Biomedical Security

Workshop

22-4 2021

- Due by 4-5 2021.
- Send your answers in a zip by e-mail to erwin@liacs.nl
- Use as filename: **<your student number>_<your name>_iBS_Workshop.zip**

PyCryptodome

In this workshop we use the Python Cryptography package PyCryptodome which you can find on https://pycryptodome.readthedocs.io/en/latest/index.html .

Preparations (Ubuntu)

Start with making a virtual environment. Issue the following commands:

```
virtualenv crypto --python=python3
source ./crypto/bin/activate
python3 -m pip install --upgrade pip
pip install pycryptodome
```

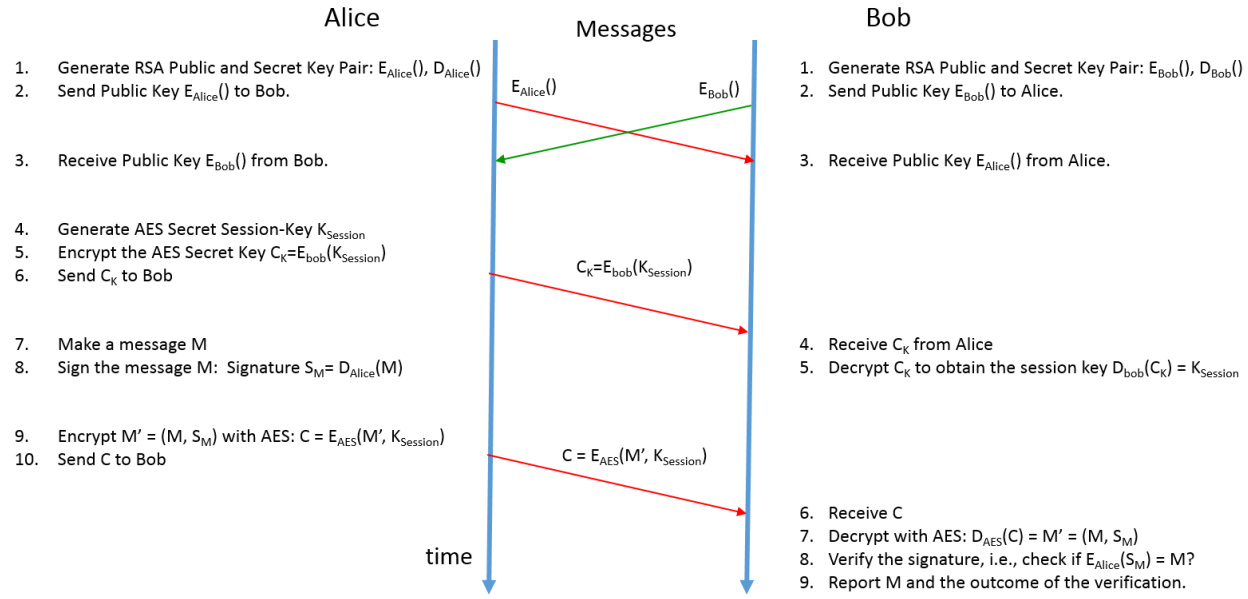
Now go to the examples page https://pycryptodome.readthedocs.io/en/latest/src/examples.html and verify that the given examples work in your environment.

Encryption and Decryption with AES:

```
from Crypto.Cipher import AES
from Crypto.Random import get_random_bytes
#Encryption
data =b'Sixteen bytedata'
key = b'16byte secretkey'
cipher = AES.new(key, AES.MODE_EAX)
ciphertext, tag = cipher.encrypt_and_digest(data)
file_out = open("encrypted.bin", "wb")
[ file_out.write(x) for x in (cipher.nonce, tag, ciphertext) ]
file_out.close()

#Decryption
file_in = open("encrypted.bin", "rb")
nonce, tag, ciphertext = [ file_in.read(x) for x in (16, 16, -1) ]

# let's assume that the key is somehow available again
cipher = AES.new(key, AES.MODE_EAX, nonce)
data = cipher.decrypt_and_verify(ciphertext, tag)

file_out = open("plaintext.bin", "wb")
file_out.write(data)
file_out.close()
```

Assignment 1

Consider the following protocol to exchange a signed message M between Alice and Bob:



Use PyCryptodome to implement the depicted protocol in a python program called ***secure*** in a way that is transparent to the user. Hereby Alice would use the program as follows:

```
secure send Bob 'message here'
```

Program Output: Message 'message here' sent to Bob.

And Bob would use the program as:

```
secure receive Alice
```

Program Output: (Non-) Authenticated Message 'message here' received from Alice

Implement the sending and receiving of the protocol-messages by respectively writing and reading files with the following names:

Write $E_{Alice}()$ in a file named *RSA.Alice.Public*
Write $E_{Bob}()$ in a file named *RSA.Bob.Public*
Write C_K in a file named *AES.SessionKey.Alice.Bob*
Write C in a file named *AES.CryptoText.Alice.Bob*

Add your Python code file *secure.py* to your .zip file.

Assignment 2

Try also to communicate an authenticated secret message using your implementation of the protocol with at least one of your fellow students by exchanging protocol-messages only.

Note: for this to work you have to agree on a common format of the message-files.

Add the messages of this communication to your .zip file.