PROTOCOLOS DE COMUNICACIÓN

Trabajo Práctico Especial

Arquitectura de la aplicación

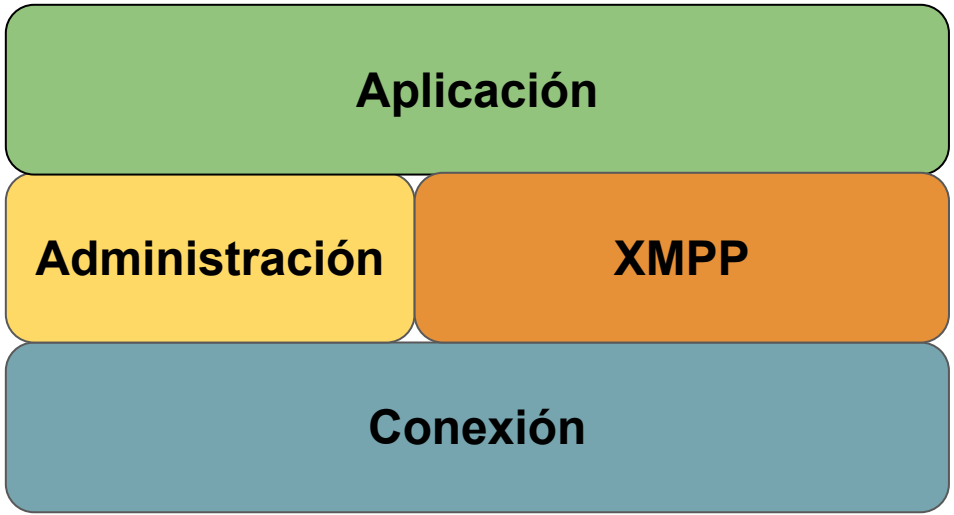
Estructura básica de la aplicación

- Separación en módulos
- *Single thread*
- Múltiples *selectors*

Separación en módulos

- Módulo de conexión
- Módulo de protocolo XMPP
- Módulo de protocolo de administración
- Módulo de aplicación

Separación en módulos



Separación en módulos

- Cada módulo tiene una tarea específica.
- Los módulos pueden estar divididos, a su vez, en subcapas.
- Por ejemplo, el módulo del protocolo XMPP puede definir una capa que *parsee* mensajes entrantes y otra que los interprete.

Single threading

- Un único thread para:
  - Atender toda la entrada entrada/salida de datos
  - Procesamiento de datos
- Posibilidad de extender a dos o más *threads*

Múltiples *selectors*

- Un *selector* dedicado al XMPP *server*.
- Un *selector* para todas las conexiones salientes (hacia servidores XMPP).
- Un *selector* dedicado al *administration server*.

Módulo de conexión

- Encargado de realizar conexiones de red.
- Utiliza el protocolo TCP.
- Multiplexa eventos de entrada y salida utilizando Selector de java.nio.
- Procesa entrada y salida en el *thread* principal.
-

Módulo de conexión

- Realiza conexiones de red.
- Utiliza el protocolo TCP.
- Multiplexa eventos de entrada y salida utilizando *Selectors*
- Define interfaces con métodos a ejecutar ante un evento de entrada/salida.
- No crea *threads* adicionales

Módulo de protocolo XMPP

- Interpreta el protocolo XMPP
- Se comunica con la capa de conexión

Módulo de protocolo de administración

- Interpreta el protocolo de configuración
- Se comunica con la capa de conexión

Módulo de aplicación

- Contiene el *main loop*.
- Decide a qué protocolo le da mayor prioridad.
- Toma decisiones en base a los mensajes entrantes.
- Contiene todos los parámetros de configuración

Protocolos de Administración/ Métricas

Protocolo de Administración

Los siguientes son los comandos del protocolo de Administración en notación ABNF:

<table>
<tr><td colspan="2">Command = authC/quitC/helpC/mtrcC /cnfgC/blckC/unblckC/mplxC/l337C/unl337C</td></tr>
<tr><td>authC = “AUTH” SP admin-name:password
CRLF
quitC = “QUIT” CRLF
helpC = “HELP” CRLF
mtrcC = “MTRC” [SP metric-name] CRLF
cnfgC =“CNFG” CRLF</td><td>blckC = “BLCK” SP user_jid CRLF
unblckC=“UNBLCK” SP username CRLF
mplxC =“MPLX” SP (username/DEFAULT)
SP “>” SP (server-name / DEFAULT) CRLF
l337C =“L337” CRLF
unl337C =“UNL337” CRLF</td></tr>
<tr><td>metric-name=1*1024 (ALPHA/DIGIT)
admin-name=1*1024 (ALPHA/DIGIT)
password = 1*1024 (ALPHA/DIGIT)</td><td>user-jid = jid; as defined in RFC 6122
server-name = ???
;UTF-8-character, a UTF-8 encoded character as defined in RFC 3629</td></tr>
</table>

Respuestas para cada comando

AUTH	if already logged in: “Must QUIT to log in” CRLF if the username/password is incorrect: “INCORRECT” CRLF else: “OK” CRLF
HELP	List of all commands, with format: *(<COMMAND_NAME> CRLF) CRLF
MTRC	If a metric is specified: <METRIC_NAME> SP <METRIC_VALUE> CRLF Otherwise: *(<METRIC_NAME> SP <METRIC_VALUE> CRLF) CRLF
CNFG	List of all current configurations, with format: *(<CONFIG_NAME> CRLF ((“NONE” CRLF) / *(<CONFIG_VALUE> CRLF))) CRLF
MPLX	If username/serverID specified, “OK” CRLF
QUIT | BLCK | UNBLOCK | L337 | UNL337: “OK” CRLF	
If user not loggedIn and command is not USER: “Must LogIn” CRLF	
If unknow command: “unknow command. Type HELP for help” CRLF	

Ejemplo de uso del protocolo

```
C: AUTH 42:42
S: OK
C: HELP
S:    Commands are:
      MTRC
      CNFG
      MPLX
      BLCK
      UNBLCK
      L337
      UNL337
C: CNFG
S: BLCK NONE
   MPLX NONE
   L337 OFF
   DEFAULT_SERVER
xmpp.example.org
C:BLCK jorge
S:OK
C:BLCK pedro
S:OK
C:CNFG
S:BLCK pedro
      jorge
      MPLX NONE
      L337 OFF
      DEFAULT_SERVER
xmpp.example.org
C:UNBLCK pedro
S:OK
C:MPLX jorge >
xmpp.b.example.org
S:OK
C:L337
S:OK
C:L337
S:OK
C:MPLX DEFAULT >
xmpp.newdefault.example.org
S:OK
C:MPLX pedro >
xmpp.example.org
S:OK
```

Ejemplo de uso del protocolo

C:MPLX pedro > DEFAULT
S:OK
C:
S:Accesses 37
Bytes: 34567
C: MTRC Accesses
S: Accesses 37
C: CNFG
S:BLCK jorge
MPLX jorge > xmpp.b.example.org
L337 ON
DEFAULT_SERVER
xmpp.newdefault.example.org

C:MPLX Javier >
javier.com.ar.edu.ar.com
S:OK
C:MTRC lalala
S:UNKNOW MTRC
C: BLCK sarasa.com.ar_hola?
S:OK
C: MPLX DEFAULT > DEFAULT
S: OK
C:UNBLCK xXJorge96Xx
S:OK
C:QUIT
S:OK

Mensajes de tamaño muy grande

- Opciones pensadas
  - Enviar mensajes antes de recibirlo entero
  - Establecer un tamaño máximo de mensaje. En caso de superarlo, se descarta y se notifica el emisor.
- Opción preferida: Enviar mensajes a medida que van llegando
  - Ahorra memoria ya que no tenemos que almacenar mensajes

Asumpciones

- Tanto el silenciado y l337 van a tomar efecto completamente o nada para cada mensaje. Mismo si se cambia durante el envío del mensaje.
- El multiplexado se decide al momento de hacer la conexión. Si se cambia y un usuario está conectado no toma efecto hasta que se desconecta y se vuelve a conectar.

Asumpciones

- Para poder ser laxos con lo que recibimos, no vamos a verificar que sean correctos los valores que envía el usuario. Todo lo que no modifiquemos vamos a suponer que esta bien (IQ sin ID, falta el namespace, contenido mixto, *body* con mismo *lang*, etc...)
- Tampoco *checkeamos* que el body no tenga hijos. Por ende si nos envian un msj de esa forma y esta prendido el l337 se va a cambiar el contenido.

Asumpciones

- Cuando un usuario está silenciado vamos a enviarle un error si intenta hablar del estilo “*not-authorized*”.
- El administrador va a poder poner cualquier dirección para los servidores a donde multiplexar. Caso que el administrador ponga un servidor inexistente, la aplicación va a tomar como que el servidor destino está caído y no va a tirar un error de que esta mal configurado pero de que el servidor no es accesible.

Asumpciones

- Todos los registros en consola. Métricas volátiles y solo cantidad de accesos y cantidad de bytes.
- Vamos a suponer que el servidor no va a pedir hacer TLS y que va a permitir hacer SASL con PLAIN.

Posibles agregados

Se podrian implementar estrategias para evitar el DOS como estan mencionadas en el RFC 6120 en la sección 13.12 como limitar las conexiones TCP de una misma IP o poner un limite para el tamaño de una stanza, limitaciones que podrian ser modificadas desde el protocolo de administracion.