Fredrik Nordlund <fnordl@kth.se> Prakash Rajagopalan<prakashr@kth.se>

IK2213 Webmail Report

Introduction

There are many different application protocols used throughout the Internet today. Some are complicated and some are simple. Some application protocols have had little success while some protocols have been hugely successful and has been used for decades (and maybe decades to come). Three of these very successful and widely (if not globally) used application protocols are the *hypertext transfer protocol* (HTTP), the *domain name service* (DNS) protocol, and the *simple mail transfer protocol* (SMTP).

The goal of this small project has been to get a greater understanding of how these above mentioned protocols are used in applications and get a more in depth knowledge of how they work. Another goal has been to get both an in depth but also a broad view perspective of how protocols, like the ones described in this report, can be used together to create a powerful application.

Problem

The problem of this project was, how can you use different protocols to create a powerful application and what do you need to do to make these protocols work. Several problems that had to be solved was for example, what are the necessary headers that has to be used when dealing with HTTP, SMTP, and MIME. How should you parse text to fit the different protocols.

Another problem was, when using HTTP, how can you create a simple web server that can handle basic HTTP. After this has been successful, how do you use DNS to connect to a SMTP server and send a message received from a user posting something on your website to the SMTP server successfully. Furthermore, if this message does contain characters not present in a global encoding, how do you encode the message using MIME so that a reader of the message still can see these characters.

Furthermore an additional field was added to specify the sending delay of an email message and all the pending messages should be able to be viewed in a requestable status web page. These two last goals were added to target the advanced level of this task.

Solution

The Webmail is designed as two subsystems. The first one is a HTTP web server and the second sub system is a SMTP client. The functionalities are as follows.

A HTTP server is implemented by creating a server socket on the HTTP port 80 and it waits for the HTTP client (A web Browser) to connect to it. Once the client connects to the server, server wait for the client to send us the HTTP request. Each line of the HTTP request is parsed by the server and various components of the request are understood. For example, if a GET request is issued, the format and the resource requested is checked and validated. The server is designed to reply back with a HTML form for composing an email when it receives a GET request.Once the user fills in email parameters and presses the send button, a POST request is received by the server containing the URL encoded post data. After the URL decoding is done and then each of the fields of the email are split up and their values are extracted and a validation check on each of the fields is carried out. If any of the mandatory fields are missing then an error page is sent back to the user. Once all the sanity checks on the user entered data is

performed, the email data is transferred to the SMTP client.

SMTP client is responsible for establishing a SMTP connection with a SMTP server and sending the email to the server. When the HTTP server sends the email data to SMTP client, it checks if the user has provided the email server to connect to, else the email server is looked up in the MX record by doing a DNS query on the domain name obtained from the “To” address. Once the mail server address is obtained a socket connection is initiated with it on standard SMTP port 25. After the initial exchange of SMTP message with the server to set up transfer of mail, the client is ready to send the mail body. Before transferring the Subject and body of the mail, it is encoded using MIME(quoted-printable encoding method) to enable transfer of non standard ASCII character in the ISO-8859-1 character set(mainly to support latin alphabets). Once the encoding is done the body of the mail is sent to the server. If an error occurred during any exchange of SMTP message exchanges with the server, an error is reported back to the user. If the mail transfer was successful, it is reported to the user using a success page. Also, a report email is also sent back to the sender of the mail stating whether the email is successful or not.

If the user has requested for a delayed mail delivery, the email data sent by the user is stored in an Array List. Once the time elapse, a new thread spawn to handle the SMTP server calls of all the emails stored in the ArrayList. At any point in time, the user can request for a static status page showing the emails scheduled to be sent. Also, a report mail is also sent back to the sender of the mail stating whether the email is successful or not.

Discussion and conclusions

Since the only thing that is needed to use this application is Internet connectivity and HTTP browser support, this application could be used as for example a light-weighted service for sending emails from any machine. It could be extra useful for devices with low performances such as low-tier mobile phones.

Possible extensions could be to add CC and BCC (sending email to multiple address) and adding other protocol support such as IMAP to retrieve emails.