Paradigmas de Sistemas Distribuídos
Sistemas Distribuídos em Grande Escala
"Armazenamento em Grande Escala"

Trabalho Prático - II

2022/2023

Informações gerais

- Cada grupo deve ser constituído por até 4 elementos, de ambas ou de uma das UCs. Os grupos que não frequentem ambas as UCs devem contactar imediatamente os docentes para adaptar o enunciado.
- Deve ser entregue o código fonte e um relatório de até 6 páginas (A4, 11pt) no formato PDF.
- O trabalho deve ser entregue até às 23:59 do dia 26 de maio de 2023 no *eLearning* de qualquer das UCs.
- A apresentação do trabalho ocorrerá em data a anunciar.
- O trabalho será classificado separadamente para cada uma das UCs, de acordo com os objetivos de cada uma delas.

Resumo

Pretende-se um sistema de armazenamento com coerência causal e particionamento da gestão de sessões e do armazenamento, de forma a suportar um número muito grande de clientes e um elevado débito de operações. Deve também limite as invocações por parte dos clientes para evitar abusos.

De forma a utilizar o serviço, um *cliente* liga-se a um *servidor de sessão*, autenticando-se, e mantendo uma ligação estabelecida, sobre a qual poderá efetuar múltiplas operações. Estas operações podem ser uma escrita (associar um valor a uma chave) ou uma leitura em bloco (devolver a lista de valores associados a uma lista de chaves). Portanto, assume-se que as operações de escrita são independentes e pedidas uma de cada vez, enquanto as operações de leitura devem ter a semântica de uma transação, devolvendo um conjunto de valores respeitando a coerência causal. Operações em diferentes transações por parte do mesmo cliente devem também respeitar a causalidade.

O *servidor de sessão* irá armazenar a informação relacionada com cada um dos clientes ligados, necessária para garantir coerência causal nas operações individuais e nas transações de leitura. Para executar estas operações, o servidor de sessão solicita os valores associados às chaves relevantes dos *servidores de dados*.

Os *servidores de dados* armazenam a associação entre chaves e valores, que se assume serem ambos *strings*. Para suportar a funcionalidade desejada, os servidores devem suportar o armazenamento de múltiplas versões de uma chave.

Funcionalidade

Cliente

O cliente emite pedidos para ler ou escrever dados armazenados no sistema como um todo, comunicando com os servidores de sessão através de *sockets* TCP/IP. Admite-se que cada cliente é *single threaded*, não emitindo um segundo pedido enquanto não receber a resposta ao primeiro. Um cliente não deve poder estar ligado a mais do que um servidor simultaneamente e a coerência causal só necessita de ser garantida dentro de cada sessão.

O trabalho deve incluir implementações de clientes que permitam demonstrar o funcionamento do sistema (por exemplo, com um menu simples) e clientes que permitam carregar o sistema e medir o seu desempenho. É admitida uma implementação centralizada que permita simular um número elevado de clientes ligados.

Servidor de Sessão

O papel do servidor de sessão é garantir a escalabilidade em termos de número de clientes ligados e de pedidos processados. Estes servidores não devem guardar estado persistente, sendo que a falha de um deles resulta apenas na interrupção da ligação aos clientes a ele ligados naquele momento. Este servidor deve:

1. Receber pedidos dos clientes, cada um contendo uma operação de escrita ou uma operação de leitura de um conjunto de chaves.
2. Obter versões apropriadas dos valores necessários para realizar um pedido do cliente, de forma a cumprir o critério de coerência.
3. No caso de uma operação de escrita, fazer chegar o valor ao servidor de dados apropriado, de acordo com a chave, de forma a cumprir o critério de coerência.

Cada servidor de sessão deve guardar o estado necessário associado a cada cliente (e.g., dependências ou relógio lógico) de forma a cumprir os requisitos.

De forma a melhorar o desempenho geral do sistema, um servidor de sessão deverá também gerir uma *cache* de versões de valores associados a chaves, de forma a reduzir a carga dos servidores de dados. A cache deve incluir valores associados a chaves para as quais ainda há clientes ligados que a elas se referiram. O número de entradas deve ser limitado, utilizando algum critério, como eliminar entradas antigas com menos clientes interessados.

Servidor de Dados

O papel do servidor de dados é garantir a persistência da informação e a escalabilidade em termos da quantidade de dados armazenados. No contexto deste trabalho, assume-se que estes servidores não falham e que podem fazer o armazenamento da informação apenas em memória. Este servidor deve:

1. Receber pedidos dos servidores de sessão contendo uma operação de escrita.
2. Receber pedidos dos servidores de sessão contendo uma operação de leitura. Tais pedidos podem especificar determinada versão de uma chave ou pedir a versão mais recente. Deve ser evitada a transferência de versões que estão em cache.

O conjunto de servidores de dados pode ser alterado acrescentando novos servidores, caso em que a carga deve ser distribuída entre eles. Assuma no entanto que existe no máximo um processo de integração de um novo servidor em curso, não começando outro até este estar concluído.

Limitador de carga

Deve ser implementado um mecanismo para limitar as invocações de clientes individuais. Assim, sempre que a média do número de invocações por segundo de determinado cliente nos últimos 60 segundos ultrapasse um valor LIMIT (e.g., LIMIT=1000), este cliente dever ser marcado como estando no estado *throttled*. Neste estado este cliente só verá respondidos no máximo BASE (e.g., BASE=100) pedidos em cada segundo. Um cliente é libertado deste estado pelo servidor de sessão que o marcou, que liberta os clientes por ordem em que foram marcados. Quando liberta um cliente, o servidor calcula o intervalo de tempo para libertar o próximo cliente como $60 + T$ segundos, sendo T o número global (aproximado) de clientes marcados como *throttled* no sistema nesse momento. Este estado não deverá poder ser contornado pelo cliente desligando-se do servidor corrente e autenticando-se noutro servidor de sessão. Sugestão: mantenha um conjunto de utilizadores *throttled*, com gestão distribuída pelo conjunto de servidores de sessão.

Objetivos

PSD O trabalho deve ser efetuado utilizando Java (servidor de dados), Erlang (servidor de sessão), ZeroMQ (ambos os servidores), sockets TCP (entre clientes e servidor de sessão). Deve em particular aproveitar a facilidade que Erlang dá a gerir o estado de sessão, e de diferentes padrões de mensagens oferecidos pelo ZeroMQ.

SDGE O sistema deve garantir o critério de coerência causal, fazer uma distribuição de carga entre os servidores de dados usando *consistent hashing* e gerir a informação relativa ao estado dos clientes de forma descentralizada com CRDTs.