Trabajo Práctico Especial

Materia: Protocolos de Comunicación

Profesores:
- Zunino, Fernando
- Codagnone, Juan
- Sotuyo Dodero, Juan

Integrantes:
- Magnorsky, Alejandro
- Merchante, Mariano
- Pose, Jimena

Fecha de entrega: 15/06/11

Índice

Descripción de los protocolos desarrollados

Protocolo de estadísticas

Este protocolo no es case-sensitive, excepto cuando se introduce el nombre de administrador y su contraseña en el comando `login`. Las respuestas a los comandos comienzan con "OK" si no hubo errores y con "ERROR" en caso contrario. Las respuestas de error van acompañadas de un mensaje informativo.
El primer comando que solicita introducir este protocolo es:

- `login <username> <password>`
  Donde "username" corresponde al nombre de administrador y "password" es su contraseña. Si la verificación se realiza correctamente, el cliente puede usar el resto de los comandos del servicio. Si el usuario o contraseña son incorrectos el usuario recibe un mensaje de error que lo notifica de eso y se cierra la conexión.

Los comandos propios del servicio son:

- `user <username>`
  Se utiliza para que las estadísticas que devuelva el servidor sean solamente aquellas que correspondan al usuario cuyo nombre es username.
- `public`
  Se utiliza para que las estadísticas que devuelva el servidor correspondan a la totalidad de usuarios como un grupo. El servidor comienza en este modo por default.

- `aq`
  Devuelve la cantidad de accesos.

- `rq`
  Devuelve la cantidad de correos leídos.

- `bt`
  Devuelve la cantidad de bytes transferidos.

- `dq`
  Devuelve la cantidad de correos borrados.

- `ah`
  Devuelve el histograma de accesos.

Protocolo de configuración

Este protocolo no es case-sensitive, excepto cuando se introduce el nombre de administrador y su contraseña en el comando `login`. Las respuestas a los comandos comienzan con “OK” si no hubo errores y con “Error” en caso contrario. Las respuestas de error van acompañadas de un mensaje informativo.
El primer comando que solicita introducir este protocolo es:

- `login <username> <password>`
  Donde “username” corresponde al nombre de administrador y “password” es su contraseña. Si la verificación se realiza correctamente, el cliente puede usar el resto de los comandos del servicio. Si el usuario o contraseña son incorrectos el usuario recibe un mensaje de error que lo notifica de eso y se cierra la conexión.

Los comandos propios del servicio son:

- `user <username>`
  Indica que los siguientes cambios serán sobre el usuario username.

- `blacklist`
  Agrega un IP bien formado a la lista negra de ips

- `commit`
  Guarda los cambios

- `exit`
  Salir del sistema

- `help`
  Muestra una pantalla de ayuda por si un usuario accede al protocolo de forma manual.

- `set rotate <boolean>`
  Rota las imagenes que retorna el proxy.

- `set leet <boolean>`
  Modifica el texto plano aplicandole una transformacion l33t.

- `set maxlogins <integer>`
  Indica la cantidad máxima de veces por dia que un usuario puede acceder

- `set server <string>`
  Modifica el servidor default del usuario

- `set schedule_min | schedule_max <integer>`
  Indica el rango de horarios en el que el usuario puede ingresar. El rango es en minutos, entre 0 y 1440.

- `set date_min | date_max <fecha>`
  Modifica el rango de fechas de envio del mail origen en el que el usuario puede eliminar mails.

- `set size_min | size_max <integer>`
  Modifica el rango de tamaños del mensaje que puede eliminar el usuario.

- `set structure <string>`
  Agrega limitaciones a la eliminación de mensaje segun la estructura de dicho mensaje. Posibles valores: ATTACH, NOATTACH o SENDERCOUNT_G <minima cantidad de remitentes>, donde SENDERCOUNT_G impide que se borren mails con menos remitentes que el indicado.

- `add content <string>`
  Agrega una restricción de eliminación por el contenido. Ejemplo: text/plain

- `add header <string>`
  Agrega una restricción de eliminación por un patrón en el header. Ejemplo: [a-zA-Z]*

- `add sender <string>`
  Agrega una restricción de eliminación según el remitente.

Problemas encontrados durante el diseño y la implementación

La utilización de XML como medio de persistencia requirió muchas más clases y manejo nuestro de lo que esperábamos. Se separaron las capas de Dominio y de Persistencia, lo cual implicó agregar una cantidad de código no esperada.

Decidimos utilizar el patrón de pipes & filters para gran parte del procesamiento que realiza el handler del cliente POP3 para modularizar la lógica e independizar los distintos chequeos y transformaciones.

Al momento de levantar el mensaje que llegaba desde el servidor POP3, tuvimos que descomponerlo en base a los distintos tipos de contenido (Content-Type), decodificar los mismos (base64 y quoted-printable) para poder aplicarle cambios y antes de devolverle el mensaje al cliente POP3, volver a rearmarlo encodeando cada contenido como estaba originalmente y poniendolo en su lugar correspondiente para que las operaciones que realizó el Proxy POP3 sea transparente al mismo.

Cuando usamos un MUA como cliente POP3 en lugar de hacerlo por la terminal, nos dimos cuenta que faltaba mandar una serie de mensajes (como el Welcome, por ejemplo) que no los estábamos enviando.

En cuanto al multiplexor de cuentas, tuvimos problemas para integrarlo con el MUA, ya que cuando este enviaba el mensaje CAPA el proxi todavía no estaba conectado al servidor (ya que para conectarse al servidor se necesita el usuario, y el MUA envía CAPA antes de enviar USER), por lo tanto el MUA no obtenía respuesta alguna.

Limitaciones de la aplicación

El Proxy POP3 no puede conectarse a servidores POP3 que sólo usen el puerto 995 ya que dicho puerto requiere encriptación.

Posibles extensiones

Se le podría agregar manejo de videos o de documentos de Office, por ejemplo, por lo que habria que implementar también las correspondientes clases que hereden de Content.

También podria agregarsele más tipos de Content-Transfer-Encoding para hacerlo compatible con otros tipos de encoding.

Conclusión

El proxy POP3 funciona de manera transparente al cliente pudiéndolo configurar, administrar y utilizar para realizar operaciones de trasformación sobre el mensaje o control sobre las operaciones que realiza el cliente. A su vez, es fácilmente modificable debido al patrón de pipes & filters usado.