Grupo 41 Miguel Geraldes nº74115 João Rebêlo nº75376 Ana Teixeira nº76969

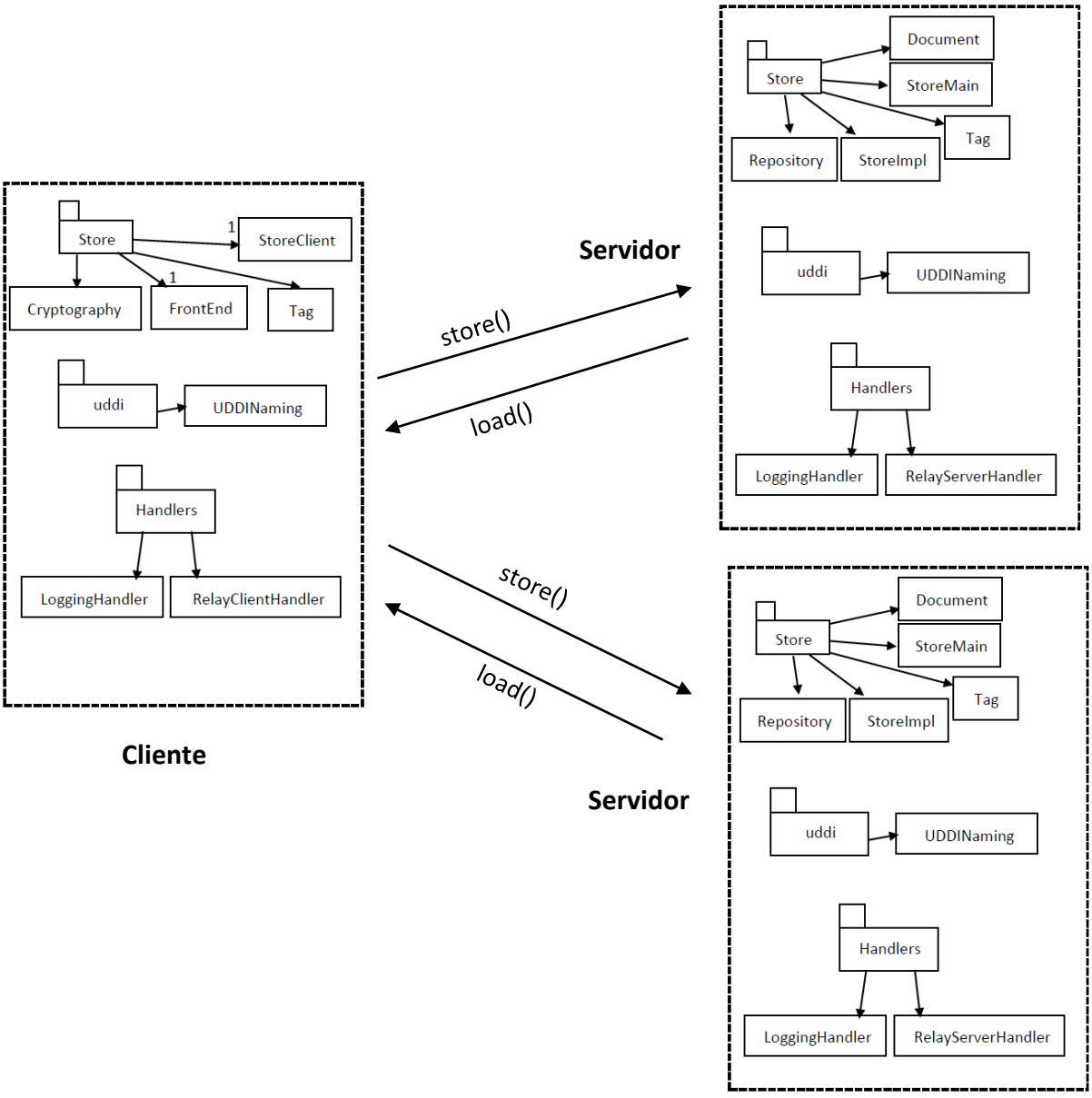
Sistemas Distribuídos
Relatório 2º Entrega



Introdução

Na entrega final do projeto de Sistemas Distribuídos foi-nos pedido que implementássemos soluções para os requisitos SD-ID.B e SD-STORE.A. O relatório encontra-se divido em duas partes, uma para cada requisito, onde fazemos uma breve explicação das soluções implementadas e também dos objetivos presentes no enunciado que não foram alcançados.

Diagrama de implementação do projeto



Grupo 41 Miguel Geraldes nº74115 João Rebêlo nº75376 Ana Teixeira nº76969
Sistemas Distribuídos
Relatório 2º Entrega

SD-ID.B

Requisitos não implementados:

- Não tirámos partido da distribuição de chaves do Kerberos implementado pelos nossos colegas no SD.ID-A. As chaves são geradas na classe FrontEnd, com acesso à classe Cryptography presente no *sd-store*-cli, utilizando as funções: getkey() e generate().

Para garantir a confidencialidade dos conteúdos dos documentos, utilizamos o método de criptografia simétrica, nomeadamente o algoritmo DES para gerar secretKeys. Para cifrar os arrays de bytes do conteúdo dos documentos utilizamos o modo de cifra ECB e padding PKCS5.

São gerados MACs para garantir a integridade dos conteúdos dos documentos a armazenar.

Quando um cliente chamar uma operação, acede à classe FrontEnd onde é gerado uma chave no seu construtor. Ao realizar uma chamada à função store() é utilizada a função Encrypt() que cifra o conteúdo destinado a armazenar. De seguida é chamada a função makeMAC() que devolve um resumo cifrado. O resumo e o array de bytes do conteúdo são guardados num outro array de bytes separados por uma sequência de bytes(`{ 0x00, 0x01, 0x00,0x02,0x00,0x00,0x03,0x04,0x00 }`) e armazenados pelo método store().

Ao chamar a operação load(), acede-se ao array de bytes(conteúdo do documento), posteriormente faz-se um *parse* ao mesmo obtendo um array com os bytes encriptados do conteúdo do documento e um array com o resumo. É chamada a função verifyMAC que retorna o valor *True* se a comparação entre o resumo e o novo resumo gerado forem iguais. Se a integridade se mantiver é chamada a função Decrypt() que decifra o conteúdo e retorna-o ao utilizador.

SD-STORE-A

Requisitos não implementados:

- Protocolo a funcionar para mais de um cliente
- O cliente (FrontEnd) não aguarda pela maioria das respostas dos servidores que enviam a tag (nº de servidores / 2), ou seja, não estamos a verificar casos quando um servidor falha subitamente.

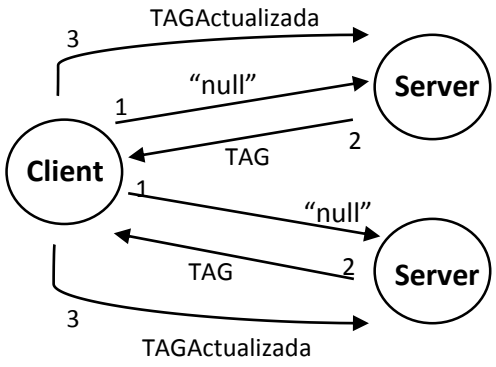


Figura ilustrativa do funcionamento do protocolo implementado

Grupo 41 Miguel Geraldes nº74115 João Rebêlo nº75376 Ana Teixeira nº76969
Sistemas Distribuídos
Relatório 2º Entrega

Em relação à operação store(), corremos todos os servidores. De seguida o cliente recebe a tag de todos, guardando-as numa lista para posteriormente escolher a tag com a versão mais actual, volta a enviar a tag actualizada para fazer update da mesma em todos os servidores, de forma a que estes tenham a versão mais recente do documento.

Na operação load(), a implementação do protocolo é idêntica à anterior, dado o cliente receber a tag de todos os servidores e escolher a tag com a versão mais recente, executando a operação load() no servidor que devolveu a tag mais actualizada, obtendo assim a ultima versão do Documento.