Trabalho - GEX635 - Tópicos especiais em computação XIII

Matheus Dias Negrão

1711100026

ARQUITETURA

Para o desenvolvimento do trabalho foram utilizadas as funções assíncronas da biblioteca Paho-MQTT, para que pudessem ser utilizadas as threads de callback e tratamento de exceções definidas na biblioteca.

Todos os usuários assim que criados tem seus tópicos de Controle e de Cliente definidos, onde eles são assinantes e recebem solicitações de sessões de chat e confirmação de chats respectivamente, apenas os próprios usuários podem escutar nesses tópicos.

Sendo assim, para realizar a comunicação um-a-um o usuário deve dizer qual é o id de quem ele quer enviar a mensagem. Fazendo com que a aplicação envie uma solicitação com seu ID no tópico de controle do destinatário. O destinatário por sua vez recebe essa solicitação e automaticamente cria um tópico onde ele é assinante e publicador, salva o tópico em uma estrutura de sessões e o envia para o solicitante por meio do tópico cliente do solicitante, que o recebe, também salva em sua estrutura de sessões e também se torna assinante e publicador.

Após a criação da sessão, o usuário pode encontrá-la ao selecionar a opção de enviar uma mensagem em chat, podendo então, enviar a mensagem naquela sessão. Se ele não estiver cadastrado em nenhum grupo ou não estiver em nenhuma sessão de chat, o usuário será notificado.

Para comunicação em grupo o usuário informa o grupo que deseja entrar, se tornando assinante daquele grupo, e assim como nas sessões, para enviar uma mensagem ele deve selecionar a opção e enviá-la.

Em todos os casos as mensagens são recebidas na thread msgarrvd, onde são definidas da seguinte forma:

- As mensagens que chegam no tópico controle são de solicitação de chat, contendo no payload apenas o id do usuário que solicita a sessão.
- As mensagens que chegam no tópico cliente são de confirmação de chat com o tópico via payload, sendo que o tópico de sessão é definido pela união dos ids dos pertencentes a sessão com um número aleatório.
- As outras mensagens de outros tópicos são as mensagens de chat de fato. Essas são separadas entre grupos e sessões. As de grupo contém no payload o grupo que ele está publicando, o id do usuário que está publicando e a mensagem em si. Nos tópicos de chat individual são recebidos no payload o id do usuário que enviou e a mensagem propriamente dita;

As estruturas principais do MQTT que foram utilizadas na comunicação são:

MQTTAsync_sendMessage e *MQTTAsync_subscribe,* que publicam e assinam em tópicos respectivamente.

As principais estruturas criadas no trabalho são:

1. **pub_msg** - Recebe um tópico e um payload. Cria a estrutura de mensagem do MQTT e envia no tópico recebido.
2. **sub_topic** - Recebe o nome do tópico e o assina;
3. **handle_new_chat** - Em caso de recebimento de solicitação para novo chat, realiza a negociação e cria o tópico para a sessão.
4. **send_msg_chat** - Recebe do usuário a mensagem e chama a função para enviá-la no tópico da sessão escolhida.
5. **send_msg_group** - Recebe do usuário a mensagem e chama a função para enviá-la no tópico do grupo escolhido.
6. **sub_group** - Pede ao usuário em qual grupo ele quer se cadastrar e chama a função para assinar o tópico correspondente;
7. **ini_chat** - Inicia o procedimento de solicitação para um novo chat, pedindo ao usuário o id do destinatário e chamando a função para publicá-la no topico de controle do destinatário.

Para tratar eventuais erros de comunicação com o broker foram utilizadas as funções fornecidas nos exemplos da biblioteca paho, onde são definidos callbacks para cada tipo de ocorrência de erro e de sucesso. São elas as funções:

- **connlost** - tenta uma reconexão em caso de desconexão
- **onDisconnectFailure** - Caso a desconexão intencional do broker falhe;
- **onDisconnect** - Caso a desconexão intencional funcione;
- **onSendFailure** - Caso o envio de alguma mensagem falhe;
- **onSend** - Caso o envio de uma mensagem ocorra corretamente;
- **onConnect** - Caso a conexão com o broker ocorra corretamente;
- **onConnectFailure** - Caso a conexão com o broker falhe;

IMPLEMENTAÇÃO

Para a implementação foram utilizadas as funções assíncronas da biblioteca Paho-MQTT-C. Utilizando o broker mosquitto para realizar a comunicação dos tópicos do cliente MQTT.

Em grande parte da aplicação são tratadas as strings de entrada do usuário para que possa ser feita a comunicação entre os usuários, tanto um-a-um como em grupo. O limite de usuários da aplicação é de 99, enquanto as mensagens e os tópicos também são strings limitadas por valores estáticos, 64 e 30 caracteres respectivamente. O usuário pode se conectar até 100 chats únicos e 99 em grupo.

Foram utilizados também mutexes para controle de áreas de acesso por múltiplas threads, como o *sub_topic* e *pub_msg*, que podem ser acessadas tanto pela thread do usuário como pela thread *msgarrvd*. Apesar das funções da biblioteca do paho serem threadsafe, outros parametros são configurados alem do envio e assinatura de topicos.

UTILIZAÇÃO

Download e instalação:

- Para utilizar a aplicação é necessario ter instalado o broker mosquitto e a biblioteca paho-MQTT. Abaixo as instruções de utilização no Ubuntu 20.04.

Mosquitto

sudo apt update
sudo apt install mosquitto
sudo apt install mosquitto-clients

Paho-MQTT-C

git clone https://github.com/eclipse/paho.mqtt.c.git
cd paho.mqtt.c
make
sudo make install

Execução:

- Para execução da aplicação deve se compilar o arquivo utilizando a biblioteca do paho mqtt e posteriormente executá-lo, como demonstrado abaixo:

gcc chat-mqtt.c -o chat -lpaho-mqtt3as -lpthread
./chat

Utilização:

- Assim que executado, para usar o aplicativo, o usuario deve inserir o seu id único (01 - 99). e Após a conexão, as funções do menu podem ser utilizadas escrevendo no terminal. A todo momento, independente da chegada de mesnagens, os comandos dados no terminal serão direcionados a seleção do menu.

Digite seu ID único:
03
Bem-vindo! Agora você está online!

-- Escolha uma oção --
1. Iniciar um chat com um usuario
2. Enviar mensagem
3. Entrar em um grupo
4. Enviar mensagem em um grupo
5. Logout