Application Layer (Web, Email) — Lab2 part1

A. Murad, D. Palma

Contents

Introduction 3
System Setup 3

1 Milestone 1 – Web Server Setup 5
1.1 Creating your own website 5
1.2 Setup a Dynamic Web server 6

2 Milestone 2 – Analyzing HTTP Traffic 7

3 Milestone 3 – Setup Mail Server 8
3.1 Creating Virtual domains and Mail Boxes 9
3.2 Setup Mail Access Protocol 10
3.2.1 Accessing Mailboxes Through Webmail 11

4 Milestone 4 – Analyzing Mail Server Traffic 11

Introduction

The goal of this lab is to learn how to setup and configure your own Web and Email server, as well as how to capture and analyze their traffic. The lab has several **milestones**. Make sure you reach each one before advancing to the next.

For delivery, submit a PDF report where you answer **only** those steps that are marked with **REPORT(%)**. Additionally, you should submit the codes or capture files, if they were **explicitly** asked for. The percent point gives you an indication of the score of that question. Lab2 (both parts) counts for **4 points** of your final score in this course.

System Setup

In this lab, we will use a system setup as shown in Figure 1, which is defined in the "docker-compose.yml" file. The same setup will be used for lab3 and lab4. Additionally, to make thing easier the required containers' ports are mapped to the Docker Host as shown in Figure 2.

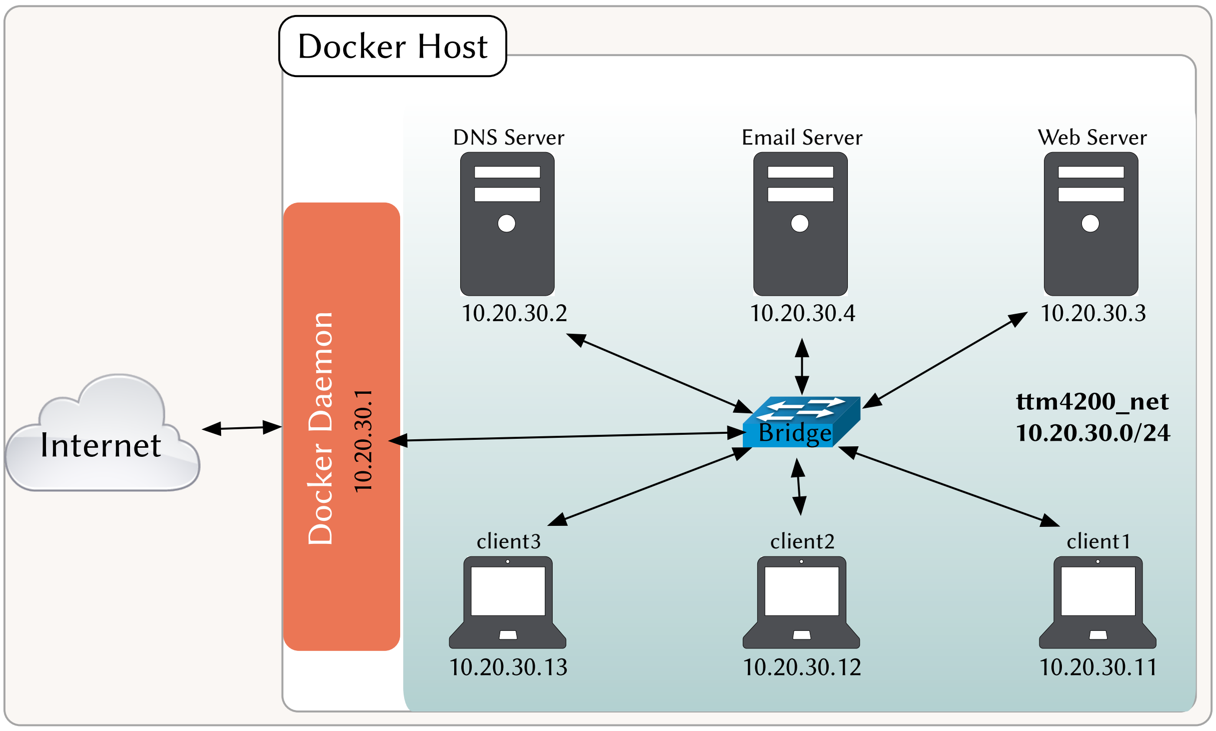


Figure 1: System Setup

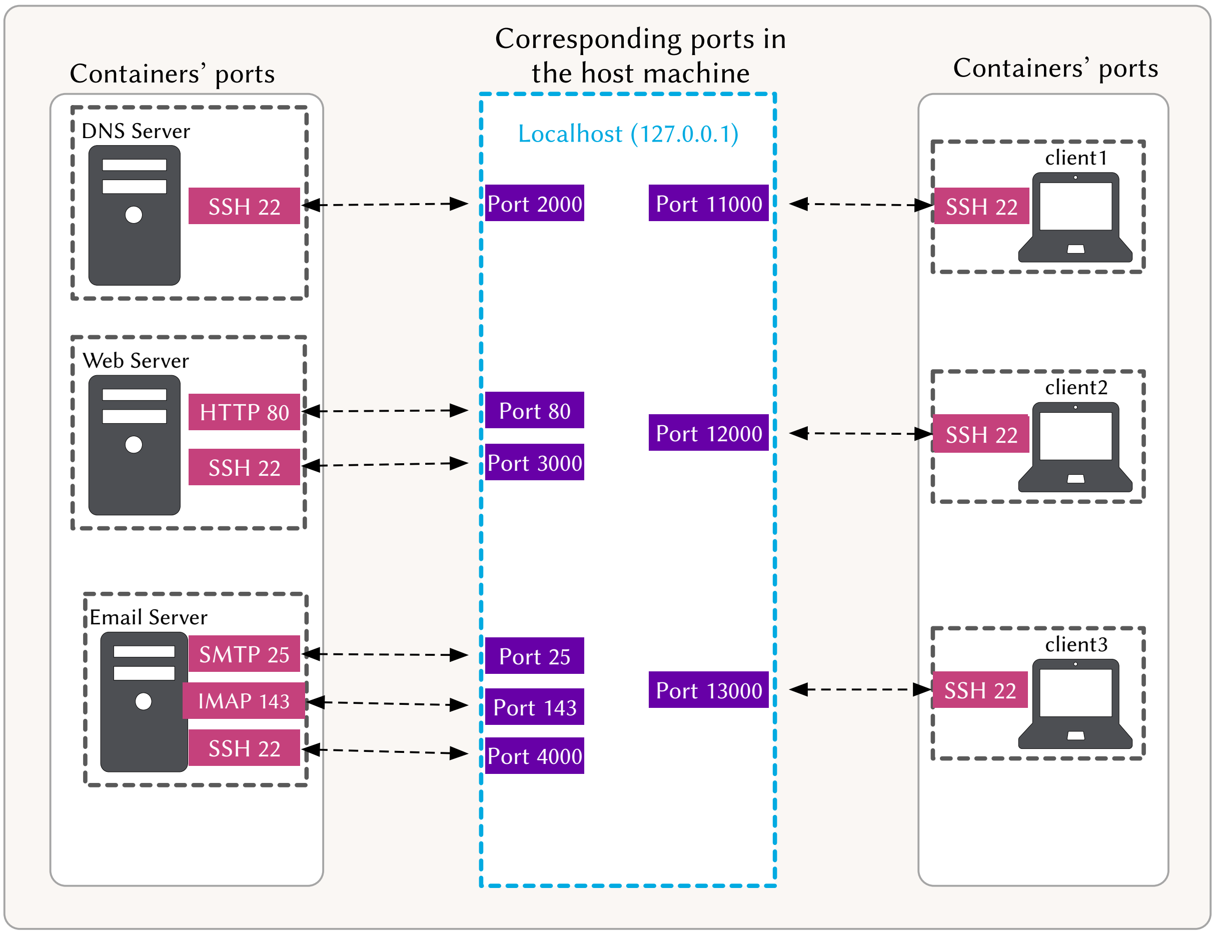


Figure 2: Port Mapping

> *You can change the port number on the Docker Host, as long as you avoid conflicts.*

- Create and start the containers using the docker-compose file:

```
docker-compose up -d --build
```

> *The first time you run this command it will take a while to build the images (less than 5 minutes). Use that time to read the "docker-compose.yml" to fully understand how the system is setup.*

> *If you are wondering about the syntax (e.g. $USER_ID), these are environment variables. By default, docker-compose will look for a file named ".env" in the directory that you run the command from to populate values inside a Compose file (https://docs.docker.com/compose/environment-variables/). The syntax ${USER_ID:-1000} assigns a default value if the vari-*

> *able is not defined (https://www.gnu.org/software/bash/manual/html_node/Shell-Parameter-Expansion.html)*

- You can connect to any container using SSH:

```
ssh ttm4200@127.0.0.1 -p [Mapped Port in the Docker Host]
```

- To copy files from or to containers, use `scp` or the mounted volumes. Read through the "docker-compose.yml" to find out which directories are mounted.
- Inside the containers, the password for the root and the user "ttm4200" is setup to "ttm4200". If you want, you can change it in the file "ttm4200_base/utilities.sh".
- To save your configuration in any container, run the script: `sudo sh ~/work_dir/save.sh` inside that specific container.

1 Milestone 1 – Web Server Setup

In this lab we will use Nginx (https://www.nginx.com/) as a web server. It is an open-source web server that is also used as a reverse proxy, HTTP cache, and load balancer. As of October 2020, *Nginx* served 34 percent of all websites, ranking it first above Apache at 27 percent (according to Netcraft (https://news.netcraft.com/archives/2020/10/21/october-2020-web-server-survey.html)).

- *Nginx* is already pre-installed in the "webserver" image. It is configured to serve a static HTML page. To access this page in your web-browser, run this command in your **VM:**

```
sudo sh -c 'echo "127.0.0.1 www.ttm4200.com" >> /etc/hosts'
```

> *This is to bypass DNS resolution in your VM. It basically lets the VM to translate the domain "www.ttm4200.com" to the IP address "127.0.0.1"(localhost) without using the DNS protocol for resolving the domain name.*

- Access the html page by typing in your browser: "www.ttm4200.com:80".

> *The port number in the domain name isn't necessary, since the default HTTP port is 80. However, if you mapped a different HTTP port to your Docker Host, you must use the port after the domain: "www.ttm4200.com:[port]/some/path/"*

1.1 Creating your own website

- Access the "webserver" container and and create a root directory for your website in "/var/www/". This will be the directory from which requests will be served from.

- Create an simple HTML page (e.g that contains your team's info) and place it in your website's root directory.
- Create a server block file that tells *Nginx* how to serve your website's contents. Name it "teamsite" and place it in "/etc/nginx/sites-available/". You can start with this template by filling it with your configuration:

```
server {
        # specify the port your webserver will listen to
        listen ====fill in here==== ;

        # specify root directory of your website
        root ====fill in here==== ;

        # specify the default page (e.g. your HTML page)
        index ====fill in here==== ;

        # Specify your server name. This will be the domain name
        #of your website. (e.g. www.team10.com). Use your team number
        server_name ====fill in here==== ;
}
```

> *By default Nginx server block configuration files are stored in "/etc/nginx/sites-available" and enabled through symbolic links to "/etc/nginx/sites-enabled/". You can have multiple configuration files for different websites with different names. The* `server_name` *directive determines which server block is used to a given request, not the file name.*

- Enable your website by creating a symbolic link from your config file to the "sites-enabled" directory, which *Nginx* reads from during startup.

```
sudo ln -s /etc/nginx/sites-available/teamsite /etc/nginx/sites-enabled/teamsite
```

- Check the syntax of the *Nginx* configuration file to to make sure that there are no syntax errors `sudo nginx -t`, then restart *Nginx* (`sudo service nginx restart`).
- Verify that your website is properly configured using a browser from your host machine (e.g. "www.team10.com:80"). Don't forget to bypass the DNS resolution in the VM. For example, if your domain name is "www.team10.com" (in the `server_name` directive):

```
sudo sh -c 'echo "127.0.0.1 www.team10.com" >> /etc/hosts'
```

Q1. **REPORT(4%):** Submit your HTML file and a screenshot of your website.

1.2 Setup a Dynamic Web server

To serve dynamic webpages (such as email login gateways), we can use a language such as PHP alongside *Nginx*. We will use the PHP Fast Processes Manager (PHP-FPM) to handle PHP scripts. PHP-FPM is already installed and configured, you just need to tell *Nginx* to use PHP for dynamic content.

- In the *Nginx* configuration file of your website, add the following directives inside the "server" block:

```
          location ~ \.php$ {
             include snippets/fastcgi-php.conf;
             fastcgi_pass unix:/run/php/php7.2-fpm.sock;
             fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
             include fastcgi_params;
          }
```

- Restart *Nginx*.
- Create a simple *php* file in your webserver's root directory (e.g. create `YourFileName.php` and add the following to it:

```
    <?php
    phpinfo();
```

- Check your configuration by accessing the *php* file you just created through a browser in you host machine (e.g. "www.team10.com:80/YourFileName.php").

2 Milestone 2 – Analyzing HTTP Traffic

- Access the container "client1" and start packet capturing with `tcpdump`. Save the captured packets to a file (e.g "client1_http.pcap").
- Open a second SSH session to "client1" and retrieve the HTML page from your website using `wget`. Because you have not configured the DNS, the container "client1" doesn't know which IP address corresponds to your domain name. Thus, you have to either bypass the DNS resolution:

```
      sudo sh -c 'echo "10.20.30.3 www.team10.com" >> /etc/hosts'
      wget www.team10.com
```

  Or provide the IP address in `wget` command:

```
    wget http://10.20.30.3 --header "Host: www.team10.com"
```

- Stop the `tcpdump` capturing and copy the capture file to your host machine.
- In your VM, open the capture file in Wireshark, and apply the display filter "http" so that only HTTP packets will be displayed.
- Examine the information in the HTTP GET and the response message. Then answer these questions and validate your answer with an **annotated** screenshot from Wireshark.

Q2. **REPORT(2%):** What is the version of HTTP client (version 1.0 or 1.1)? What version of HTTP is the server running?

Q3. **REPORT(2%):** Show the IP address for the client and the server.

Q3. **REPORT(2%):** What is the status code returned from the server to the client?

Q5. **REPORT(2%):** When was the HTML file that you are retrieving last modified at the server?

Q6. **REPORT(2%):** How many bytes of content are being returned to the client?

Q7. **REPORT(2%):** By Inspecting the raw data in the packet content, do you see the response in cleartext (line-based text data)?

Q8. **Extra:** Suppose there is a "Man-in-the-middle" between the client and the server who can sniff packets. Is it possible that they can read the contents of the HTTP messages (eavesdropping)? Can he alter the contents of the HTTP messages (active eavesdropping)? What do you think should be be done to circumvent these vulnerabilities?

Q9. **REPORT(4%):** Submit your capture file "client1_http.pcap" along with the report. We will check it and make sure that it is unique and corresponds to your answers above.

3 Milestone 3 – Setup Mail Server

One possible way to setup a mail server is use Postfix (http://www.postfix.org/) which is a free and open-source mail transfer agent. It is already installed and configured in the "mailserver" container. Additionally, an MySQL database is configured for storing virtual domains and mailboxes.

> *If you are interested in how to configure postfix in linux, look at this tutorial: https://linoxide.com/mail/install-configure-postfix-ubuntu/, or see the file "mailserver/utilities" for configuring it in a docker container.*

In this milestone, you have to only create a virtual domain for your team (e.g "team10.com") and a mailbox for everyone in you team (e.g "Alex@team10.com"). Then access these mailboxes through a Mail Access Protocol (IMAP) and Webmail.

We have setup the "webserver" to act as a proxy server for the "mailserver" as shown in figure . It will pass specific web requests to the "mailserver", fetch the response and sends it back to the client.

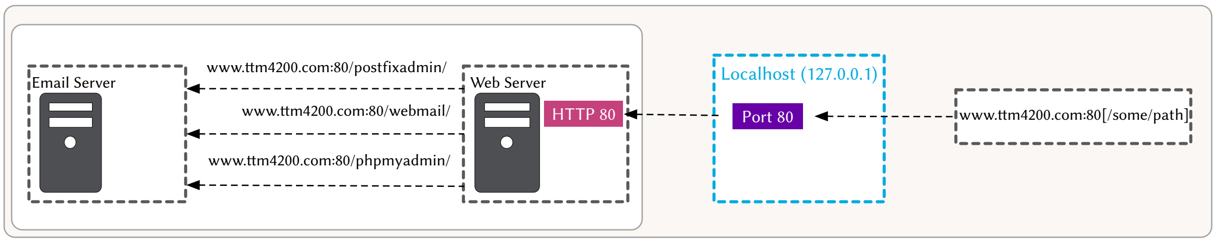

Email Server
www.ttm4200.com:80/postfixadmin/
www.ttm4200.com:80/webmail/
www.ttm4200.com:80/phpmyadmin/
Web Server
HTTP 80
Localhost (127.0.0.1)
Port 80
www.ttm4200.com:80[/some/path]

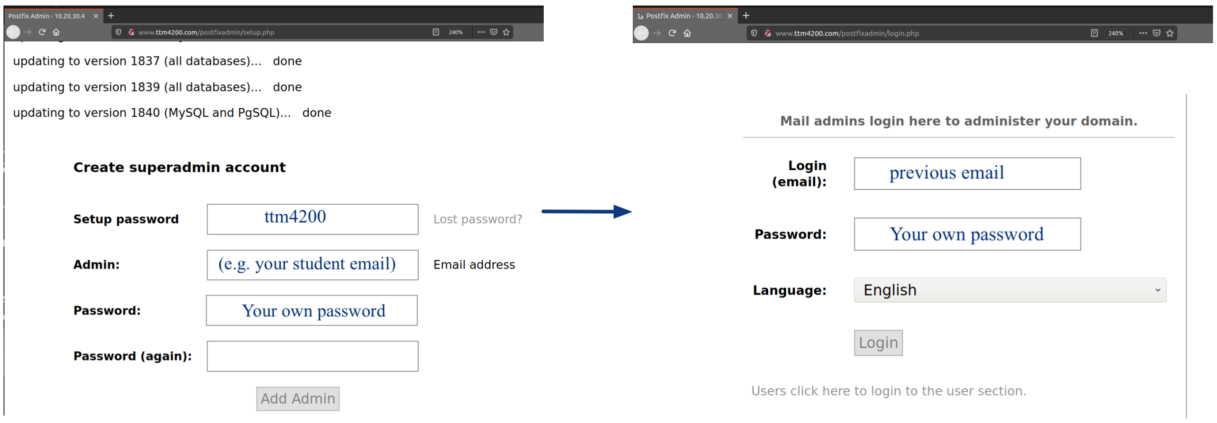
Figure 3: The webserver is acting a proxy for the mailserver

> *If you are interested in how the the proxy server is setup, check out "/etc/nginx/sites-available/ttm4200" in the webserver.*

3.1 Creating Virtual domains and Mail Boxes

You can use *postfixadmin* which is a web module that allows you to manipulate your virtual domains and users in a database.

- Access *postfixadmin* setup page ("www.ttm4200.com/postfixadmin/setup.php") and create a super administrator account. The setup password is already set to "ttm4200" and you can use any email address of your choice. This account will have administrative capabilities of signing up users and setting up the domains. Then login to your super administrator account ("www.ttm4200.com/postfixadmin/login.php") as shown in figure.



updating to version 1837 (all databases)... done
updating to version 1839 (all databases)... done
updating to version 1840 (MySQL and PgSQL)... done

Create superadmin account

Setup password	ttm4200	Lost password?
Admin:	(e.g. your student email)	Email address
Password:	Your own password	
Password (again):		

Add Admin

Mail admins login here to administer your domain.

Login (email):	previous email
Password:	Your own password
Language:	English

Login

Users click here to login to the user section.

Postfix Admin - 10.20.30.4
www.ttm4200.com/postfixadmin/setup.php
Postfix Admin - 10.20.30
www.ttm4200.com/postfixadmin/login.php

Figure 4: *Postfixadmin* setup page: Creating a super administrator account

- Create a domain for your team (e.g. "team10.com") by navigating to **Domain List** → **New Domain**). Then create a mailbox for every member of your team (e.g "alex@team10.com") by navigating to **Virtual List** → **Add Mailbox**).
- To check the database hosting your domain and mailboxes, you can use phpMyAdmin

(https://www.phpmyadmin.net/), which is an open source administration tool for MySQL database. It is already installed and configured in the “mailserver”. You can access it through “www.ttm4200.com/phpmyadmin/”. The database (username) is called “postfix” and the password is setup to “ttm4200”. On the left hand side, navigate to **postfix→domain**, you can see your domain data. If you navigate to **postfix→mailbox**, you can see your mailbox’s data.

Q10. **REPORT(4%):** Submit a screenshot of your domain and mailboxes from phpMyAdmin.

3.2 Setup Mail Access Protocol

In order to retrieve email messages form the mail server over a TCP/IP connection, we can use the IMAP protocol. One popular and open source IMAP server is Dovecot (https://www.dovecot.org/). It is already installed and configured in the “webserver”. You have to only setup an email client in your **VM**. You can use Thunderbird (https://en.wikipedia.org/wiki/Mozilla_Thunderbird) as your email client/agent.

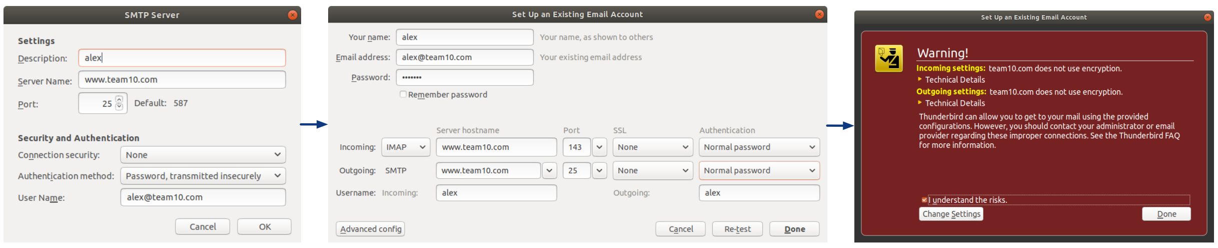
In your VM, connect with the "mailserver: using one of your created mailboxes by following these steps:

- Edit your hosts file to map your domain name to the localhost IP address (if you use a different domain than in Milestone 1):

```
sudo sh -c 'echo "127.0.0.1 www.team10.com" >> /etc/hosts'
```

- In Thunderbird, navigate to **Preferences→Account Settings→Outgoing Server (SMTP)→Add**
- Set the *Server Name* to your domain name and set the SMTP port (25).
- Set *Connection security* to “None”, and *Authentication method* to “Password, transmitted insecurely”.
- Set the *User Name* to the created mailbox .
- Go to **Account Actions→Add Mail Account**, set your email created *Email address* and *Password*, then **Manual config**.
- Set the Mail Access Protocol to “IMAP” on “localhost” with port 143.

  > *In this lab, we are using insecure IMAP (over port 143), but it is possible to use IMAPS (IMAP over SSL) in which traffic travels over a secure socket to a secure port (port 993). Therefore, Thunderbird will warn you as shown in figure 5*


SMTP Server

Settings
Description: alex
Server Name: www.team10.com
Port: 25 Default: 587

Security and Authentication
Connection security: None
Authentication method: Password, transmitted insecurely
User Name: alex@team10.com

Set Up an Existing Email Account

Your name: alex — Your name, as shown to others
Email address: alex@team10.com — Your existing email address
Password: •••••••
☐ Remember password

		Server hostname	Port	SSL	Authentication
Incoming:	IMAP	www.team10.com	143	None	Normal password
Outgoing:	SMTP	www.team10.com	25	None	Normal password

Username: Incoming: alex Outgoing: alex

Set Up an Existing Email Account

Warning!
Incoming settings: team10.com does not use encryption.
Outgoing settings: team10.com does not use encryption.
Thunderbird can allow you to get to your mail using the provided configurations. However, you should contact your administrator or email provider regarding these improper connections. See the Thunderbird FAQ for more information.
☑ I understand the risks.

Figure 5: An example of thunderbird settings for the domain name “team10.com”

- Send an email to another mailbox. You can check the received message through different methods but in the following task we will do it using webmail on a browser.

3.2.1 Accessing Mailboxes Through Webmail

To enable webmail access, we have installed and configured Roundcube (https://en.wikipedia.org/wiki/Roundcube) in the “mailserver”. It is a web-based IMAP email client.

- Access the webmail page (“www.ttm4200.com/webmail/”) and login to the “other” mailbox. Check that you have received the email sent from Thunderbird.

Q11. **REPORT(3%):** Include a screenshot of the sent and received emails in your report.

4 Milestone 4 – Analyzing Mail Server Traffic

- Start packet capture with `tcpdump` on your mail server and save the captured packets to a file (e.g “mailserver.pcap”).
- Send an email from your Thunderbird to the other account opened through webmail, and reply to this email.
- Stop the `tcpdump` capturing and copy the capture file to your host machine. Open the capture file in Wireshark.
- Apply a display filter to show only SMTP based traffic. Follow TCP Stream of one email and examine SMTP transport.By closing the dialog, it will apply a display filter which selects all the packets in the current stream. Then answer these questions and validate your answer with an **annotated** screenshot from Wireshark:

  Q12. **REPORT(3%):** Find TCP Connection Traffic (three way handshake). What is the destination port?

  Q13. **REPORT(3%):** Find SMTP Service Ready Traffic (packet labeled 220) and the client TCP acknowledgement of receiving the Service Ready message.

Q14. **REPORT(3%):** Find SMTP EHLO (extended hello) Traffic. These are the packets corresponding to the client initiating a session with the server with a *EHLO* command.

Q15. **REPORT(3%):** Find packets corresponding to transmission of the body of the mail message, initiated with a *DATA* command.

Q16. **REPORT(3%):** Find SMTP Completed Traffic (labelled 250).

Q17. **REPORT(3%):** Find SMTP QUIT Traffic. These are packets corresponding to a session end, initiated with *QUIT*.

Q18. **REPORT(3%):** Find SMTP Closing Traffic (packet labelled 221).

- Apply a display filter to show only IMF based traffic. It will show the email text.

Q19. **REPORT(3%):** Submit a screenshot of your email text from Wireshark.

- Apply a display filter to show only the IMAP based traffic and Follow TCP Stream. Then answer these questions and validate your answer with an **annotated** screenshot from Wireshark:

Q20. **REPORT(3%):** Show the packets corresponding to the client establishing a TCP connection. Show the port number used for connection.

Q21. **REPORT(3%):** Show the packets corresponding to the server respond of its IMAP capabilities

Q22. **REPORT(3%):** Show the packets corresponding to the access authentication process. Then find the server reply with code 3 (login success).

Q23. **REPORT(3%):** Show packets corresponding the client sending IMAP *FETCH* command to fetch any mails from the server.

Q24. **REPORT(4%):** Submit your capture file “mailserver.pcap” along with the report.