School of Electrical and Information Engineering
University of the Witwatersrand, Johannesburg.

ELEN4017: Network Fundamentals
Instructor : Dr. Asad Mahmood

Email Application

1. Introduction

The goal of this project is to create an Email application system. The email application system will make use of the networking principles studied in the class e.g. protocols, application and transport layer, socket programming etc. You can imagine the components of the email application system that we use in our daily lives e.g. email clients (outlook etc.) and email servers (e.g. the one hosted in our school) to have an idea of basic functions/entities in an email application system, however the project requires implementation of only some of these functions/entities so as to limit the scope.

Just like other email application systems, your application should allow any user to send and receive emails via the Email Client. Email client will interact with the local mail servers, both local SMTP server for outgoing mails and local IMAP/POP3 server for incoming emails. Finally, the client is able to either send the email to the desired destination or receive its email and display it to the user.

Note that although programming languages like Python provide libraries that implement the required protocols e.g. SMTP, IMAP and POP3 for you but this hides away the functionality of these protocols. Hence, you are not required to use those standard libraries for these protocols and should only use the basic socket methods. This would also imply that you would need to consult the RFCs for SMTP, POP3 and IMAP protocols in order to implement these protocols in the standard manner. Once these protocols are implemented in the right manner, your mail client and server should not only be intercommunicate but the mail client should be able to send email to a standard SMTP server and receive email from a standard POP3/IMAP mail server. Similarly, a standard email client e.g. Outlook, Thunderbird etc. should be able to send emails to your implemented SMTP server which can then forward the email to destination.

2. Project Tasks

Developing the email application system would require implementation of the following features. These are minimal features to be implemented. You can extend them based on your needs:

A. Features of the Email-client

1. A simplistic user interface allowing the user to enlist, read, write and delete emails and to send/receive them.
2. Upon 'send' request, the email client contacts the concerned email server and forwards the email to it using SMTP protocol so that it can eventually send it to the desired email address.

3. Upon ‘receive’ request, the client should also be able to retrieve the desired emails from the concerned server by making use of the POP3 and IMAP protocols.
4. Your email client should be able to interact with a standard Email Server hosted in our school. Details of this standard server are given in section C below. Destination email address when using this email server could be your wits email address or one of your web email addresses.
5. The client should implement at least a basic commands of client side of SMTP protocol. Section 4.5.1 for RFC 5321 discusses a set of commands for minimal implementation.
6. The client should implement at least a basic commands of client side of POP3 protocol e.g. Optional commands as mentioned in section 7 of RFC 1939 need not be implemented.
7. The client should implement at least a basic commands of client side of IMAP protocol (RFC 3501). IMAP protocol has different commands with respect to the state of the system. A minimal implementation could consist of implementation of
    1. Universal commands = CAPABILITY and LOGOUT
    2. Not Authenticated State commands = LOGIN
    3. Authenticated state commands = SELECT, CREATE, DELETE, LIST, SUBSCRIBE and UNSUBSCRIBE
    4. Selected state commands = CLOSE, FETCH and COPY

   You are welcome to implement more commands from the RFC, if needed.

B. Features of the Email-server

1. Your email server should act as a relay to the destination email server for outgoing emails.
2. For incoming emails, it must have provisions to store emails for different users in their respective mailboxes.
3. A client is able to access his/her mailbox upon authentication and can request for his/her email.

4. Hence, a standard email client (e.g. Outlook etc.) should be able to send and receive emails via your developed email server.
5. Implement the server-side of the concerned commands of the SMTP, POP3 and IMAP protocols.
6. A Multi-threaded version of the web-server must also be created

C. Other Features

1. Your server must be able to deal with different types of emails i.e. text only, text with document and text with images.
2. The email server should be able to correspond with the client, for both sending and receiving, in both the cases i.e. 1) client and the server are both

hosted on the same PC/host and 2) client are server are on different hosts in the network.

3. Note that many of the protocol commands would solicit a reply from the other side see e.g. section 4.2 in RFC5321, section 9 in RFC 939 and section 7 in RFC 3501. Hence, you are also required to implement 3 digit reply codes corresponding to these commands. You are not required to implement all the reply codes but the reply codes relevant and essential for the above commands. Please specify the reply codes implemented in your final report as well as justify your choice of implemented reply codes.
4. The client and the server should be able to deal with common errors and generate appropriate responses as per RFC. Please specify the errors catered for in your final report.
5. You should be able to show that Wireshark can be used to monitor protocol messages between the client and the server.
6. The protocol implementations should strictly follow the respective RFC for the format of all the request and response messages and the order and types of messages exchanged. You do not have to go through all the details of these RFCs but only the relevant sections concerning your targeted commands and actions. This is important for communicating with the standard clients and servers, for which it is also important to know the type and version of protocols implemented by those standard clients and servers.
7. You are ONLY allowed to use low level socket class and associated methods for implementing the protocols needed for your email application, and the use of higher-level classes that implement these protocols for you is not allowed. However, for other aspects such as parsing, dealing with multimedia etc. you are allowed to use built-in libraries.
8. The information about the standard email server hosted in the school as given to me by the network supervisor are as follows.
   - Outgoing/SMTP server = Shannon.eie.wits.ac.za
     IP address/ Port No. = 146.141.16.134 / SMTP port 25
   - Incoming email server = I believe our Wits email mailboxes are in clymene.ds.wits.ac.za. You will have to see if you can interact with the server using IMAP to see your email.
   - If you are not able to access the above servers/ports from your laptops you would need to try it from a recognized host e.g. try D-lab PCs

Finally, any suggestions to modify/improve the above system are welcome and can be discussed in class. Also, any cool additional features can get you bonus marks.

3. Project Assessment

This project contributes 30% to the total mark for the course. The assessment policy is as shown in the following table:

Outcomes	**Checklist**	**Marks**
Features Implemented as per section ‘Project Tasks’ above.	1. Email client basic features 2. Email Server basic features	14%

	3. Interaction with standard email server and client 4. Ability to deal with different types of emails 5. Multithreading 6. Use of Multiple PCs 7. Use of Wireshark	
Demo + Q&A	- Questions related to concepts that are involved in the project as well as about the coding part will be asked individually.	4%
Report	- Each group should submit a single report - Maximum 7 double-column pages excluding the appendices. Should contain: 1. Introduction 2. Description of the implemented system and the implemented features. What features could not be implemented and why. 3. Brief description of commands/replies for each protocol that were implemented. 4. Implementation of features in detail **5. Division of Tasks and Individual Contribution (in code and report)** 6. Results (Wireshark Screenshots etc.) 7. Critical Analysis 8. Basic structure of the code (Classes, methods etc.) and how to use the code 9. References 10. Appendix	10%
Source Code	– Should be well commented – Must have a readme file detailing how to use the code – Should send it electronically	2%

4. Other Details

- You can form a group of 2 students. Groups must be **finalized by March 15th and details of group members must be sent to the course instructor** at asad.mahmood@wits.ac.za so that a Group Number can be allocated to each group.
- Slots for demos will be published later and each group will select a slot. You will be required to give a demo of the implemented features for different cases. You are expected to know both the theory as well as the coding aspects of the tasks and you might be asked about any relevant concept, any portion of the code and/or to modify the code during demo.
- I would recommend doing the implementation in Python language, but in case you prefer to use another language, kindly let me know beforehand. **Please specify your language, OS and IDE before your demo. It would be your responsibility**

to make sure beforehand that your software is working on the systems that will be used for demos.

- Please also note that there have been many updates/extensions on the relevant protocols of SMTP, POP3, and IMAP as defined by other RFCs. You need not consult these unless absolutely necessary for a basic working.
- **Individual Contribution**
  - The final report must describe all work undertaken in the project, and must explicitly describe the individual's contribution
  - A good description would be "wrote function xyz() in server implementation." A bad description would be "wrote some server code".
  - The individual must discuss the implementation of their own code in the report.
  - The source code must clearly indicate which individual has authored each code segment.
- Plagiarism in any form will be strictly dealt with as per School's policy.

5. Submission Details

The submission deadline, as per deadlines given by School for 4th year projects, is at **07h50 on Monday, May 8th, 2017**. You are required to submit the following project components by the given deadline:

1. Electronic-copy of all the project material (report, source code etc.) via Sakai. All project material should be zipped as a single file and submitted with the name Group_X where X would be the group number allocated to you.

Late submission will be penalised according to guidelines specified in the "*Red Book*".

References:

1. https://en.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol#Related_requests_for_comments
2. https://en.wikipedia.org/wiki/Post_Office_Protocol#Related_requests_for_comments_.28RFCs.29
3. http://ccm.net/contents/116-how-email-works-mta-mda-mua