1 - 48	4 - 192 GB	Ampere Altra
○	T2D	Scale-out workloads	1 - 60	4 - 240 GB	AMD EPYC M
●	N1	Balanced price & performance	0.25 - 96	0.6 - 624 GB	Intel Skylake

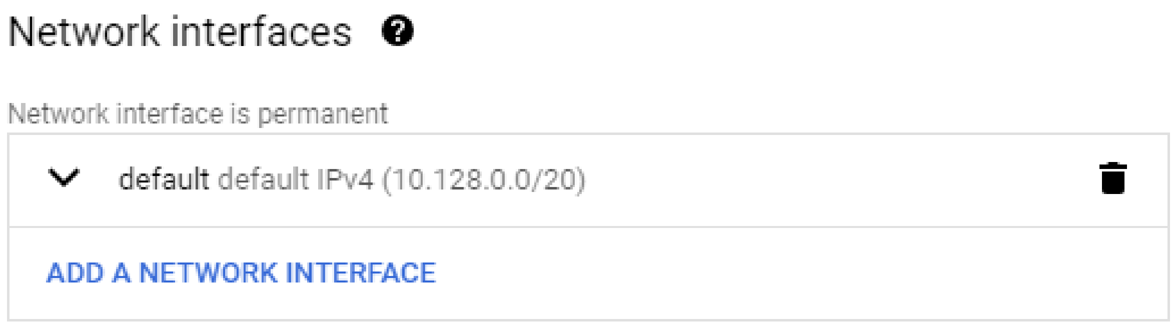
Machine type

Figure 9:



Shared-core	f1-micro 0.25-1 vCPU (1 shared core), 614 MB memory
Standard	g1-small 0.5-1 vCPU (1 shared core), 1.7 GB memory
High memory	
High CPU	

Figure 10:

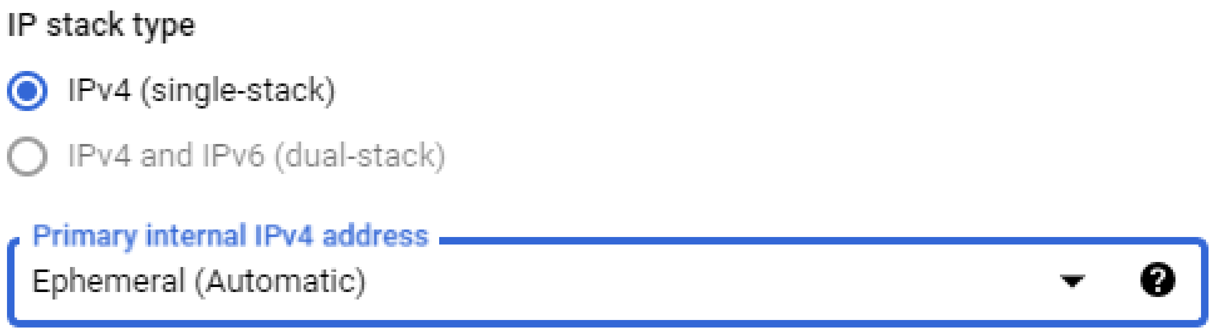


Network interfaces

Network interface is permanent

default default IPv4 (10.128.0.0/20)

Figure 11:



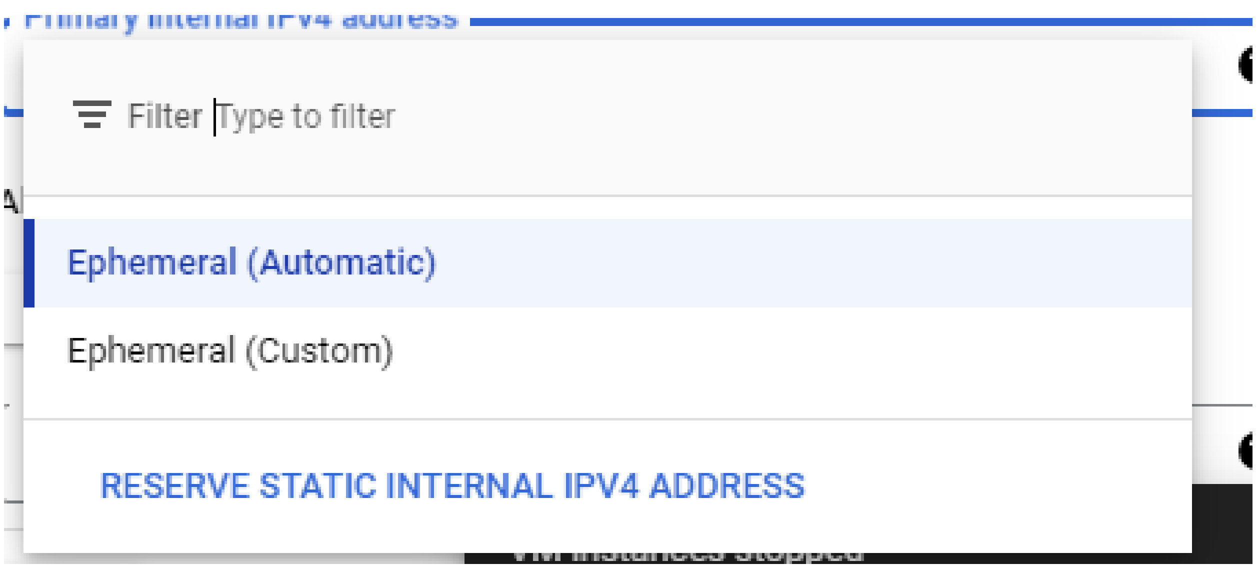
IP stack type

[✓ IPv4 (single-stack)]

IPv4 and IPv6 (dual-stack)

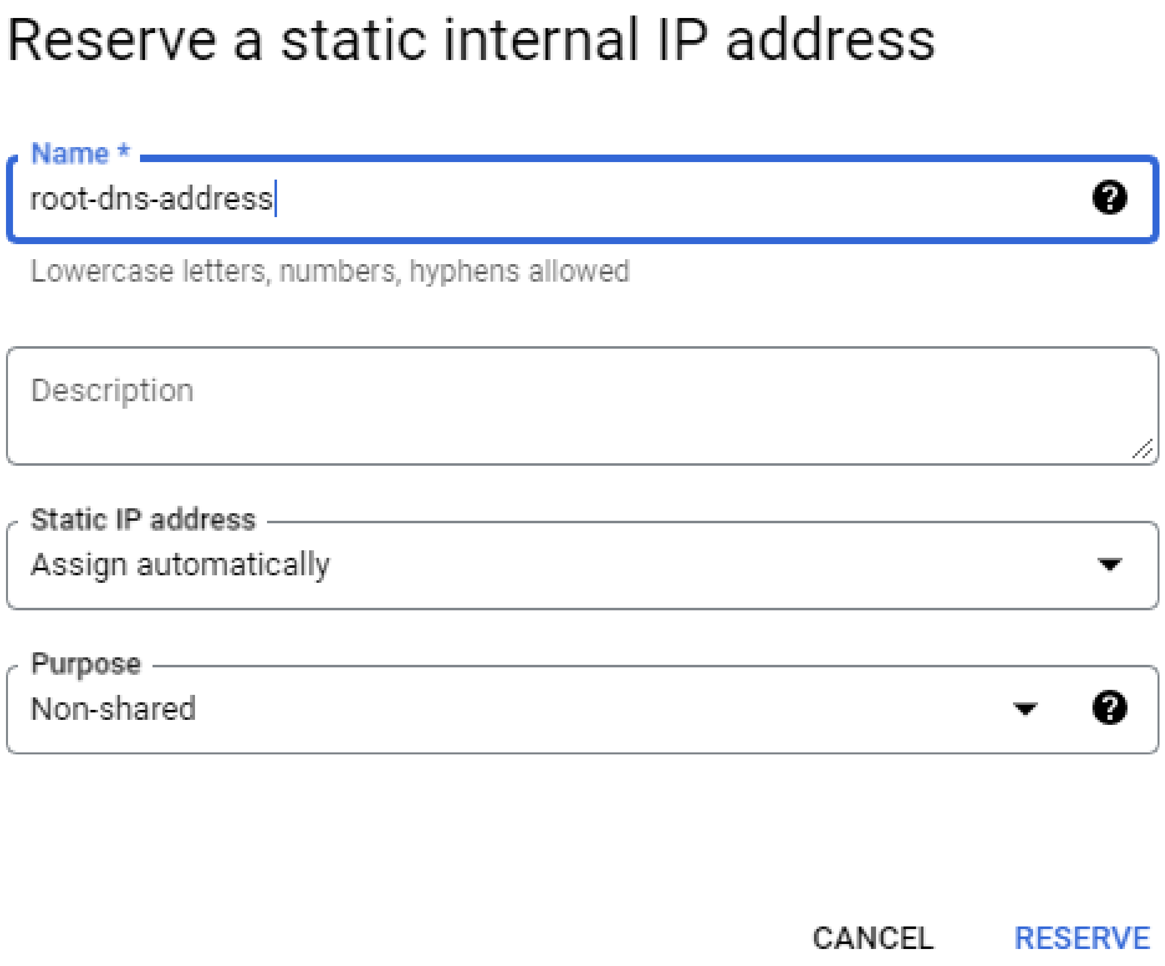
Primary internal IPv4 address: Ephemeral (Automatic)

Figure 12:



Primary internal IPv4 address
Filter Type to filter
Ephemeral (Automatic)
Ephemeral (Custom)
RESERVE STATIC INTERNAL IPV4 ADDRESS

Figure 13:



Reserve a static internal IP address

Name *
root-dns-address

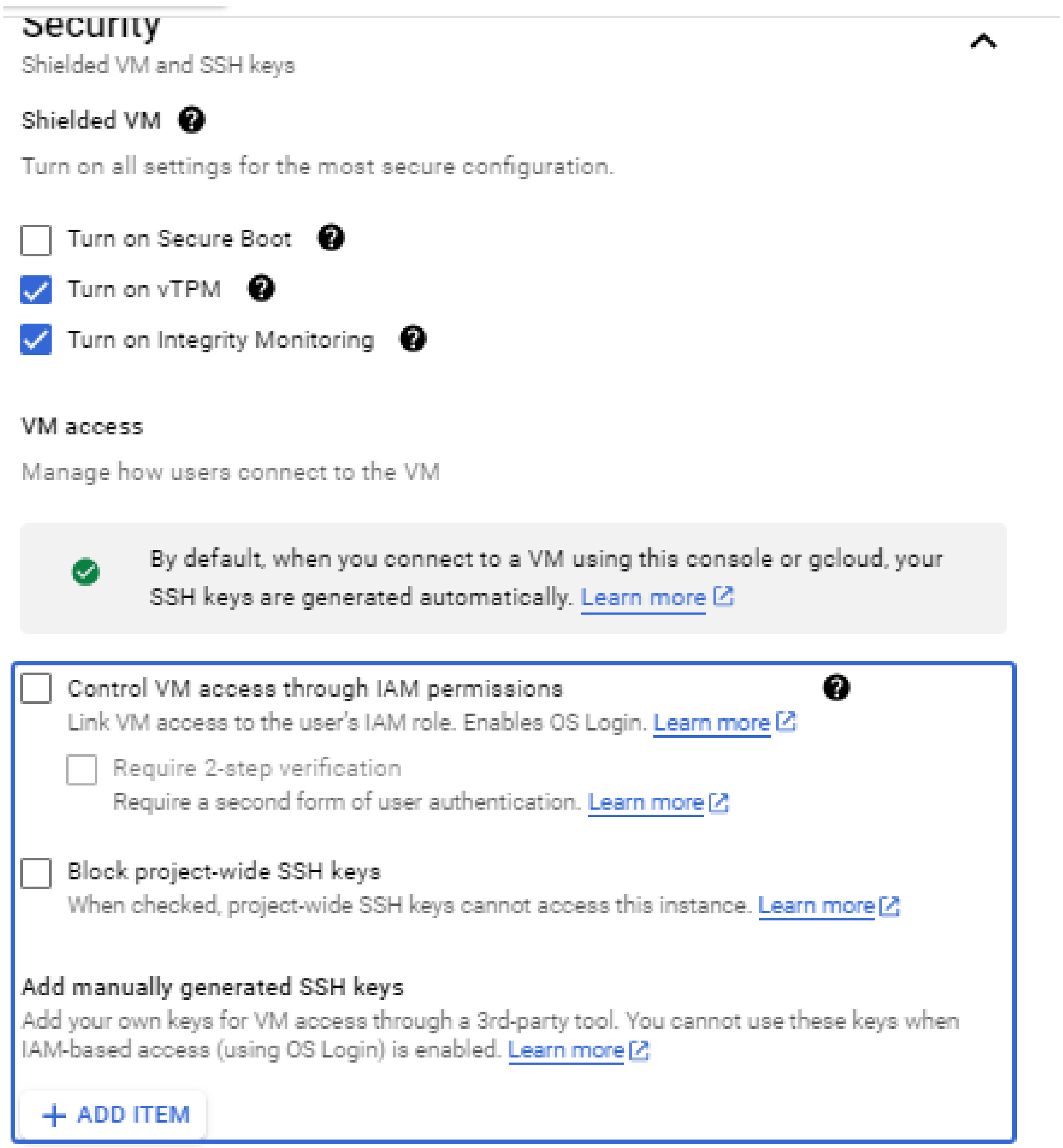
Lowercase letters, numbers, hyphens allowed

Description

Static IP address
Assign automatically

Purpose
Non-shared

Figure 14:



Security

Shielded VM and SSH keys

Shielded VM

Turn on all settings for the most secure configuration.

- [ ] Turn on Secure Boot
- [✓] Turn on vTPM
- [✓] Turn on Integrity Monitoring

VM access

Manage how users connect to the VM

By default, when you connect to a VM using this console or gcloud, your SSH keys are generated automatically. Learn more

- [ ] Control VM access through IAM permissions
  Link VM access to the user's IAM role. Enables OS Login. Learn more
  - [ ] Require 2-step verification
    Require a second form of user authentication. Learn more
- [ ] Block project-wide SSH keys
  When checked, project-wide SSH keys cannot access this instance. Learn more

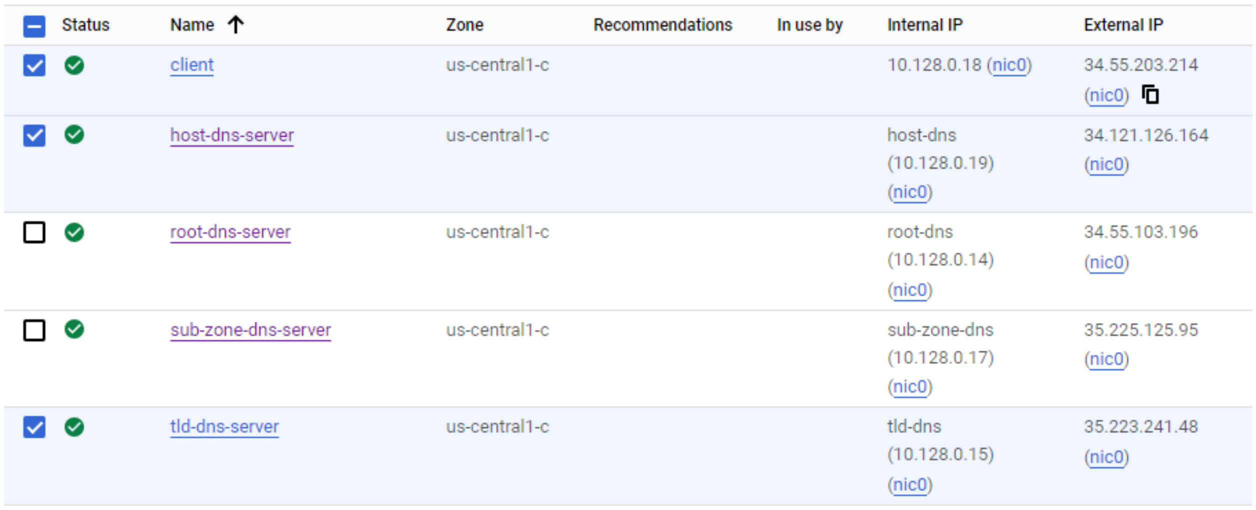
Add manually generated SSH keys

Add your own keys for VM access through a 3rd-party tool. You cannot use these keys when IAM-based access (using OS Login) is enabled. Learn more

Figure 15:

5.2 More details about the lab

All computers in the lab are on the same network. They have the following addresses (servers have a static address):



	Status	Name ↑	Zone	Recommendations	In use by	Internal IP	External IP
✓	✓	client	us-central1-c			10.128.0.18 (nic0)	34.55.203.214 (nic0)
✓	✓	host-dns-server	us-central1-c			host-dns (10.128.0.19) (nic0)	34.121.126.164 (nic0)
☐	✓	root-dns-server	us-central1-c			root-dns (10.128.0.14) (nic0)	34.55.103.196 (nic0)
☐	✓	sub-zone-dns-server	us-central1-c			sub-zone-dns (10.128.0.17) (nic0)	35.225.125.95 (nic0)
✓	✓	tld-dns-server	us-central1-c			tld-dns (10.128.0.15) (nic0)	35.223.241.48 (nic0)

Figure 16:

The lab will configure the domain name k.j., and the host host.k.j.

5.3 Making Installations

Make sure for the duration of this lab, you are a super user, it can be done by running the following command:

```
sudo -i
```

To install BIND, run the following command:

```
apt-get install -y bind9
```

To install BIND documentation, run the following command:

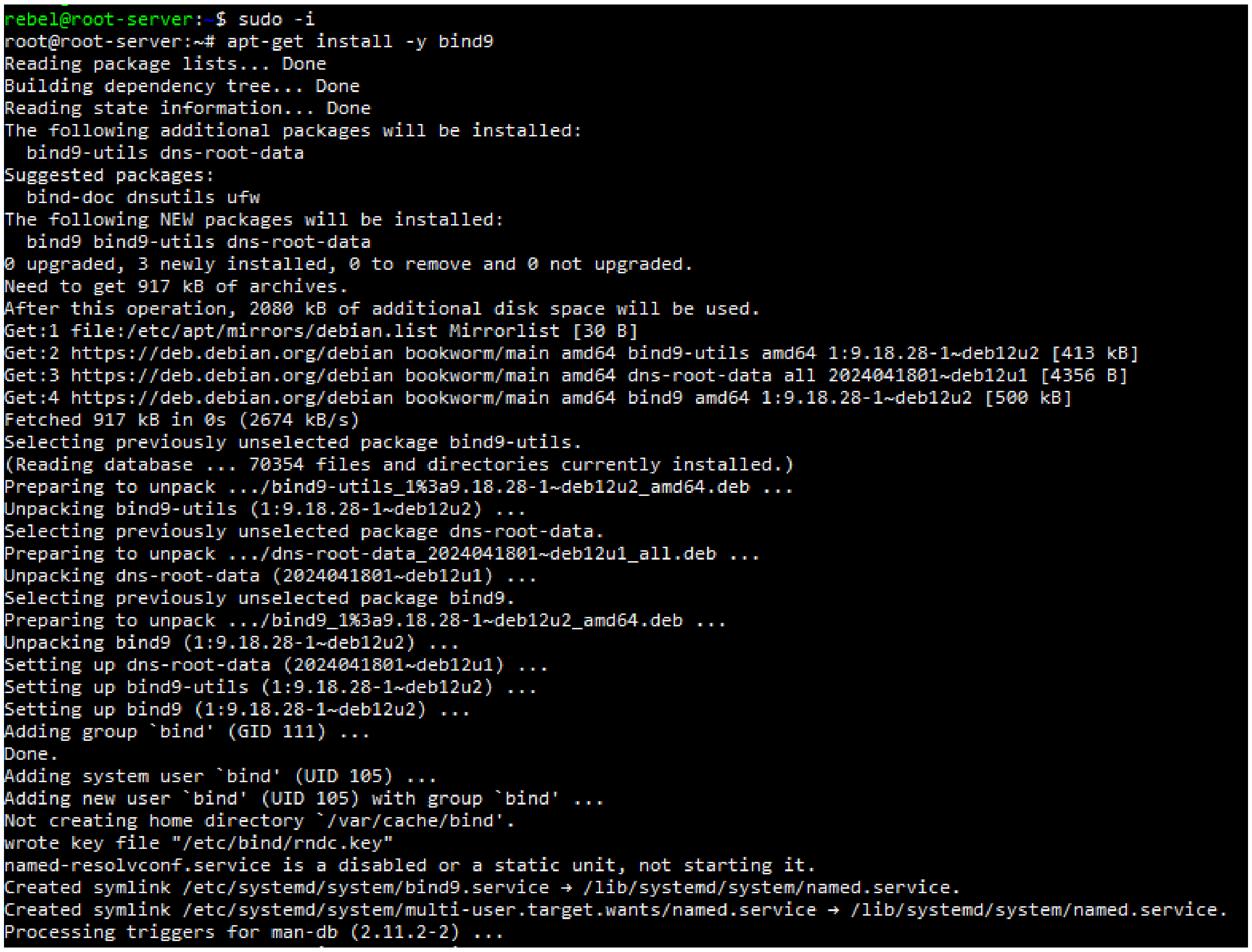
```
apt-get install -y bind9-doc
```

To install dig and other DNS tools, run the following command:

```
apt-get install -y dnsutils
```

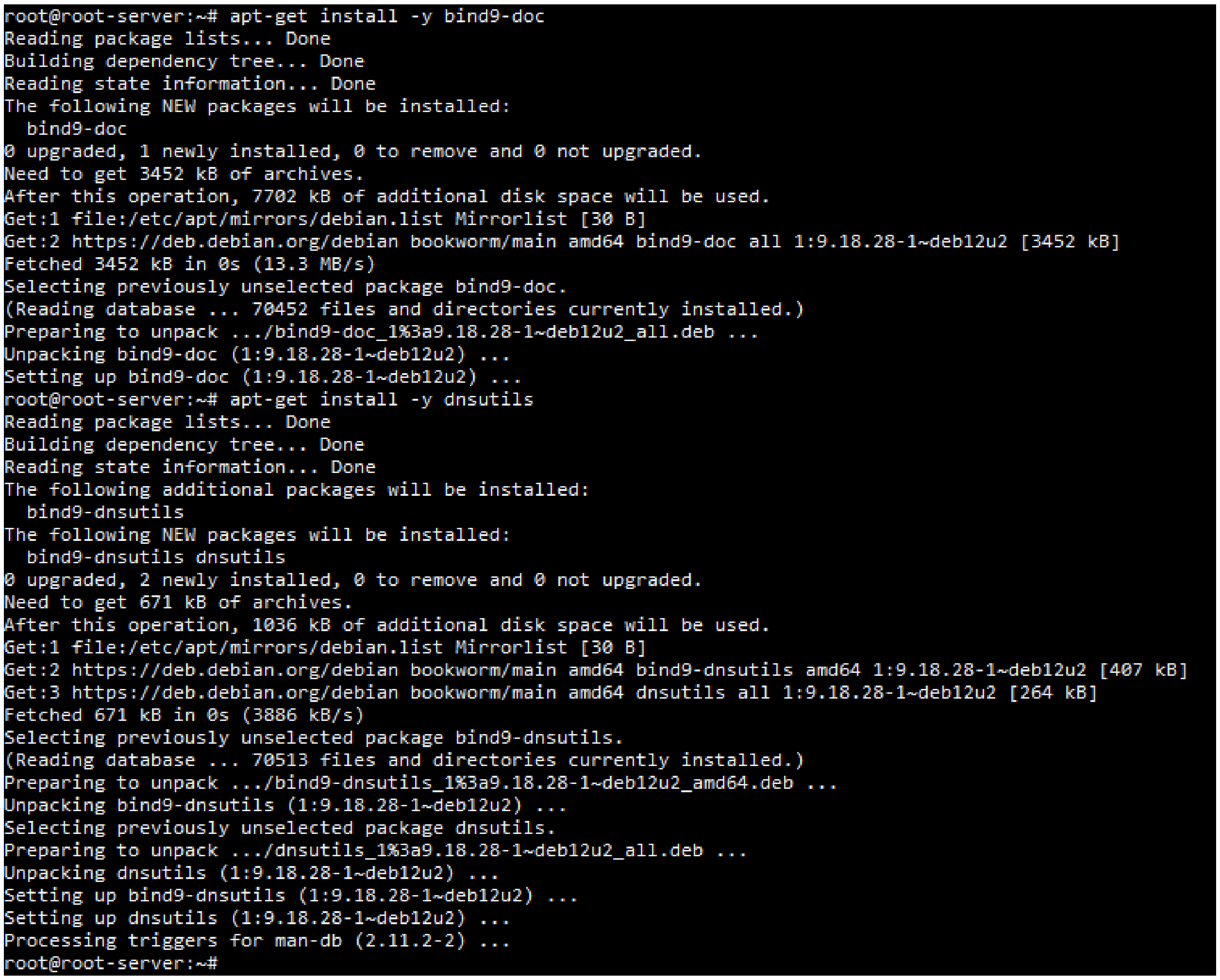
To install the tool used for keeping our /etc/resolv.conf configuration persistent, run the following command:

```
sudo apt install -y resolvconf
```



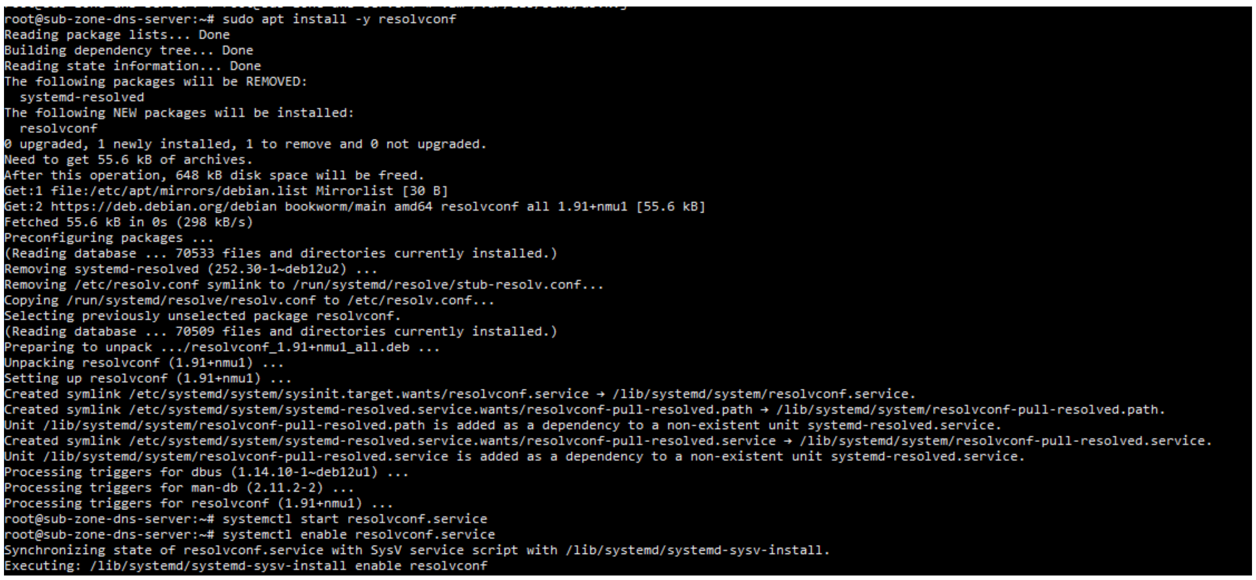
```
rebel@root-server:~$ sudo -i
root@root-server:~# apt-get install -y bind9
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bind9-utils dns-root-data
Suggested packages:
  bind-doc dnsutils ufw
The following NEW packages will be installed:
  bind9 bind9-utils dns-root-data
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 917 kB of archives.
After this operation, 2080 kB of additional disk space will be used.
Get:1 file:/etc/apt/mirrors/debian.list Mirrorlist [30 B]
Get:2 https://deb.debian.org/debian bookworm/main amd64 bind9-utils amd64 1:9.18.28-1~deb12u2 [413 kB]
Get:3 https://deb.debian.org/debian bookworm/main amd64 dns-root-data all 2024041801~deb12u1 [4356 B]
Get:4 https://deb.debian.org/debian bookworm/main amd64 bind9 amd64 1:9.18.28-1~deb12u2 [500 kB]
Fetched 917 kB in 0s (2674 kB/s)
Selecting previously unselected package bind9-utils.
(Reading database ... 70354 files and directories currently installed.)
Preparing to unpack .../bind9-utils_1%3a9.18.28-1~deb12u2_amd64.deb ...
Unpacking bind9-utils (1:9.18.28-1~deb12u2) ...
Selecting previously unselected package dns-root-data.
Preparing to unpack .../dns-root-data_2024041801~deb12u1_all.deb ...
Unpacking dns-root-data (2024041801~deb12u1) ...
Selecting previously unselected package bind9.
Preparing to unpack .../bind9_1%3a9.18.28-1~deb12u2_amd64.deb ...
Unpacking bind9 (1:9.18.28-1~deb12u2) ...
Setting up dns-root-data (2024041801~deb12u1) ...
Setting up bind9-utils (1:9.18.28-1~deb12u2) ...
Setting up bind9 (1:9.18.28-1~deb12u2) ...
Adding group `bind' (GID 111) ...
Done.
Adding system user `bind' (UID 105) ...
Adding new user `bind' (UID 105) with group `bind' ...
Not creating home directory `/var/cache/bind'.
wrote key file "/etc/bind/rndc.key"
named-resolvconf.service is a disabled or a static unit, not starting it.
Created symlink /etc/systemd/system/bind9.service → /lib/systemd/system/named.service.
Created symlink /etc/systemd/system/multi-user.target.wants/named.service → /lib/systemd/system/named.service.
Processing triggers for man-db (2.11.2-2) ...
```

Figure 17:



```
root@root-server:~# apt-get install -y bind9-doc
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  bind9-doc
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 3452 kB of archives.
After this operation, 7702 kB of additional disk space will be used.
Get:1 file:/etc/apt/mirrors/debian.list Mirrorlist [30 B]
Get:2 https://deb.debian.org/debian bookworm/main amd64 bind9-doc all 1:9.18.28-1~deb12u2 [3452 kB]
Fetched 3452 kB in 0s (13.3 MB/s)
Selecting previously unselected package bind9-doc.
(Reading database ... 70452 files and directories currently installed.)
Preparing to unpack .../bind9-doc_1%3a9.18.28-1~deb12u2_all.deb ...
Unpacking bind9-doc (1:9.18.28-1~deb12u2) ...
Setting up bind9-doc (1:9.18.28-1~deb12u2) ...
root@root-server:~# apt-get install -y dnsutils
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bind9-dnsutils
The following NEW packages will be installed:
  bind9-dnsutils dnsutils
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 671 kB of archives.
After this operation, 1036 kB of additional disk space will be used.
Get:1 file:/etc/apt/mirrors/debian.list Mirrorlist [30 B]
Get:2 https://deb.debian.org/debian bookworm/main amd64 bind9-dnsutils amd64 1:9.18.28-1~deb12u2 [407 kB]
Get:3 https://deb.debian.org/debian bookworm/main amd64 dnsutils all 1:9.18.28-1~deb12u2 [264 kB]
Fetched 671 kB in 0s (3886 kB/s)
Selecting previously unselected package bind9-dnsutils.
(Reading database ... 70513 files and directories currently installed.)
Preparing to unpack .../bind9-dnsutils_1%3a9.18.28-1~deb12u2_amd64.deb ...
Unpacking bind9-dnsutils (1:9.18.28-1~deb12u2) ...
Selecting previously unselected package dnsutils.
Preparing to unpack .../dnsutils_1%3a9.18.28-1~deb12u2_all.deb ...
Unpacking dnsutils (1:9.18.28-1~deb12u2) ...
Setting up bind9-dnsutils (1:9.18.28-1~deb12u2) ...
Setting up dnsutils (1:9.18.28-1~deb12u2) ...
Processing triggers for man-db (2.11.2-2) ...
root@root-server:~#
```

Figure 18:



```
root@sub-zone-dns-server:~# sudo apt install -y resolvconf
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages will be REMOVED:
  systemd-resolved
The following NEW packages will be installed:
  resolvconf
0 upgraded, 1 newly installed, 1 to remove and 0 not upgraded.
Need to get 55.6 kB of archives.
After this operation, 648 kB disk space will be freed.
Get:1 file:/etc/apt/mirrors/debian.list Mirrorlist [30 B]
Get:2 https://deb.debian.org/debian bookworm/main amd64 resolvconf all 1.91+nmu1 [55.6 kB]
Fetched 55.6 kB in 0s (298 kB/s)
Preconfiguring packages ...
(Reading database ... 70533 files and directories currently installed.)
Removing systemd-resolved (252.30-1~deb12u2) ...
Removing /etc/resolv.conf symlink to /run/systemd/resolve/stub-resolv.conf...
Copying /run/systemd/resolve/resolv.conf to /etc/resolv.conf...
Selecting previously unselected package resolvconf.
(Reading database ... 70509 files and directories currently installed.)
Preparing to unpack .../resolvconf_1.91+nmu1_all.deb ...
Unpacking resolvconf (1.91+nmu1) ...
Setting up resolvconf (1.91+nmu1) ...
Created symlink /etc/systemd/system/sysinit.target.wants/resolvconf.service → /lib/systemd/system/resolvconf.service.
Created symlink /etc/systemd/system/systemd-resolved.service.wants/resolvconf-pull-resolved.path → /lib/systemd/system/resolvconf-pull-resolved.path.
Unit /lib/systemd/system/resolvconf-pull-resolved.path is added as a dependency to a non-existent unit systemd-resolved.service.
Created symlink /etc/systemd/system/systemd-resolved.service.wants/resolvconf-pull-resolved.service → /lib/systemd/system/resolvconf-pull-resolved.service.
Unit /lib/systemd/system/resolvconf-pull-resolved.service is added as a dependency to a non-existent unit systemd-resolved.service.
Processing triggers for dbus (1.14.10-1~deb12u1) ...
Processing triggers for man-db (2.11.2-2) ...
Processing triggers for resolvconf (1.91+nmu1) ...
root@sub-zone-dns-server:~# systemctl start resolvconf.service
root@sub-zone-dns-server:~# systemctl enable resolvconf.service
Synchronizing state of resolvconf.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable resolvconf
```

Figure 19:

5.4 Configuring DNS Options

For the Root, TLD, and Sub Zone server, add/make sure the following items are listed in options in the ”/etc/bind/named.conf.options” file, comments (anything following //) can be ignored.

```
directory "/var/cache/bind";
dnssec-validation no;
```

Syntax for opening a file is shown below, usable text editors are vim and nano:

```
<txt editor> <file>
```

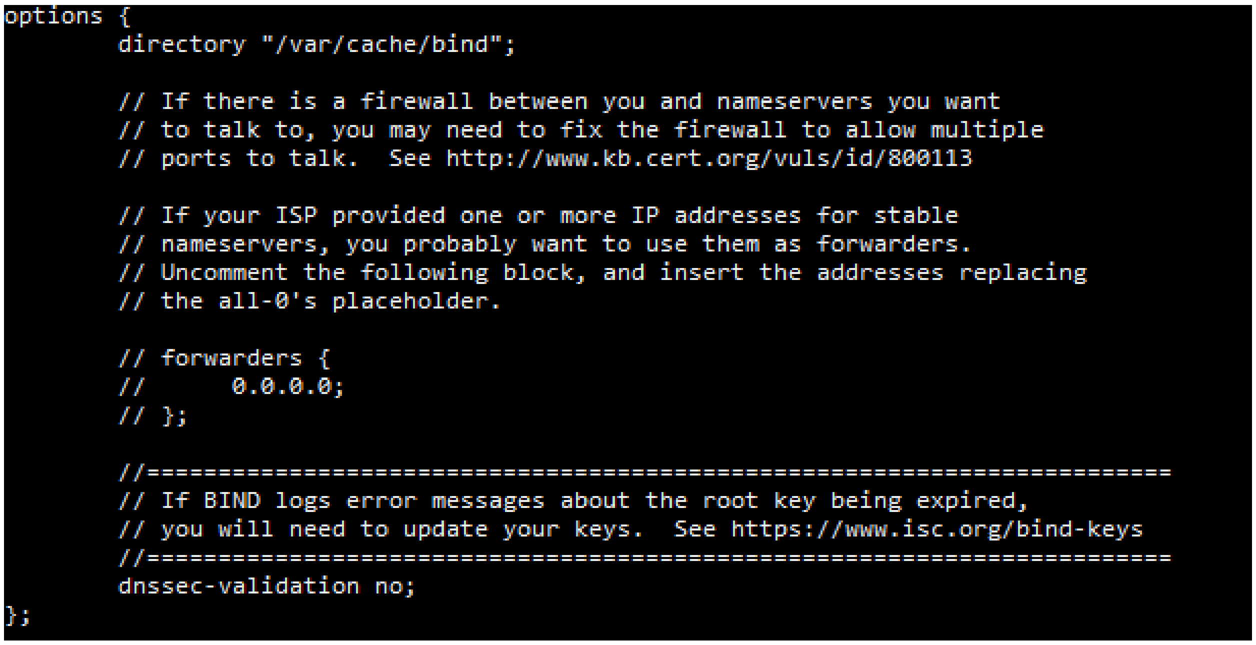
The first line is the location of the cache used by BIND, and the second is ensuring dnssec is turned off.

Next, for only the Root server, add the following line:

```
allow-recursion { any; };
```

This option allows the client to use the DNS server.

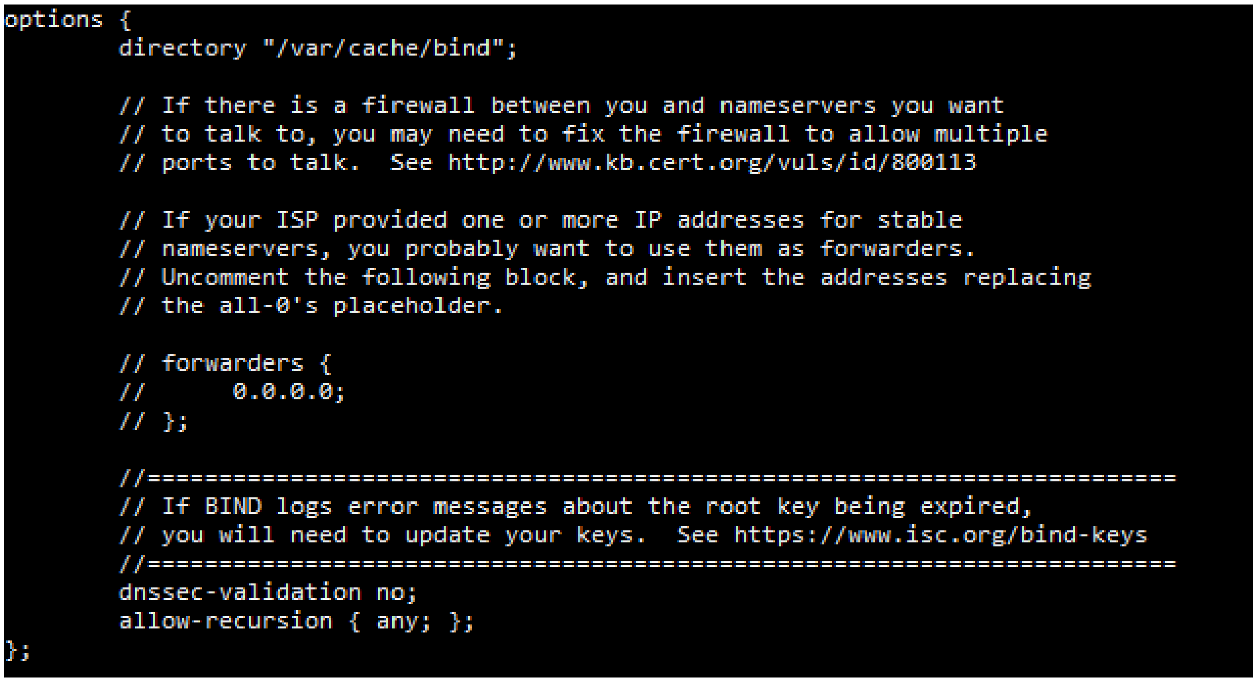
Base Configuration:



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

        // If there is a firewall between you and nameservers you want
        // to talk to, you may need to fix the firewall to allow multiple
        // ports to talk.  See http://www.kb.cert.org/vuls/id/800113

        // If your ISP provided one or more IP addresses for stable
        // nameservers, you probably want to use them as forwarders.
        // Uncomment the following block, and insert the addresses replacing
        // the all-0's placeholder.

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

        //========================================================================
        // If BIND logs error messages about the root key being expired,
        // you will need to update your keys.  See https://www.isc.org/bind-keys
        //========================================================================
        dnssec-validation no;
};
```

Figure 20:

Root Configuration:



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

        // If there is a firewall between you and nameservers you want
        // to talk to, you may need to fix the firewall to allow multiple
        // ports to talk.  See http://www.kb.cert.org/vuls/id/800113

        // If your ISP provided one or more IP addresses for stable
        // nameservers, you probably want to use them as forwarders.
        // Uncomment the following block, and insert the addresses replacing
        // the all-0's placeholder.

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

        //========================================================================
        // If BIND logs error messages about the root key being expired,
        // you will need to update your keys.  See https://www.isc.org/bind-keys
        //========================================================================
        dnssec-validation no;
        allow-recursion { any; };
};
```

Figure 21:

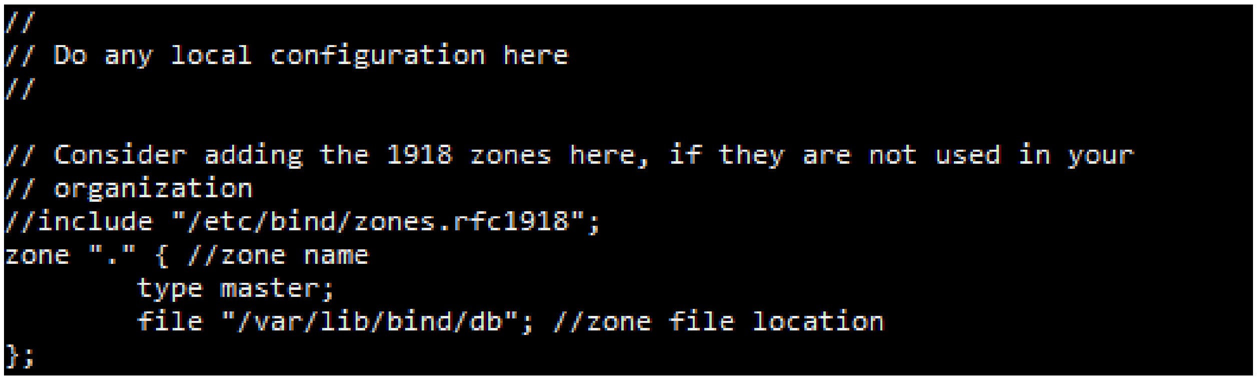
5.5 Configuring DNS Zones

The next step is to configure the zones themselves. The file to configure the zones is "/etc/bind/named.conf.local". Be sure to pay attention to the zone name and zone file location, the zone file has the configurations for the zone which is done in the next section.

See Section 1.4 for syntax for opening a file.

5.5.1 Root Server Zone Declaration

```
zone "." { //zone name
        type master;
        file "/var/lib/bind/db"; //zone file location
};
```

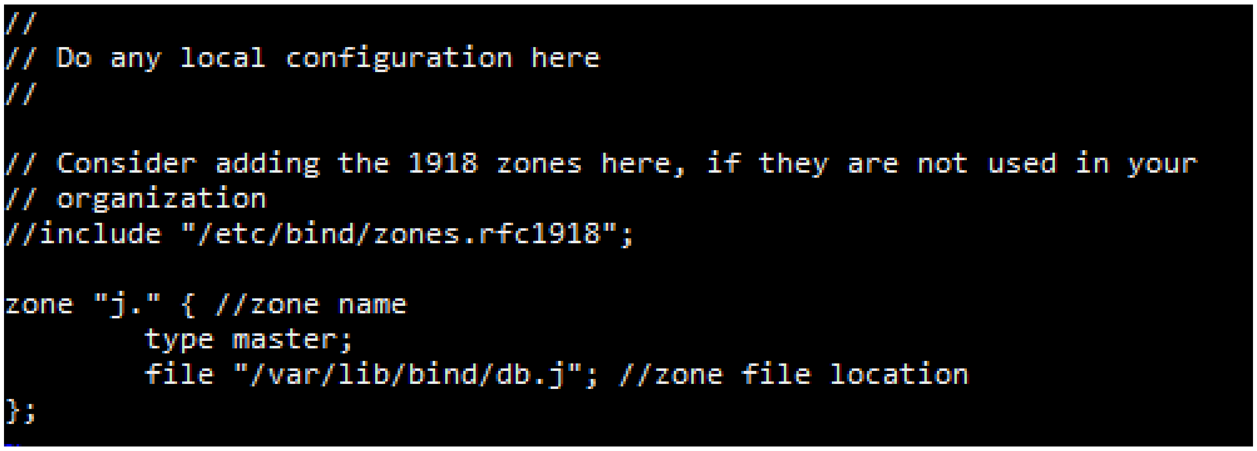


```
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";
zone "." { //zone name
        type master;
        file "/var/lib/bind/db"; //zone file location
};
```

Figure 22:

5.5.2 TLD Zone Declaration

```
zone "j." { //zone name
        type master;
        file "/var/lib/bind/db.j"; //zone file location
};
```

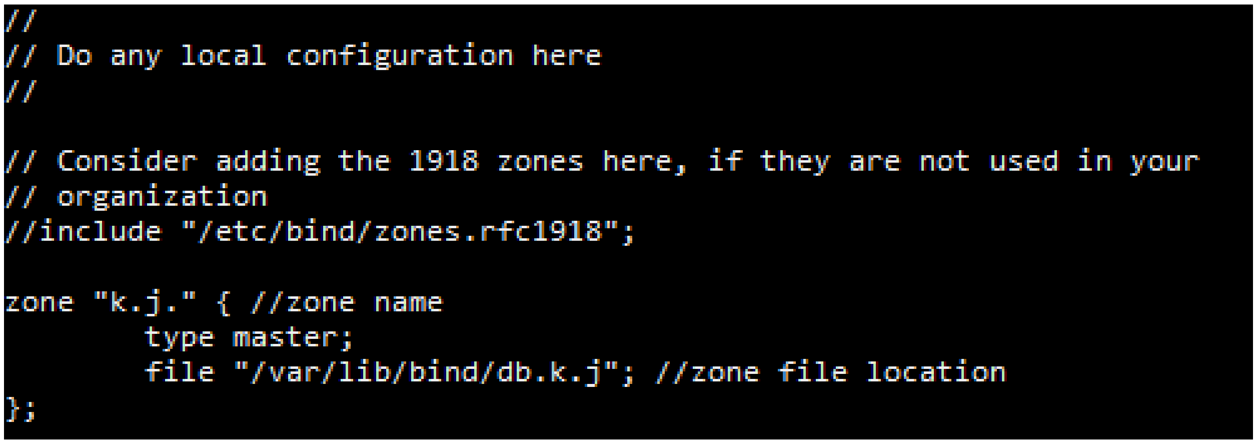


```
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone "j." { //zone name
        type master;
        file "/var/lib/bind/db.j"; //zone file location
};
```

Figure 23:

5.5.3 Sub Zone Declaration

```
zone "k.j." { //zone name
        type master;
        file "/var/lib/bind/db.k.j"; //zone file location
};
```



```
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone "k.j." { //zone name
        type master;
        file "/var/lib/bind/db.k.j"; //zone file location
};
```

Figure 24:

5.6 Configuring DNS Zone Files

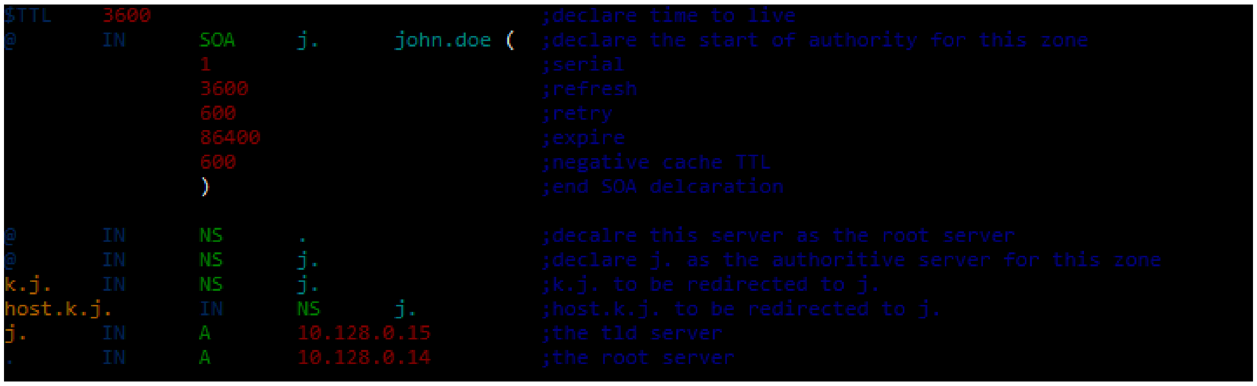
The files used throughout this section are created by the user, this can be done using touch, or it can be created when using a text editor. The location is decided by the zone configuration shown in the previous section.

See Section 1.4 for syntax for opening a file.

Pay attention to comments (any text following ;) to understand how the files are configured

5.6.1 Root Server

```
$TTL    3600                                    ;declare time to live
@       IN      SOA     j.      john.doe (      ;declare the start of authority for this zone
                1                               ;serial
                3600                            ;refresh
                600                             ;retry
                86400                           ;expire
                600                             ;negative cache TTL
                )                               ;end SOA delcaration

@       IN      NS      .                       ;decalre this server as the root server
@       IN      NS      j.                      ;declare j. as the authoritive server for this zone
k.j.    IN      NS      j.                      ;k.j. to be redirected to j.
host.k.j.       IN      NS      j.              ;host.k.j. to be redirected to j.
j.      IN      A       10.128.0.15             ;the tld server
.       IN      A       10.128.0.14             ;the root server
```

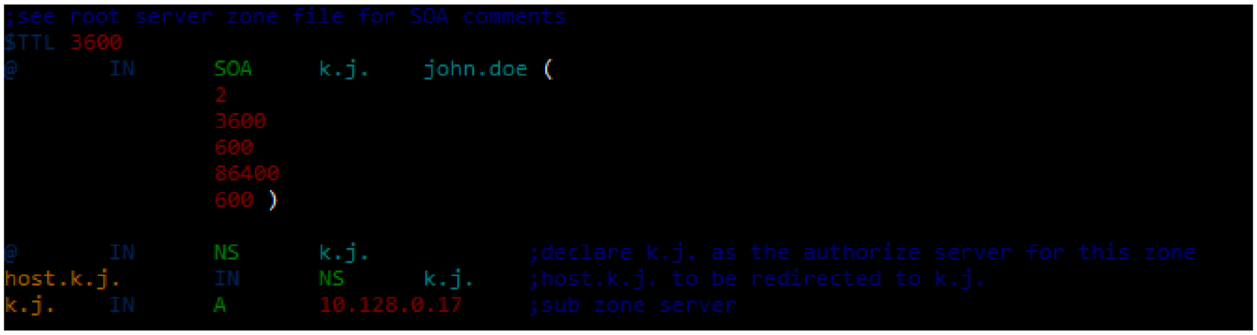


```
$TTL    3600                                    ;declare time to live
@       IN      SOA     j.      john.doe (      ;declare the start of authority for this zone
                1                               ;serial
                3600                            ;refresh
                600                             ;retry
                86400                           ;expire
                600                             ;negative cache TTL
                )                               ;end SOA delcaration

@       IN      NS      .                       ;decalre this server as the root server
@       IN      NS      j.                      ;declare j. as the authoritive server for this zone
k.j.    IN      NS      j.                      ;k.j. to be redirected to j.
host.k.j.       IN      NS      j.              ;host.k.j. to be redirected to j.
j.      IN      A       10.128.0.15             ;the tld server
.       IN      A       10.128.0.14             ;the root server
```

Figure 25:

5.6.2 TLD Server

```
;see root server zone file for SOA comments
$TTL 3600
@       IN      SOA     k.j.    john.doe (
                2
                3600
                600
                86400
                600 )

@       IN      NS      k.j.            ;declare k.j. as the authorize server for this zone
host.k.j.       IN      NS      k.j.    ;host.k.j. to be redirected to k.j.
k.j.    IN      A       10.128.0.17     ;sub zone server
```

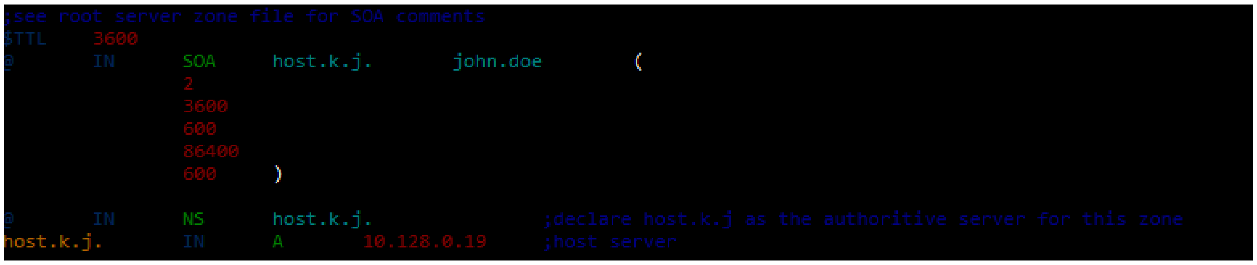


```
;see root server zone file for SOA comments
$TTL 3600
@       IN      SOA     k.j.    john.doe (
                2
                3600
                600
                86400
                600 )

@       IN      NS      k.j.            ;declare k.j. as the authorize server for this zone
host.k.j.       IN      NS      k.j.    ;host.k.j. to be redirected to k.j.
k.j.    IN      A       10.128.0.17     ;sub zone server
```

Figure 26:

5.6.3 Sub Zone Server

```
;see root server zone file for SOA comments
$TTL    3600
@       IN      SOA     host.k.j.       john.doe        (
                2
                3600
                600
                86400
                600     )

@       IN      NS      host.k.j.               ;declare host.k.j. as the authoritive server for this zone
host.k.j.       IN      A       10.128.0.19     ;host server
```



```
;see root server zone file for SOA comments
$TTL    3600
@       IN      SOA     host.k.j.       john.doe        (
                2
                3600
                600
                86400
                600     )

@       IN      NS      host.k.j.               ;declare host.k.j as the authoritive server for this zone
host.k.j.       IN      A       10.128.0.19     ;host server
```

Figure 27:

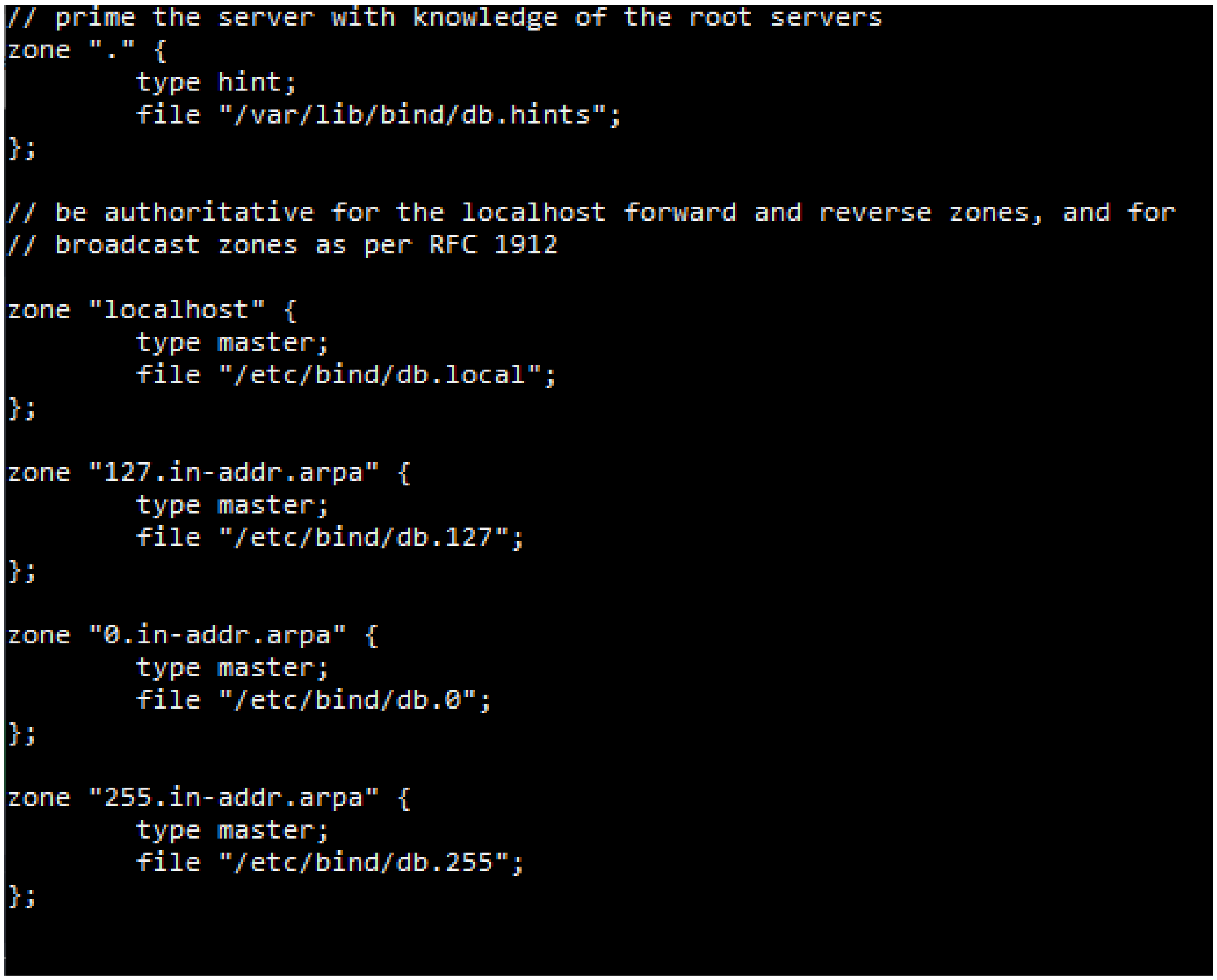
5.7 Adding Root Hints

For the TLD and Sub Zone file, open the "/etc/bind/named.conf.default-zones" (see section 1.4 for syntax opening a file) and change the file in the "." zone to your choosing.

Open that file and add the following configurations:

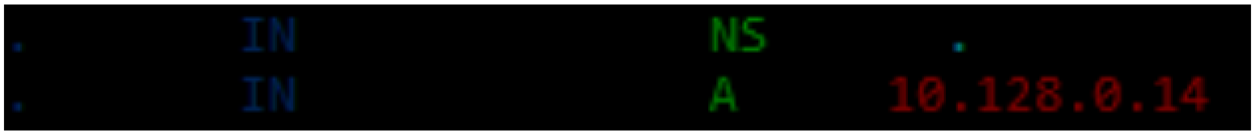
```
.       IN              NS      .
.       IN              A    <IP address of root>
```

This configures our TLD and Sub Zone server for root hints using the root server. Whenever a request is received from either server and it cannot be found, it will send the request to the root server.



```
// prime the server with knowledge of the root servers
zone "." {
        type hint;
        file "/var/lib/bind/db.hints";
};

// be authoritative for the localhost forward and reverse zones, and for
// broadcast zones as per RFC 1912

zone "localhost" {
        type master;
        file "/etc/bind/db.local";
};

zone "127.in-addr.arpa" {
        type master;
        file "/etc/bind/db.127";
};

zone "0.in-addr.arpa" {
        type master;
        file "/etc/bind/db.0";
};

zone "255.in-addr.arpa" {
        type master;
        file "/etc/bind/db.255";
};
```

Figure 28:



```
.       IN              NS      .
.       IN              A       10.128.0.14
```

Figure 29:

5.8 Activating BIND and /etc/resolv.conf

Make sure to do the following steps in this section for every server (except host) and client.

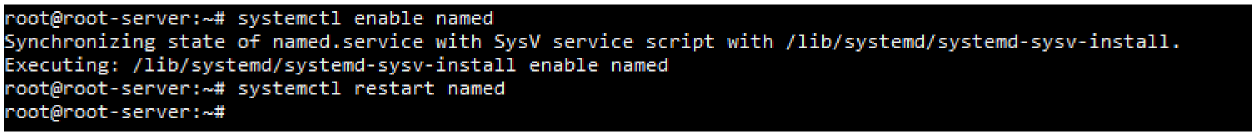
5.8.1 Activating BIND (exclude client)

In order to activate bind on startup, run the following command:

```
systemctl enable named
```

Next, to use it right now, restart the service by running the following command:

```
systemctl restart named
```



```
root@root-server:~# systemctl enable named
Synchronizing state of named.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable named
root@root-server:~# systemctl restart named
root@root-server:~#
```

Figure 30:

5.8.2 configure /etc/resolv.conf

Run the following line to enable the service:

```
systemctl enable resolvconf.service
```

Open the "/etc/resolvconf/resolv.conf.d/head" file.

See Section 1.4 for syntax for opening a file.

On the Root, TLD, and Sub Zone server, add the following line (we are making the name-server, localhost):

```
nameserver 127.0.0.1
```

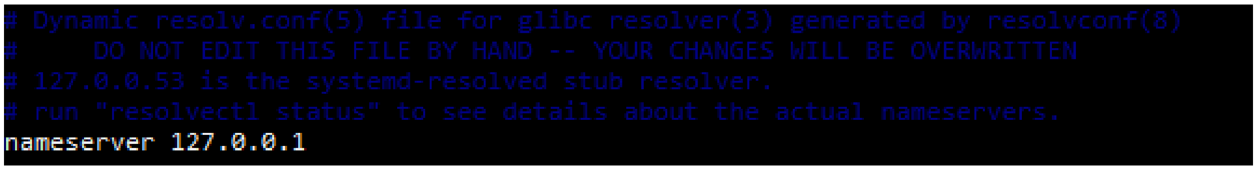
On the client, change the nameserver line to the IP address of the Root server.

Run the following line to start the resolve service on startup, then restart your system with the following command.

```
shutdown -r now
```

Now the /etc/resolv.conf file has your configurations, and will not change.

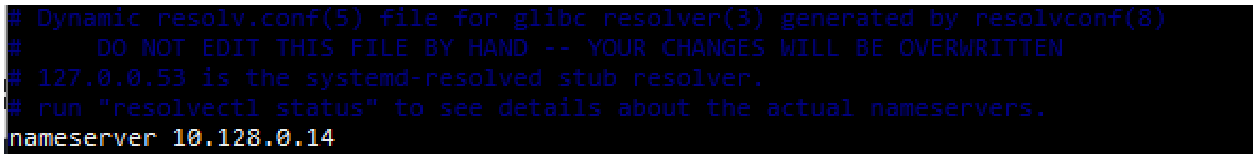
Server configuration:



```
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)
#     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
# 127.0.0.53 is the systemd-resolved stub resolver.
# run "resolvectl status" to see details about the actual nameservers.
nameserver 127.0.0.1
```

Figure 31:

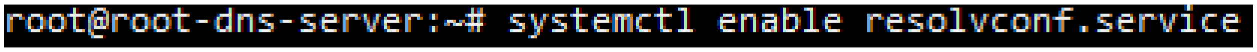
Client configuration:



```
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)
#     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
# 127.0.0.53 is the systemd-resolved stub resolver.
# run "resolvectl status" to see details about the actual nameservers.
nameserver 10.128.0.14
```

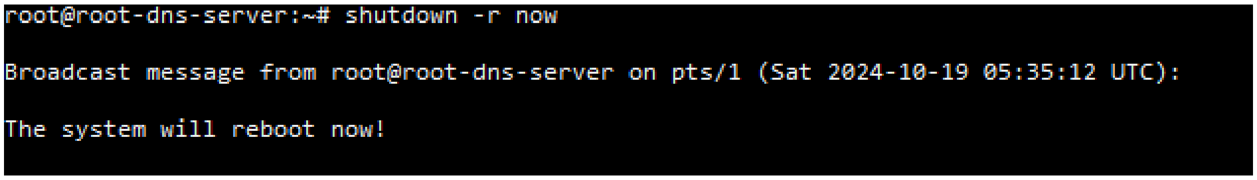
Figure 32:

Enabling the service:



```
root@root-dns-server:~# systemctl enable resolvconf.service
```

Figure 33:



```
root@root-dns-server:~# shutdown -r now

Broadcast message from root@root-dns-server on pts/1 (Sat 2024-10-19 05:35:12 UTC):

The system will reboot now!
```

Figure 34:

5.9 Testing and Debugging DNS

5.9.1 Debugging

Syntax errors in a zone file can be found by running the following command:

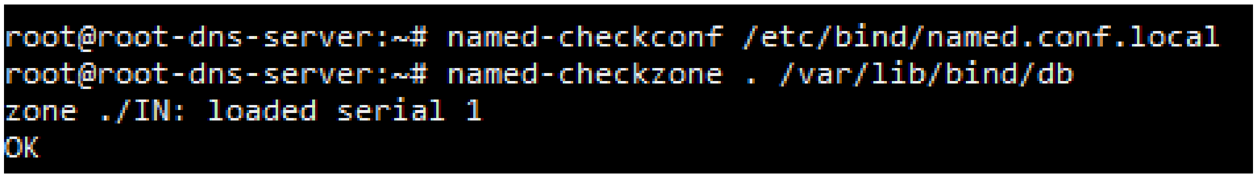
```
named-checkzone <zone> <zone file location>
```

Syntax error for the configuration file (local and options) can be found by running the following command:

```
named-checkconf <conf file location>
```

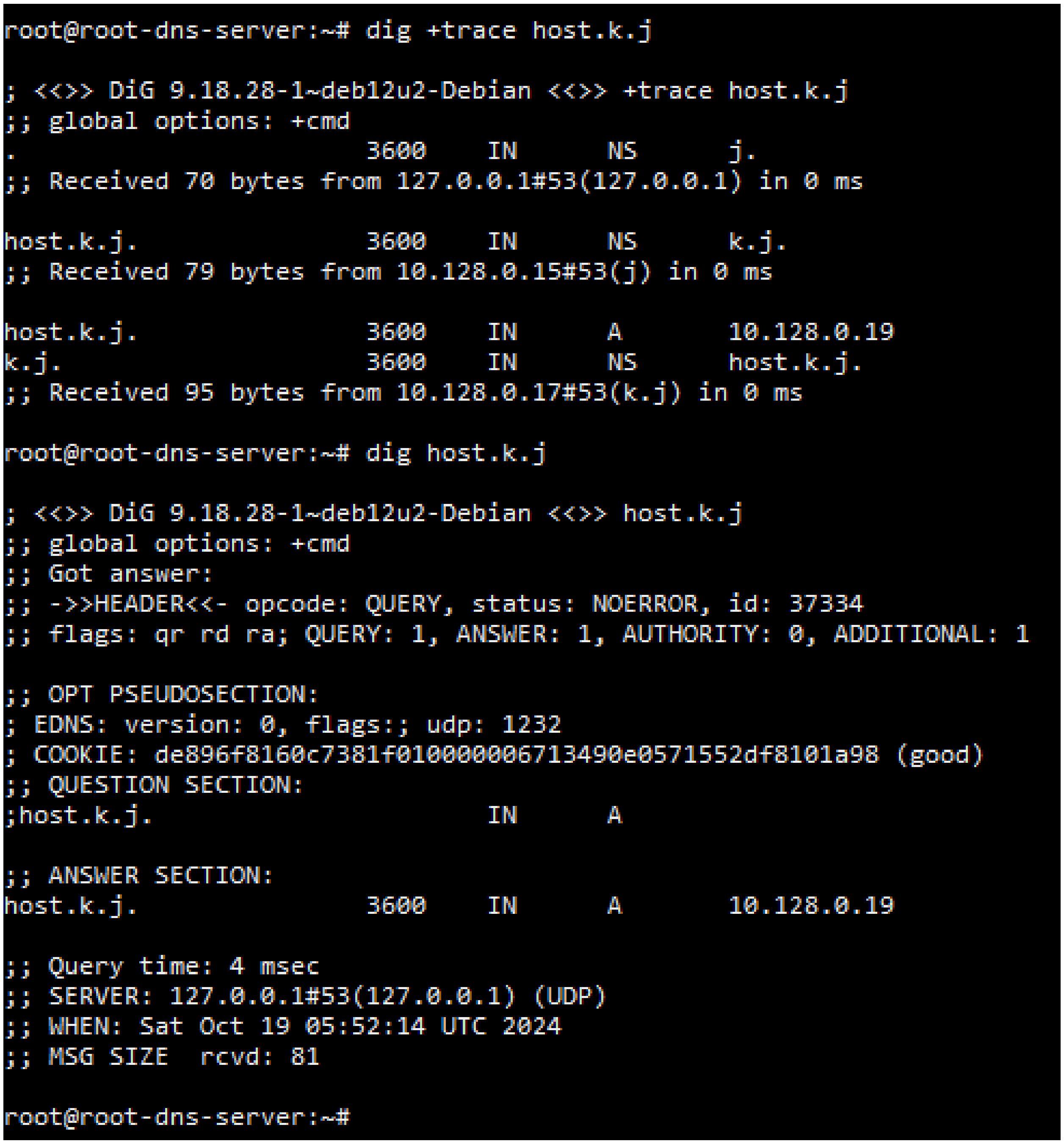
The dig command can be used for debugging, more information may be gathered when using +trace, the syntax is shown below:

```
dig <domain>
dig +trace <domain>
```



```
root@root-dns-server:~# named-checkconf /etc/bind/named.conf.local
root@root-dns-server:~# named-checkzone . /var/lib/bind/db
zone ./IN: loaded serial 1
OK
```

Figure 35:



```
root@root-dns-server:~# dig +trace host.k.j

; <<>> DiG 9.18.28-1~deb12u2-Debian <<>> +trace host.k.j
;; global options: +cmd
.                       3600    IN      NS      j.
;; Received 70 bytes from 127.0.0.1#53(127.0.0.1) in 0 ms

host.k.j.               3600    IN      NS      k.j.
;; Received 79 bytes from 10.128.0.15#53(j) in 0 ms

host.k.j.               3600    IN      A       10.128.0.19
k.j.                    3600    IN      NS      host.k.j.
;; Received 95 bytes from 10.128.0.17#53(k.j) in 0 ms

root@root-dns-server:~# dig host.k.j

; <<>> DiG 9.18.28-1~deb12u2-Debian <<>> host.k.j
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37334
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: de896f8160c7381f010000006713490e0571552df8101a98 (good)
;; QUESTION SECTION:
;host.k.j.                      IN      A

;; ANSWER SECTION:
host.k.j.               3600    IN      A       10.128.0.19

;; Query time: 4 msec
;; SERVER: 127.0.0.1#53(127.0.0.1) (UDP)
;; WHEN: Sat Oct 19 05:52:14 UTC 2024
;; MSG SIZE  rcvd: 81

root@root-dns-server:~#
```

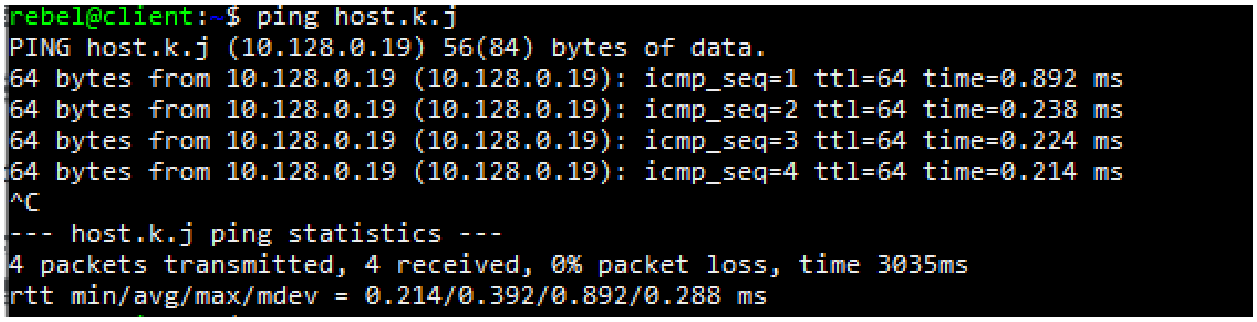
Figure 36:

5.9.2 Testing

From the client, try to ping the host: host.k.j, an example is shown below

```
ping host.k.j
```

If the ping is successful, the lab is finished.



```
rebel@client:~$ ping host.k.j
PING host.k.j (10.128.0.19) 56(84) bytes of data.
64 bytes from 10.128.0.19 (10.128.0.19): icmp_seq=1 ttl=64 time=0.892 ms
64 bytes from 10.128.0.19 (10.128.0.19): icmp_seq=2 ttl=64 time=0.238 ms
64 bytes from 10.128.0.19 (10.128.0.19): icmp_seq=3 ttl=64 time=0.224 ms
64 bytes from 10.128.0.19 (10.128.0.19): icmp_seq=4 ttl=64 time=0.214 ms
^C
--- host.k.j ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3035ms
rtt min/avg/max/mdev = 0.214/0.392/0.892/0.288 ms
```

Figure 37:

5.10 More Reading

For more details of how this works, check out the following links:

https://wiki.debian.org/Bind9

https://kb.isc.org/docs/aa-01309

https://www.tecmint.com/set-permanent-dns-nameservers-in-ubuntu-debian/

https://bind9.readthedocs.io/en/latest/reference.html#namedconf-statement-options