Sistemas Distribuidos 23/24

Logo

Trabalho 2

Prazos

2º Trabalho (online - código + formulário):

- Data de entrega: 1 de junho, 23h59, 1 hora de tolerância. Entregas atrasada, no dia seguinte, terão uma penalização de 1 valor.

Form para submissão do trabalho: https://forms.gle/pXNqxZNNmWfpsCNe8

Objetivo

O objetivo do segundo trabalho é estender o primeiro trabalho, com as seguintes funcionalidades:

- segurança;
- interação com serviços externos;
- tolerância a falhas - vamos implementar a tolerância a falhas no servidor de shorts e no servidor de blobs.

O desenho da solução, incluindo a arquitetura e as interfaces dos serviços permanecem idênticas às do primeiro trabalho, salvo as indicações em contrário neste enunciado.

A compatibilidade com as interfaces e operações pré-definidas tem que ser observada.

Funcionalidades

Segurança (máx: 3,5 valores)

O objetivo desta funcionalidade é tornar o sistema seguro, impedindo que elementos não autorizados executem operações no serviço de feeds e serviço de utilizadores. Para alcançar este objetivo, a solução deve incluir os seguintes mecanismos:

1. utilizar canais seguros, usando TLS, com autenticação do servidor. As operações dos clientes incluem uma password para verificar que o cliente está autorizado a efetuar a operação indicada (esta última parte já se verificava nas interfaces definidas no sistema);
2. caso existam operações executadas apenas entre os servidores, garantir que estas não podem ser invocadas por um cliente – sugere-se a utilização dum segredo partilhado entre os servidores, o qual pode ser passado como parâmtero ao arrancar o programa.

Tests: TBD.

Segurança - controlo de acessos nos servidores de blobs (máx: 1 valor)

O objetivo desta funcionalidade é garantir que operações não autorizadas não podem ser efetuadas nos servidores de blobs, i.e., que apenas os utilizadores autorizados podem executar operações.

Nesta alternativa permitem-se soluções em que há pedidos adicionais entre os servidores ou em que se usam tokens com duração limitada para codificar as permissões.

NOTA: O tester enviará nos pedidos efetuados para o servidor de blobs os query parameters que estiverem nos URLs devolvidos no objeto Short devolvido anteriormente. Na versão GRPC, os query parameters serão incluídos no blobid.

Tests: TBD.

Interação com um serviço externo (alternativa E1) (máx: 4,5 valores)

O objetivo desta funcionalidade é ter um servidor de blobs que interaja com um serviço externo que disponibilize um serviço REST com autenticação O.Auth para armazenar o conteúdo dos blobs. Sugere-se a utilização do sistema Dropbox (outras opções: Google Drive, etc.).

Para implementação desta funcionalidade sugere-se a utilização da biblioteca ScribeJava, como apresentado nas aulas práticas.

NOTA: Apenas é necessário fazer este servidor com uma interface REST.

NOTA: Deve ser possível lançar mais do que um destes servidores.

NOTA: Para que seja possível testar o serviço de forma automática, usando o Tester, é necessário poder indicar ao servidor, quando arranca, que deve iniciar com o estado do serviço externo *limpo*. Para tal, o Tester passará como primeiro parâmetro do servidor que interage com o serviço externo o valor **true** para indicar que o estado anterior deve ser ignorado. Se o Tester passar o valor **false**, o estado gravado deve ser usado pelo servidor.

NOTA: Este servidor de Blobs deve expor uma interface REST.

Tests: TBD.

Tolerância a falhas - servidor de blobs (máx: 2 valores)

Implementar uma solução que permita tolerar falhas em 1 máquina em que esteja a correr um servidor de blobs numa configuração com pelo menos 2 servidores de blobs. A solução deve tolerar a falha de qualquer servidor de blobs e ser desenhada para uma configuração em que existe um número elevado de servidores de blobs.

Como propriedade valorativa da solução pode-se considerar manter o nível de tolerância a falhas em situações em que servidores de blobs falham.

NOTA: quando existe replicação de blobs, um short deve devolver no `blobURL` uma lista de URLs separado pelo carater `|`.

Tests: TBD.

Tolerância a falhas - servidor de shorts (alternativa D1) (máx: 7 valores)

Implementar uma solução que permita tolerar falhas numa máquina em que esteja a correr um servidor de shorts, replicando o servidor de shorts com uma solução de replicação de máquina de estados, recorrendo a um sistema de comunicação indireta - e.g. Kafka. A solução deve tolerar a falha de qualquer servidor de shorts.

A solução deve garantir que um cliente lê sempre o estado dum servidor que tem uma versão tão atual quanto a versão do servidor que foi acedido anteriormente. Para tal, o servidor pode adicionar *headers* às resposta a enviar aos clientes, os quais serão enviados nas próximas operações executadas pelo mesmo cliente (o Tester reenviará todos os headers começados em **X-SHORTS**).

NOTA: Este servidor deve expor uma interface REST.

Tests: TBD.

Tolerância a falhas - servidor de shorts (alternativa D2a) (máx: 6 valores)

Implementar uma solução que permita tolerar falhas numa máquina em que esteja a correr um servidor de shorts, replicando o servidor de shorts com uma solução de replicação de máquina de estados, implementando o protocolo primário/secundário. A solução deve tolerar a falha de qualquer servidor; no caso da falha do servidor primário, o sistema deve garantir que continua a ser possível fazer leituras, mas não necessita permitir a execução de escritas.

A solução deve garantir que um cliente lê sempre o estado dum servidor que tem uma versão tão atual quanto a versão do servidor que foi acedido anteriormente. Para tal, o servidor pode adicionar *headers* às resposta a enviar aos clientes, os quais serão enviados nas próximas operações executadas pelo mesmo cliente (o Tester reenviará todos os headers começados em **X-SHORTS**).

Nota: O programa deve suportar a falha de 1 servidor, estando os protocolos implementados configurados para tal.

NOTA: Este servidor deve expor uma interface REST.

Tests: TBD.

Tolerância a falhas - servidor de shorts (alternativa D2b) (máx: 9 valores)

Implementar uma solução que permita tolerar falhas numa máquina em que esteja a correr um servidor de shorts dum domínio, replicando o servidor de shorts com uma solução de replicação de máquina de estados, implementando o protocolo primário/secundário. A solução deve tolerar a falha de qualquer servidor, incluindo o servidor primário – para tal, deve eleger um novo primário.

A solução deve garantir que um cliente lê sempre o estado dum servidor que tem uma versão tão atual quanto a versão do servidor que foi acedido anteriormente. Para tal, o servidor pode adicionar *headers* às resposta a enviar aos clientes, os quais serão enviados nas próximas operações executadas pelo mesmo cliente (o Tester reenviará todos os headers começados em **X-SHORTS**).

Nota: O servidor deve suportar a falha de 1 servidor, estando os protocolos implementados configurados para tal.

NOTA: Este servidor deve expor uma interface REST.

Tests: TBD.

Fatores depreciativos

O código entregue deverá seguir boas práticas de programação. A repetição desnecessária de código, inclusão de constantes mágicas, o uso de estruturas de dados inadequadas, etc., poderá incorrer numa penalização. (max: 2 valores)

Falta de robustez e comportamentos erráticos da solução são motivo para penalização.

NOTA: A solução deve continuar a contemplar as falhas temporárias dos canais de comunicação.

Execução

O trabalho pode ser executado em grupo, de 1 ou 2 alunos. Os alunos do mesmo grupo não precisam de pertencer ao mesmo turno prático, embora tal seja fortemente recomendado.

Os grupos neste segundo trabalho podem ser diferentes do primeiro trabalho.

Como base deste segundo trabalho, os alunos podem usar a sua implementação do primeiro trabalho ou a implementação disponível no seguinte link, fornecida como tal:

https://github.com/smduarte/sd2324-proj

Avaliação

A avaliação do trabalho terá em conta os seguintes critérios:

- Funcionalidades desenvolvidas e a sua conformidade com a especificação, tendo como base os resultados da bateria de testes automáticos;
- Qualidade da solução;
- Qualidade do código desenvolvido.

A classificação final do aluno é individual e será menor ou igual à classificação do trabalho, em função da discussão do trabalho, a realizar durante a seguir à entrega do trabalho.

Bateria de testes

TBA

Ambiente de desenvolvimento

Todo o material de apoio fornecido pressupõe que o desenvolvimento será em ambiente Linux e Java 17. A validação do trabalho por via da bateria de testes automática fará uso de tecnologia Docker.

Histórico de alterações

27/4/24

Enunciado

Versão inicial.