# INFA723 Cryptography and Network Security

# Lab7 Use OpenSSL to Create RSA Public/Private Key (4096bits) with Password Protection

# 1 Introduction

OpenSSL toolkit provides a command line tool and a crypto library used for a wide range of crypto operations. In this exercise, we are going to learn how to use the command line tool to create RSA public key and private key with password protection. The lab will further demonstrate how to use OpenSSL to encrypt/decrypt a file using an RSA public/private key. Note that for all the functions implemented in the lab, you can find a way to implement them using the OpenSSL crypto library.

Note that the lab has a limitation of the size of the plaintext since it must be smaller than the modulo n we generated in the lab. If you want to encrypt a large file using the 4096bits keys, you have to split the message in small chunks and encrypt each chunk separately.

A lab package has been created for the lab. Go to the class website and download the lab package. Assume your Cygwin installation folder is c:\cygwin. Unzip the package to your Cygwin home folder, e.g., C:\cygwin\home\user name.

# 2 Objectives

* Learn how to use OpenSSL command tool to generate a RSA public key and a private key with password protection
* Learn how to use OpenSSL command tool to encrypt and decrypt a file using RSA public-key cryptography

# 3 Use OpenSSL to Create RSA Public/Private Key (4096bits) with Password Protection

1. Generate a 4096 bits RSA private key with password protection.

**$ openssl genrsa -des3 -out private4096.key 4096**

**Generating RSA private key, 4096 bit long modulus**

**...................................................++**

**....................................++**

**e is 65537 (0x10001)**

**Enter pass phrase for private4096.key:**

**Verifying - Enter pass phrase for private4096.key:**

Enter a password to protect the private key. The same password will be required when decrypting a message.

1. Extract public key from the private key.

**$ $ openssl rsa -in private4096.key -pubout -out public4096.key**

**Enter pass phrase for private4096.key:**

**writing RSA key**

1. Encrypt a file using public key

**$ openssl rsautl -encrypt -pubin -inkey public4096.key -in plaintext1.txt -out encrypted.txt**

Note that the OpenSSL RSA utility does not automatically separate the message to make sure m<n. Thus, the size of the plaintext here cannot be too long. If the plaintext is too long, an error will happen when encrypting the file.

1. Decrypt the file using private key

**$ openssl rsautl -decrypt -inkey private4096.key -in encrypted.txt -out c.txt**

**Enter pass phrase for private4096.key:**

To decrypt the message, enter the password you put in Step 1.

1. Compare plaintext1.txt and c.txt

**$ diff plaintext1.txt c.txt**

# 4 Question

Confidentiality, integrity, authentication are desired security services when transmitting messages over communication networks. Use the same notations and similar diagrams as we discussed in the “Implementation OSI Security Services” to design scenarios to demonstrate how to provide confidentiality, integrity, and authentication using the following provided cryptographic operations.

1. (8 points) Use symmetric cipher and message authentication code to provide confidentiality, integrity, and authentication. (use the giving cryptographic operations only).
2. (8 points) Use symmetric cipher and digital signature to provide confidentiality, integrity, and authentication. (use the giving cryptographic operations only).
3. (4 points) Compare the two scenarios as designed in a) and b). What are their advantages and limitations?