# Assignment 4: Developing a Simple email Application

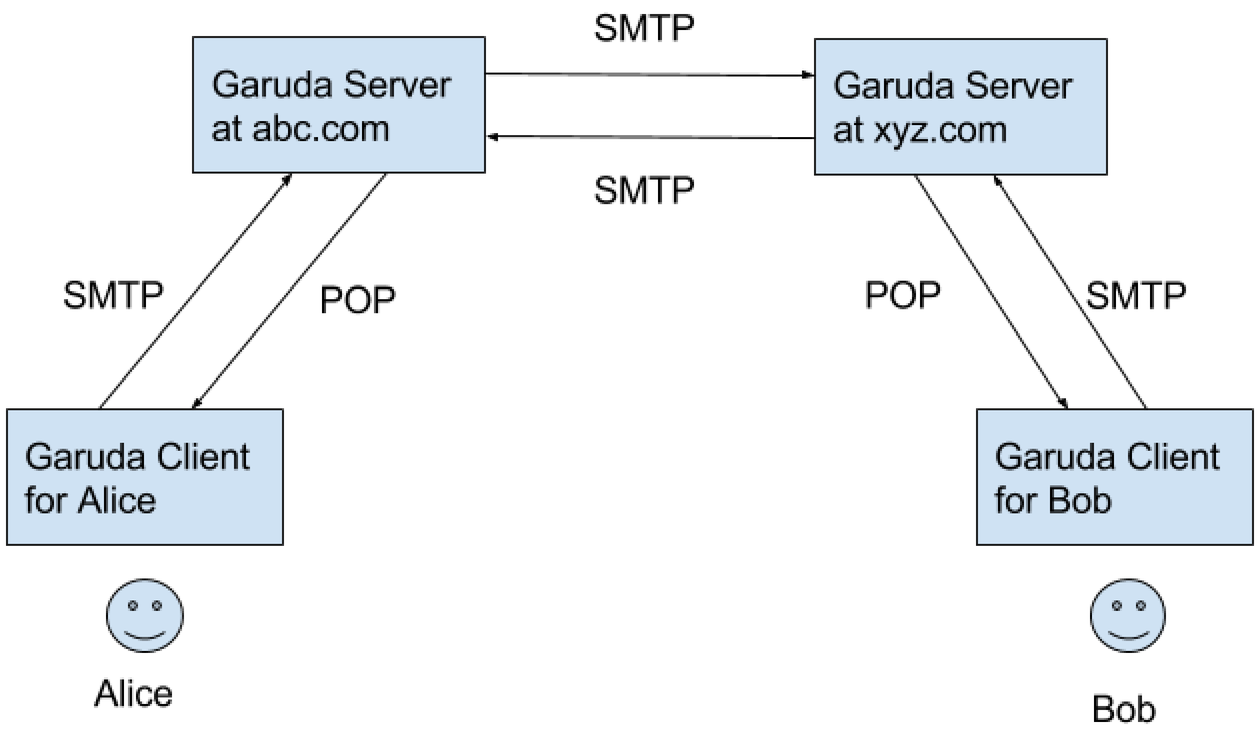
**Objective:**
The objective of this assignment is to develop an end-to-end email application which enables the users to send emails to other users and retrieve the emails destined for them. The current email applications we use everyday (gmail, yahoo mail, microsoft outlook or mozilla thunderbird) are complex and their architectural designs are often proprietary and beyond the scope of this lab. Hence, your task is to develop a simpler email application which provides only the basic functionalities. We call the application 'Garuda', named after the mythological humanoid bird. If you wish to know more about the currently deployed email applications, you can look into the open source email application Thunderbird (https://developer.mozilla.org/en-US/docs/Mozilla/Thunderbird).

**Description:**
The Garuda infrastructure consists of several components that work together to send, relay, receive, store, and deliver email. Garuda uses the following Internet standard protocols for sending and retrieving email:

- Simple Mail Transfer Protocol (SMTP - 25): the Internet standard protocol for sending email.
- Post Office Protocol (POP - 110): an Internet standard protocol for retrieving email.

Email is delivered in Garuda using a client/server architecture. An email message is created using a Garuda client program. This client program then sends the message to a Garuda server. The server then forwards the message to the recipient's Garuda server, where the message is then supplied to the recipient's Garuda client.

The following figure shows the Garuda components and the flow of email through those components. Suppose there are two users Alice and Bob, having email addresses in two domains: abc.com and xyz.com. Alice wants to send an email to Bob. That email flows through the Garuda components as follows:



1. From her Garuda client, Alice creates an email and asks the client to send.
2. The client uses SMTP to send the email to the Garuda server at abc.com.
3. Then the Garuda server at abc.com relays and routes the email to the Garuda server at xyz.com which is the domain of the recipient Bob.
4. The Garuda server at xyz.com stores the email for Bob.
5. When Bob is free, he uses his Garuda client to query the Garuda server at xyz.com for any incoming emails.
6. The client uses POP to retrieve the email for Bob from the Garuda server at xyz.com.
7. From his Garuda client, Bob reads the email created by Alice.

In conjunction with Garuda server and clients, you can use additional text editors (like gedit or vim) to preprocess and postprocess the email. A description of deploying such applications is outside the scope of this assignment. You may choose to select solutions to work in conjunction with Garuda for such additional functions. Otherwise, you can use only the terminal to compose and display the emails for the users.

**Architecture:**
Garuda client consists of two independent parts: one SMTP client and one POP client. Garuda Server has three such parts: one SMTP server, one SMTP client and one POP server. The first one receives emails from the Garuda clients and puts them into the email queue. The second one picks up messages from the queue, sends them to the destination Garuda

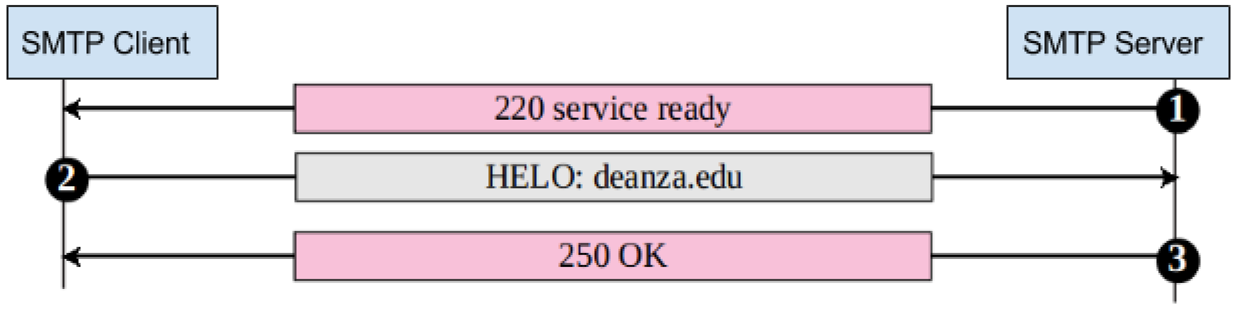
server. The third part delivers the email to the final recipient Garuda clients. Now let's see how SMTP and POP protocols work.

**SMTP:**

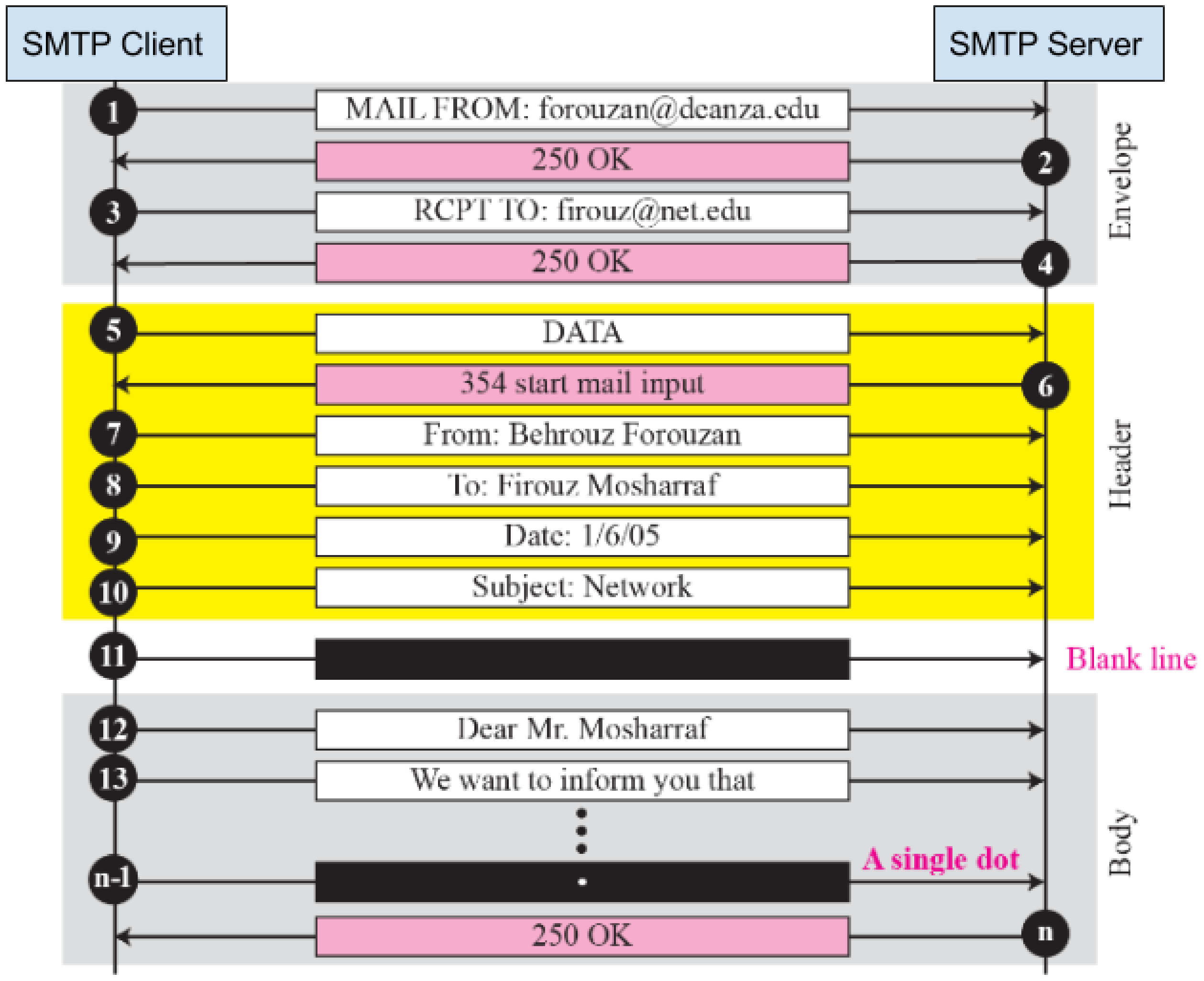
SMTP is a connection-oriented, text-based protocol in which an email sender communicates with an email receiver by issuing command strings and supplying necessary data over a reliable ordered data stream channel (i.e. a TCP connection). Initiating an SMTP session consists of commands originated by an SMTP client and corresponding responses from the SMTP server so that the session is opened, and session parameters are exchanged. This SMTP session stays until connection termination messages are exchanged. In between these connection establishment and termination, email message is exchanged between the sender and the receiver.

See the example below for the steps involved in a successful SMTP communication.

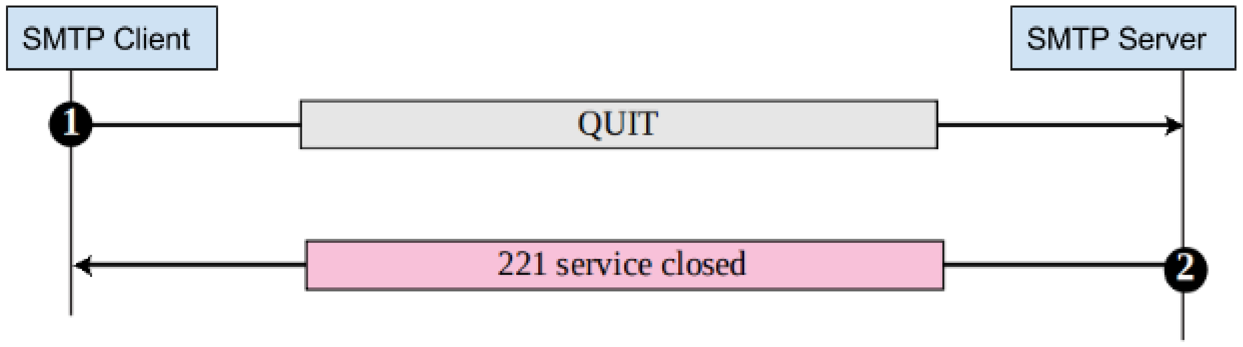
**Connection Establishment:**



**Message Transfer:**

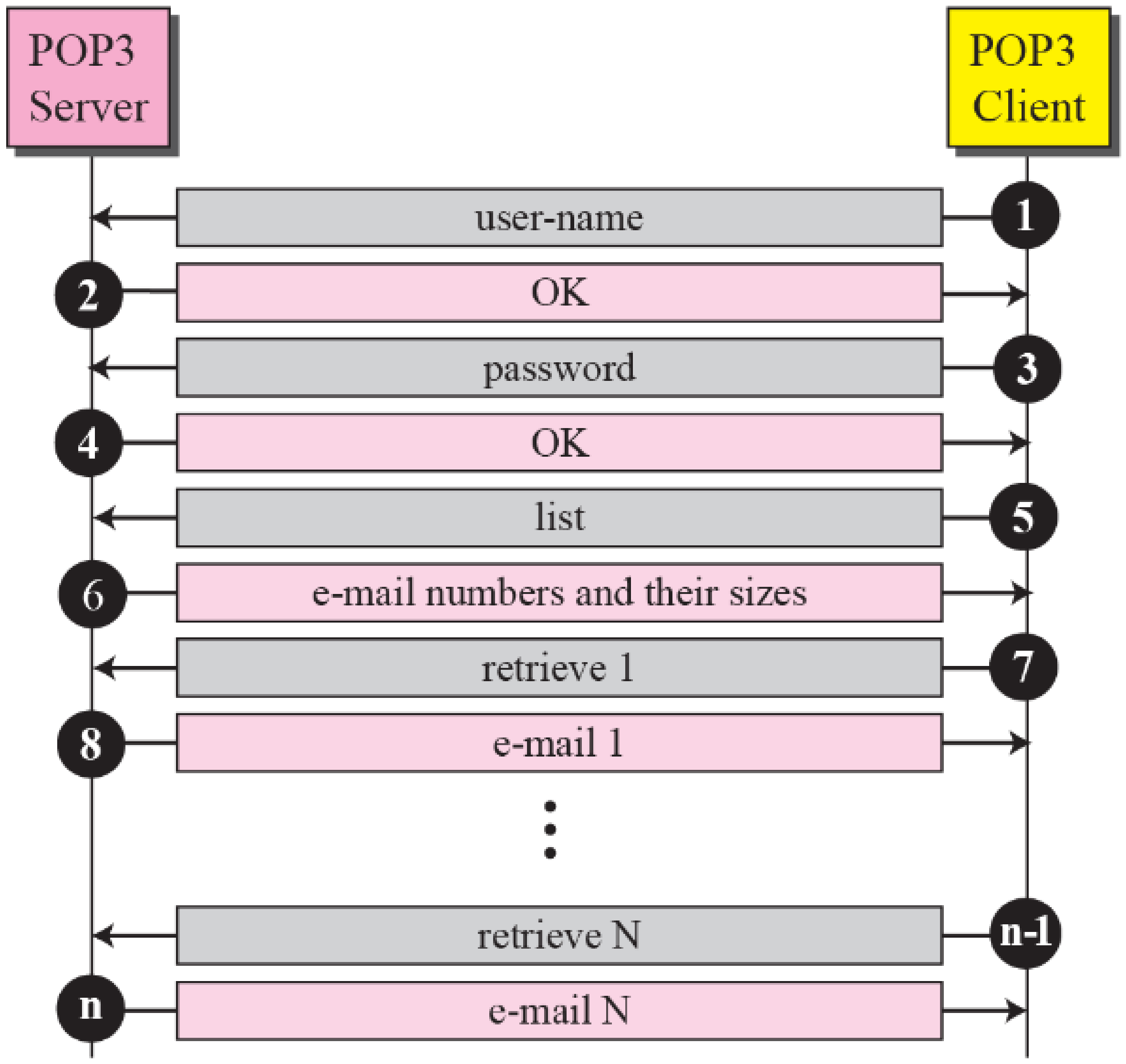


**Connection Termination:**



**POP:**

Please note that SMTP is a delivery protocol only. In normal use of SMTP, email is "pushed" to a destination mail server as it arrives. Therefore, we need POP specifically for pulling (or retrieving) the messages by individual users. POP supports simple download-and-delete requirements for access to remote mailboxes. First a POP client authenticates with the POP server and then get the list of all emails available for the particular user. It then retrieves the emails one by one. Once the emails are downloaded by the client, POP server deletes the emails. The figure below gives an example workflow.



**Deliverables:**

In this assignment, you need to implement the complete Garuda application (including the client and the server). As part of Garuda client and server, you have to implement SMTP server-client as well as POP server-client instances. Both SMTP and POP use TCP as the transport layer protocol. For simplicity, assume the emails to be only textual (i.e. no need to implement MIME), and implement only the following SMTP and POP commands required for basic email sending and retrieval.

SMTP Commands: HELO, MAIL FROM, RCPT TO, DATA, SAML, VRFY, RSET, QUIT
POP Commands: USER, PASS, STAT, LIST, RETR, QUIT

**Deliverable 1:** Implement and run one Garuda server instance abc.com running on a particular machine. Assume that there are four users of abc.com: Alice, Arun, Ananya, and Alex. These users should be able to connect to abc.com from different machines and send emails to each other. The users should also able to retrieve the messages destined for them from different machines.

**Deliverable 2:** Now instead of one Garuda server, implement and run two Garuda server instances abc.com and xyz.com running on two different machines. Assume that there are four users of abc.com: Alice, Arun, Ananya, and Alex and four users of xyz.com: Bob, Bilal, Alex and Bernee. Note that Alex has accounts on both abc.com and xyz.com. Similar to deliverable 1, these users should be able to connect their respective Garuda servers from different machines and send emails to each other. The users should also be able to retrieve the messages destined for them from different machines.

**Deliverable 3:** For users (e.g. Alex) having multiple accounts at multiple Garuda servers, their Garuda client should be able to fork multiple processes each communicating with one Garuda server instance.

**PS:**
As both SMTP and POP are internet standards, there are multiple RFCs explaining the protocols in detail. We encourage you to go through the following RFCs to get a better understanding of these protocols and to understand in which order the commands should be processed.

RFC 821 for SMTP: https://tools.ietf.org/html/rfc821
RFC 1081 for POP: https://tools.ietf.org/html/rfc1081