

## Worksheet 3 – Symmetric Cryptography

### Symmetric Algorithms @.NET

#### Covered topics:

- Confidentiality: symmetric encryption
- Encoding UTF8 and BASE64
- The key exchange problem

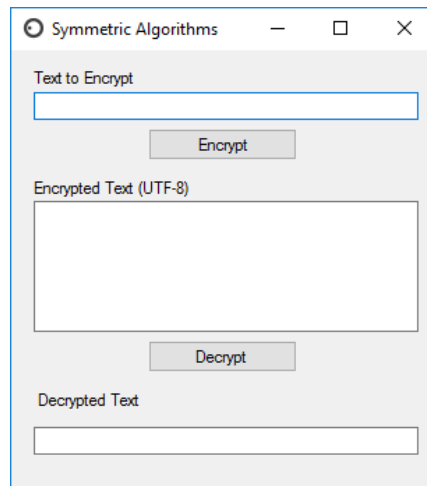
©2020: {rui.ferreira,nuno.costa,vitor.fernandes,ricardo.p.gomes,nuno.reis, marisa.maximiano}@ipleiria.pt

## 1. Symmetric Encryption

The following exercises are intended to show how we can use symmetric algorithms, implemented in .NET, to ensure data confidentiality.

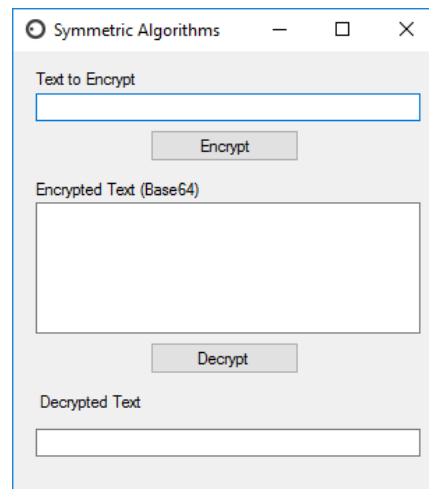
#### Exercises

1. Download the project "ei.si-worksheet3-ex1.1" and use the existing components in the form to:



- a) Encrypt the data of the first textbox ("Text to Encrypt") and put the result in the second textbox ("Encrypted Text"), using UTF-8 encoding;
- b) Decrypt the data from the second textbox ("Encrypted Text"), which were previously encrypted, and put the result in the third textbox ("Decrypted Text").

2. Change the project from the previous exercise to now use the BASE64 encoding instead of UTF-8. Note: Use the class "SymmetricsSI" (import the "SymmetricsSI.dll" file).

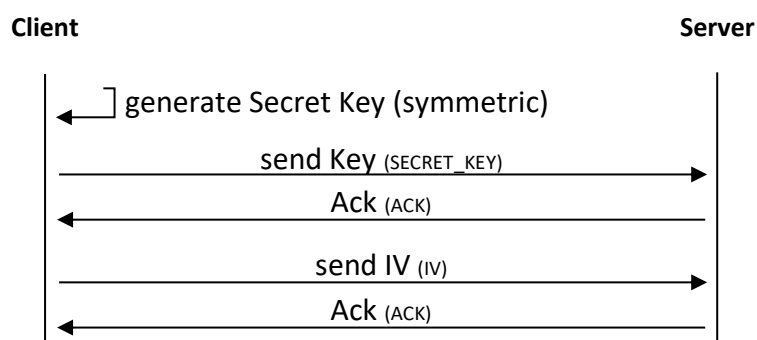


## 2. Key Exchange

Next is presented an exercise where only the **exchange of symmetric key** is required, and some data needs to be transferred. The goal is to confirm that the generated key was properly transmitted to another entity, through the network over a non-secure channel.

### Exercises

1. Download the project "ei.si-worksheet3-ex2.1" and, using the existing base code, implement the concept of symmetric key exchange, where the protocol to adopt is shown in the next figure:



### Notes:

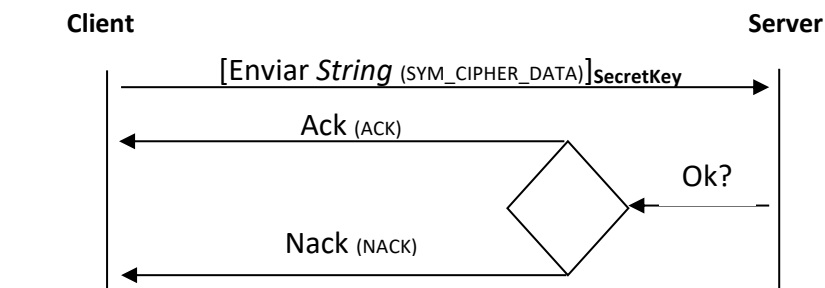
- Choose: encryption algorithm, encryption mode and padding mode;
- No data should be exchanged;
- This key exchange is not supposed to be done securely;
- The use of the ProtocolSI is required.

### 3. Confidentiality

Now that the key exchange was implemented (although in a non-secure way), it is possible to ensure that data can be securely transmitted through the network.

#### Exercises

1. Using the project from exercise 2.1, implement the next protocol to achieve confidentiality in the exchange of data:



#### Notes:

- Send an encrypted string using **SymmetricsSI** class;
- An ACK must be returned to confirm the correct receipt of the message, or a NACK otherwise;
- The exchange of keys is not yet done securely (to be solved in next worksheet).

### 4. Extra Class

#### Exercises

1. Download the project "ei.si-worksheet3-ex4.1" and use the existing components in the form to allow the user to choose the encryption algorithm (AES or TripleDES).

The screenshot shows a Java Swing window titled "Symmetric Algorithms". It has a standard title bar with minimize, maximize, and close buttons. The window contains the following components:

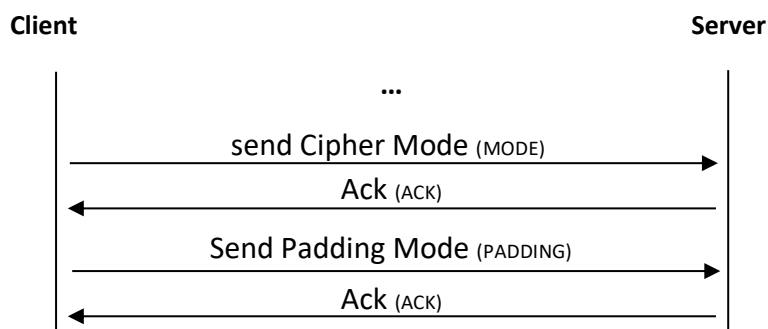
- A text field labeled "Text to Encrypt".
- A dropdown menu labeled "Algorithm" with a blue border and a downward arrow. The menu is open, showing two options: "AES" and "TripleDES".
- A button labeled "Encrypt" to the right of the algorithm dropdown.
- A text area labeled "Encrypted Text (Base64)".
- A button labeled "Decrypt" below the encrypted text area.
- A text field labeled "Decrypted Text" at the bottom.

2. Download the project "ei.si-worksheet3-ex4.2" and use the existing components in the form to encrypt and decrypt a file. Please note:

- You must use the *StreamReader* and *StreamWriter* classes to pass the data into symmetric algorithm elements (*MemoryStream* and *CryptoStream*);
- The "Choose File" button already has code (`openFileDialog1`) to choose a file through a classic Windows window. It should also show, in the appropriate textbox, the first 200 bytes of that file;

- Use the "Encrypt ..." button to encrypt the contents of the selected file and save this information in a new temporary file "*c:\temp\temp.dat*". It should also show, in the appropriate textbox, the first 100 bytes of the encrypted file;
- Decrypt the contents of the temporary file "*c:\temp\temp.dat*" with the "Decrypt ..." button and display the first 200 bytes in the appropriate textbox.

3. Using the project from exercise 3.1, allow the client to choose the encryption and the padding mode and implement the following protocol:



4. Using the project from exercise 4.3, (using ProtocolSI) implement a Bitcoin currency conversion service for Euros and vice-versa (conversion factor: 100.00). The fundamental requirement is that all communication transmitted by the network is confidential.

Note: the key exchange should still not be done securely.