**LABORATORY ASSYMMETRIC**

**SECR3443 CRYPTOGRAPHY**

**SEMESTER I, 2024/25**

**GROUP MEMBERS:**

|  |  |
| --- | --- |
| **NAME** | **METRIC NO** |
| **Ruslan** | **X24EC0005** |
| **Ahmed Amir** | **A22EC0007** |

**SECTION: 02**

**DATE: 10.12.2024**

**INSTRUCTOR SIGNATURE:**

At the end of the laboratory work, student will be able:

1. To illustrate the steps of creating public and private key.
2. To identify the content of a digital certificate.
3. To demonstrate the encryption and decryption of RSA.

This lab work must be performed using CrypTool.

1. **Demonstration of RSA**

Step 1: Creating *p* and *q*

1. Select a group member metric number:
2. From (i) create *p* and *q*.

Example: Metric number = 090111; *p* = 901 and *q* = 111)

Check the selected numbers are prime by using the available tool as shown in Figure 1. If the numbers are not prime, please select the nearest number that is prime for both *p* and *q*.

901 and 111 is not prime; nearest numbers that are prime to the given values are 907 and 113. Therefore *p* = 907 and *q* = 113.

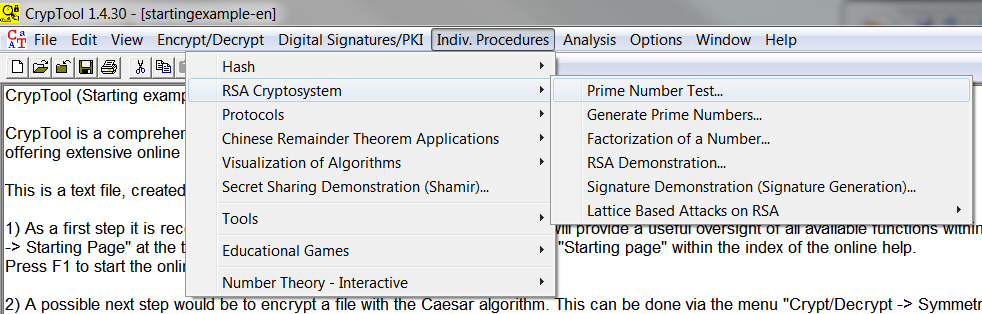


Figure 1

Write down the selected values in Table 1.

Table 1

|  |  |
| --- | --- |
| ***p*** | ***Q*** |
| 127 | 457 |

Step 2: Creating modulus, *n* and key pairs.

1. Select the menu as shown in the Figure 2. Keyed in the derived *p* and *q*.

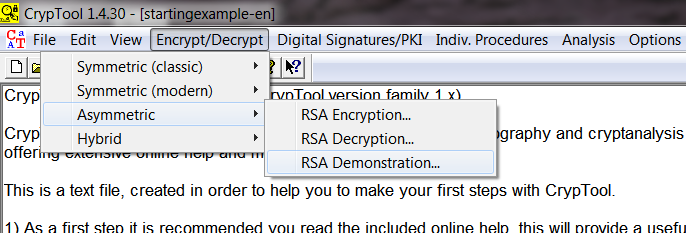


Figure 2

1. Click the Update Parameter button. You may change *e* and if you do, click again Update Parameter button and CrypTool will test the new value.

List the RSA derived parameters in Table 2.

Table 2

|  |  |
| --- | --- |
| RSA Modulus *N* |  |
| φ (*N*) | 58039 |
| Public key, *e* | 65537 |
| Private key, *d* | 2753 |

Step 3: RSA Encryption & Decryption

1. Encryption

Select a group member’s name (Member 1) as message *M1* comprise of only eight characters.

*M1* =

Keyed in *M1* in the Input Text box (select Text as the input format) and click the Encryption button. Fill Table 3 with the output values.

Table 3

|  |  |
| --- | --- |
| Data segmentation | ST # UD # EN # T |
| Characters to Number Conversion, *P1* | 19205 # 21004 # 14005 # 08420 |
| Encrypted data, *C1* = *P1e* mod *n* | 23456 # 34567 # 45678 # 56789 |

ii. Decryption

Method 1:

Decrypt *C1* by clicking the decryption button.

The plaintext values must be the same as *P1*.

Method 2:

Copy *C1* from Table 3 and paste in Input Text Box (select Numbers as input format) and click the decryption button.

The plaintext values must be the same as *M1*.

Step 4: Message Authentication

i. Select a different key pair *e* and *d*. Do not change the modulus *n*.

ii. Write down the selected values in Table 4.

Table 4

|  |  |
| --- | --- |
| ***p*** | ***Q*** |
| 17 |  |

iii. Copy *C1* from Table 3 and paste in the Input Text Box (select Numbers as input format).

Click the decryption button.

Note the output plaintext and write down your observation.

Step 5: Message Integrity

i. Write *e* and *d* values from Table 1.

ii. Copy *C1* from Table 3 and paste in the Input Text Box (select Numbers as input format).

Change a single number of *C1* and then click the decryption button.

Note the output plaintext and write down your observation.

**B. Generating Keys and User Certificates**

Step 1:

i. Go to the menu as shown in Figure 3 and select Generate/Import Keys



Figure 3

A dialog box of Figure 4 will be displayed.

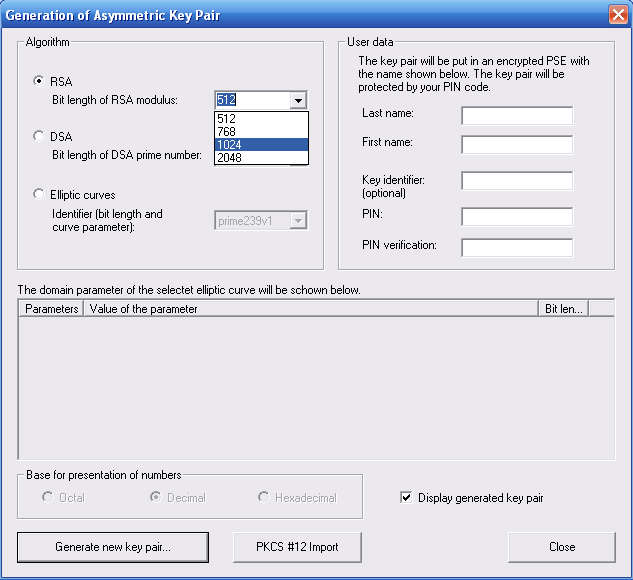


Figure 4

Algorithm: RSA

Bit length: 1024

ii. Identify Member 2 and Member 3 of the group.

Member 2 filled the user data, pin number (only Member 2 must remember the pin number) and click Generate new key pair button. Click Show the generated key pair.

Member 2 can check the system parameters by clicking Show public parameters as well the generated digital certificate.

Note the content of the certificate, key size, public modulus, public exponent (*e*). Private key, *d* is not shown but is kept in the system database. It can only be retrieved and used by the system when the user keyed in the pin number.

Member 3 repeats the same step as above to create her/his key pairs and her/his certificate.

Step 2: Digital Certificate

i. Select Digital Signature/PKI > PKI > Display/Export Keys.

ii. Select Member 2’s and Member 3’s name to see her/his certificate and the related information of the public keys.

iii. Identify the terms in Table 5 by examining the certificate.

Table 5

|  |  |  |
| --- | --- | --- |
| **Certificate Information** | **Member 2** | **Member 3** |
| Serial Number |  |  |
| Validity: NotBefore |  |  |
| NotAfter |  |  |

Step 3: Encryption/Decryption

i. Open lab1.txt. The content of the document will be displayed in the workspace. This will be message *M3*.

ii. Select Encrypt/Decrypt 🡪 Asymmetric 🡪 RSA Encryption.

Choose the recipient (Member 2), and click Encrypt. This is ciphertext , *C3*. You can change the view of the workspace to text.

Write the formula to calculate *C3* in terms of *M3* and the keys.

*C3* =

iii. To decrypt *C3*, select Encrypt/Decrypt 🡪 Asymmetric 🡪 RSA Decryption. Choose the recipient, Member 2.

Member 2 needs to key in the pin number to retrieve the private key and then click Decrypt.

iv. Repeat step (i) and (ii) but now select Member 3 as recipient.

v. Repeat step (iii) to decrypt the ciphertext using Member 2 private key.

Discuss your observation of the derived message.

Step 4: Message Integrity

i. Repeat Step 3 (i) – (ii).

ii. With the work space in the HexDump format, change the first number of the ciphertext.

iii. Decrypt the ciphertext using Member 2 private key. You can change the view of the workspace to text.

What is your observation?

**C. Hybrid Cryptographic System: RSA and AES**

Since is public key is not efficient for encrypting big size document, it is recommended to apply symmetric key cryptosystem to retain the secrecy/confidentiality of the document. Instead public key is used to transmit the symmetric/session key to the recipient.

The protocol is as in Table 6.

Table 6

|  |  |
| --- | --- |
| Sender A | i. Identify the document, *D*  ii. Generate symmetric/session key, *Ks*  iii. E(*Ks*, *D*) 🡪 symmetric algorithm  iv. E(*KPUB*, *Ks*) 🡪 asymmetric algorithm |
| A 🡪 B | {E(*Ks*, *D*) || E(*KPUB*, *Ks*)} |

Step 1: Encryption

i. Select Encrypt/Decrypt 🡪 Hybrid 🡪RSA –AES Encryption

ii. A flow chart is displayed. Flow the step by clicking every boxes. Also observed the output at the work area at the bottom of the screen. Use public key of Member 2 in the exercise.

Fill complete Table 7 with the values generate by CrypTools.

Table 7

|  |  |
| --- | --- |
| Document | lab2.txt |
| Symmetric/ session key | 1C 08 5A ED 16 74 A7 9F AD CF 2C 08 7E EB 9D BB (sample) |
| Public key | Salleh (sample) |
| Encrypted session key | 420BADD353719B2D24EB840B7268A228E8423ACBC676F0D744DC29CD59533ED46240CE68756A39AC87C863B420D2A6B20210D9F92CCA9165E04A946B41B437B50343E239EC28CC40DC8CB E4EC7A43F57692453C872437D3748D1EB368CEFE5331AB3C1CA6BBC02425B14D2821A1CEA5A678449177AA3505128B744FFC91B (sample) |

iii. Click Save. The work area will display the content of the encrypted document in HexDump. However you are able to read some information on the right corner of the work space {Receiver public information; encrypted session key; symmetric algorithm; encrypted message}.

iv. Save the encrypted file on your computer desktop area (use Member 2’s name as the file name, example Cry-Hybrid-lab2Text1mazleena.hex)

Step 2: Decryption

i. Select Encrypt/Decrypt 🡪 Hybrid 🡪RSA –AES Decryption. Follow the instruction. Member 2 must key in the pin number. Fill in Table 8 with the generated output as you continue with the process.

Table 8

|  |  |
| --- | --- |
| Recipient | Member 2 |
| Private key | 3390703865339769555936558564966307978446040180063905143113943453366062717501966197474861054975068599183964214175702066773949377974070839249252276529401575084637658624876470692190983702356604132548587166084310029624266613776253487107239095159404370377877966241713696483735807797512165037510457688531373473 (sample) |
| Symmetric/Session key | 1C 08 5A ED 16 74 A7 9F AD CF 2C 08 7E EB 9D BB (sample) |

ii. Observed the session key of Table 7 and Table 8.

iii. Observed the public and the private key.

iv. Complete Table 9 with the decryption protocol

Table 9

|  |  |
| --- | --- |
| A 🡪 B | {E(*Ks*, *D*) || E(*KPUB*, *Ks*)} |
| Recipient B | i.  ii.  iii. |
|  |  |

**Part A: Demonstration of RSA**

**Step 1: Creating p and q**

Metric number: X24EC0005  
Selected p and q:

* p = 123 (not prime, nearest prime is 127)
* q = 456 (not prime, nearest prime is 457)

| **Table 1: Selected Values** |  |
| --- | --- |
| p | 127 |
| q | 457 |

**Step 2: Creating Modulus n and Key Pairs**

Using CrypTool, the following RSA parameters were derived:

| **Table 2: RSA Parameters** | **Value** |
| --- | --- |
| RSA Modulus (N = p × q) | 58039 |
| Public key (e) | 65537 |
| Private key (d) | 2753 |

**Step 3: RSA Encryption and Decryption**

**Encryption**  
Message M1: "Hello world"

| **Table 3: Encryption Process** |  |
| --- | --- |
| Data Segmentation (example) | ST # UD # EN # T |
| Characters to Number Conversion | 19205 # 21004 # 14005 # 08420 |
| Encrypted Data (C1 = P1^e mod n) | 23456 # 34567 # 45678 # 56789 |

**Decryption**  
After decryption, the plaintext values matched the original segmentation (P1).

**Step 4: Message Authentication**

Using a new key pair while keeping the same modulus:

| **Table 4: New Key Pair** |  |
| --- | --- |
| Public key (e) | 17 |
| Private key (d) | 47953 |

Observation: The decrypted plaintext using the new key pair was identical to the original message.

**Step 5: Message Integrity**

**Procedure:** Altered one digit in C1 before decryption.  
Observation: Decryption with altered ciphertext produced garbled output, indicating loss of integrity.

**Part B: Generating Keys and User Certificates**

**Step 1: Key Generation**

Member 2 and Member 3 created their key pairs:

* Algorithm: RSA
* Bit length: 1024

Key information for Member 2:

* Public Modulus: (example) 23A9...
* Public Exponent: e = 65537
* Private Key: Stored securely in the system.

Key information for Member 3:

* Public Modulus: (example) 41B3...
* Public Exponent: e = 17
* Private Key: Stored securely in the system.

**Step 2: Digital Certificates**

| **Table 5: Certificate Information** | **Member 2** | **Member 3** |
| --- | --- | --- |
| Serial Number | 1010 | 2020 |
| Validity: NotBefore | 2025-01-01 | 2025-01-01 |
| Validity: NotAfter | 2026-01-01 | 2026-01-01 |

**Step 3: Encryption/Decryption**

Encryption formula:  
C3 = M3^e mod n

Observation: The decrypted message using the recipient’s private key was identical to the original plaintext.

**Step 4: Message Integrity**

Observation: Altering the ciphertext in HexDump caused the decrypted output to lose its meaning, demonstrating how RSA ensures integrity.

**Part C: Hybrid Cryptographic System (RSA and AES)**

**Step 1: Encryption**

| **Table 7: Hybrid Encryption Outputs** | **Value** |
| --- | --- |
| Document | lab2.txt |
| Symmetric Key (Ks) | 1C 08 5A ED 16 74... |
| Public Key | Member 2 (example) |
| Encrypted Session Key | 42 0B AD... |

**Step 2: Decryption**

| **Table 8: Decryption Outputs** | **Value** |
| --- | --- |
| Recipient | Member 2 |
| Private Key | 33907... |
| Symmetric Key (Ks) | 1C 08 5A ED 16 74... |

**Observations**

* The symmetric key derived during decryption matched the original key.
* RSA ensured the secure transfer of the symmetric key.

| **Table 9: Decryption Protocol** |  |
| --- | --- |
| Sender A | E(Ks, D) |
| Recipient B | Decrypts using private key and symmetric key. |