Segurança de Redes e Sistemas de Computadores 2019/2020, 1º Semestre

Aula Prática (LAB 3)

(Guião de Orientação)

Objetivo: Nesta aula endereçaremos os seguintes aspetos de verificação e implementação prática para programação com JAVA/JCE:

- Aprender a usar na prática construções do tipo *Password-Based Encryption Schemes* no ambiente Java/JCE, com utilização de métodos criptográficos simétricos e implementações base de referencia em componentes de provedores criptográficos.
- Compreender os aspetos práticos de geração e uso (com exposição controlada) de chaves criptográficas no uso de criptografia simétrica e gestão de *keystores* JAVA para chaves criptográficas simétricas (ou também para chaves secretas MAC, HMAC ou CMAC)
- Discussão da abordagem de um protocolo para disseminação de mensagens multiponto, em que as hipóteses do modelo de adversário que possa atacar as comunicações exige defesas com provas mínimas de propriedades de autenticação de mensagens (*message-authentication*), integridade (*connectionless integrity control withou recovery*) e confidencialidade (*connectionless confidentiality*).
    - Pense na tipologia de ataques da framework X.800 que a implementação defenderá como contra-medidas de segurança.
- Desafios para consolidação da aprendizagem da prática

Sequência de Atividades (Hands-On):

1. Para os próximos pontos, comece por obter o arquivo lab3.tgz. Abra o arquivo de modo a obter o material que lhe é fornecido. **1. Geração de chaves criptográficas para algoritmos criptográficos simétricos** No arquivo preparado para a aula prática ver o código da diretoria **Keygeneration** A partir do código base (ver **KeyGeneratorExample.java**) verificar o funcionamento e treinar o uso de geradores de chaves para algoritmos criptográfico simétricos.

2. **Password Based Encryption** No arquivo preparado para a aula prática ver o código da diretoria **pbencryption** Ser-lhe-á feita uma apresentação sobre os esquemas de cifra com passwords, a sua utilização e os seus princípios de funcionamento. Deverá seguir essa apresentação (que pode aceder em documento PDF que também encontra no arquivo lab3. De seguida deverá praticar os exemplos e exercícios sugeridos na aula, a partir do código inicialmente disponibilizado, analisando o código seguinte e verificando as diferenças de utilização prática.
    - **PBEWithParamsExample.java**
    - **PBEWithoutParamsExample.java**
    - **BEOtherExample.java**

    Com base nas implementações de esquemas do tipo PBE que possui, deve verificar que é possível usar qualquer implementação normalizada deste tipo de algoritmos e

esquemas criptográficos que encontra nos diversos provedores criptográficos para a JCE.

3. **Keystores para armazenamento de chaves criptográficas para algoritmos simétricos**

Ser-lhe-á feita uma demonstração do uso de Keystores (normalizadas para Java) e que têm em vista sistematizar e normalizar o processo de gestão de chaves secretas para criptografia simétrica. Para este efeito:

a) vamos usar a ferramenta keytool (ou outras ferramentas congéneres), bem como os aspeto relativos às diversas opções para geração de chaves criptográficas bem como seu armazenamento em keystores.

b) devemos compreender a estrutura interna dos objectos "keystore", nomeadamente keystores do tipo JCEKS, bem como pode manipular esses objetos no contexto de programas JAVA que usam o suporte JCA/JCE.

Desafios (trabalho de casa)
Exercício (para programar)

A partir dos exercícios dos desafios lançados no LAB-1 (*challenges*) tente agora iterar esses desafios que visavam proteger mensagens no canal com propriedades de CONFIDENCIALIDADE E INTEGRIDADE, usando MACs, mas fazendo agora a gestão das chaves (chaves de confidencialidade e chaves MAC) em *keystores*. Para tal, dependendo do exercício que quiser considerar

Hipótese 1: enriquecer o protocolo C/S da aplicação CAPITALIZATION, passando a ter informação da *keystore* ou parâmetros (*security association parameters*) de forma configurável, partilhados entre cliente e o servidor mas configuráveis fora das respetivas implementações.

Hipótese 2: Usar a aplicação fornecida (MulticastQuoteServer e MulticastQuoteClient), protegendo o respetivo canal multi-ponto (servidor > grupo de clientes) nas propriedades desejadas

Hipótese 3: Usar a aplicação MCHAT (Multicast Chat), a keystore deverá estar partilhada entre todos os utilizadores (endpoints) da sessão de multicasting-chat, em que apenas temos um contexto de canal *Peer-Group* (com os vários clientes constituindo os multiparticipantes desse canal)

Note que a proteção de cada *keystore* em cada cliente pode ser feita de modo que cada utilizador usará as suas próprias passwords. Note que no caso da última hipótese, diferentes CHAT- SESSIONS (definidas com base em suporte de diferentes grupos Multicast), podem ter diferentes *keystores*, ou corresponder a múltiplas entradas de uma mesma *keystore*.