Trabajo Práctico Especial 2024

Diseño e Implementación de un SERVIDOR SMTP

1. Equipo 1
2. Descripción detallada de los protocolos y aplicaciones desarrolladas. 2
3. Problemas encontrados durante el diseño y la implementación 2
4. Limitaciones de la aplicación 2
5. Posibles extensiones 2
6. Conclusiones 2
7. Ejemplos de prueba 3
8. Guía de instalación 3
9. Instrucciones para la configuración 4
10. Ejemplos de configuración y monitoreo 4
11. Diseño del proyecto: arquitectura de la aplicación 4

1. Equipo

Nombre	Apellido	Legajo	E-mail
Nicolas	Casella	62.311	**ncasella@itba.edu.ar**
Timoteo	Smart	62.844	**tsmart@itba.edu.ar**
Catalina	Müller	63.199	**camuller@itba.edu.ar**
Manuel	Quesada	63.580	**mquesada@itba.edu.ar**

2. Descripción detallada de los protocolos y aplicaciones desarrolladas.

En esta ocasión, se desarrolló la implementación de un servidor para Simple Mail Transfer Protocol (RFC 5321), con el objetivo de enviar correos electrónicos de forma rentable y eficiente.

El servidor va transicionando hacia diferentes estados a lo largo de las sesiones de conexión que inician los clientes, a través de comandos que son respondidos por el servidor. Empezando por el comando EHLO (o en su defecto HELO para mantener compatibilidad con clientes antiguos) para autenticarse ante el servidor con un nombre de usuario; MAIL FROM, que indica el remitente del correo electrónico; RCPT que recibe por parámetro el receptor, siendo posible ejecutar este comando varias veces para indicar más de un receptor. Al transicionar al estado DATA, es cuando se introduce el contenido a enviar en el correo electrónico, según en el formato establecido en el RFC mencionado. Finalmente, QUIT termina la conexión con el servidor, abortando cualquier transacción en curso.

3. Problemas encontrados durante el diseño y la implementación

Un problema que surgió fue la concurrencia de múltiples clientes simultáneamente. Para implementarlo se hizo uso de threads, atendiendo a cada cliente en un thread. El problema que surgió de esto fue que las variables globales son compartidas entre threads, por lo que varios clientes las accedían y modificaban simultáneamente, y se pisaban los valores entre distintos threads.

Para solucionarlo, se creó una estructura *client_state*, la cual contiene varias de las variables necesarias para el funcionamiento del protocolo, y se agregó a la estructura *smtp*. Por cada conexión se crea una estructura smtp, por lo que las variables pasaron a ser únicas para cada thread.

4. Limitaciones de la aplicación

- Hay un máximo de 600 usuarios concurrentes.
- No permite autenticación de usuario.

5. Posibles extensiones

- **Autenticación de usuario**: el servidor podría soportar autenticación usuario/contraseña (AUTH PLAIN) [RFC4954[1]].
- **Implementación de POP3**: se podría diseñar una implementación de POP3 compatible con el servidor SMTP.
- **Servidor como relay**: el servidor podría aceptar correspondencia para usuarios externos, no solamente locales.

6. Conclusiones

A pesar de las dificultades encontradas a lo largo del desarrollo, el proyecto fue de gran utilidad para adentrarse y familiarizarse con el uso, creación e implementación de protocolos de red.

7. Ejemplos de prueba

Para todos los casos de prueba es necesario

- Seguir los pasos detallados en la guía de instalación

- Abrir una terminal y correr el comando:

  nc -C localhost 2525

Enviar un correo genérico desde localhost

- Dentro de la terminal ejecutar los siguientes comandos:

  EHLO <User>
  MAIL FROM: <User>@smtpd.com
  RCPT TO: <Recipient>
  DATA
  <Contenido del correo>

  <CR><LF>.<CR><LF>

- El correo debería estar encolado y presente en la carpeta mail_dir/<User>

[1] https://datatracker.ietf.org/doc/html/rfc4954

Obtener respuestas de error de parte del servidor

- Dentro de la terminal ejecutar los siguientes comandos:

  RCPT TO: rcpt@smtpd.com
  MAIL FROM: user@smtp.com
  DATA
  EHLO user (*esta línea no devuelve un mensaje de error*)
  MAIL FROM: user
  RCPT TO:
  INVALID COMMAND

Mandar dos correos diferentes concurrentemente desde 2 terminales

- Abrir otra terminal y correr en ambas el comando:

  nc -C localhost 2525

- En la primera terminal ejecutar los siguientes comandos:

  EHLO user1
  MAIL FROM: user1@smtpd.com
  RCPT TO: user2
  DATA
  Subject: para user2
  Este correo está destinado al user2.
  <CR><LF>.<CR><LF>

- En la segunda terminal ejecutar los siguientes comandos:

  EHLO user2
  MAIL FROM: user2@smtpd.com
  RCPT TO: user1
  DATA
  Subject: para user1
  Este correo está destinado al user1.
  <CR><LF>.<CR><LF>

Mandar un correo a distintos remitentes

- Dentro de la terminal ejecutar los siguientes comandos:

EHLO <User>
MAIL FROM: <User>@smtpd.com
RCPT TO: <Recipient1>
RCPT TO: <Recipient2>
RCPT TO: <Recipient3>
RCPT TO: <Recipient4>
RCPT TO: <Recipient5>
DATA
<Contenido del correo>

<CR><LF>.<CR><LF>

Escribir un correo vacío

- Dentro de la terminal ejecutar los siguientes comandos:

EHLO <User>
MAIL FROM: <User>@smtpd.com
RCPT TO: <Recipient>
DATA

<CR><LF>.<CR><LF>

Mandar una secuencia repetida de comandos

- Dentro de la terminal ejecutar los siguientes comandos:

EHLO <User>
EHLO <User>
MAIL FROM: <User>@smtpd.com
MAIL FROM: <User>@smtpd.com
RCPT TO: <Recipient>
RCPT TO: <Recipient>
DATA
<Contenido del correo>

<CR><LF>.<CR><LF>

8. Guía de instalación

Para poder compilar y correr el servidor hace falta descargar o clonar el repositorio de Github: TPE - PDC[2].

Para compilar el código, es necesario correr el siguiente comando dentro de la carpeta */src*:

make clean all

Una vez compilado se creará el archivo smtpd dentro de la misma carpeta. Para ejecutarlo, correr:

./smtpd -<flag> <param>

Para ver una lista de parámetros posibles, se puede usar el flag *-h*

Es necesario disponer de un entorno Linux para poder compilar y ejecutar el servidor.

9. Instrucciones para la configuración

Al momento de escribir documento (21/06/2024), no es posible configurar el servidor. En el futuro esto va a cambiar.

10. Ejemplos de configuración y monitoreo

Un registro de las acciones del servidor puede ser encontrado dentro del archivo *logs/log.txt*.

11. Diseño del proyecto: arquitectura de la aplicación

En proceso.

[2] https://github.com/catamuller/TPE-PDC