Segurança em Sistemas e Redes

Manuel Barbosa at dcc.fc.up.pt

MIEEC – 2015/2016

Tutorial 3 — Client-Server Encryption

The goal of this tutorial is to explore the performance of different block cipher modes of operation in network communication.

As a starting point, create a small client-server system composed of two command-line applications:

- These applications should be implemented as two Java classes: Client and Server.
- The system should allow the Client to upload a file to the Server, which is listening on a given `port` (e.g. 4567).
- The Server will accept one connection at a time, but many files may be uploaded by running the Client applicatin multiple times (sequencially).
- The Client application will receive the file to upload as a parameter on startup.
- The Server application will receive the upload folder as a parameter on startup.
- Whenever the Client connects to the Server, it will simply send the file contents (in binary).
- The Server will automatically assign a name to the incoming file, e.g. `file1`, `file2`, etc. and store each uploaded file in the upload folder.
- All communications should be encrypted.
- The secret key should be stored in a file on both sides of communication.

Experiment with the following encryption schemes and modes of operation and explain the impact of these options on buffering and padding:

- RC4
- AES/CBC/NoPadding
- AES/CBC/PKCS5Padding
- AES/CFB8/PKCS5Padding
- AES/CFB8/NoPadding
- AES/CFB/NoPadding

Make sure to characterize in detail the operation of the `update` and `doFinal` methods in the `Cipher` class.

Note that most of the above modes of operation require an initialization vector to be randomly generated prior to communication and transmitted in the clear.

Addicional relevant classes:

- javax.crypto.spec.IvParameterSpec
- java.security.SecureRandom