UNIVERSIDADE da MADEIRA

2021/2022
Programação Avançada – Projeto prático 2

1. Introdução

Com este projeto espera-se que os alunos demonstrem o conhecimento adquirido no segundo terço da disciplina de Programação Avançada. Assim espera-se a entrega de um projeto em JAVA que resolva o problema de comunicação descrito abaixo. Especificamente espera-se a utilização de:

- Encriptação simétrica;
- Encriptação assimétrica;
- Criação de MAC e Hash;
- Geração de chaves;
- Setup da comunicação.

2. Descrição do problema

O projeto proposto, definido de agora em diante como "pa-tls-chat", tem como objetivo a criação de um servidor de Chat em JAVA. Tendo como ponto de partida os vários projetos disponibilizados no GitHub da disciplina (stream_ciphers_and_HMAC, aes_ecb_cbc, asymmetric_encryption, chat_group, public-private, socket-with-hash, symmetric_encryption e ascii_encryption), espera-se que os alunos estendam e combinem estes projetos de forma a implementar um chat seguro, em que cada cliente possui diferentes requisitos de comunicação.

O funcionamento deste projeto deverá replicar a filosofia do protocolo TLS:

- Cada cliente possui uma configuração que deverá ser definida na sua criação. Esta configuração define:
  - Tamanho de chaves suportada
  - Algoritmo de encriptação suportado (AES, DES, 3DES ou RSA)
  - Utilização de Hash
  - Algoritmo de hash suportado (SHA-256, SHA-512, MD4 ou MD5)
  - …
- A definição destes parâmetros deverá ser responsabilidade do utilizador de cada cliente, e deverá ser definida através da consola.
- O servidor deverá suportar todos os algoritmos de encriptação discutidos na disciplina

Sempre que um novo cliente liga-se ao servidor este deverá realizar um *handshake* com o servidor, para que os parâmetros definidos acima sejam acertados antes da troca de mensagens no chat. A troca de mensagens no *handshake* deverá necessitar de o mínimo de mensagens possível, contudo inicia-se sempre com uma mensagem **CLIENT_HELLO**, e finaliza sempre com as mensagens **SERVER_OK** e **CLIENT_OK.**

A organização e a formatação das restantes mensagens do *handshake*, e das mensagens com o conteúdo do chat deverá ser implementada da forma que o grupo achar mais adequada, contudo consideramos que uma abordagem similar ao TLS será a implementação mais robusta. Após a finalização do *handshake*, o servidor deverá enviar para todos os clientes a seguinte mensagem:

```
[TIMESTAMP]:O cliente "NOME DO CLIENTE" ligou-se ao Chat.
```

O servidor deverá suportar 2 tipos de mensagens, Broadcasts e mensagens diretas. Por defeito sempre que um cliente envia uma mensagem esta mensagem é direcionada para todos os clientes ligados (Broadcast). Contudo um cliente poderá enviar uma mensagem apenas para outro(s) cliente(s) no servidor, para isso, no ínicio da mensagem o cliente deverá colocar:

```
@ClienteDestino "Mensagem a enviar"
```

Da mesma forma, se o cliente decidir enviar esta mensagem para vários clientes específicos o nome destes deverão ser indicados, separados por vírgulas:

```
@ClienteDestino1,@ClienteDestino2,@ClienteDestino10 "Mensagem a enviar"
```

2.1 Funcionamento Detalhado

Toda a comunicação no **pa-tls-chat** é mediada pelo servidor. Assim, o servidor será responsável por desencriptar mensagens de um cliente, para que estas sejam novamente encriptadas antes de serem reencaminhadas para outro cliente. Pois, visto que cada cliente terá o seu *setup,* a comunicação "direta" não será possível. Desta forma o servidor terá que manter um registo das configurações de encriptação de todos os clientes mesmo após a finalização do *handshake*. Ver Figura 1.

A comunicação deverá ser implementada com recurso a Sockets, e a troca de mensagens deverá ocorrer através da consola.

É óbvio que esta abordagem, onde o servidor tem que desencriptar/encriptar a mensagem antes de a encaminhar para o cliente(s) apropriado(s) apresenta uma vulnerabilidade de segurança. Assim espera-se que cada grupo encontre e implemente possíveis abordagens para mitigar este problema. Esta implementação deverá ser externa ao procedimento descrito no *handshake*.

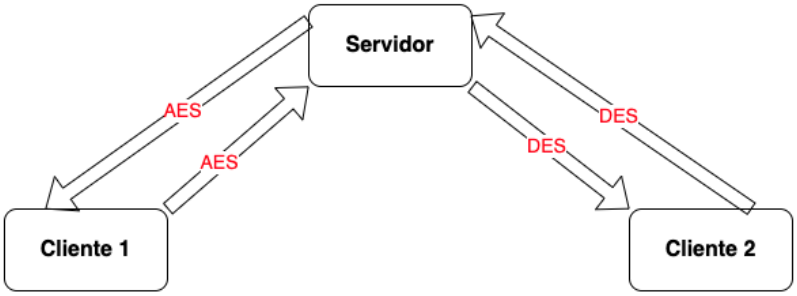

Figura 1: Exemplo de comunicação suportada pelo pa-tls-chat

Será dado um crédito extra aos grupos que implementarem corretamente o esquemas de distribuição de chaves de *Elliptic-Curve Diffie–Hellman* (ECDH), como alternativa ao método *Diffie–Hellman* tradicional.

3. Requisitos de entrega

1. O projeto deverá ser entregue até ao **dia 15 de Maio de 2022**. A entrega deverá ser realizada através do repositório da disciplina.
2. Juntamente com a implementação, deverão ser entregues testes unitários e relatório de *code coverage* gerado pela biblioteca JaCoCo. O ficheiro `pom.xml` está disponibilizado aqui, e já

contém as configurações para as bibliotecas JUnit, JaCoCo e para os *plugins* Maven Surefire e Javadoc.

3. Projetos que não compilem, e cuja sua execução seja, assim, impossível não serão considerados para avaliação.
4. Todas as classes, e os métodos do programa, deverão apresentar documentação de suporte de acordo com as práticas de documentação JAVA (Javadoc) discutidas na aula TP2

De forma a garantir os requisitos n.º 2, 3 e 4 é obrigatório que os alunos implementem GitHub Actions para a execução dos testes, criação do relatório de *code-coverage,* e criação da documentação de suporte. Poderão seguir os exemplos criados na aula TP2.

4. Avaliação Individual

A contribuição individual de cada aluno será avaliada tendo em conta o histórico de atividade do mesmo no repositório do projeto. Logo que iniciem a implementação, **todos** os grupos deverão partilhar os seus repositórios privados com os docentes da disciplina (**Dntfreitas** e **Lipegno**).

5.Código de ética e honestidade académica

Nesta disciplina, espera-se que cada aluno subscreva os mais altos padrões de honestidade académica. Isto significa que cada ideia que não seja do aluno deve ser explicitamente creditada ao(s) respetivo(s) autor(es). O não cumprimento do disposto constitui uma prática de plágio. O plágio inclui a utilização de ideias, código ou conjuntos de soluções de outros alunos ou indivíduos, ou qualquer outra fonte para além dos textos de apoio à disciplina, sem dar o respectivo crédito a essas fontes. A menção das fontes não altera a classificação, mas os alunos não devem copiar código de outros colegas, ou dar o seu próprio código a outros colegas em qualquer circunstância. De notar que a responsabilidade de manter o acesso ao código somente para os colegas de grupo é de todos os elementos.