Instituto de Matemática e Estatística da USP

MAC0352 - Redes de Computadores e Sistemas Distribuídos - 1s2022

EP1
Entrega até 8:00 de 26/4/2022
(INDIVIDUAL)

Prof. Daniel Macêdo Batista

1 Problema

Neste EP você deverá implementar a interpretação e o processamento de algumas mensagens da camada de aplicação de um broker[1] MQTT na versão 3.1.1 do protocolo. O código referente às camadas inferiores não precisa ser escrito se você não quiser, pois o código de um servidor de eco está disponibilizado no e-Disciplinas e pode ser usado como base. Caso você pegue o código do servidor de eco bastará[2] modificar os trechos referentes à camada de aplicação para transformá-lo em um broker MQTT[3].

Seu broker não precisa ser um `mosquitto` [4]! **Ele só precisa aceitar conexões e desconexões de clientes, receber e enviar mensagens em um tópico específico sem se preocupar com falhas e sem se preocupar com autenticação ou criptografia**.

Para entender mais sobre isso, consulte a especificação do protocolo no site da OASIS em `http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html` e a página da wikipedia sobre o MQTT em `https://pt.wikipedia.org/wiki/MQTT`. A correção do broker será feita utilizando os programas `mosquitto_pub` e `mosquitto_sub`, dois clientes MQTT disponibilizados no pacote debian `mosquitto-clients`[5]. É altamente recomendável que você entenda como a comunicação com um broker MQTT funciona instalando o `mosquitto` na sua máquina e realizando comunicações usando o `mosquitto_sub` e o `mosquitto_pub` enquanto o `Wireshark` é executado. Observar os pacotes no `Wireshark` vai ajudar a entender o protocolo. Talvez mais do que ler a especificação.

[1] no fim das contas um broker age como um servidor, da arquitetura cliente-servidor, então entenda que você vai precisar implementar um servidor

[2] da forma como está escrito, parece que é algo trivial, mas não é :-)

[3] Na verdade há dois servidores de eco disponibilizados. Comece a fazer o EP e você vai entender porque o segundo servidor de eco fornecido como exemplo poderá ajudar você.

[4] `https://mosquitto.org/`

[5] `https://packages.debian.org/bullseye/mosquitto-clients`

2 Requisitos

2.1 Comportamento do broker MQTT

O servidor, ou *broker*, como é mais conhecido um servidor de um sistema *publish-subscribe*, deve se comportar de forma similar a como o broker `mosquitto` se comporta em quatro situações:

- Conexão de vários clientes simultaneamente (cada cliente simultâneo pode publicar ou requisitar mensagens do mesmo tópico ou de tópicos distintos);
- Inscrição de cliente em um tópico e consequente envio das mensagens deste tópico para o cliente;
- Publicação de mensagem em um tópico;
- Desconexão de cliente.

As mensagens e os tópicos devem respeitar os limites de tamanho do protocolo e devem possuir apenas caracteres ASCII[6].

Para saber como o `mosquitto` se comporta em todos os casos, instale-lo em algum computador e faça os testes rodando o `Wireshark` em paralelo.

Caso você não tenha GNU/Linux no seu computador ou se você não quer instalar o `mosquitto` com medo de esquecer de desinstalá-lo e ele ficar aceitando conexões no seu computador, crie uma máquina virtual no seu próprio computador, instale alguma distribuição de GNU/Linux, instale o `mosquitto` e capture os pacotes com o `Wireshark`. Informações sobre a utilização de máquinas virtuais podem ser encontradas em `https://www.virtualbox.org/`, `https://xenproject.org/` ou `https://www.docker.com/`.

Seu EP **não** precisa se preocupar com falhas. Por exemplo, se um cliente que estiver assinando um tópico for interrompido por qualquer motivo e nesse meio tempo alguma mensagem for enviada para aquele tópico, quando ele reconectar ele não precisa receber as mensagens que foram enviadas enquanto ele esteve desconectado.

2.2 Linguagem

O broker deve ser escrito em C. Certifique-se de que ele funciona no GNU/Linux pois ele será compilado e avaliado apenas neste sistema operacional.

Os códigos `mac0352-servidor-exemplo-ep1.c` e `mac0352-servidor-exemplo-ep1+pipe.c` disponíveis no e-Disciplinas podem (não é obrigatório) ser usados como base. Eles são servidores de eco. Leia os comentários no início dos códigos para entender como fazer para executá-los. Toda a parte de gerência da conexão no código pode ser ignorada. Basta focar no trecho onde devem ser feitas as mudanças para o EP, que está identificado no código. Note que esses servidores de exemplo fazem a comunicação utilizando mensagens em modo texto. Apesar da implementação do broker MQTT feita por você ter que transferir mensagens de texto, algumas informações em binário precisarão ser trocadas. Logo, isso precisará ser levado em conta na sua implementação.

[6] `man ascii` em uma máquina rodando GNU/Linux vai mostrar quais são esses 128 caracteres

2.3 Slides

Você deverá entregar, além dos códigos, um .pdf com slides que você usaria caso você fosse apresentar o seu trabalho. Os slides deverão descrever, em alto nível, a sua implementação destacando os pontos principais de cada comando do MQTT que foi implementado. Decisões de projeto que você julgar que merecem ser apresentadas também podem ser incluídas. Além das informações sobre a implementação, os slides também devem apresentar gráficos de análise de desempenho comparando o desempenho do broker, em termos de uso de rede e de CPU, em três cenários: (i) apenas com o broker, sem nenhum cliente conectado; (ii) com o broker e com cem clientes publicando e recebendo mensagens simultaneamente e (iii) com o broker e mil clientes publicando e recebendo mensagens simultaneamente. Nos slides inclua informações sobre o ambiente computacional e de rede que você usou para realizar os experimentos e como você avaliou a carga na rede (Certifique-se de manter todos os outros softwares que utilizem a Internet fechados enquanto estiver realizando os experimentos). Os experimentos devem fazer a comunicação entre dois computadores diferentes, ou seja, não podem ser feitos utilizando o localhost. Note que você não precisa ter 1000 computadores reais para ir testando o EP e para fazer os experimentos. Vários dos clientes podem estar rodando em uma máquina virtual no seu próprio computador. Todos os resultados precisam ter validade estatística. Logo, faça uma quantidade razoável de medições e apresente os resultados com média e alguma informação de dispersão, como desvio padrão ou intervalo de confiança por exemplo.

Esses slides não serão apresentados mas considere que eles seriam apresentados em um tempo máximo de 10 minutos. Portanto, antes de enviar seu .pdf, faça um ensaio da apresentação para garantir que a quantidade de conteúdo cabe dentro de 10 minutos de apresentação.

Entregas sem o .pdf da apresentação não serão corrigidas e receberão nota ZERO.

3 Entrega

Você deverá entregar um arquivo .tar.gz contendo os seguintes itens:

- fonte;
- Makefile (ou similar);
- arquivo LEIAME;
- .pdf dos slides.

O desempacotamento do arquivo .tar.gz deve produzir um diretório contendo os itens. O nome do diretório deve ser ep1-seu_nome. Por exemplo: ep1-joao_dos_santos.

A entrega do .tar.gz deve ser feita no e-Disciplinas.

O EP deve ser feito individualmente.

Obs.1: Serão descontados 2,0 pontos de EPs com arquivos que não estejam nomeados como solicitado ou que não criem o diretório com o nome correto após serem descompactados. Confirme que o seu .tar.gz está correto, descompactando ele no shell (não confie em interfaces gráficas na hora de testar seu .tar.gz pois alguns gerenciadores de arquivos criam o diretório automaticamente mesmo quando esse diretório não existe. Se você nunca usou o comando tar, leia a manpage e "brinque" um pouco com ele para entender o funcionamento).

Obs.2: A depender da qualidade do conteúdo que for entregue, o EP pode ser considerado como não entregue, implicando em MF=0,0. Isso acontecerá por exemplo se for enviado um .tar.gz corrompido, ou códigos fonte vazios.

Obs.3: O prazo de entrega expira às 8:00:00 do dia 26/4/2022.

4 Avaliação

60% da nota será dada pela implementação, 10% pelo LEIAME e 30% pelos slides. Os critérios detalhados da correção serão disponibilizados apenas quando as notas forem liberadas.