ASSIGNMENT 2 DNS, DHCP, and FTP Configuration Report

Prepared by: Guillermo Padilla Keymole

John Abbott College | Network Administration Program

Table of Figures (Captions)

Introduction

This report presents the installation, configuration, and validation of three essential network services: **DNS (Domain Name System)**, **DHCP (Dynamic Host Configuration Protocol)**, and **FTP (File Transfer Protocol)**, all performed on an AlmaLinux server and tested from an Ubuntu client. These services are fundamental to any networked environment, providing name resolution, dynamic IP address management, and file transfer capabilities.

The steps outlined in this report include detailed command execution, file configurations, service validation, and client testing. All relevant outputs are supported with clear screenshots and concise explanations to demonstrate the proper functioning of each service.

The assignment was completed individually using the same virtual machines from Assignment 1, ensuring a consistent and practical learning environment throughout the configuration process.

Topology

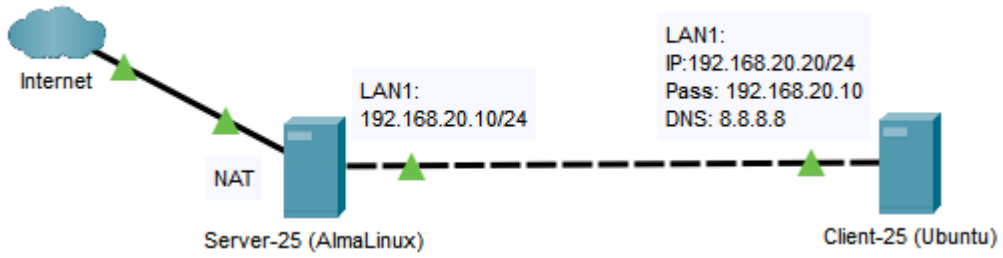

Figure 1 Assignment 2 network topology with IP, gateway, and DNS setup.

PART 1 – DNS Configuration & Validation

1.1 DNS Configuration

To begin the DNS service setup, I connected to the AlmaLinux server from my host machine using PowerShell with an SSH session:

```
PS C:\Users\Guill> ssh root@192.168.5.134
Welcome to Server-7. Authorized access only.
root@192.168.5.134's password:
Last login: Sat Apr  5 06:38:50 2025 from 192.168.5.1
[root@Server-7 ~]#
```

Figure 2 SSH connection from Windows PowerShell to the AlmaLinux server using root access.

Install Bind (DNS)

To get started with the DNS setup, I installed the bind package on my AlmaLinux server using dnf. I used the -y option so I wouldn't have to confirm each step manually. This installed BIND along with all the necessary dependencies so the server can handle DNS requests.

```
[root@Server-7 ~]# dnf install -y bind
AlmaLinux 9 - AppStream
AlmaLinux 9 - AppStream
AlmaLinux 9 - BaseOS
AlmaLinux 9 - BaseOS
AlmaLinux 9 - Extras
AlmaLinux 9 - Extras
Dependencies resolved.
================================================================
 Package                                        Architecture
================================================================
Installing:
 bind                                           x86_64
```

Figure 3 Installation of the BIND service and its dependencies using dnf install -y bind on AlmaLinux.

Configuring BIND to Listen on the Server and Accept Local Queries

To configure BIND to listen on the server IP and allow DNS queries from our local network, I edited the main configuration file. Inside the options block, I added the server's IP address 192.168.20.10 to the listen-on directive and specified the network 192.168.20.0/24 for allow-query, so that devices on the same subnet can make DNS requests

```
[root@Server-7 ~]# vim /etc/named.conf
[root@Server-7 ~]#
```

```
options {
        listen-on port 53 { 192.168.20.10; };
        listen-on-v6 port 53 { ::1; };
        directory       "/var/named";
        dump-file       "/var/named/data/cache_dump.db";
        statistics-file "/var/named/data/named_stats.txt";
        memstatistics-file "/var/named/data/named_mem_stats.txt";
        secroots-file   "/var/named/data/named.secroots";
        recursing-file  "/var/named/data/named.recursing";
        allow-query     { 192.168.20.0/24; };
```

Figure 4 Modified named.conf file to set the DNS server's IP and allow queries from the local 192.168.20.0/24 network

Checking the Syntax of the BIND Configuration

Before starting the DNS service, it's important to verify that the BIND configuration file is free of syntax errors. To do this, I used the named-checkconf utility, which checks the main /etc/named.conf file for any misconfigurations.

```
[root@Server-7 ~]# named-checkconf
[root@Server-7 ~]#
```

Figure 5 Verifying BIND configuration file syntax with named-checkconf

Forward Master DNS Zone Configuration

To define the DNS zone for the domain local.itmt.qc.ca, I edited the /etc/named.conf file and added a new zone entry. This designates the server as the primary DNS server (master) for the domain and specifies the location of the zone file.

```
zone "." IN {
        type hint;
        file "named.ca";
};

zone "local.itmt.qc.ca" IN {
        type master;
        file "/var/named/local.itmt.qc.ca.zone";
        allow-update { none; };
};

include "/etc/named.rfc1912.zones";
include "/etc/named.root.key";
```

Figure 6 Forward zone definition added to /etc/named.conf for local.itmt.qc.ca.

DNS Configuration File Syntax Recheck

After defining the forward master zone for local.itmt.qc.ca in /etc/named.conf, I ran the named-checkconf command again to ensure that no syntax errors were introduced.

```
[root@Server-7 ~]# named-checkconf
[root@Server-7 ~]#
```

Figure 7 Verifying the syntax of the BIND configuration file after adding the forward zone

Creating the Forward Zone File

To define the records for our DNS zone, I created and edited the forward master zone file using Vim. These entries ensure that the DNS server can resolve the required local hostnames to their respective Ips.

```
[root@Server-7 ~]# vim /var/named/local.itmt.qc.ca.zone
[root@Server-7 ~]#
```

```
$TTL 80000
@               IN      SOA     server-7.local.itmt.qc.ca. root.local.itmt.qc.ca. (
                                40      ; Serial
                                1D      ; Refresh
                                1H      ; Retry
                                1W      ; Expire
                                3H)     ; Minimum TTL

                IN      NS      server-7.local.itmt.qc.ca.
server-7        IN      A       192.168.20.10
client-7        IN      A       192.168.20.20
web-server      IN      A       192.168.20.2
email-server    IN      A       192.168.20.3
                IN      MX      20 email-server.local.itmt.qc.ca.
www             IN      CNAME   web-server.local.itmt.qc.ca.
```

Figure 8 Forward DNS zone file configuration for local.itmt.qc.ca zone on AlmaLinux.

Verifying Forward DNS Zone Syntax

To ensure that the forward DNS zone file was correctly configured, I used the named-checkzone command with the appropriate zone name and file path.

```
[root@Server-7 ~]# named-checkzone local.itmt.qc.ca /var/named/local.itmt.qc.ca.zone
zone local.itmt.qc.ca/IN: loaded serial 40
OK
[root@Server-7 ~]#
```

Figure 9 Successful forward DNS zone validation with named-checkzone.

Starting and Enabling the DNS Server

Once the zone file was verified successfully, I enabled and started the named DNS service using the following command. Then, I checked the service status to make sure it was active and running properly.

```
[root@Server-7 ~]# systemctl enable --now named
Created symlink /etc/systemd/system/multi-user.target.wants/named.service → /usr/lib/systemd/system/named.service.
[root@Server-7 ~]# systemctl status named
● named.service - Berkeley Internet Name Domain (DNS)
     Loaded: loaded (/usr/lib/systemd/system/named.service; enabled; preset: disabled)
     Active: active (running) since Wed 2025-04-09 20:06:40 EDT; 8s ago
    Process: 2710 ExecStartPre=/bin/bash -c if [ ! "$DISABLE_ZONE_CHECKING" == "yes" ]; then /usr/sbin/named-checkconf
    Process: 2714 ExecStart=/usr/sbin/named -u named -c ${NAMEDCONF} $OPTIONS (code=exited, status=0/SUCCESS)
   Main PID: 2715 (named)
```

Figure 10 The named (DNS) service is active and enabled successfully

Creating the Reverse DNS Zone

To support reverse DNS lookups for the 192.168.20.0/24 network, I edited the BIND configuration file /etc/named.conf and added a reverse master zone definition. This zone allows the DNS server to resolve IP addresses back to hostnames.

```
[root@Server-7 ~]# vim /etc/named.conf
[root@Server-7 ~]#
```

```
};

zone "20.168.192.in-addr.arpa" IN {
        type master;
        file "/var/named/local.itmt.qc.ca.zone.rev";
        allow-update { none; };
};

include "/etc/named.rfc1912.zones";
include "/etc/named.root.key";
```

Figure 11 Reverse zone definition added to the /etc/named.conf file.

Syntax Validation of BIND Configuration (Post Reverse Zone Edit)

After adding the reverse DNS zone to the /etc/named.conf file, I ran the named-checkconf command to verify that there were no syntax errors in the configuration.

```
[root@Server-7 ~]# named-checkconf
[root@Server-7 ~]#
```

Figure 12 Verifying named configuration syntax using named-checkconf after reverse zone definition.

Creating the Reverse DNS Zone File

To define reverse DNS mappings for the 192.168.20.0/24 network, I created the zone file /var/named/local.itmt.qc.ca.zone.rev. This file will allow PTR record resolution for hosts within our domain.

```
[root@Server-7 ~]# vim /var/named/local.itmt.qc.ca.zone.rev
[root@Server-7 ~]#
```

```
$TTL 80000
@       IN      SOA     server-7.local.itmt.qc.ca. root.local.itmt.qc.ca. (
                        40      ; Serial
                        1D      ; Refresh
                        1H      ; Retry
                        1W      ; Expire
                        3H )    ; Minimum

server-7.local.itmt.qc.ca.      IN      A       192.168.20.10
@       IN      NS      server-7.local.itmt.qc.ca.
10      IN      PTR     server-7.local.itmt.qc.ca.
20      IN      PTR     client-7.local.itmt.qc.ca.
4       IN      PTR     web-server.local.itmt.qc.ca.
5       IN      PTR     email-server.local.itmt.qc.ca.
```

Figure 13 Reverse DNS zone file created for the 192.168.20.0/24 network.

Note: I added an A record for server-7.local.itmt.qc.ca in the forward zone to avoid errors when validating the reverse zone file.

After creating the reverse zone file, I verified its syntax using the named-checkzone command

```
[root@Server-7 ~]# named-checkzone local.itmt.qc.ca /var/named/local.itmt.qc.ca.zone.rev
zone local.itmt.qc.ca/IN: loaded serial 40
OK
[root@Server-7 ~]#
```

Figure 14 Verifying reverse DNS zone file syntax with named-checkzone

Adjusting DNS Zone File Group Ownership

Before running the DNS service, I changed the group ownership of both zone files to named to ensure that the BIND service can access them without permission issues.

```
[root@Server-7 ~]# cd /var/named
[root@Server-7 named]# ls -la
total 28
drwxrwx--T.  5 root  named  189 Apr  9 21:03 .
drwxr-xr-x. 20 root  root  4096 Apr  9 18:58 ..
drwxrwx---.  2 named named   23 Apr  9 20:06 data
drwxrwx---.  2 named named   60 Apr  9 20:07 dynamic
-rw-r--r--.  1 root  root   396 Apr  9 19:51 local.itmt.qc.ca.zone
-rw-r--r--.  1 root  root   389 Apr  9 20:54 local.itmt.qc.ca.zone.rev
-rw-r-----.  1 root  named 2112 Feb 19 11:04 named.ca
-rw-r-----.  1 root  named  152 Feb 19 11:04 named.empty
-rw-r-----.  1 root  named  152 Feb 19 11:04 named.localhost
-rw-r-----.  1 root  named  168 Feb 19 11:04 named.loopback
drwxrwx---.  2 named named    6 Feb 19 11:04 slaves
[root@Server-7 named]#
```

Figure 15 Changing the group ownership of the zone files to named

I used the chgrp command followed by ls -la to confirm the changes

```
[root@Server-7 named]# chgrp named local*
[root@Server-7 named]# ls -la
total 28
drwxrwx--T.  5 root  named  189 Apr  9 21:03 .
drwxr-xr-x. 20 root  root  4096 Apr  9 18:58 ..
drwxrwx---.  2 named named   23 Apr  9 20:06 data
drwxrwx---.  2 named named   60 Apr  9 21:07 dynamic
-rw-r--r--.  1 root  named  396 Apr  9 19:51 local.itmt.qc.ca.zone
-rw-r--r--.  1 root  named  389 Apr  9 20:54 local.itmt.qc.ca.zone.rev
-rw-r-----.  1 root  named 2112 Feb 19 11:04 named.ca
-rw-r-----.  1 root  named  152 Feb 19 11:04 named.empty
-rw-r-----.  1 root  named  152 Feb 19 11:04 named.localhost
-rw-r-----.  1 root  named  168 Feb 19 11:04 named.loopback
drwxrwx---.  2 named named    6 Feb 19 11:04 slaves
[root@Server-7 named]#
```

Figure 16 Listing the zone files with updated group ownership

Restarting the DNS Service

After making changes to the zone files and permissions, I restarted the named service to apply the configuration updates. I then used systemctl status to confirm the service is active and running without errors.

```
[root@Server-7 named]# systemctl restart named
[root@Server-7 named]# systemctl status named
● named.service - Berkeley Internet Name Domain (DNS)
     Loaded: loaded (/usr/lib/systemd/system/named.service; enabled; preset: disabled)
     Active: active (running) since Wed 2025-04-09 21:12:47 EDT; 14s ago
    Process: 2821 ExecStartPre=/bin/bash -c if [ ! "$DISABLE_ZONE_CHECKING" == "yes" ];
    Process: 2823 ExecStart=/usr/sbin/named -u named -c ${NAMEDCONF} $OPTIONS (code=exi
   Main PID: 2824 (named)
```

Figure 17 Restarting the DNS service and confirming it is active and running

1.2 DNS Validation

On Client-7 (Ubuntu)

Preventing DNS Overwrites on Ubuntu

Before configuring DNS resolution manually on the Ubuntu client, I made sure that NetworkManager wouldn't automatically overwrite my changes. I used the nmcli command to disable automatic DNS updates for the LAN1 connection and then reloaded the connection to apply the change.

```
atohme@Client-7:~$ nmcli con mod LAN1 ipv4.ignore-auto-dns yes
atohme@Client-7:~$ nmcli con d LAN1 ; nmcli con u LAN1
Connection 'LAN1' successfully deactivated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnectio
n/1)
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/2)
atohme@Client-7:~$
```

Figure 18 Disabling auto DNS updates and reloading the LAN1 connection using nmcli.

DNS Resolver Configuration on the Ubuntu Client

To allow the Ubuntu client to properly resolve hostnames using the DNS server we configured, I edited the /etc/resolv.conf file. I added the IP address of the AlmaLinux DNS server (192.168.20.10) as the primary nameserver, Google's 8.8.8.8 as a fallback, and specified local.itmt.qc.ca as the search domain for local queries.

```
atohme@Client-7:~$ sudo vim /etc/resolv.conf
atohme@Client-7:~$
```

```
nameserver 192.168.20.10
nameserver 8.8.8.8
search local.itmt.qc.ca
~
```

Figure 19 Edited /etc/resolv.conf with the local DNS server, Google DNS fallback, and local search domain.

Forward DNS Lookup with nslookup

To validate the DNS server's forward resolution capabilities, I used the nslookup command on the Ubuntu client to test name-to-IP address resolution for multiple hostnames defined in the DNS zone.

```
atohme@Client-7:~$ nslookup server-7.local.itmt.qc.ca
Server:         192.168.20.10
Address:        192.168.20.10#53

Name:   server-7.local.itmt.qc.ca
Address: 192.168.20.10

atohme@Client-7:~$ nslookup client-7.local.itmt.qc.ca
Server:         192.168.20.10
Address:        192.168.20.10#53

Name:   client-7.local.itmt.qc.ca
Address: 192.168.20.20

atohme@Client-7:~$ nslookup email-server.local.itmt.qc.ca
Server:         192.168.20.10
Address:        192.168.20.10#53

Name:   email-server.local.itmt.qc.ca
Address: 192.168.20.3

atohme@Client-7:~$
```

Figure 20 Verifying forward DNS resolution of server-7, client-7, and email-server using nslookup on the Ubuntu client.

Testing Forward DNS Resolution with dig

To verify that forward DNS resolution works correctly, I used the dig command on the Ubuntu client. I queried the DNS server for the hostname server-7.local.itmt.qc.ca, and the response confirmed that it correctly maps to IP address 192.168.20.10.

```
atohme@Client-7:~$ dig server-7.local.itmt.qc.ca

; <<>> DiG 9.18.30-0ubuntu0.22.04.2-Ubuntu <<>> server-7.local.itmt.qc.ca
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 20555
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: ce11331601231c350100000067f72a00bf1a16488a51b75f (good)
;; QUESTION SECTION:
;server-7.local.itmt.qc.ca.     IN      A

;; ANSWER SECTION:
server-7.local.itmt.qc.ca. 80000 IN     A       192.168.20.10

;; Query time: 0 msec
;; SERVER: 192.168.20.10#53(192.168.20.10) (UDP)
;; WHEN: Wed Apr 09 22:16:32 EDT 2025
;; MSG SIZE  rcvd: 98
```

Figure 21 dig command resolving server-7.local.itmt.qc.ca to its IP address

Testing Reverse DNS Resolution with dig -x

Next, I tested reverse DNS resolution using the dig -x command. This confirmed that the IP address 192.168.20.10 successfully resolves back to server-7.local.itmt.qc.ca, proving that the reverse zone is functioning properly.

```
atohme@Client-7:~$ dig -x 192.168.20.10

; <<>> DiG 9.18.30-0ubuntu0.22.04.2-Ubuntu <<>> -x 192.168.20.10
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 13537
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: ae9352eed40fc2310100000067f72a24e3a47da996b362a3 (good)
;; QUESTION SECTION:
;10.20.168.192.in-addr.arpa.    IN      PTR

;; ANSWER SECTION:
10.20.168.192.in-addr.arpa. 80000 IN    PTR     server-7.local.itmt.qc.ca.

;; Query time: 0 msec
;; SERVER: 192.168.20.10#53(192.168.20.10) (UDP)
;; WHEN: Wed Apr 09 22:17:08 EDT 2025
;; MSG SIZE  rcvd: 122
```

Figure 22 dig -x command resolving 192.168.20.10 back to server-7.local.itmt.qc.ca

Testing Mail Exchange (MX) Record Resolution

To test if the DNS server is properly handling mail exchange (MX) records, I used the dig command from the Ubuntu client to query the domain local.itmt.qc.ca. This test confirms that our DNS server is correctly configured to provide mail routing information.

```
atohme@Client-7:~$ dig local.itmt.qc.ca MX

; <<>> DiG 9.18.30-0ubuntu0.22.04.2-Ubuntu <<>> local.itmt.qc.ca MX
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21568
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 2227563eee0a57b00100000067f7350782726584e7d6bf9a (good)
;; QUESTION SECTION:
;local.itmt.qc.ca.              IN      MX

;; ANSWER SECTION:
local.itmt.qc.ca.       80000   IN      MX      20 email-server.local.itmt.qc.ca.

;; ADDITIONAL SECTION:
email-server.local.itmt.qc.ca. 80000 IN A       192.168.20.3

;; Query time: 0 msec
;; SERVER: 192.168.20.10#53(192.168.20.10) (UDP)
;; WHEN: Thu Apr 10 14:42:48 EDT 2025
;; MSG SIZE  rcvd: 118
```

Figure 23 Successful MX record lookup for local.itmt.qc.ca using dig from Ubuntu client

Checking All DNS Records from the Client

To make sure that all the DNS entries I added to the forward zone were working, I used the host -l command on the Ubuntu client. This command lists all the DNS records for the domain local.itmt.qc.ca. It's a quick way to check if the DNS server is properly configured and responding with the right information.

```
atohme@Client-7:~$ host -l local.itmt.qc.ca
local.itmt.qc.ca name server server-7.local.itmt.qc.ca.
client-7.local.itmt.qc.ca has address 192.168.20.20
email-server.local.itmt.qc.ca has address 192.168.20.3
server-7.local.itmt.qc.ca has address 192.168.20.10
web-server.local.itmt.qc.ca has address 192.168.20.2
atohme@Client-7:~$
```

Figure 24 Output of host -l local.itmt.qc.ca showing all DNS records from the client

Verifying DNS Resolution with Ping

To make sure that my DNS setup was actually working in practice, I ran a ping test from the Ubuntu client to the server using its fully qualified domain name: server-7.local.itmt.qc.ca. This was to check both name resolution and connectivity.

```
atohme@Client-7:~$ ping -c 4 server-7.local.itmt.qc.ca
PING server-7.local.itmt.qc.ca (192.168.20.10) 56(84) bytes of data.
64 bytes from server-7.local.itmt.qc.ca (192.168.20.10): icmp_seq=1 ttl=64 time=0.444 ms
64 bytes from server-7.local.itmt.qc.ca (192.168.20.10): icmp_seq=2 ttl=64 time=0.320 ms
64 bytes from server-7.local.itmt.qc.ca (192.168.20.10): icmp_seq=3 ttl=64 time=1.05 ms
64 bytes from server-7.local.itmt.qc.ca (192.168.20.10): icmp_seq=4 ttl=64 time=0.384 ms

--- server-7.local.itmt.qc.ca ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3041ms
rtt min/avg/max/mdev = 0.320/0.550/1.053/0.293 ms
atohme@Client-7:~$
```

Figure 25 Successful ping from Ubuntu client to DNS server

PART 2 – DHCP Configuration & Validation

2.1 DHCP Configuration

On Server-7(AlmaLinux)

Installing the DHCP Server

To begin configuring dynamic IP address assignment, I installed the dhcp-server package on my AlmaLinux server using the dnf command. I included the -y flag so I wouldn't have to confirm the installation manually.

```
[root@Server-7 ~]# dnf -y install dhcp-server
Last metadata expiration check: 0:34:46 ago on Wed Apr  9 22:57:18 2025.
Dependencies resolved.
================================================================================
 Package                         Architecture                    Version
================================================================================
Installing:
 dhcp-server                     x86_64                          12:4.4.2-19.b1.el9
Installing dependencies:
 dhcp-common                     noarch                          12:4.4.2-19.b1.el9

Transaction Summary
```

Figure 26 Installing the DHCP server (dhcp-server) on AlmaLinux

DHCP Configuration File Location

After installing the DHCP package, I verified the location of the configuration file. The main file used for defining DHCP settings is /etc/dhcp/dhcpd.conf, which I opened using Vim for editing.

```
[root@Server-7 ~]# tree /etc/dhcp/
/etc/dhcp/
├── dhclient.d
│   └── chrony.sh
├── dhcpd.conf
└── dhcpd6.conf

1 directory, 3 files
[root@Server-7 ~]# vim /etc/dhcp/dhcpd.conf
```

Figure 27 Locating and opening the DHCP configuration file

Editing the DHCP Server Configuration

To set up the DHCP server with the right parameters, I edited the /etc/dhcp/dhcpd.conf file. I configured it to assign IP addresses from 192.168.20.200 to 192.168.20.220, set the default gateway to 192.168.20.10, and included two DNS servers: the local DNS server and Google's 8.8.8.8. I also set the lease duration to 1 day (86400 seconds).

```
subnet 192.168.20.0 netmask 255.255.255.0 {
  range 192.168.20.200 192.168.20.220;
  option routers 192.168.20.10;
option domain-name "local.itmt.qc.ca";
option domain-name-servers 192.168.20.10, 8.8.8.8;
default-lease-time 86400;
max-lease-time 86400;
}
```

Figure 28 DHCP configuration for subnet 192.168.20.0/24 in /etc/dhcp/dhcpd.conf

Validating the DHCP Configuration

Before starting the DHCP service, I made sure the configuration file had no syntax errors by using the dhcpd -cf /etc/dhcp/dhcpd.conf command.

```
[root@Server-7 ~]# dhcpd -cf /etc/dhcp/dhcpd.conf
Internet Systems Consortium DHCP Server 4.4.2b1
Copyright 2004-2019 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/
ldap_gssapi_principal is not set,GSSAPI Authentication for LDAP will not be used
Not searching LDAP since ldap-server, ldap-port and ldap-base-dn were not specified in the config file
Config file: /etc/dhcp/dhcpd.conf
Database file: /var/lib/dhcpd/dhcpd.leases
PID file: /var/run/dhcpd.pid
Source compiled to use binary-leases
Wrote 0 leases to leases file.
Listening on LPF/ens192/00:0c:29:46:51:dc/192.168.20.0/24
Sending on   LPF/ens192/00:0c:29:46:51:dc/192.168.20.0/24

No subnet declaration for ens160 (192.168.5.134).
** Ignoring requests on ens160.  If this is not what
   you want, please write a subnet declaration
   in your dhcpd.conf file for the network segment
   to which interface ens160 is attached. **

Sending on   Socket/fallback/fallback-net
[root@Server-7 ~]#
```

Figure 29 Syntax check for /etc/dhcp/dhcpd.conf using dhcpd -cf

Starting and Enabling the DHCP Service

After verifying the configuration, I started and enabled the DHCP service using the systemctl enable --now dhcpd command. This made sure the service launched immediately and would also start automatically on boot. I then confirmed it was active and running by checking its status.

```
[root@Server-7 ~]# systemctl enable --now dhcpd
Created symlink /etc/systemd/system/multi-user.target.wants/dhcpd.service → /usr/lib/systemd/system/dhcpd.service.
[root@Server-7 ~]# systemctl status dhcpd
● dhcpd.service - DHCPv4 Server Daemon
     Loaded: loaded (/usr/lib/systemd/system/dhcpd.service; enabled; preset: disabled)
     Active: active (running) since Wed 2025-04-09 23:54:13 EDT; 6s ago
```

Figure 30 DHCP service successfully started and enabled on AlmaLinux using systemctl

2.2 DHCP Validation

On Client-7(Ubuntu)

Testing DHCP Functionality

To test if the DHCP server was correctly assigning network settings to clients, I first set the connection method back to automatic on the Ubuntu client. This ensures the system would request its IP settings from the DHCP server on the network.

```
atohme@Client-7:~$ nmcli con mod LAN1 ipv4.method auto
atohme@Client-7:~$ nmcli con down LAN1 ; nmcli con up LAN1
Connection 'LAN1' successfully deactivated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnectio
n/1)
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/2)
```

Figure 31 Switching to DHCP method and restarting connection

Next, I ran the nmcli command to verify the new configuration details received via DHCP.

```
atohme@Client-7:~$ nmcli
ens33: connected to LAN1
        "Intel 82545EM"
        ethernet (e1000), 00:0C:29:E5:8B:FE, hw, mtu 1500
        ip4 default
        inet4 192.168.20.200/24
        inet4 192.168.20.20/24
        route4 192.168.20.0/24 metric 100
        route4 192.168.20.0/24 metric 100
        route4 169.254.0.0/16 metric 1000
        route4 default via 192.168.20.10 metric 100
        inet6 fe80::451d:6e3f:5ff1:47b8/64
        route6 fe80::/64 metric 1024

lo: unmanaged
        "lo"
        loopback (unknown), 00:00:00:00:00:00, sw, mtu 65536

DNS configuration:
        servers: 192.168.20.10 8.8.8.8
        domains: local.itmt.qc.ca
        interface: ens33
```

Figure 32 IP, DNS, and domain information received from DHCP

Verifying Active DHCP Lease on the Server

To make sure the DHCP server had successfully assigned an IP address to my Ubuntu client, I checked the DHCP lease file located at /var/lib/dhcpd/dhcpd.leases on the AlmaLinux server.

```
[root@Server-7 ~]# cat /var/lib/dhcpd/dhcpd.leases
# The format of this file is documented in the dhcpd.leases(5) manual page.
# This lease file was written by isc-dhcp-4.4.2b1

# authoring-byte-order entry is generated, DO NOT DELETE
authoring-byte-order little-endian;

server-duid "\000\001\000\001/\211\374\223\000\014)FQ\334";

lease 192.168.20.200 {
  starts 4 2025/04/10 04:08:10;
  ends 5 2025/04/11 04:08:10;
  cltt 4 2025/04/10 04:08:10;
  binding state active;
  next binding state free;
  rewind binding state free;
  hardware ethernet 00:0c:29:e5:8b:fe;
  uid "\001\000\014)\345\213\376";
  client-hostname "Client-7";
}
[root@Server-7 ~]#
```

Figure 33 DHCP lease file showing client lease information

PART 3 – FTP Configuration & Validation

3.1 FTP Configuration

On Server-7(AlmaLinux)

Installing the FTP Server (vsftpd)

To begin configuring the FTP service, I installed the vsftpd (Very Secure FTP Daemon) package on my AlmaLinux server using dnf. The -y flag was included to automatically confirm the installation prompt.

```
[root@Server-7 ~]# dnf install -y vsftpd
Last metadata expiration check: 0:45:42 ago on Thu Apr 10 01:59:37 2025.
Dependencies resolved.
================================================================================
 Package                    Architecture                Version
================================================================================
Installing:
 vsftpd                     x86_64                      3.0.5-6.el9

Transaction Summary
================================================================================
Install  1 Package

Total download size: 157 k
Installed size: 347 k
Downloading Packages:
vsftpd-3.0.5-6.el9.x86_64.rpm
```

Figure 34 Installation of the vsftpd FTP server on AlmaLinux.

Enabling Anonymous FTP Access on AlmaLinux

To allow anonymous users to connect and upload files to the FTP server, I edited the /etc/vsftpd/vsftpd.conf file and made the following changes, these changes allow users to connect without a username and upload files to writable directories. I uncommented and added the following parameters:

- **anonymous_enable=YES**
  This allows users to connect to the FTP server without needing a username or password.
- **write_enable=YES**
  This lets any connected user (including anonymous users) perform actions like uploading files or creating folders.
- **anon_upload_enable=YES**
  This allows anonymous users to upload files to directories where they have write permissions.

- **anon_mkdir_write_enable=YES**
  With this setting, anonymous users can also create new directories on the FTP server.

```
[root@Server-7 ~]# vim /etc/vsftpd/vsftpd.conf
[root@Server-7 ~]#
```

```
# Allow anonymous FTP? (Beware - allowed by default if you comment this out).
anonymous_enable=YES
#
# Uncomment this to allow local users to log in.
local_enable=YES
#
# Uncomment this to enable any form of FTP write command.
write_enable=YES
#
# Default umask for local users is 077. You may wish to change this to 022,
# if your users expect that (022 is used by most other ftpd's)
local_umask=022
#
# Uncomment this to allow the anonymous FTP user to upload files. This only
# has an effect if the above global write enable is activated. Also, you will
# obviously need to create a directory writable by the FTP user.
# When SELinux is enforcing check for SE bool allow_ftpd_anon_write, allow_ftpd_full_access
anon_upload_enable=YES
#
# Uncomment this if you want the anonymous FTP user to be able to create
# new directories.
anon_mkdir_write_enable=YES
#
# Activate directory messages - messages given to remote users when they
# go into a certain directory.
dirmessage_enable=YES
#
# Activate logging of uploads/downloads.
xferlog_enable=YES
#
# Make sure PORT transfer connections originate from port 20 (ftp-data).
connect_from_port_20=YES
#
# If you want, you can arrange for uploaded anonymous files to be owned by
# a different user. Note! Using "root" for uploaded files is not
# recommended!
#chown_uploads=YES
#chown_username=whoever
#
# You may override where the log file goes if you like. The default is shown
# below.
#xferlog_file=/var/log/xferlog
#
# If you want, you can have your log file in standard ftpd xferlog format.
# Note that the default log file location is /var/log/xferlog in this case.
xferlog_std_format=YES
#
# You may change the default value for timing out an idle session.
#idle_session_timeout=600
```

Figure 35 Configuration changes to /etc/vsftpd/vsftpd.conf enabling anonymous FTP access and uploads.

Allowing FTP Through the Firewall

To allow FTP access between the server and client, I added the FTP service to the nm-shared firewall zone and reloaded the firewall. I then verified that the FTP service was correctly listed.

```
[root@Server-7 ~]# firewall-cmd --permanent --add-service=ftp --zone=nm-shared
success
[root@Server-7 ~]# firewall-cmd --reload
success
[root@Server-7 ~]# firewall-cmd --list-services --zone=nm-shared
dhcp dns ftp mountd nfs rpc-bind samba ssh
[root@Server-7 ~]#
```

Figure 36 FTP service successfully added to the firewall in the nm-shared zone and verified.

Changing the Owner of the FTP Public Directory

To allow the anonymous FTP user to upload files, I changed the ownership of the /var/ftp/pub directory to the ftp account. This is required so that files can be written by anonymous users.

```
[root@Server-7 ~]# cd /var/ftp/pub
[root@Server-7 pub]# ls -ld /var/ftp/pub
drwxr-xr-x. 2 root root 6 Oct  2  2024 /var/ftp/pub
[root@Server-7 pub]# chown -R ftp /var/ftp/pub
[root@Server-7 pub]# ls -ld /var/ftp/pub
drwxr-xr-x. 2 ftp root 6 Oct  2  2024 /var/ftp/pub
[root@Server-7 pub]#
```

Figure 37 Ownership of the /var/ftp/pub directory was changed from root to ftp to allow anonymous FTP uploads.

Enabling SELinux Write Access for Anonymous FTP Uploads

To allow anonymous users to upload files to the /var/ftp/pub directory under SELinux, I assigned the correct security context using the chcon command. This step is necessary when SELinux is enforcing access control.

```
[root@Server-7 pub]# chcon -R -t public_content_rw_t /var/ftp/pub
[root@Server-7 pub]#
```

Figure 38 SELinux context set for anonymous FTP write access.

Starting and Enabling the FTP Service

To make the FTP service available and persistent across reboots, I enabled and started the vsftpd service using systemctl. I then verified that the service is active and running.

```
[root@Server-7 pub]# systemctl enable --now vsftpd
Created symlink /etc/systemd/system/multi-user.target.wants/vsftpd.service → /usr/lib/systemd/system/vsftpd.service.
[root@Server-7 pub]# systemctl status vsftpd
● vsftpd.service - Vsftpd ftp daemon
     Loaded: loaded (/usr/lib/systemd/system/vsftpd.service; enabled; preset: disabled)
     Active: active (running) since Thu 2025-04-10 04:40:01 EDT; 9s ago
    Process: 4142 ExecStart=/usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf (code=exited, status=0/SUCCESS)
   Main PID: 4143 (vsftpd)
```

Figure 39 FTP service successfully enabled and running on AlmaLinux.

SELinux Configuration for FTP Uploads

To allow anonymous FTP users to upload files, we need to change a specific SELinux boolean setting. This ensures SELinux permits write operations from the anonymous user.

```
[root@Server-7 pub]# setsebool -P allow_ftpd_anon_write=1
[root@Server-7 pub]#
```

Figure 40 Enabling SELinux boolean to allow anonymous FTP uploads permanently.

Part of Command	Description
setsebool	Command used to change SELinux boolean values
-P	Makes the change persistent across reboots
allow_ftpd_anon_write=1	Enables write access for anonymous users via the FTP server (vsftpd)

Note. Even if you set correct permissions and context (like with chcon), SELinux will still block uploads from anonymous users unless this setting is turned on.

3.2 FTP Validation

On Client-7(Ubuntu)

FTP Upload Test from Ubuntu Client

To validate the functionality of the FTP server, I used the command-line ftp client on my Ubuntu machine to perform an anonymous login and upload a test file.

- First, I created a file named file.txt using the touch command.
- Then, I connected to the FTP server at 192.168.20.10 using the ftp command and logged in as an anonymous user.
- After successfully logging in, I navigated to the pub directory using the cd pub command.
- I used put file.txt to upload the file to the FTP server.
- The transfer completed successfully, confirming that anonymous uploads are now permitted.

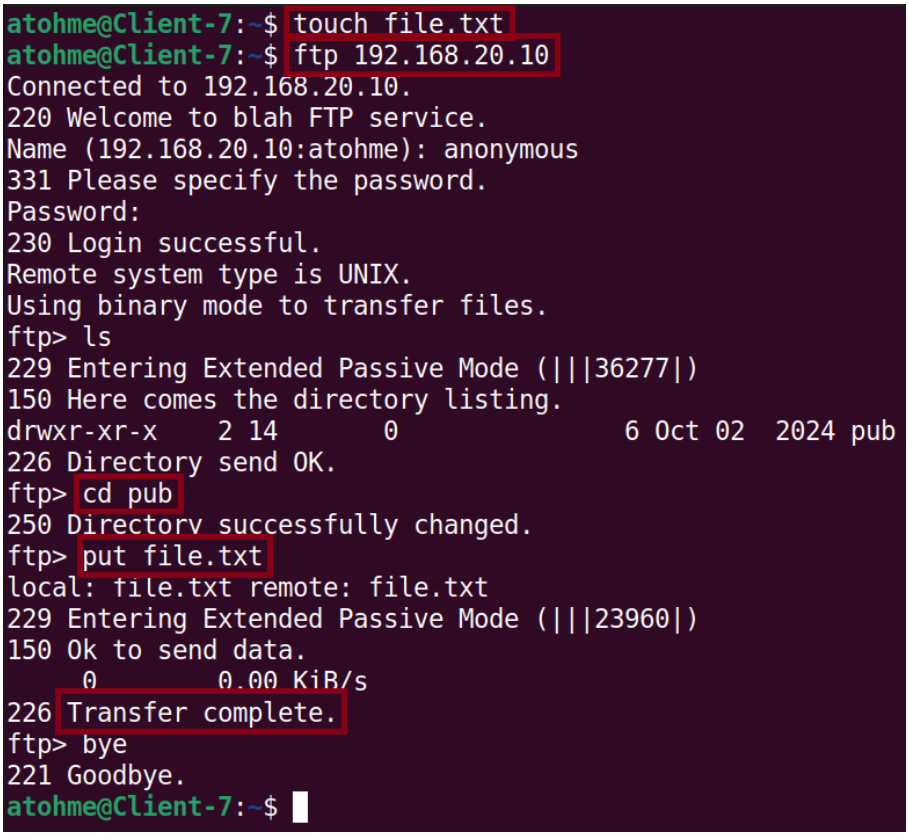

Figure 41 Successful upload of file.txt via the command-line FTP client.

FTP Server-7 Validation

To confirm the file transfer and verify the FTP service is working correctly:

- I logged into the FTP server and navigated to /var/ftp/pub, where I verified the uploaded file file.txt is present with the correct ownership (ftp:ftp) and permissions.
- I also ran netstat -tunap | grep ftp to check the FTP server is actively listening on port 21 and that an established session exists, confirming a successful FTP connection from the client.

```
[root@Server-7 ~]# cd /var/ftp/pub
[root@Server-7 pub]# ls -l
total 0
-rw-------. 1 ftp ftp 0 Apr 10 05:18 file.txt
[root@Server-7 pub]# netstat -tunap | grep ftp
tcp6       0      0 :::21                   :::*                    LISTEN      4143/vsftpd
tcp6       0      0 192.168.20.10:21        192.168.20.20:59420     ESTABLISHED 4379/vsftpd
[root@Server-7 pub]#
```

Figure 42 Verification of file upload and active FTP session on the AlmaLinux server.

FileZilla Installation on Client-7 (Ubuntu)

To perform GUI-based FTP testing, I installed the FileZilla client on my Ubuntu machine using the following commands:

```
atohme@Client-7:~$ sudo apt update
[sudo] password for atohme:
Hit:1 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:2 https://dl.google.com/linux/chrome/deb stable InRelease
Hit:3 http://ca.archive.ubuntu.com/ubuntu jammy InRelease
Hit:4 http://ca.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:5 http://ca.archive.ubuntu.com/ubuntu jammy-backports InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
36 packages can be upgraded. Run 'apt list --upgradable' to see them.
atohme@Client-7:~$ sudo apt install filezilla -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

Figure 43 Installing FileZilla using apt on Ubuntu

Connecting to the FTP Server Using FileZilla

To validate FTP server access using a graphical interface, I installed and launched FileZilla on my Ubuntu client. I opened FileZilla from the terminal using the following command:

```
atohme@Client-7:~$ filezilla
Reading locale option from /home/atohme/.config/filezilla/filezilla.xml
```

Figure 44 Launching FileZilla from the terminal.

I entered the FTP server details in the Quickconnect bar. Then, I clicked on **Quickconnect** to initiate the session:

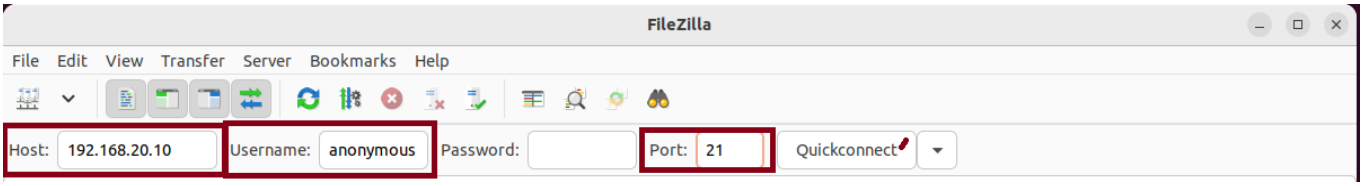

Figure 45 Entering FTP server connection details in FileZilla.

Navigating the FTP Server with FileZilla

After successfully connecting to the FTP server as the anonymous user, I used FileZilla to browse the directory structure. The local file system (/home/atohme) is displayed on the left, while the FTP server's root directory / is shown on the right. The pub folder is visible and accessible, which confirms that directory browsing permissions are working correctly.

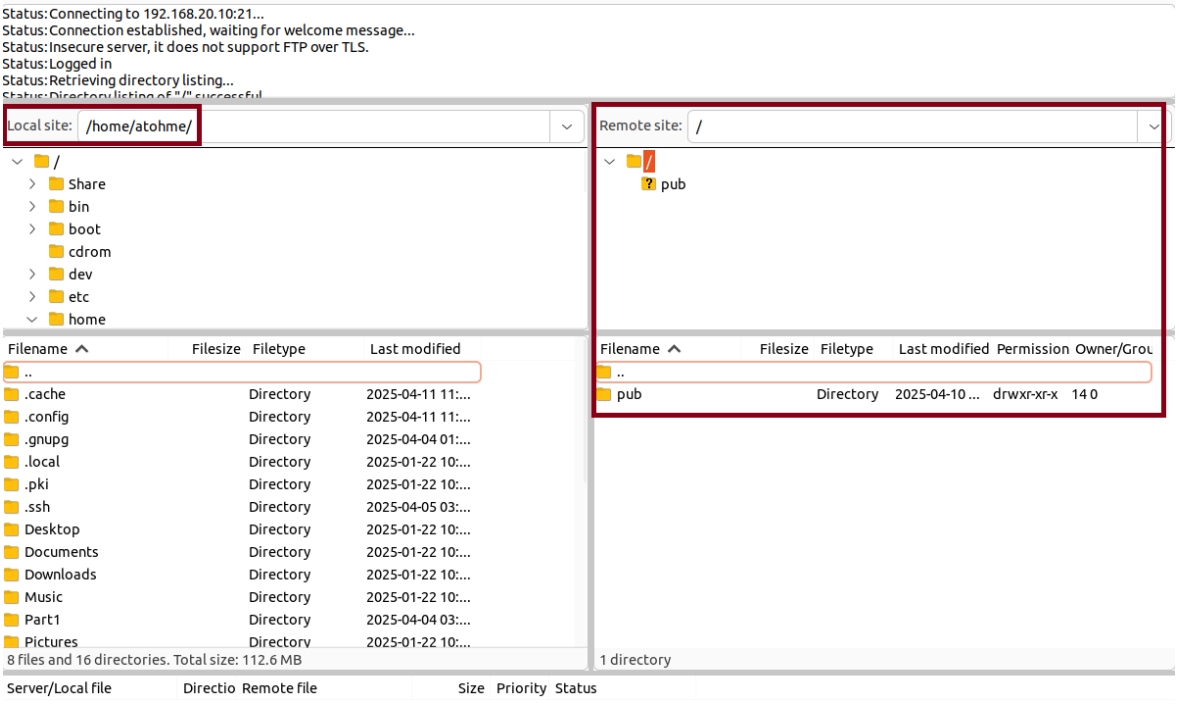

Figure 46 Connected to the FTP server in FileZilla with access to the pub directory.

FTP File Transfer Using FileZilla

After connecting to the FTP server with **anonymous login** through the FileZilla client, I tested the file upload functionality by transferring a new test file named filezilla_test.txt.

- On the **left pane**, I navigated to the local directory /home/atohme and selected the file filezilla_test.txt.
- I then dragged and dropped the file into the /pub directory shown in the **right pane**, which represents the remote FTP server's accessible upload location.

The transfer was successfully completed, and the file appeared on the server side alongside file.txt, confirming that anonymous users are allowed to upload files via FTP.

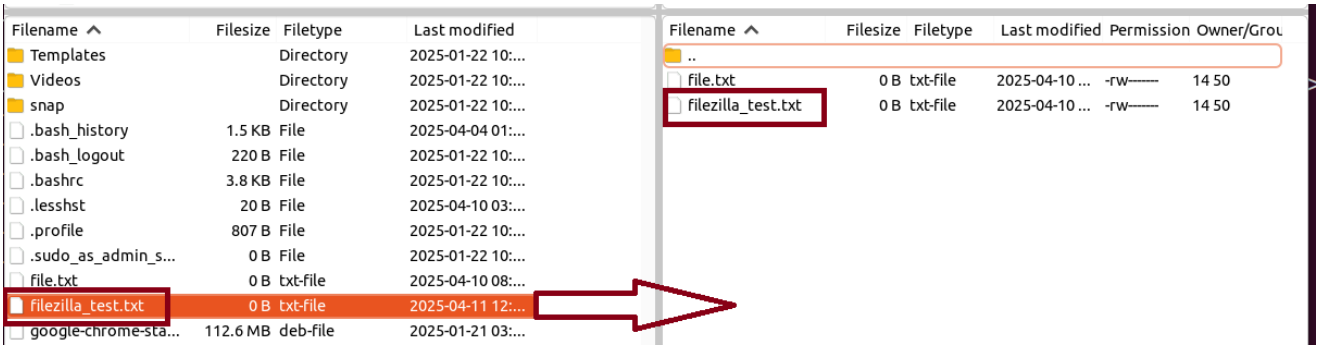

Figure 47 Successful upload of filezilla_test.txt from the Ubuntu client to the FTP server's

Conclusion

This lab helped me understand how to set up and configure essential network services in a Linux environment. I successfully installed and configured **DNS, DHCP, and FTP servers** on AlmaLinux and tested them from an Ubuntu client.

Through this assignment, I learned how to:

- Configure **DNS** to resolve domain names within a local network.
- Set up a **DHCP server** to dynamically assign IP addresses and network settings.
- Deploy and configure an **FTP server** that allows **anonymous access** and **file uploads**.
- Use tools like dig, nslookup, and ftp in the CLI for testing.
- Transfer files both through the **terminal** and **FileZilla**, reinforcing the practical use of these services.

Troubleshooting SELinux and firewall rules gave me a better understanding of how **security policies** affect service functionality. Overall, this assignment provided valuable hands-on experience with configuring real-world network services and reinforced my confidence in working with Linux-based server administration.