UFPI - Ciências da Computação - Sistemas Distribuídos - 2020.2

Daniel Leite Dantas

Estevam Amorim de Carvalho Alves

Multicast: Desenvolvimento de aplicação tolerante a falhas

O objetivo deste trabalho era a criação de uma aplicação tolerante a falhas que permitisse a resolução de expressões aritméticas simples, utilizando para isso o conceito de Multicast.

1. **Instruções para Execução**

   As instruções abaixo são voltadas para execução em ambiente **Linux**, utilizando **Python 2 ou 3**:

   - ➔ Em todas as máquinas, **habilite o uso de multicast** através do comando `route add -net 224.0.0.0 netmask 224.0.0.0 eth0`
   - ➔ **Mude para o diretório** onde estão os arquivos fonte disponibilizados juntamente com este relatório;
   - ➔ No(s) **servidor(es)**, execute o script `server.py` passando como argumento um **número entre (0, 5**], único por máquina, indicando o índice do servidor. É possível alterar o número máximo de servidores através da variável `MAX_NUMBER_OF_SERVERS` no script mencionado;
   - ➔ Na(s) máquina(s) **cliente**, execute o script `client.py`. Em seguida, digite uma expressão aritmética (Ex: 3+5-4/3*4, sem aspas) e aperte enter para enviar. Uma mensagem com a resposta da expressão deverá ser recebida de um dos servidores.

2. **Informações exibidas nos terminais**

   No terminal da máquina do cliente é exibida uma mensagem pedindo para que seja digitada a expressão, no servidor, além de atualizações dos estados dos servidores, a cada 20 segundos é mostrado o estado de atividade de todos os servidores do cluster.

3. Comunicação entre Servidores

Os servidores se comunicam entre si utilizando um grupo multicast. Executando a rotina em uma thread, cada servidor envia uma mensagem de confirmação no grupo passando o seu índice e informando que ainda está conectado e ativo, a cada 1 segundo.

Todos os servidores do cluster escutam e reagem às mensagens trocadas no grupo multicast, atualizando localmente os estados de cada servidor, que podem variar entre: **Ativo**, **Aguardando Confirmação, Irresponsivo** e **Desconectado**, de acordo com o tempo em que a última mensagem de confirmação foi recebida.

Os estados de todos os servidores (incluindo o da máquina local) são inicializados como Desconectado. A cada mensagem de confirmação recebida, o estado do servidor cujo índice é informado na mensagem é atualizado para Ativo. Vale ressaltar que um servidor também escuta suas próprias mensagens, logo o seu estado será condizente com a sua capacidade de se comunicar com o grupo multicast.

Ao *enviar* uma mensagem na rotina mencionada inicialmente, cada servidor também irá decrementar um contador (TTL) de todos os servidores do grupo, de forma que o estado destes passará gradualmente a representar uma condição menos responsiva caso novas mensagens de confirmação não sejam recebidas.

Na configuração padrão, considera-se que servidores dos quais não se recebe mensagem de confirmação de estado há mais de 2 ticks (segundos) estão irresponsivos e, portanto, não estão aptos a responderem requisições de clientes. A frequência na qual as mensagens são trocadas e os limiares de conectividade podem ser configurados no *script* do servidor.

4. Comunicação entre Cliente(s) e Servidores

A aplicação distribuída desenvolvida utiliza dois grupos multicast, enquanto um deles é exclusivo para os servidores se comunicarem entre si, o outro serve para prover a funcionalidade ofertada aos clientes, nele estão tanto os clientes quanto os servidores.

O funcionamento da aplicação consiste no cliente enviar uma mensagem em multicast contendo uma expressão aritmética que será recebida por todos os servidores pertencentes ao grupo. Cada servidor irá verificar, a partir da informação salva localmente e constantemente atualizada do estado de atividade dos outros servidores, se existe algum servidor ativo com o número de identificação menor que o dele e, caso exista, o servidor não irá responder a requisição. Caso contrário, ele irá responder o cliente em unicast informando o resultado da expressão recebida.

Com base nisso pode-se concluir que apenas o servidor ativo e com o menor número de identificação dentro do cluster será responsável por responder às requisições. É importante destacar que a aplicação oferece alta tolerância a falhas, já que ela só deixa de ofertar seu serviço caso todos os servidores pertencentes ao cluster estejam inativos.