**SECURITY IN COMPUTING AND IT COSC2536/2537**

**ASSIGNMENT #2**

**Total Marks: 35 (Contributes 35% of the total Grade)**

**Submission Deadline: Week 11, Tuesday the 21st May 2019 11:59pm**

**Special Instructions:**

* **You are required to submit answers as a PDF or Word Document format by uploading in the CANVAS on or before due date.**
* **You are required to show all of the steps and intermediate results for each of the questions.**
* **Please DO NOT provide ONLY codes as answers.**

**PART-1 (Submit in the CANVAS)**

**Q**

**1**

**.**

**(**

**Privacy**

**-**

**Preserving**

**Computation)**

**[6**

**Marks**

**]**

Suppose there are

**9**

**voters**

to vote for

**YES**

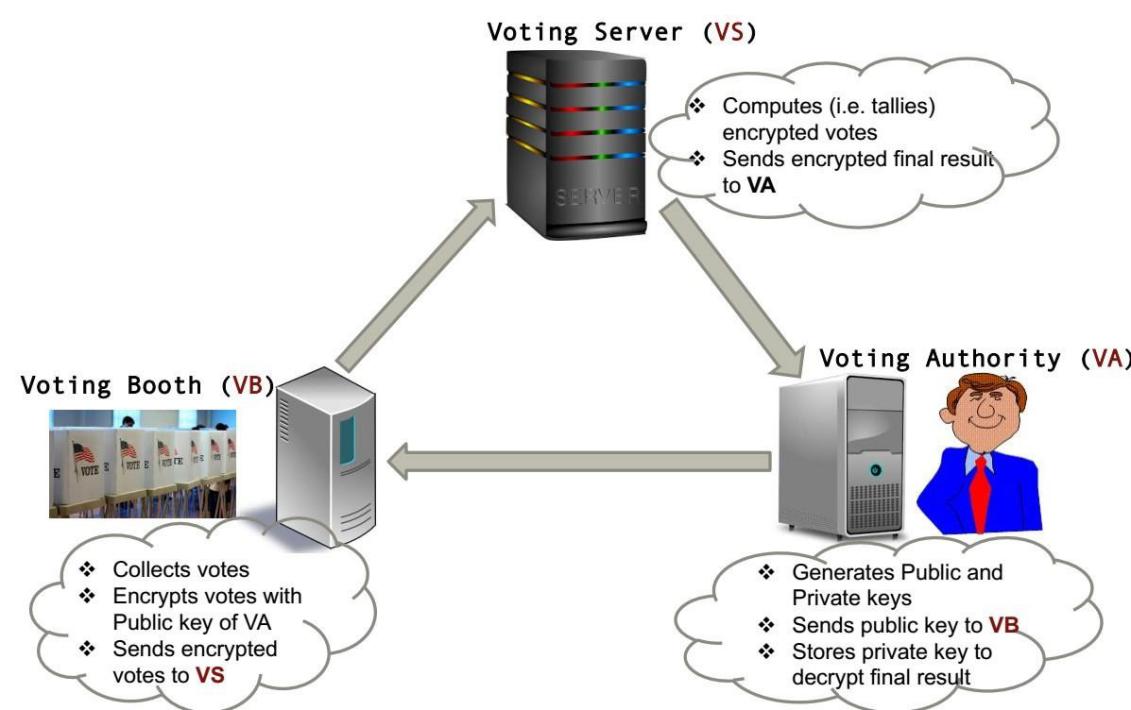
or

**NO**

to give their opinions.



Design a secure voting prototype as shown in **Figure-Q1** using **Paillier cryptosystem** where the votes must be **encrypted** from **Voting Booth** before sending them to the **Voting Server**.



**Figure-Q1: Secure voting system**

Assume, **five voters** will vote for **YES** and **four voters** will vote for **NO**. The **Voting Authority** should find **five YESs** and **four NOs** after counting the votes. The **Voting Authority** chooses ***p=37***, ***q=73*** and select ***g=9868***. The private numbers chosen by **9 voters** and their votes are as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| **Voter No.** | **Voter’s Private Number, *r*** | **Vote** | **Voting message, *m*** |
| 1 | 20 | YES | **00010000 = 16** |
| 2 | 21 | YES | **00010000 = 16** |
| 3 | 22 | YES | **00010000 = 16** |
| 4 | 23 | YES | **00010000 = 16** |
| 5 | 24 | YES | **00010000 = 16** |
| 6 | 25 | NO | **00000001 = 1** |
| 7 | 26 | NO | **00000001 = 1** |
| 8 | 27 | NO | **00000001 = 1** |
| 9 | 28 | NO | **00000001 = 1** |

Show the **encryption**, **homomorphic computations** **and decryption** processes. [***Hints***: Refer to the lecture-5 Secure e-voting. You need to represent the total number of votes by **8-bit string**. The first **4 (four)** bits should represent the votes for **YES** and the rests for **NO.** When adding a vote for **YES**, the system adds **00010000**, which is **16** in integer. Similarly, the system adds **00000001** when voting for **NO**, which is **1** in the integer form.]

• Receiver computes n = p\*q = 37\*73 = 2701

• Receiver sends public key (n, g) = (2701, 9868) to the sender

• Receiver computes private key parameter λ= lcm(p-1, q-1) = lcm(36, 72) = 72

• Receiver computes k = L(g^λ mod n^2) using function L(u)=(u 1)/n.

Here, g^λmodn^2=9868^72mod2701^2=2565951

Therefore, k=2565950/2701=950

• Receiver computes private key parameter µ as follows

µ = k^-1 mod n = 950^-1 mod 2701 = 2556

• Receiver saves private key (λ, µ) = (72, 2556)

To encrypt the vote the sender does the followings:

• Sender encrypts plain-text m as follows

1st voter c =g^m\*r^n mod n^2=9868^16\*20^2701mod2701^2

=3081436

2nd voter c =g^m\*r^n mod n^2=759231

3rd voter c=g^m\*r^n mod n^2=6642664

4th voter c=g^m\*r^n mod n^2=3965333

5th voter c=g^m\*r^n mod n^2=357748

6th voter c =g^m\*r^n mod n^2=181876

7th voter c =g^m\*r^n mod n^2=5495102

8th voter c =g^m\*r^n mod n^2=5409140

9th voter c =g^m\*r^n mod n^2=125196

C=(3081436\*759231\*6642664\*3965333\*357748\*181876\*5495102\*5409140\*125196)mod 2701^2=1075754

• The Election Authority decrypts the encrypted resultant votes C = 509 as follows m = L(C^λ mod n^2).µ mod n = L(3973172)\*2556 mod 2701

=1471\*2556mod2701

=84=

We convert 84 into binary it becomes 01010100

So, Convert binary 0101 into decimal it becomes 5, Convert binary 0100into decimal it becomes 4.

# Q2. (Digital Signatures) [2+2+2+1 = 7 Marks]

Suppose Bob and Alice, two business partners, use their smart phones to communicate with each other regarding their business decisions. Hence, their messages are very sensitive and require to be authenticated. Otherwise, an attacker, say one of their business rivals, may perform phone number porting fraud attack. In this attack, the attacker may use another SIM card to port Alice or Bob’s phone number for pretending as Alice or Bob, respectively. Therefore, BoB and Alice uses digital signature scheme in their smart phone to sign messages for ensuring authenticity. The working procedure of the digital signature is illustrated in **FigureQ2.**



**Verification**

**Bob**

**Sender**

**)**

**(**

**Message**

**Signing with Bob’s**

**private**

**-**

**key**

**Signed Message**



**Verification with**

**Bob’s public**

**-**

**key**

**Verified**

**Message**



**Alice**

**(Receiver)**

Answer **Q2.1**, **Q2.2** and **Q2.3** using the scenario mentioned above.

## Q2.1

Suppose Bob (the sender) wants to send a message ***m***=**654321** to Alice (the receiver). However, before sending the message he would like sign the message. When Alice receives the signed message, she would like to verify that the message is indeed from Bob. To facilitate signing and verification Bob generates public and private keys using **RSA encryption algorithm** and sends the public key to Alice. Bob uses parameter ***p* = 9973** and ***q* = 7541**, and chooses a suitable public key parameter ***e=7321*.**  How would Bob sign message ***m*=654321**? How would Alice verify the signed message from Bob?

[***Hints:*** Refer to the lecture-6 and tutorial-6. You do not need to generate hash of the message ***m***.]

• Bob calculates n = p \* q = 9973\*7541 = 75206393

Bob Calculates Φ (n)=(p-1)\* (q-1) = (9972) \*(7540) = 75188880

Bob’s public key (n, e)=(75206393,7321)

d\*e=1 mod Φ (n) , e=7321,so d=12622201. This is our secret private key.

• Bob signs the message using private key d = 12622201 as follows:

s = m^d mod n =654321^12622201mod 75206393 = 15764124

• Bob sends (m, s) = (654321, 15764124) to Alice

• Alice verifies using public key= (75206393,7321)as follows: m’ = s^e mod n = 654321

## Q2.2

Suppose Bob (the sender) wants to send a message ***m***=**9876** to Alice (the receiver). However, before sending the message he would like sign the message. When Alice receives the signed message, she would like to verify that the message is indeed from Bob. To facilitate signing and verification Bob generates public and private keys using **ElGamal encryption algorithm** and sends the public key to Alice. Bob chooses ***p*=8369***,* ***g*=3031***,* ***x*=61.** How would Bob sign message ***m*=9876**? How would Alice verify the signed message from Bob?

[***Hints:*** Refer to the lecture-6 and tutorial-6. You do not need to generate hash of the message ***m***.]

•Bob generates a public key parameter y as follows: y = g^x mod p = 3031^61 mod 8369= 3400. So, Bob’s public key is p =8369, g =3031, y = 3400

•Bob sends public key (p = 8369, g = 3031, y = 3400) to Alice

•Bob selects a random number k while satisfying gcd(k, p 1) = 1. Let’s pick k = 9.

•Bob computes signature parameters as follows: r = g^k mod p = 3031^9 mod 8369

=4090

s = k^-1 (m- x\*r) mod (p -1) =7778.

Bob sends signed message (m = 9876, r = 4090, s = 7778)

•Verification (by receiver)

Alice computes verification parameters v and w as follows:

v = g^m mod p = 3031^9876mod 8369 = 6346

and w = y^r\*r^s mod p = 3400^4090\*4090^7778mod 8369 = 6346

Since v = w = 6346, signature is accepted by Alice.

## Q2.3

Suppose Bob (the sender) wants to send a large text message ***M*** to Alice (the receiver). You should download the text message file “**Message.txt**” from the CANVAS. The text message ***M*** is as follows:

***The invention of computers in the 20th century revolutionized cryptology. IBM Corporation created a code, Data Encryption Standard (DES) that has not been broken to this day. Thousands of complex codes and ciphers have been programmed into computers so that computers can algorithmically unscramble secret messages and encrypted files.***

Before sending the message, Bob generates a **hash *h(M)*** of the text message ***M*** using MD5 hash algorithm, and converts ***h(M)*** into integer message ***m***.Then, he signs the ***m*** and sends it to Alice. When Alice receives the signed message, she would like to verify that the message is indeed from Bob. To facilitate signing and verification Bob generates public and private keys using **RSA encryption algorithm** and sends the public key to Alice. Bob uses the following parameters: ***p* =** **330620674058481288123238031776544995151 *q* =** **273416090781657641961913344431645866621**

Bob chooses a suitable public key parameter ***e=37*.**  How would Bob sign message ***M***? How would Alice verify the signed message from Bob?

[***Hints:*** Refer to the “*Running Example of RSA Signature for Text Message”* of lecture6. The document can be found here: <https://rmit.instructure.com/courses/46189/files/3608593/download?wrap=1>**Use the following links:**

*For generating MD5 hash:* <http://www.miraclesalad.com/webtools/md5.php> *For converting hexadecimal to decimal:*

<https://www.rapidtables.com/convert/number/hex-to-decimal.html>]

•The original message is M=***The invention of computers in the 20th century revolutionized cryptology. IBM Corporation created a code, Data Encryption Standard (DES) that has not been broken to this day. Thousands of complex codes and ciphers have been programmed into computers so that computers can algorithmically unscramble secret messages and encrypted files.***

•The hash of the message h(M)=d66ebc48f4caf3f1f1b4d0e5277ae8a0.If we do hex to decimal conversion h(M)=285029762625603712791732860606117243040

•So, the message to be signed is 285029762625603712791732860606117243040

•Bob selects a random prime p=330620674058481288123238031776544995151

q=273416090781657641961913344431645866621

calculate n=p\*q=90397012232666561605229998826529078280036287731309230718015295597153637754771(public key)

•Bob calculates

φ(n)=(p-1)\*(q-1)= 90397012232666561605229998826529078279432250966469091787930144220945446893000

•Bob chooses a suitable public key parameter e=37,de=1modφ(n)

d=41533762377171122899700269731107954885144547741350663794454390588001962085973

•Bob signs the message using private key as follows

s=m^d mod n

=1005436109973011866628156397480407084517393977589283926001393886744969398659

•Bob sends M and the signature of the message s

•Alice verifies the signature using the public key of Bob

m’=s^e mod n

=285029762625603712791732860606117243040

Change to hexadecimal is d66ebc48f4caf3f1f1b4d0e5277ae8a0

•Alice finds h(M)=m=m’ , so Alice verifies the signed message from Bob and accept.

## Q2.4

Suppose, Bob (the sender) wants to send a message to Alice. Before sending the message, Bob generates a digital signature. Assume that Bob and Alice use both of the *public-key cryptography system* and *hash function* when computing digital signatures. Say, the *hash function* used to compute and verify signatures is **insecure**, but the *public-key cryptography system* is **secure**. Show that Trudy can forge signatures. [***Hints:*** Refer to the lecture-2 and tutorial-2]

# Q3 (BlockChain Technology) [15 Marks]

Write a **well-organized** **report** on how the blockchain technology can impact **Smart Industry systems**. Please consider the followings in your report sequentially:

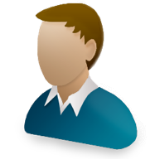
1. Explain a motivating scenario of a *smart industry* where the blockchain can be applied.
2. Explain your understanding with necessary diagrams on how the specified *smart industry* can be designed using blockchain.
3. Discuss some popular *consensus mechanisms* for blockchain. Among the popular consensus mechanisms, which one can be applied in your specified blockchain based *smart industry* and why? Justify your answer.
4. Explain how the **integrity** and **traceability** of *smart industry* data are obtained using blockchain in your specified scenario.
5. What are the advantages and disadvantages of using blockchain technology in your specified *smart industry*?

[***Hints:*** Resources on existing blockchain based smart industry should be reviewed from online sources. One of the sample projects can be found in the following link:

<https://ico.productivist.com/en/>]

# Q4 (Authentication Protocol) [3 Marks]

The following mutual authentication protocol is proposed based on a symmetric-key cryptography algorithm. We assume that the cryptography algorithm that is used here is secure. Given that the following protocol does not provide mutual authentication. Give **two different attack scenarios** where Trudy can convince Bob that she is Alice. Briefly explain each attack scenario performed by **Trudy**with proper diagram which on the protocol.



**Alice**

**Bob**

**“Alice”, R**

**E(**

**R**

**,**

**K**

**AB**

**)**

**E(**

**R**

**, K**

**+1**

**AB**

**)**

**[*Hints*:** You need to show **two** attack scenarios performed by **Trudy**with proper diagram on the protocol. Additionally, provide brief explanation of attacks to justify your answer. Refer to attack scenarios on mutual authentication protocols that were discussed during the Lecture-7 and Tutorial-7.**]**

# Q5 (OpenSSL and IPFS) [5 Marks]

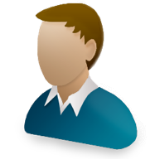
Assume that the School of Scienceof RMIT University is planning to use IPFS-based repository of sensitive files for sharing among staffs. An owner of a particular file, say **Alice** wants to share the file to her supervisor, say ***Bob***.Therefore, **Alice** encrypts the file with Alice and Bob’s shared ***AES*** ***secret key*** ***(KAB)*** using **OpenSSL**, uploads the encrypted file in the **IPFS-based repository**, and receives a ***Unique Hash Identifier*** ***(UHI)***. Next, **Alice** encrypts  ***KAB*** and ***UHI*** with Bob’s **RSA public key** using **OpenSSL** and gets ciphertext ***C.*** Further, Alice generates a *message digest* ***M*** using **OpenSSL** with her **RSA private key** for the ciphertext ***C.***Finally, Alice sends ***{M, C}*** to Bob through email.

Upon receiving them, Bob verifies the *message digest* ***M*** using **OpenSSL** with ***Alice’s* *RSA* *public-key***. If the verification fails, Bob discards further steps and informs Alice that the verification has been failed. If the verification is successful, Bob decrypts ***C*** using **OpenSSL** with his ***RSA private-key***. If the decryption is successful, Bob retrieves the shared ***AES*** ***secret key*** ***(KAB)*** and ***unique hash identifier*** ***(UHI)*** for the file.

Bob downloads the file from IPFS-based repository with the file’s ***UHI***. Bob uses IPFS commands to download the file. Upon receiving the file from IPFS network, Bob decrypts the encrypted AES secret key. Finally, Bob decrypts the download file from IPFS network using the shared ***AES*** ***secret key*** ***(KAB)***. The scenario is illustrated in the **Figure-Q5** below. Show the required OpenSSL and IPFS commands sequentially for each step stated above**. Please provide screenshot of the outcome for each command.**

* Stores files and provides a ***unique***  ***hash ID (UHI)*** for each file.
* Sends the ***unique hash ID (UHI)***to

Alice





Returns a file to the requestor

here, Bob) based on received

(

***U***

***H***

***I***

.

**Alice**

**Owner**

**)**

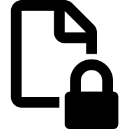
**(**

**Bob**

**(**

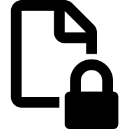
**User**

**)**



**Encrypted**

**File**



{

***M,***

***C***

**}**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | * Encrypts a file using ***AES*** ***secret key*** ***(KAB)*** * Sends encrypted file to IPFS for storing and collects ***unique hash ID (UHI)***of the   file   * Encrypts ***KAB*** and ***UHI***with Bob’s ***RSA public-key*** , and gets ciphertext ***C*** * Generates a digest ***M*** of ***C*** * Sends {***M,*** ***C*}** to Bob | | |  | | --- | | * Verifies signed digest M using Alice’s ***RSA public-key.*** * If verified, decrypts ***C*** to get ***AES*** ***secret key*** ***(KAB)*** using his ***RSA private-key*** * Downloads encrypted file from IPFS using its ***unique hash ID (UHI)*** * Decrypt encrypted file using ***AES*** ***secret key*** ***(KAB)*** | |

## Figure Q5: IPFS based encrypted file sharing