Segurança de Redes e Sistemas de Computadores 2021/2022, 1º Semestre

LAB 2

Materiais disponibilizados e guião de orientação

Objetivo: Programação em JAVA / JCE e observações relevantes para compreensão de parametrizações de construções para cifras simétricas, incluindo configurações de modo de processamento de blocos e *padding* bem como as respetivas implicações. Neste objetivo pretende-se obter sensibilização para que possam usar-se parametrizações seguras e adequadas aos fins em vista, evitando vulnerabilidades devidas ao uso incorreto de parametrizações ou configurações na programação. Os materiais e trabalho em aula ainda cobrem ainda a programação com funções de sínteses de segurança bem como de construções MAC (HMAC e CMAC) para construção de provas de autenticidade e integridade de mensagens. Finalmente, aborda-se também como programar com algoritmos para cifras simétricas em cadeia.

Sequência de Atividades (Hands-On):

1-Verif-JCE-CriptoProviders-Policy

Discussão (tutorial e observação prática) de aspetos relativos a detalhes sobre a arquitetura do pacote Java JCE (Java Cryptographic Extension), provedores criptográficos e algoritmos neles disponibilizados, bem como observações sobre o respetivo suporte *runtime*.

2-block-ciphers

Exercícios de Programação com Criptografia Simétrica e suas parametrizações com modos de cifra de blocos e *padding* normalizado e verificação que isso apenas lhe dá a base para abordar requisitos de confidencialidade (mas não de integridade) no que aprendeu nas aulas teóricas. Nesta parte deve seguir-se em sequência os exemplos/exercícios disponibilizados.

Comece por obter o arquivo lab2.tgz. Abra o arquivo de modo a obter o material que lhe é fornecido. Encontra código em diferentes diretorias no arquivo, devendo seguir em sequência como se segue:

O objetivo aqui é verificar como utiliza a JCE para programar (parametrizar para cifrar e decifrar) com diferentes algoritmos da criptografia simétrica (cifras de bloco) existentes nos seus provedores disponíveis, e como pode e deve usar os modos criptográficos simétricos, e para que fim ex: ECB (SimpleECBExample), CBC (SimpleCBCExample), CTR (SimpleCTRExample) ou CFB. Verifique os padrões de programação mostrados, identificando os efeitos diferentes

Deve ensaiar os programas e pode modificar de modo a compreender o funcionamento, bem, diferenças e impacto de usar os vários modos, bem como o papel da parametrização de *padding*. Tente compreender as implicações nas parametrizações das funções de cifra

bem como o impacto no tamanho dos blocos (*Plaintext* vs. *Ciphertext*), sabendo quais são os tamanhos dos blocos base processados pelos algoritmos. Faça os exercícios que lhe vão ser sugeridos na aula prática pelo docente.

Analise depois o código em InlineIvCBCExample.java e NonceIvCBCExmaple.java. Analise o que encontra de interessante nestes programas para aprender em relação à gestão de vetores de inicialização (IVs) ou *nonces* usados como IVs (vetores de inicialização). Deverá observar em mais detalhe na prática a utilização da parametrização de *padding e IVs* e s sua importância quando usa de forma segura os algoritmos criptográficos simétricos por blocos, bem como a importância desta parametrização e sua normalização.

Para o efeito, verificar a diretoria: **ExemplosVerifPadding**

Esteja atento à discussão e explicação que será feita para verificar o impacto desta parametrização quer do ponto de vista da segurança quer do ponto de vista do impacto em relação à dimensão da informação *plaintext* bem como da informação *ciphertext*.

3-Streaming

Material associado a um sistema de Streaming para visualização de conteúdos (filmes) em tempo real, com uso da ferramenta de visualização vlc. Este material será explicado na aula (Lab) para lançamento de um exercício inicial que consistirá em construir uma solução de segurança que tem como objetivo disseminar os conteúdos para visualização com base em canais de comunicação seguros. Esta solução permitirá disseminar os conteúdos cifrados de modo a que possam continuar a ser vistos em condições de streaming e visualização em tempo real (com a ferramente vlc).

4-OTHER-OBSERVATIONS-using-openssl

Conhecimento sobre utilização da ferramenta openssl (*comand-line tool*). Esta ferramenta está associada ao suporte de uma biblioteca criptográfica (openssl) implementada na lingugem C. Na aula discutir-se-ão exemplos de uso da ferramenta openssl que será depois usada para algumas demonstrações em aulas seguintes teóricas e práticas.

5-secure-hashing

Secure Hashing (ou uso de Funções para Síntese Segura de Mensagens)

Vamos utilizar o conteúdo da diretoria (seguindo e ensaiando os vários exemplos e demonstrações bem como exercícios sugeridos na aula).

a) Exercicios-secure-hashing

Deve seguir a seguinte sequência para poder compreender o que está envolvido:

TamperedExample.java

TamperedExample2.java

Tratam-se de exemplos que demonstram um ataque do tipo *Message Tampering* que atenta

contra a INTEGRIDADE das mensagens. Verifique e compreenda o problema deste tipo de ataques e como não os protege usando cifras simétricas.

Verifique que pode fazer ataques deste tipo, a partir dos programas fornecidos, sem que o "emissor" e o "receptor" se apercebessem do que estão em presença de um atacante.

b) Secure Hashing (Funções para Síntese Segura de Mensagens) para proteção de integridade

Verificaremos de seguida como resolver o problema da INTEGRIDADE. Comece por compreender o código em: **TamperedWithDigestExample.java**

Utilize este programa para ver como os ataques de *Message Tampering* podem agora ser defendidos ... Tudo parece assim permitir proteger a INTEGRIDADE.

... Ou será que não? Estude e verifique o programa: **TamperedDigestExample.java**

Exercício: Esteja atento à demonstração que lhe será feita e tire as suas conclusões do que aprendeu.

O que há a fazer para conseguir ter uma proteção definitiva face a ataques que visam quebrar a integridade e a autenticidade das mensagens ?

Questão importante a reter: como usar corretamente a combinação de cifras simétricas (para proteger a CONFIDENCIALIDADE) e sínteses seguras (para proteger a INTEGRIDADE) ? Qual a melhor solução e como tratar na prática o TRADEOF entre segurança e desempenho ?

6-secure-macs

Exercícios de Programação com Criptografia Simétrica, Funções de Síntese Segura de Mensagens (*Secure Hashing*) e Códigos de Autenticação de Mensagens (*MACs, HMACs e CMACs*)

MACs (*Message Authentication Codes*), HMACs (ou *Hash-Based MACs*) e CMACs (ou Cryptographic MACs)

Vamos agora utilizar o conteúdo da última diretoria

Exercicios-MAC

Esta diretoria tem um único exercício.

Verifique como pode usar MACs. Neste caso podemos usar HMACs (ver o caso de **TamperedWithHMacExample.java**) ou CMACs (no caso de **CipherMacExample.java**) para prevenir em definitivo problemas de proteção das comunicações para resistir a ataques à integridade de mensagens com provas incluídas de autenticidade das mesmas (ou seja *Message Aauthentication Codes*).

Desafios (trabalho de casa e reflexão)

Exercício 1 (para programar)

A partir dos exercícios dos desafios lançados no LAB-1, será que consegue iterar esses desafios protegendo também as mensagens agora com CONFIDENCIALIDADE E INTEGRIDADE, usando MACs (com HMACs ou CMACs) para implicitamente verificar a integridade com autenticidade das mensagens. Será que consegue arranjar forma da sua solução ser parametrizável em relação ao uso de algoritmos criptográficos (bem como a possível necessidade de IVs ou parametrizações de Padding), bem como funções de sínteses de segruanças e construções MAC ? Como acha que também podia evitar ataqus de Message-Replaying" ?

Exercício 2 (para definir uma solução para discutir no próximo Lab3)

Como construir um programa que possa ser usado como simples ferramenta de auditoria que verificasse ou atestasse a integridade (ou quebras de integridade) do file-system dos computadores dos laboratórios do DI (ou do seu próprio computador), ou de diretorias críticas específicas no seu file-system, de modo a prevenir ataques por intrusões que tivessem em vista modificar ficheiros considerados críticos (ex., aplicações, ficheiros com configurações críticas, ficheiros associados a devices, etc). Como o faria ? Quais as diretorias ou partes do file-system que iria privilegiar ? Porquê ? Como proporia uma solução para o efeito ? Com que segurança efetiva a sua ferramenta produziria o efeito desejado ? Qual era a arquitetura que estenderia para introduzir mais segurança numa auditoria desse tipo? Como a programaria uma tal ferramenta?

7-stream-ciphers

Exemplos e exercícios que mostram como programar com algoritmos criptográficos simétricos para cifras em cadeira.