Grupo A07

TÉCNICO LISBOA

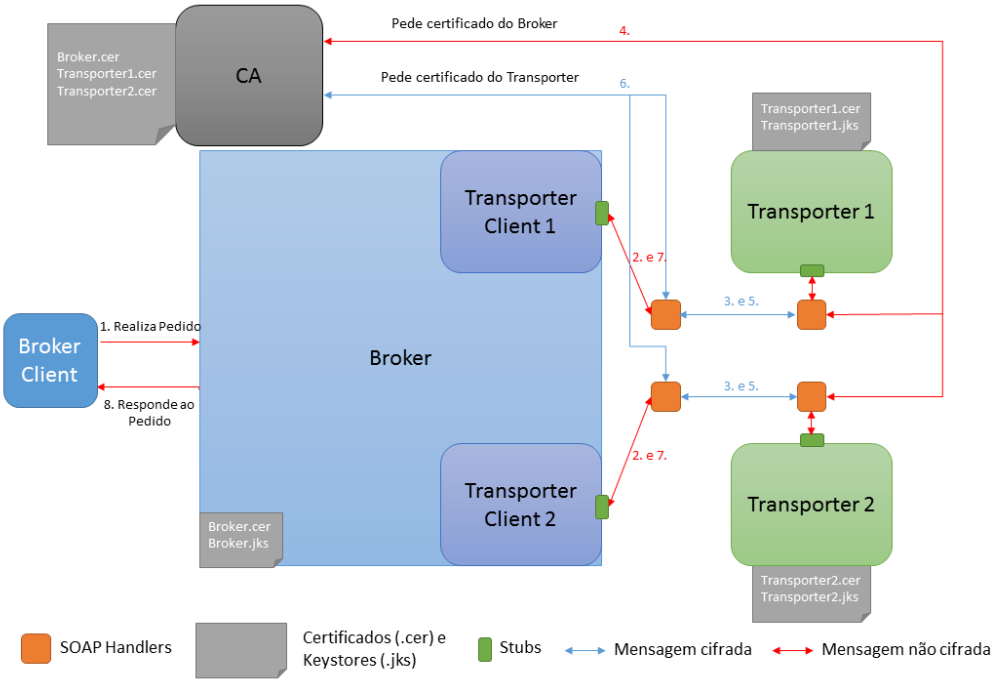
Sistemas Distribuídos 2015-2016

Relatório de Projecto

https://github.com/tecnico-distsys/A_07-project

	**Daniel Fermoselle, 78207**
	**Pedro Ribeiro, 79055**
	**Tiago Rodrigues, 78692**

Segurança



Inicialmente, tal como indicado na figura, o *Broker Client* faz um pedido ao *Broker*. Posteriormente, no *handler*, utiliza-se a função de *hash* sobre o conteúdo do body mais o nounce gerando o *digest*, este por sua vez é cifrado com a chave privada do Broker. O Broker usando Transporter Clients envia o pedido a um *Transporter.* Os *Transporters* irão decifrar a mensagem com a chave pública do Broker obtendo-a através do CA. De seguida os *Transporters* envia uma mensagem ao *Broker* por um processo semelhante ao anterior.

Assim garantimos cada uma das seguintes componentes da seguinte forma:

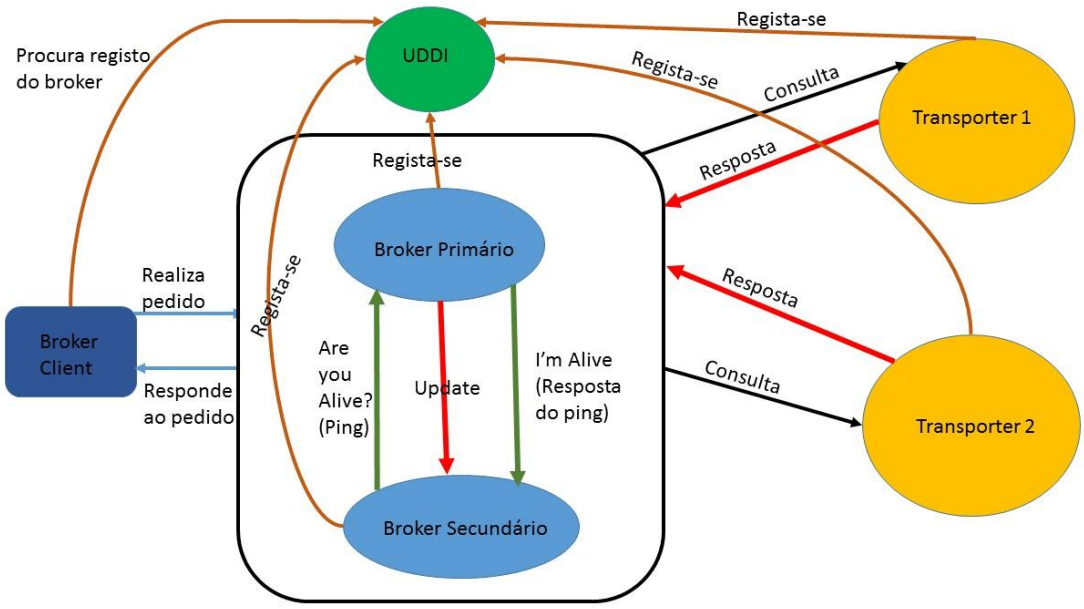
- **Frescura:** Para conseguirmos garantir a frescura de cada mensagem, criamos e usamos um *nounce*, que será enviado no *header* da mensagem.

- **Autenticidade:** De forma a garantirmos a autenticidade da mensagem, isto é, que foi enviada a partir de uma fonte credível (broker-ws e transporter-ws) recorre-se a uma *digital signature*.

- **Integridade:** Se o *digest* gerado para verificar a autenticidade for diferente do *digest* recebido, implica que o *nounce* ou o corpo foram alterados e, portanto, a mensagem perdeu a sua integridade.

- **Não-Repúdio:** Através da assinatura garante-se o não repúdio, pois só a entidade que enviou a mensagem é que sabe a sua chave privada.

Replicação



Nesta figura temos um *Broker Client* que realiza pedidos a um *Broker*. O cliente não sabe se está a falar com o primário ou com o secundário, apenas sabe que está a falar com um *Broker*. O *Broker* primário sempre que realiza uma operação que altera o seu estado actualiza o estado do *Broker* secundário. Periodicamente o *Broker* secundário "pergunta" ao *Broker* primário se ele ainda está vivo e caso este não responda até ao fim de um *timeout* então o *Broker* secundário registar-se-á no UDDI como sendo o novo *Broker* primário e o cliente passará a realizar os pedidos com o *Broker* secundário.

Na nossa implementação tivemos de adicionar ao WSDL do *Broker* uma nova operação denominada *update* e uma auxiliar chamada *addJobs*. A operação de update actualizará o *Broker* secundário sempre que for necessário. Na nossa implementação temos o *Broker* principal denominado de *UpaBroker* e o secundário *UpaBrokerSlave*. O *UpaBrokerSlave* irá enviar pings para o *UpaBroker* de 4 em 4 segundos e caso o ping não tenha recebido resposta até um *timeout* de 3 segundos então o servidor secundário irá substituir o primário. No *Broker Client* também foi introduzido um *timeout* de resposta com 8 segundos para os pedidos realizados pelo cliente, e assim caso um pedido de um cliente demore mais do que 8 segundos a ser respondido, então o cliente voltará a consultar o UDDI para se conectar ao novo *Broker* e repetir o pedido.

Com isto garantimos a tolerância a 1 falta silenciosa. Temos de ter 1+1 servidores de modo a garantir esta tolerância pois o grau de replicação das faltas silenciosas é **f**+1 sendo **f** o número de faltas toleradas.