



Trabajo Práctico Especial Segundo Cuatrimestre 2024

72.07 Protocolos de Comunicación

Integrantes del grupo

Diego Badín	63551
Diego Rabinovich	63155
Magdalena Taurian	62828
Julieta Techenski	62547

Fecha de entrega: 26 de Noviembre de 2024

Índice

1. Descripción detallada de los protocolos y aplicaciones desarrolladas	3
1.1 Server POP3	3
1.2 Protocolo de Management	5
1.2.1 Ejemplo de uso de los comandos	5
1.3 Aplicación Cliente	8
2. Problemas encontrados durante el diseño y la implementación	9
3. Limitaciones de la aplicación	9
4. Posibles extensiones	9
5. Conclusiones	9
6. Ejemplos de prueba	10
7. Guía de instalación detallada y precisa	16
8. Instrucciones para la configuración	17
9. Ejemplos de configuración y monitoreo	18
10. Documento de diseño del proyecto	20
11. Referencias	20

1. Descripción detallada de los protocolos y aplicaciones desarrolladas

1.1 Server POP3

El servidor POP3 implementado se inicializa junto con sus estructuras que luego van a ser cargadas con información recibida por parámetros cuando se ejecuta de la siguiente forma:

```
./server -d <maildir_path> -U <user>:<pass> [[-u  
<user>:<pass>]...] [OPTION]...
```

Las opciones a agregar son:

- **-h** imprime la ayuda y termina.
- **-l <POP3 addr>** es la dirección donde servirá el servidor POP3. Si no se especifica, su valor default es 0.0.0.0..
- **-L <MGMT addr>** es la dirección donde servirá el servicio de management. Si no se especifica, su valor default es 127.0.0.1..
- **-p <POP3 port>** es el puerto donde correrá el servidor. Si no se especifica, su valor default es 1080.
- **-P <MGMT port>** es el puerto entrante conexiones configuración. Si no se especifica, su valor default es 8080.
- **-u <user>:<password>** son los distintos usuarios que hay en el servidor. En esta opción se deben incluir TODOS los usuarios que aparecen dentro del directorio mails y asignarles una contraseña.
- **-U <user>:<password>** es el usuario y contraseña del administrador del servidor. Estas credenciales son las que se deben mandar siempre que se manden requests con el cliente desarrollado.
- **-v** imprime información sobre la versión y termina.
- **-d <Maildir path>** es el directorio de mails del servidor. Aquí dentro se encontrarán los directorios de los distintos usuarios. Este respeta una cierta estructura, ver sección "Estructura del directorio de mails" del README para más información.
- **-t <cmd>** es el ejecutable para llevar a cabo transformaciones.

Luego, se van extrayendo los argumentos pasados por línea de comandos. Estos argumentos permiten especificar los puertos y direcciones con las que se van a poder conectar los usuarios. También se puede definir usuarios y contraseñas presentes en el directorio, también pasado por argumentos. De no coincidir un directorio con un usuario pasado por parámetros, entonces no se va a poder autenticar, es

decir, no va a poder acceder a los mails. Este directorio es el que contendrá los directorios de los distintos usuarios. Cabe destacar que los directorios siguen una estructura predeterminada y previamente creada.

Posteriormente, se almacena en su estructura los datos y las credenciales de los usuarios autorizados en el servidor. Se prepara el socket para poder recibir conexiones. Dado que se suscribe al servidor para para lectura en el selector y se agrega un handler que se encarga de aceptar de manera pasiva nuevos clientes, se previene un potencial bloqueo. Además se crea un handler para cerrar el servidor, y de esta forma liberar todos los recursos del sistema que fueron reservados.

En el momento en el que llegan nuevos clientes, se establece la conexión para cada uno de ellos. Esta representa una sesión que luego será liberada una vez que el cliente cierre su conexión con el servidor. La sesión se suscribe al select para escritura. Se definen los estados iniciales y finales y sus respectivos handlers

Esta implementación de POP3 soporta los comandos: USER, PASS, STAT, LIST, RETR, DELE, NOOP, RSET y QUIT, siguiendo las especificaciones del RFC 1939. Los mensajes de respuesta se dividen en mensajes de éxito, precedidos de un “+OK”, y mensajes de error, precedidos de un “-ERR”.

La conexión comienza con un estado inicial, INITIAL, que le informa al cliente que la conexión se establece de forma exitosa. Avanza al estado USER_AUTH. En el estado AUTHORIZATION_USER, solo se aceptan dos comandos, USER y QUIT. Luego en el estado AUTHORIZATION_PASSWORD, se aceptan los comandos PASS y QUIT. El estado TRANSACTION tiene los comandos STAT, LIST, RETR, DELE, NOOP, RSET y QUIT. Luego se encuentra el estado UPDATE que solo se accede cuando un cliente realiza un QUIT desde el estado TRANSACTION ya que desde este estado se puede modificar la casilla de mails.

Cuando un cliente escribe un comando, este se almacena temporalmente en un buffer. Mientras todavía queden pedidos por correr (con pedidos se hace alusión a comandos realizados por el cliente), estos se van a ir consumiendo hasta que el buffer no tenga más para leer y si no tiene más, se espera a que el cliente genere otro pedido. Luego se llama a un parser que se fija si es un comando válido o no. De no serlo, consume el resto del pedido y retorna que es INVALID. De ser válido, se verifica si necesita argumentos y los almacena en una estructura con el comando ejecutado.

Luego, en cada estado se verifica si es un comando invalido dentro de sus posibilidades y maneja la función que devolverá un mensaje de éxito o de error. Finalmente, avanza al estado correspondiente luego de la ejecución.

Finalmente, cuando se cierra una conexión se liberan todos los recursos asociados a la misma.

1.2 Protocolo de Management

El protocolo utiliza el protocolo de comunicación **TCP** debido a la fiabilidad y orden que ofrece. TCP asegura que la transmisión de datos entre el cliente y el servidor sea completa, sin errores y en el orden correcto, lo cual es importante porque se realiza el manejo de credenciales sensibles como nombres de usuario y contraseñas.

El objetivo de este protocolo es permitir mediante una aplicación cliente acceso a métricas y la posibilidad de modificar la configuración del servidor.

Para el protocolo de management se armó un cliente que permite al especificar las credenciales válidas por línea de comando como se desarrolla en el README del proyecto, junto con el comando de management que se desea ejecutar. Se establecieron 5 comandos para permitir al usuario administrador realizar modificaciones en tiempo de ejecución del servidor:

- **logs:** Este comando permite al administrador obtener información sobre los accesos de los usuarios al servidor. Se provee información sobre el día hora y el tipo de acceso (login o logout).
- **addu:** Este comando permite al administrador agregar credenciales de usuarios al servidor. El mismo crea carpetas para el usuario agregado.
- **deleu:** Este comando permite al administrador desregistrar usuarios en el servidor. En caso de intentar registrarse nuevamente con los mismos, el servidor no reconocerá las credenciales al reintentar ingresar. En caso de encontrarse activo el mismo en tiempo de ejecución del comando, al intentar correr un comando se cerrará la sesión.
- **metrics:** Este comando permite al administrador obtener información sobre las conexiones históricas, los bytes totales transferidos y la cantidad de usuarios conectados.
- **users:** Este comando permite al administrador obtener un listado de los usuarios conectados y no conectados del servidor de pop3.

Adicionalmente, se implementó una conexión continua que permite que el manager establezca una conexión continua con el servidor y se corran múltiples comandos sin necesitar correr nuevamente el cliente. Esta conexión se establece mediante el uso de netcat en el puerto y dirección especificados para el manager.

1.2.1 Ejemplo de uso de los comandos

Al ejecutar el comando:

```
/bin/server -u p:p -u q:q -u r:r -u s:s -u t:t -u u:u -u v:v -u w:w -u x:x -u y:y -u z:z -d ../maildir/ -p 1081 -P 1082 -L ::1 -U a:a -t /bin/base64
```

Primero se tiene que conectar el cliente administrador:

```
manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTOS$ nc ::1 1082
+OK MGMT server ready.
login a:a
+OK Logged in.
```

Imagen 1.2.1.1: Usuario administrador logueado.

Se ven los usuarios, se agrega un usuario con el comando ADDU <user>:<pass> y se verifica:

```
users
+OK Users quantity: 11
0 -> p           offline
1 -> q           offline
2 -> r           offline
3 -> s           online
4 -> t           online
5 -> u           offline
6 -> v           offline
7 -> w           offline
8 -> x           offline
9 -> y           offline
10 -> z          offline

.
addu k:k
+OK User added successfully.

.
users
+OK Users quantity: 12
0 -> p           offline
1 -> q           offline
2 -> r           offline
3 -> s           online
4 -> t           online
5 -> u           offline
6 -> v           offline
7 -> w           offline
8 -> x           offline
9 -> y           offline
10 -> z          offline
11 -> k          offline
```

Imagen 1.2.1.2: Se agrega un usuario.

Ahora vamos a ver los movimientos de los clientes del POP3, con los logs:

```

logs
+OK 2024-11-27 15:19:59 q LOGIN
2024-11-27 15:20:06 t LOGIN
2024-11-27 15:20:12 s LOGIN
2024-11-27 15:22:16 t LOGOUT
2024-11-27 15:22:25 t LOGIN
2024-11-27 15:26:22 q LOGOUT
.

```

Imagen 1.2.1.3: Se ven los logs de los usuarios.

Luego vemos, nuevamente, los usuarios y eliminamos el que se crea anteriormente:

```

users
+OK Users quantity: 12
0 -> p offline
1 -> q offline
2 -> r offline
3 -> s online
4 -> t online
5 -> u offline
6 -> v offline
7 -> w offline
8 -> x offline
9 -> y offline
10 -> z offline
11 -> k offline
.
deleu k
+OK User deleted successfully.
.
users
+OK Users quantity: 11
0 -> p offline
1 -> q offline
2 -> r offline
3 -> s online
4 -> t online
5 -> u offline
6 -> v offline
7 -> w offline
8 -> x offline
9 -> y offline
10 -> z offline
.

```

Imagen 1.2.1.4: Se elimina un usuario.

Finalmente, vemos las métricas de todo lo sucedido hasta el momento y cerramos la conexión:

```
metrics
+OK
Current connections: 2
Bytes transferred: 4575
Historic connections: 4

.
quit
+OK Logging out.
.
```

Imagen 1.2.1.5: Se ven las métricas y la salida.

1.3 Aplicación Cliente

La aplicación cliente recibe los argumentos por entrada estándar y se encarga de procesarlos correctamente o, de no haber, termina la ejecución e informa al cliente como se debe ejecutar y sus opciones. A continuación explicaremos cómo se ejecuta:

Teniendo el servidor en ejecución y corriendo el siguiente comando

```
./manager_client -S <user>:<password> [OPTION]...
```

- **-S <user>:<password>** es el usuario y contraseña del administrador del servidor. Este debió haber sido especificado al ejecutar el servidor con el flag -U.
- **-h** imprime la ayuda y termina.
- **-H <host>** es la dirección donde servirá el servicio de management. Si no se especifica, su valor default es 127.0.0.1..
- **-P <port>** es el puerto entrante conexiones configuración. Si no se especifica, su valor default es 8080.
- **-U** envía una solicitud para obtener los usuarios registrados.
- **-A <user>:<password>** envía una solicitud para agregar un usuario al servidor.
- **-D <user>** envía una solicitud para eliminar un usuario al servidor.
- **-M** envía una solicitud para obtener métricas específicas del servidor.
- **-L** envía una solicitud para obtener los logs del servidor.

Luego de obtener los argumentos, se los almacena en una estructura y se realiza una conexión con el socket correspondiente al protocolo de monitoreo. Este le envía el comando para su ejecución y luego se liberan los recursos.

2. Problemas encontrados durante el diseño y la implementación

En la implementación los problemas encontrados fueron principalmente en relación a la primera sección de definición del funcionamiento básico del servidor al migrar a la configuración no bloqueante del mismo con selects. También se tomó la decisión de implementar un parser propio que tuvo que ser modificado en varias ocasiones durante el desarrollo del trabajo para su optimización de recursos y para la incorporación de pipelining.

Sin embargo, ambas dificultades pudieron ser resueltas sin problemas adicionales.

3. Limitaciones de la aplicación

La principal limitación de la implementación del servidor POP3 es que, a pesar de tener los comandos necesarios para su ejecución, no se incorporaron todos los comandos que el protocolo permite. Esto se debe a que el equipo prioriza el protocolo de Management.

La aplicación fue implementada de forma que se deberían soportar como máximo 500 usuarios conectados.

4. Posibles extensiones

La principal adición a la implementación serían aquellos comandos faltantes del protocolo POP3 como TOP, UIDL y APOP. Luego se podrían agrupar los logs en un archivo para una mayor organización. Además, actualmente los mensajes informativos del servidor son genéricos, se podría indicar el usuario que realiza cada pedido.

Otra de las posibles extensiones podría ser incorporar comandos adicionales al server de management que permite realizar configuraciones adicionales como incrementar cantidad de usuarios soportados o modificar los tamaños de los buffers. También podría ser un agregado interesante el cálculo de métricas no volátiles para poder mantener los datos independientemente del estado del servidor.

5. Conclusiones

En el desarrollo del trabajo logramos implementar un protocolo existente y, además, diseñar e implementar un protocolo propio, lo que nos permitió abordar de manera práctica los conceptos teóricos estudiados y trabajados durante la materia. Esta experiencia nos brindó una comprensión más profunda de los principios de las comunicaciones en red, desde el diseño lógico hasta la implementación y prueba de un protocolo funcional.

Por un lado, al trabajar con un protocolo ya existente, adquirimos experiencia en la interpretación y aplicación de estándares definidos, lo que nos permitió entender cómo los protocolos establecen reglas claras para la transmisión de datos.

Por otro lado, el desarrollo de un protocolo propio representó un reto aún mayor, ya que debimos diseñar desde cero las reglas y procedimientos para la transmisión de información.

En conclusión, este trabajo integró aspectos teóricos y prácticos que reforzaron los contenidos abordados durante la materia y poder observar cómo se realizan las aplicaciones prácticas de los mismos.

6. Ejemplos de prueba

A continuación se enumeran las pruebas realizadas para demostrar el correcto funcionamiento del servidor POP3.

Al ejecutar el comando:

```
./bin/server -u x:x -u y:y -u z:z -d ../maildir -p 1081 -P 1082 -L  
::1 -U a:a -t /bin/base64
```

- Soporte de IPv4 e IPv6 tanto del manager como del pop3

Imagen 6.3: Conexión IPv6

En las Imágenes 6.2 y 6.3 se muestra que efectivamente funcionan utilizando el comando `netcat`.

- Intento de autenticación fallido y uno exitoso:

```
manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTOS$ nc localhost 1081
+OK POP3 server ready.
user y
+OK
pass z
-ERR [AUTH] Authentication failed.
user y
+OK
pass y
+OK Logged in.
```

Imagen 6.4: Autenticación en POP3.

- Se listan todos los mensajes, se lista la información del mensaje 1 y se obtiene dicho mensaje en base 64 pues estaba esa transformación:

```
list
+OK 5 messages (2497 octets)
1 507
2 452
3 530
4 472
5 536
.
list 1
+OK 1 507
retr 1
+OK message follows
QXN1bnRvOjBQZXJkw60gdW4gYXJjaGl2by4uLiDCoXkgbGEgZGlnbmlkYWQhIPCfkrSKSG9sYSBb
Tm9tYnJlIGRlbCBZNX0aW5hdGFyaW9dLAoKRXN0b3kgZW4gYnVzY2EgZGUgdW4gYXJjaGl2byBx
dWUsIGp1cmFyw61hLCBndWFyZM0pIGVuIGFsZ806biBsdWdhciBzw7pwZXIgbM0zZ2ljby4uLiBo
YXN0YSBxdWUgbWUgZGkgY3VlbnRhIGRlIHf1ZSBsbyBndWFyZM0pIGNvbW8gImFzZmtnamRrX2Zp
bmFsX2ZpbmFsX0ZJTkFMX3YyLmRvY3giLgoKU2kgcG9yIGNhc3VhbGlkYWQgbG8gZW5jdWVudHJh
cyBlbiBlbCBzZXJ2aWRvciBvIGVzY3VjaGFzIHJlbW9yZXMGZGUgc3UgcGFyYWRLcm8sIHBvciBm
YXZvciBhdsc0tc2FtZS4gTWllbnRyYXMgdGFudG8sIHBvciB21ldG8gcmVub21icmFyIG1pcyBhcmNo
aXZvcyBjb21vIHVuYSBwZXJzb25hIG5vcmlhbCAobyBlc28gaW50ZW50YXLDqSkuCgpHcmFjaWFz
IHBvciB0dSBheXVksYswKQ2FybGl0b3MsIGVsIGRlc29yZ2FuaXphZG8gZGlnaXRhbC4K
.
```

Imagen 6.5: Listado y obtención de mensajes en POP3.

- El cliente borra el mensaje 1, se ve que ya no pertenece a los listados pero cuando lo recuperamos, vuelve a ser una opción.

```
dele 1
+OK Marked to be deleted.
dele 1
-ERR Message is deleted.
list
+OK 4 messages (1990 octets)
2 452
3 530
4 472
5 536
.
rset
+OK
list
+OK 5 messages (2497 octets)
1 507
2 452
3 530
4 472
5 536
.
```

Imagen 6.6: Eliminación y recuperación de mensajes en POP3.

- El cliente ejecuta los comandos `noop` y `stat`:

```
noop
+OK
stat
+OK 5 2497
```

Imagen 6.7: Comandos 'noop' y 'stat' en POP3.

- El cliente cierra la conexión:

```
quit
+OK Logging out.
```

Imagen 6.8: Cierre de conexión en POP3.

- Ahora veamos algunos mensajes de error:

```

list
+OK 5 messages (2497 octets)
1 507
2 452
3 530
4 472
5 536
.
list 7
-ERR There's no message 7
list 4vrej1berl
-ERR Noise after message number: vrej1berl
retr 7
-ERR There's no message 7
retr
-ERR Invalid message number:

```

Imagen 6.9: Algunos posibles errores en POP3.

- Obtener listado de mensajes a través de `curl`:

```

manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTO$ curl -u z:z pop3://localhost:1081
1 507
2 452
3 530
4 472
5 536
manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTO$ curl -v -u z:z pop3://localhost:1081
* Host localhost:1081 was resolved.
* IPv6: ::1
* IPv4: 127.0.0.1
* Trying [::1]:1081...
* connect to ::1 port 1081 from ::1 port 44926 failed: Connection refused
* Trying 127.0.0.1:1081...
* Connected to localhost (127.0.0.1) port 1081
< +OK POP3 server ready.
> CAPA
< -ERR Unknown command.
> USER z
< +OK
> PASS z
< +OK Logged in.
> LIST
< +OK 5 messages (2497 octets)
1 507
2 452
3 530
4 472
5 536
* Connection #0 to host localhost left intact

```

Imagen 6.10: Output de listar los mensajes mediante 'curl'.

- Obtener un mensaje a través de `curl`:

```

manuorodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTOS$ curl -u z:z pop3://localhost:1081/1
QXN1bnRvOiBQZXJkw60gdW4gYXJjaGl2by4uLiDCoXkgbGEgZGlbnmlkYWQhIPCfkrsKSG9sYSBb
Tm9tYnJlIGRlbcBEZXN0aw5hdGFyaW9dLAoKRXN0b3kgZW4gYnVzY2EgZGUgdW4gYXJjaGl2byBx
dWUsIGp1cmFyw61hLCBndWfyZM0pIGVuIGFsZ806biBsdWdhciBzw7pwZXIgbM0zZ2ljby4uLiBo
YXN0YSBxdWUgbWUgZGkgY3VlbnRhIGRlIHf1ZSBSbyBndWfyZM0pIGNvbW8gImFzZmtnamRrX2Zp
bmFsX2ZpbmFsX0ZJTkFMX3YyLmRvY3giLgoKU2kgcG9yIGNhc3VhbGkYwQgbG8gZW5jdWVudHJh
cyBlbiBlbCBzZXJ2aWRvciBvIGVzY3VjaGFzIHJlbW9yZXMGZGUgc3UgcGFyYWRlcm8sIHBvciBm
YXZvciBhds0tc2FtZS4gTWlbnRyYXMGdGFudG8sIHB5b21ldG8gcmVub21icmFyIG1pcyBhcmNo
aXZvcyBjb21vIHVuYSBwZXJzb25hIG5vcm1hbCAobyBlc28gaW50ZW50YXLDqSkuCgpHcmFjaWFz
IHBvciB0dSBheXVkySwKQ2FybGl0b3MsIGVsIGRlc29yZ2FuaXphZG8gZGlnaXRhbC4K

manuorodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTOS$ curl -v -u z:z pop3://localhost:1081/1
* Host localhost:1081 was resolved.
* IPv6: ::1
* IPv4: 127.0.0.1
* Trying [::1]:1081...
* connect to ::1 port 1081 from ::1 port 58138 failed: Connection refused
* Trying 127.0.0.1:1081...
* Connected to localhost (127.0.0.1) port 1081
< +OK POP3 server ready.
> CAPA
< -ERR Unknown command.
> USER z
< +OK
> PASS z
< +OK Logged in.
> RETR 1
< +OK message follows
QXN1bnRvOiBQZXJkw60gdW4gYXJjaGl2by4uLiDCoXkgbGEgZGlbnmlkYWQhIPCfkrsKSG9sYSBb
Tm9tYnJlIGRlbcBEZXN0aw5hdGFyaW9dLAoKRXN0b3kgZW4gYnVzY2EgZGUgdW4gYXJjaGl2byBx
dWUsIGp1cmFyw61hLCBndWfyZM0pIGVuIGFsZ806biBsdWdhciBzw7pwZXIgbM0zZ2ljby4uLiBo
YXN0YSBxdWUgbWUgZGkgY3VlbnRhIGRlIHf1ZSBSbyBndWfyZM0pIGNvbW8gImFzZmtnamRrX2Zp
bmFsX2ZpbmFsX0ZJTkFMX3YyLmRvY3giLgoKU2kgcG9yIGNhc3VhbGkYwQgbG8gZW5jdWVudHJh
cyBlbiBlbCBzZXJ2aWRvciBvIGVzY3VjaGFzIHJlbW9yZXMGZGUgc3UgcGFyYWRlcm8sIHBvciBm
YXZvciBhds0tc2FtZS4gTWlbnRyYXMGdGFudG8sIHB5b21ldG8gcmVub21icmFyIG1pcyBhcmNo
aXZvcyBjb21vIHVuYSBwZXJzb25hIG5vcm1hbCAobyBlc28gaW50ZW50YXLDqSkuCgpHcmFjaWFz
IHBvciB0dSBheXVkySwKQ2FybGl0b3MsIGVsIGRlc29yZ2FuaXphZG8gZGlnaXRhbC4K

* Connection #0 to host localhost left intact

```

Imagen 6.11: Output de obtener el mensaje 1 mediante 'curl'.

- Transformaciones utilizando transformaciones con el comando base64.

```

manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTOS$ nc localhost 1081
+OK POP3 server ready.
user s
+OK
pass s
+OK Logged in.
retr 1
+OK message follows
QXN1bnRvOibQZXJkw60gdW4gYXJjaGl2by4uLiDCoXkgbGEgZGlnbmlkYWQhIPCfkrsKSG9sYSBb
Tm9tYnJlIGRlbCBZNXN0aw5hdGFyaW9dLAoKRXN0b3kgZW4gYnVzY2EgZGUgdW4gYXJjaGl2byBx
dWUsIGp1cmFyw61hLCBndWFyZM0pIGVuIGFsZ806biBsdWdhciBzw7pwZXIgbM0zZ2ljby4uLiBo
YXN0YSBxdWUgbWUgZGkgY3VlbnRhIGRlIHf1ZSBsbyBndWFyZM0pIGNvbW8gImFzZmtnamRrX2Zp
bmFsX2ZpbmFsX0ZJTtFMX3YyLmRvY3giLgoKU2kgcG9yIGNhc3VhbGlkYWQgbG8gZW5jdWVudHJh
cyBlbiBlbCBzZXJ2aWRvciBvIGVzY3VjaGFzIHJlbW9yZXMGZGUgc3UgcGFyYWRLcm8sIHBvciBm
YXZvciBhds0tc2FtZS4gTWllbnRyYXMgdGFudG8sIHByb21ldG8gcmVub2licmFyIGlpcyBhcmNo
aXZvcyBjb21vIHVuYSBwZXJzb25hIG5vcm1hbCAobyBlc28gaW50ZW50YXLDqSkuCgpHcmFjaWFz
IHBvciB0dSBheXVKSXVwKQ2FybGl0b3MsIGVsIGRlc29yZ2FuaXphZG8gZGlnaXRhbC4K

.
list
+OK 5 messages (2497 octets)
1 507
2 452
3 530
4 472
5 536
.

```

Imagen 6.12: Obtención de un mail mediante el servidor POP3 con transformación de /bin/base64.

```

manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/maildir/s/new$ cat mail4.txt | base64
QXN1bnRvOibQZXJkw60gdW4gYXJjaGl2by4uLiDCoXkgbGEgZGlnbmlkYWQhIPCfkrsKSG9sYSBb
Tm9tYnJlIGRlbCBZNXN0aw5hdGFyaW9dLAoKRXN0b3kgZW4gYnVzY2EgZGUgdW4gYXJjaGl2byBx
dWUsIGp1cmFyw61hLCBndWFyZM0pIGVuIGFsZ806biBsdWdhciBzw7pwZXIgbM0zZ2ljby4uLiBo
YXN0YSBxdWUgbWUgZGkgY3VlbnRhIGRlIHf1ZSBsbyBndWFyZM0pIGNvbW8gImFzZmtnamRrX2Zp
bmFsX2ZpbmFsX0ZJTtFMX3YyLmRvY3giLgoKU2kgcG9yIGNhc3VhbGlkYWQgbG8gZW5jdWVudHJh
cyBlbiBlbCBzZXJ2aWRvciBvIGVzY3VjaGFzIHJlbW9yZXMGZGUgc3UgcGFyYWRLcm8sIHBvciBm
YXZvciBhds0tc2FtZS4gTWllbnRyYXMgdGFudG8sIHByb21ldG8gcmVub2licmFyIGlpcyBhcmNo
aXZvcyBjb21vIHVuYSBwZXJzb25hIG5vcm1hbCAobyBlc28gaW50ZW50YXLDqSkuCgpHcmFjaWFz
IHBvciB0dSBheXVKSXVwKQ2FybGl0b3MsIGVsIGRlc29yZ2FuaXphZG8gZGlnaXRhbC4K
manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/maildir/s/new$

```

Imagen 6.13: Obtención de un mail en base64.

- Se realizaron pruebas de más de 100 usuarios para verificar las conexiones exitosas con el archivo `/src/testing/test_selects.py`

7. Guia de instalación detallada y precisa

Antes de ejecutar el servidor, se debe correr el comando `make all` en la raíz del proyecto. En la carpeta `bin` se habrán generado los archivos `server` y `manager_client`.

Para ejecutar el servidor se debe correr los siguientes comandos dentro del directorio *bin*:

```
./server -d <maildir_path> -U <user>:<pass> [[-u  
<user>:<pass>]...] [OPTION]...
```

Las opciones a agregar son:

- **-h** imprime la ayuda y termina.
- **-l <POP3 addr>** es la dirección donde servirá el servidor POP3. Si no se especifica, su valor default es 0.0.0.0..
- **-L <MGMT addr>** es la dirección donde servirá el servicio de management. Si no se especifica, su valor default es 127.0.0.1..
- **-p <POP3 port>** es el puerto donde correrá el servidor. Si no se especifica, su valor default es 1080.
- **-P <MGMT port>** es el puerto entrante conexiones configuración. Si no se especifica, su valor default es 8080.
- **-u <user>:<password>** son los distintos usuarios que hay en el servidor. En esta opción se deben incluir TODOS los usuarios que aparecen dentro del directorio mails y asignarles una contraseña.
- **-U <user>:<password>** es el usuario y contraseña del administrador del servidor. Estas credenciales son las que se deben mandar siempre que se manden requests con el cliente desarrollado.
- **-v** imprime información sobre la versión versión y termina.
- **-d <Maildir path>** es el directorio de mails del servidor. Aquí dentro se encontrarán los directorios de los distintos usuarios. Este respeta una cierta estructura, ver sección "Estructura del directorio de mails" del README para más información.
- **-t <cmd>** es el ejecutable para llevar a cabo transformaciones.

8. Instrucciones para la configuración

La aplicación cliente (monitoreo) se puede ejecutar de dos formas:

1. Utilizando netcat y siguiendo los comandos descritos en la [sección 1.2](#).
2. Corriendo el siguiente comando

```
./manager_client -S <user>:<password> [OPTION]...
```

- **-S <user>:<password>** es el usuario y contraseña del administrador del servidor. Este debió haber sido especificado al ejecutar el servidor con el flag -U.
- **-h** imprime la ayuda y termina.

- **-H <host>** es la dirección donde servirá el servicio de management. Si no se especifica, su valor default es 127.0.0.1..
- **-P <port>** es el puerto entrante conexiones configuración. Si no se especifica, su valor default es 8080.
- **-U** envía una solicitud para obtener los usuarios registrados.
- **-A <user>:<password>** envía una solicitud para agregar un usuario al servidor.
- **-D <user>** envía una solicitud para eliminar un usuario al servidor.
- **-M** envía una solicitud para obtener métricas específicas del servidor.
- **-L** envía una solicitud para obtener los logs del servidor.

Finalmente, el programa finaliza exitosamente. Si se desea correr otro comando, se debe ejecutar el programa nuevamente de la manera descrita anteriormente.

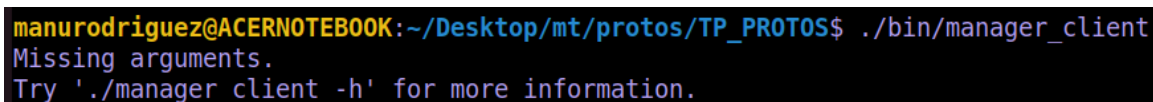
9. Ejemplos de configuración y monitoreo

Al ejecutar el comando:

```
/bin/server -u p:p -u q:q -u r:r -u s:s -u t:t -u u:u -u v:v -u w:w -u x:x -u y:y -u z:z -d ../maildir/ -p 1081 -P 1082 -L ::1 -U a:a -t /bin/base64
```

Utilizando la aplicación cliente, se mostrarán ejemplos de configuración y de monitoreo del servidor a través del protocolo implementado.

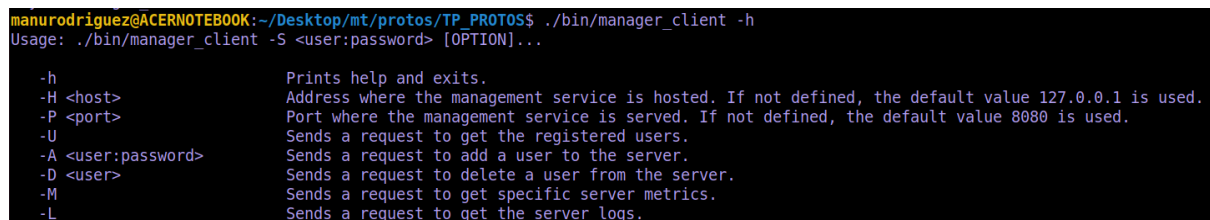
- Un cliente nuevo intenta usar la aplicación cliente pero no sabe cómo es su funcionamiento.



```
manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTO$ ./bin/manager_client
Missing arguments.
Try './manager_client -h' for more information.
```

Imagen 9.1: Ejecución de ./bin/manager_client

- Luego el cliente ejecuta el comando sugerido y obtiene más información sobre los comandos disponibles:



```
manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTO$ ./bin/manager_client -h
Usage: ./bin/manager_client -S <user:password> [OPTION]...

-h          Prints help and exits.
-H <host>   Address where the management service is hosted. If not defined, the default value 127.0.0.1 is used.
-P <port>   Port where the management service is served. If not defined, the default value 8080 is used.
-U          Sends a request to get the registered users.
-A <user:password> Sends a request to add a user to the server.
-D <user>   Sends a request to delete a user from the server.
-M          Sends a request to get specific server metrics.
-L          Sends a request to get the server logs.
```

Imagen 9.2: Ejecución de ./bin/manager_client -h

- Luego el cliente puede ver una lista de los usuarios activos al ejecutar:

```
manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTOS$ ./bin/manager_client -S a:a -P 1082 -H ::1 -U
+OK Users quantity: 11
0 -> p          offline
1 -> q          online
2 -> r          offline
3 -> s          online
4 -> t          online
5 -> u          offline
6 -> v          offline
7 -> w          offline
8 -> x          offline
9 -> y          offline
10 -> z         offline
.
```

Imagen 9.3: Ejecución con el flag -U.

- Luego podríamos ver las métricas del momento:

```
manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTOS$ ./bin/manager_client -S a:a -P 1082 -H ::1 -M
+OK
Current connections: 3
Bytes transferred: 618
Historic connections: 3
.
```

Imagen 9.4: Ejecución con el flag -M.

- También los movimientos de los usuarios en el server:

```
manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTOS$ ./bin/manager_client -S a:a -P 1082 -H ::1 -L
+OK 2024-11-27 15:19:59 q          LOGIN
2024-11-27 15:20:06 t          LOGIN
2024-11-27 15:20:12 s          LOGIN
2024-11-27 15:22:16 t          LOGOUT
2024-11-27 15:22:25 t          LOGIN
.
```

Imagen 9.5: Ejecución con el flag -L.

- Ya finalizando se agrega un usuario:

```
manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTOS$ ./bin/manager_client -S a:a -P 1082 -H ::1 -A k:k
+OK User added successfully.
.
manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTOS$ ./bin/manager_client -S a:a -P 1082 -H ::1 -U
+OK Users quantity: 12
0 -> p          offline
1 -> q          online
2 -> r          offline
3 -> s          online
4 -> t          online
5 -> u          offline
6 -> v          offline
7 -> w          offline
8 -> x          offline
9 -> y          offline
10 -> z         offline
11 -> k         offline
.
```

Imagen 9.6: Ejecución con el flag -A.

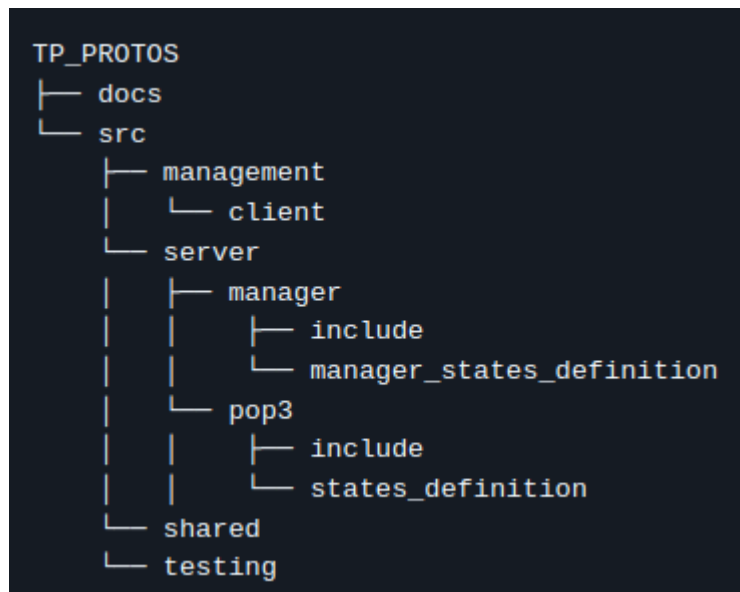
- Finalmente se elimina el usuario

```
manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTOS$ ./bin/manager_client -S a:a -P 1082 -H ::1 -D k
+OK User deleted successfully.
.
manurodriguez@ACERNOTEBOOK:~/Desktop/mt/protos/TP_PROTOS$ ./bin/manager_client -S a:a -P 1082 -H ::1 -U
+OK Users quantity: 11
0 -> p      offline
1 -> q      online
2 -> r      offline
3 -> s      online
4 -> t      online
5 -> u      offline
6 -> v      offline
7 -> w      offline
8 -> x      offline
9 -> y      offline
10 -> z     offline
.
```

Imagen 9.7: Ejecución con el flag -D.

10. Documento de diseño del proyecto

El proyecto tiene la siguiente estructura:



11. Referencias

- [RFC1939] Myers, J. and M. Rose, "Post Office Protocol - Version 3", STD 53, RFC 1939, May 1996.