**Grupo 14**
**Requisito SD-ID.A:**

Na utilização da nossa aplicação, é apresentado uma interface de seleção de comandos, entre os quais a requestAuthentication, associada ao requisito SD-ID.A. Na nossa implemetação, este método tem como argumentos o User em questão e um vetor de bytes reservados. Neste vetor de bytes, vão o nome do serviço em bytes e um número gerado aleatoriamente de 16 bytes, ambos concatenados. É feita a invocação remota no servidor ID deste método. Primeiramente, o servidor verifica se tem registado esse User, e em caso positivo, faz um get da sua password e de seguida faz um hash MD5 desta password e assim é criado o Kcliente. Nesta altura é também gerada a Ksession aleatoriamente. É então concatenado este Ksession e o numero aleatório recebido e é tudo cifrado em AES com a chave uma Kcliente. Temos aqui um package cifrado. Para formar o ticket, é concatenado entre "/" o User, o Serviço, um TimeStamp de criação do ticket, e outro TimeStamp = TimeStamp de criação + 5 min (300000 ms). São passados este elementos todos para bytes e concatenados à Ksession. É gerado uma chave aleatória Kserver, que vai ser partilhada com o server STORE por FileOutputStream. Com esta chave Server é cifrado o Ticket. Por fim, é concatenado o primeiro package (de tamanho fixo 48 bytes) mais o Ticket. Isto é o retorno do requestAuthentication.
De volta ao cliente, este faz o parse do primeiro package e do ticket (0-48 bytes é o package, de 48-fim é o ticket). O cliente, sabendo a sua password, faz o mesmo hash MD5 para gerar a Kcliente. Com esta chave o cliente decifra o package e retira o KSession e o número. Se este número for igual ao enviado como argumento, a autenticação foi bem sucedida.

**Requisito SD-STORE.B:**

É gerado uma classe de FrontEnd para o User. Esta classe regista as réplicas do server STORE no UDDI. O nr de Réplicas, o nome do serviço, o ticket, o KSession, e o URL do UDDI entram como argumento.
O cliente pode agora invoncar os métodos Create Docs e List Docs a partir desta FrontEnd. Aquando de cada de invocação, é criada uma Autenticação com o user e com o Time Stamp da invocação e cifrada em kSession. É também criado um MAC com os argumentos do pedido. Juntamente com o Ticket, todos este elementos são adicionados ao Header do SOAP, para serem enviados para as réplicas juntamente com o pedido. De notar que estas invocação para o server STORE são assincronas!
Do lado do Server STORE, este recebe o Header do SOAP e executa um método verify que verifica se: 1-o Time Stamp do pedido (dentro da Autenticação) está dentro dos Time Stamps do Ticket. 2- Se o user enviado pelo Ticket é igual ao user dentro da Autenticação. 3- Se os argumentos do MAC são os mesmo do pedido.
Em caso afirmativo, o server STORE executa o método e retorna os resultados para o FrontEnd que interpreta os resultados das múltiplas réplicas e apresenta ao Cliente.

**Notas:** De notar que em caso negativo de alguma verificação, o sistema lança excepção do devido erro. Não implemtámos os teste Mockit e alguns testes Unit falham na sua realização.