Robert Gunawan's Algorithm

The New Secure Data Sharing Algorithm

Robert Gunawan - 13508038
Informatics Engineering
School of Electrics and Informatics
Bandung Institute of Technology, Ganeca Street 10, Bandung 40132, Indonesia
If18038@students.if.itb.ac.id ; gabriel_robert_gunawan@yahoo.com

The Asymmetric Cryptography algorithm is the one of the invention in cryptography that can be used to encrypt and decrypt with different encrypt-decrypt key. One of the asymmetric cryptography algorithm is Public key algorithm. The principal is creating 2 keys, public and private key that correspondence each other and do the encryption and decryption with one of the keys. Public key is the key that can be published in society, and private key is the key that must be kept personally.

There are several instance of Public key algorithm created nowadays. But many of the algorithm use the same number of public and private keys, used to do the encryption and decryption process. This can't be used as data sharing encryption because to unlock an encrypted data, we must use the same number key that used to encrypt the data and that means all the people must be unlocking in the same time. Using Shamir secret sharing technique, the author, Robert Gunawan, create the asymmetric cryptography algorithm for data sharing

Keywords: asymmetric cryptography algorithm, encrypt, encryption, decrypt, decryption, public key, private key, public key algorithm, new data sharing encryption algorithm, Robert Gunawan's Algorithm

I. BACKGROUND

The rapid development in informatics leads us to a new world digital era. Right now, people can do many things in computer and internet for example buying, selling, browsing, sharing, etc. The one aspect that affects people in doing many things in the computer is security. Security helps people secure what they need to share in the computer and internet.

For a long times, people develop the ways to secure data, called cryptography algorithm. There are many algorithms in the world and we can classify them into 2 categories, the symmetric algorithm and asymmetric algorithm.

The symmetric algorithm uses the same key for encryption and decryption. The key must be store in a secure place in order to prevent the "jail breaking" process. The primacy of this algorithm is the speed, because the symmetric algorithm can do the encryption and decryption in a relatively rapid speed. The weakness is the key that used to do the process is the same, so whenever somebody has the key, somebody has the data. This algorithm usually used to encrypt the common data, and the size is relatively big.

The asymmetric algorithm is the next invention in the cryptography algorithm. The asymmetric algorithm use more than 1 key, usually use 2 keys to do the encryption and decryption process (called public and private key). The primacy of this algorithm is the security provided by using different keys to encrypt and decrypt the data. The weakness is this process takes a long time to complete a single data processing. This algorithm usually used for securing a key used for the symmetric algorithm because of the times taken (The size of the key used in symmetric algorithm is relatively small).

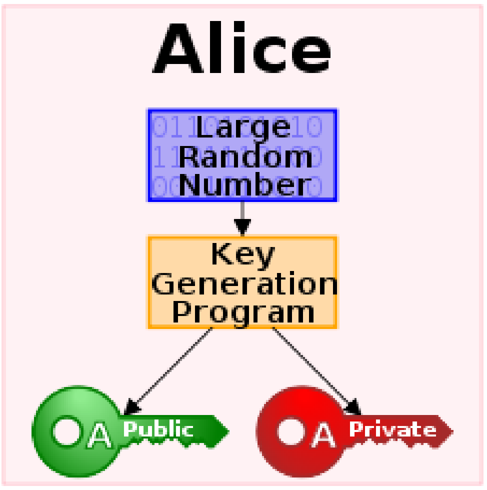
Robert Gunawan has learned this 2 kind of algorithm and has a new invention of combining this 2 algorithms into a single algorithm. Using of the asymmetric algorithm to secure the key, the symmetric algorithm to do the encrypt-decrypt process, and Shamir secret sharing technique, people can have a new data sharing algorithm. This algorithm created because people need to share data without sharing the key that used especially in decryption process (called private key). The main idea is using more than 1 public key for the encryption and using the amount defined of each correspondence private key to do the decryption.

II. THEORY

Public Key Algorithm

The distinguishing technique used in public key algorithm is the use of asymmetric key algorithms, where the key used to encrypt a message is not the same as the key used to decrypt it. Each user has a pair of cryptographic keys—a public encryption key and aprivate decryption key. The publicly available encrypting-key is widely distributed, while the private decrypting-key is known only to the recipient. Messages are encrypted with the recipient's public key and can only be decrypted with the corresponding private key. The keys are related mathematically, but parameters are chosen so that determining the private key from the public key is prohibitively expensive. The discovery of algorithms that could produce public/private key pairs revolutionized the practice of cryptography beginning in the middle 1970s.

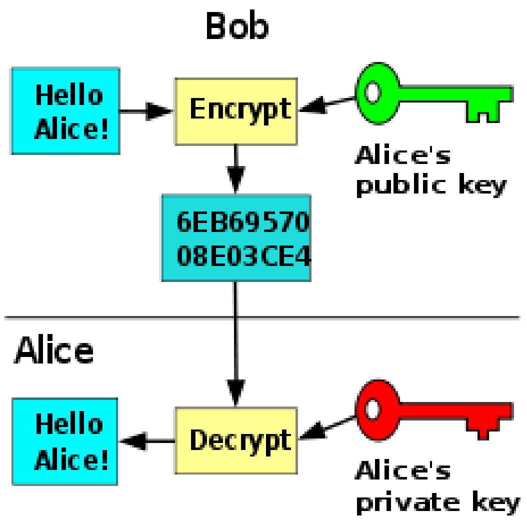
The illustration of key generator can be seen in the picture below



Picture 1. The Key Generator

The process above creates 2 keys from a large random number. Usually, the key used for asymmetric algorithm is created with randomness for confusing the "jail-breaker".

The illustration of encrypt-decrypt process using asymmetric algorithm can be seen in picture below



Picture 2. The Asymmetric Encypt-Decrypt Process

The illustration above shows 2 persons, Alice and Bob do the asymmetric algorithm. Bob wants to give Alice some greetings and decide to give it in encrypted ways. Bob use the Alice's public key to encrypt the data and send the encrypted data (shown as numbers) to Alice. Alice can't read the message she received from Bob, so Alice use her private key to decrypt the Bob's message. After that Alice can see Bob's greeting.

Shamir Secret Sharing Technique

Shamir's Secret Sharing is an algorithm in cryptography developed by Adi Shamir. It is a form of secret sharing, where a secret is divided into parts, giving each participant its own unique part, where some of the parts or all of them are needed in order to reconstruct the secret.

Counting on all participants to combine together the secret might be impractical, and therefore we sometimes use the threshold scheme where any k of the parts are sufficient to reconstruct the original secret.

Formally, our goal is to divide some data *D* (e.g., the safe combination) into n pieces D1,…,Dn in such a way that:

1. Knowledge of any k or more D pieces makes D easily computable.
2. Knowledge of any k-1 or fewer Di pieces leaves D completely undetermined (in the sense that all its possible values are equally likely).

This scheme is called (k,n) threshold scheme. If k=n then all participants are required to reconstruct the secret.

The essential idea of Adi Shamir's threshold scheme is that 2 points are sufficient to define a line, 3 points are sufficient to define a parabola, 4 points to define a cubic curve and so forth. That is, it takes k points to define a polynomial of degree k-1.

Suppose we want to use (k,n) threshold scheme to share our secret S, without loss of generality assumed to be an element in a finite field F.

Choose at random k-1 coefficients $a_1,\ldots,a_{k-1}$ in F, and let a_0 = S. Build the polynomial .

Let us construct any n points out of it, for instance set I = 1,…,n to retrieve (i,f(i)). Every participant is given a point (a pair of input to the polynomial and output). Given any subset of k of these pairs, we can find the coefficients of the polynomial using interpolation and the secret is the constant term a_0.

Hash Function

A hash function is any well-defined procedure or mathematical function that converts a large, possibly variable-sized amount of data into a small datum, usually a single integer that may serve as an index to an array (cf. associative array). The values returned by a hash function are called hash values, hash codes, hash sums, checksums or simply hashes.

Hash functions are mostly used to accelerate table lookup or data comparison tasks—such as finding items in a database, detecting duplicated or similar records in a large file, finding similar stretches in DNA sequences, and so on.

A hash function may map two or more keys to the same hash value. In many applications, it is desirable to minimize the occurrence of such collisions, which means that the hash function must map the keys to the hash values as evenly as possible. Depending on the application, other properties may be required as well. Although the idea was conceived in the 1950s, the design of good hash functions is still a topic of active research.

Hash functions are related to (and often confused with) checksums, check digits, fingerprints, randomization functions, error correcting codes, and cryptographic hash functions. Although these concepts overlap to some extent, each has its own uses and requirements and is designed and optimized differently. The Hash Keeper database maintained by the American National Drug Intelligence Center, for instance, is more aptly described as a catalog of file fingerprints than of hash values.

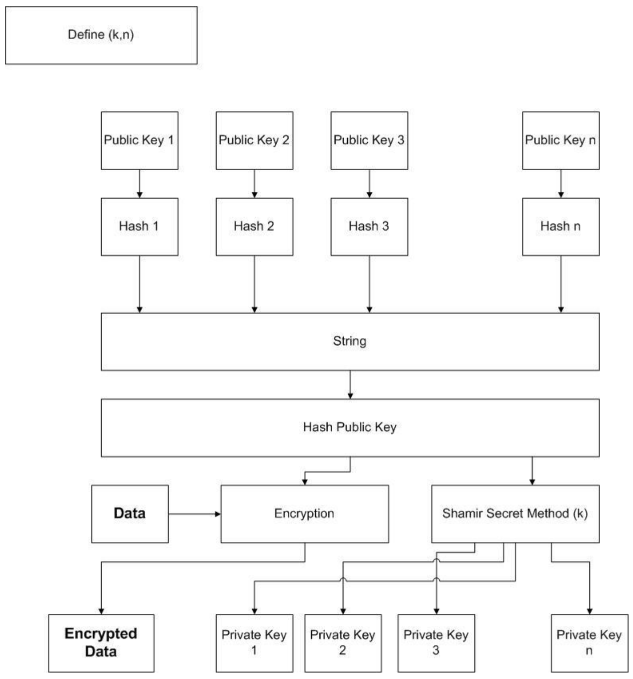
III. The Main Protocol

There are 2 protocol here, the encryption and decryption protocol.

The protocol to do Robert Gunawan's Algorithm's Encryption described as:

1. Define how many persons involved as n and how many person needed to reconstruct the data as k. n is greater or equal as k and the min k is 1.
2. Define the data that must be secured with Robert Gunawan's Algorithm.
3. For each person involved, gather the person's public key. The public key can be anything, from a single word, sentence, or another data that can be converted into a string.
4. Do hash function for each public key gathered, and append it each other in a new string.
5. Do hash function for the new string that contains appended hash from public key called hash public key.
6. Encrypt the data using symmetric algorithm with hash public key.
7. For the same number of person involved, create the private key for each person using Shamir Secret Sharing Technique (k,n) and hash public key. Give it back to each person.
8. Give the encrypted data back to all persons.

The encryption schema can be seen in picture aside.

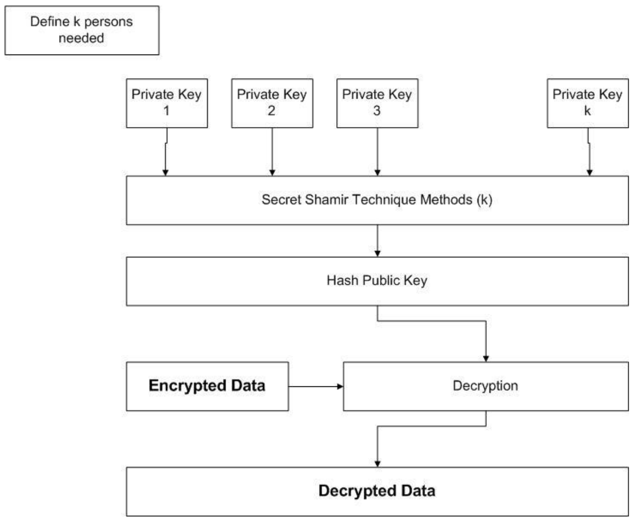


Picture 3. Encryption Schema

The protocol to do Robert Gunawan's Algorithm's Decryption described as:

1. Define k persons needed to decrypt the data.
2. Define the private key from k persons needed to unlock the encrypted data as public keys.
3. Do the reverse Shamir Secret Sharing Technique to get the a_0 (The hash public key that encrypted the data).
4. Decrypt the data using the hash public key.
5. Give the decrypted data back to the persons that decrypt the data.

The decryption schema can be seen in picture below.



Picture 4. Decryption Schema

IV. The Main Algorithm

General Algorithm

Hashing Algorithm

```
Function Hash (string: input) : string
Local Variable

Algorithm

/* Do hash algorithm to hash input, the algorithm used to hash input can be any hash function algorithm*/
/*return hashed input*/
```

In this function, programmer can choose any hash function algorithm to be implemented. The function is simple, to hash an input string.

Appended String Algorithm

```
Function AppendString (array of string: input) : string
Local Variable

integer: i //for iteration
string: result //result
Algorithm

for i=1 to input.size do
    result = result + Hash(input[i])
endfor
return result
```

In this function, programmer must iterate the array string input from beginning to the end, hash each of them and append it into a temporary variable called result. After the process finished, return the result

Encryption

Shamir Secret Technique Method (Encryption)

```
Function ShamirEncrypt (string: input, integer: k, integer n) : array of  Point(X,Y)
Local Variable

integer:  i //for iteration
array of  integer: Variable
array of Point: result
Algorithm

//first insert integer value of input into the first element of Variable
Variable[1] = ToInteger(input) //convert input to integer

//fill all other variable with random number
for i=2 to k do
    Variable[i] = Random(0,infinite) //do the random function that generates an integer from 0 to infinite
endfor

//the key creation
for i=1 to k do
    for j=2 to k do
        result[i].Y = result[i].Y + Pangkat(i,j-1)*Variable[j]
    endfor
    result[i].X = i
endfor
    return result
```

In this function, programmer must use the Shamir Secret Technique to divide a string input to a several X and Y value in Point that called Private key.

Encryption Algorithm

```
Function Encryption (string input, string key): string
Local Variable


Algorithm

/*Encrypt the input with key, programmer can use any symmetric algorithm to encrypt data*/
/*return encrypted input*/
```

In this function, programmer must decide what symmetric algorithm used to encrypt the input with the key. The more secure algorithm selected, the more secure the data.

Robert Gunawan's Encryption Algorithm

```
Procedure RGAEncryption (void)
Local Variable

integer: i //for iteration
integer: k
integer: n
string: input
array of string: public_key
array of Point: P
string Hash_Public_Key
string result
Algoritma

read(k) //read how many people needed to unlock the data
read(n) //read max people
read(input) //read the input data

//read each people involved's public key
for i=1 to n do
    read(public_key[i])
endfor

//append and hash the key
Hash_Public_Key = AppendString(public_key)
```

```
//do the Shamir secret sharing technique, return P[i] to each people involved
P = ShamirEncrypt(Hash_Public_Key, k, n)

//Return the result
result = Encryption(input, Hash_Public_Key)
```

This is the core of Robert Gunawan's Encryption Algorithm

Decryption

Shamir Secret Technique Method (Decryption)

```
Function ShamirDecrypt (Array of Point: P): string
Local Variable

integer: i //for iteration
array of array of integer: Variable
integer result = 0
Algorithm

/*get the Langrange Polynomial from each X value in P and insert it in Variable*/
for i=1 to P.Max do
        /*do the langrange function*/
        for j=1 to P.Max do
                Variable[i][j] = Variable that langrange produce number j
        endfor
endfor

//for each Variable[i] compute the last constant (the constant in front of x^0) with each Y in P
for i=1 to P.Max do
        result = result + P[i].Y * Variable[i][P.Max]
endfor

//return the string value of  result
return StringValueOf(result)
```

This function is the reverse algorithm from Shamir Secret Technique Encryption. This function used for getting the Hashed Public key for decryption process. The Langrange used to create the variable is the Langrange function, function that create a line equation from some points located in Cartesian diagram.

The Example for Langrange function described below

Consider

$$(x_0, y_0) = (2, 1942); (x_1, y_1) = (4, 3402); (x_2, y_2) = (5, 4414)$$

The Langrange function generated is

$$\ell_0 = \frac{x - x_1}{x_0 - x_1} \cdot \frac{x - x_2}{x_0 - x_2} = \frac{x-4}{2-4} \cdot \frac{x-5}{2-5} = \frac{1}{6}x^2 - 1\frac{1}{2}x + 3\frac{1}{3}$$

$$\ell_1 = \frac{x - x_0}{x_1 - x_0} \cdot \frac{x - x_2}{x_1 - x_2} = \frac{x-2}{4-2} \cdot \frac{x-5}{4-5} = -\frac{1}{2}x^2 + 3\frac{1}{2}x - 5$$

$$\ell_2 = \frac{x - x_0}{x_2 - x_0} \cdot \frac{x - x_1}{x_2 - x_1} = \frac{x-2}{5-2} \cdot \frac{x-4}{5-4} = \frac{1}{3}x^2 - 2x + 2\frac{2}{3}$$

Decryption Algorithm

```
Function Decription (string input, string key): string
Local Variable


Algorithm

/*Decrypt the input with key, programmer can use any symmetric algorithm to decrypt data*/
/*return decrypted input*/
```

In this function, programmer must decide what symmetric algorithm used to decrypt the input with the key. The algorithm selected must be the same with the one to encrypt the input

Robert Gunawan's Decription Algorithm

```
Procedure Decription (void)
Local Variable

integer: i
integer: k
string: input
array of Point: private_key
string: key
string: result

read(input) //read the encrypted data
read(k) //read the max private key

//read private key
for i=1 to k do
        read(private_key[i])
endfor

//get the real key with Shamir secret sharing technique
key = ShamirDecrypt(private_key)

//return the result
result = Decrypt(input, key)
```

This is the core of Robert Gunawan's Decryption Algorithm

V. Analysis

Comparation With Usual Symmetric Algorithm

No	Usual Symmetric Algorithm	Robert Gunawan's Algorithm
1	Less secure	More secure

2	Use the same key to do encryption and decryption	Use different key for encryption and decryption
3	Number of keys needed for encryption and decryption are same	Number of keys needed for encryption and decryption can be different
4	Time for encryption and decryption is relatively same	Time for encryption and decryption is not same because the decryption process take more time to construct Langrange function
5	-	Can be used as secure data sharing algorithm

From the security aspect, Robert Gunawan's Algorithm relatively more secure because the key is generated by the algorithm from each public keys user have entered.

From the key aspect, Robert Gunawan's Algorithm relatively better because the algorithm can use the different number of keys for each decryption and encryption and also different keys.

From the time aspect, we can't judge which is better, but the time taken in Robert Gunawan's Encryption and Decryption is different.

Comparation With Usual Asymmetric Algorithm

No	Usual Asymmetric Algorithm	Robert Gunawan's Algorithm
1	The ammount of data is relatively smaller	The amount of data is relatively larger
2	The time taken is relatively longer	The time taken is relatively smaller
3	Number of keys needed for encryption and decryption are same	Number of keys needed for encryption and decryption can be different
4	Time for encryption and decryption is relatively same	Time for encryption and decryption is not same because the decryption process take more time to construct Langrange function
5	-	Can be used as secure data sharing algorithm

From the comparation, Robert Gunawan's Algorithm is better than usual asymmetric algorithm because the data is larger, time taken is smaller, and the other else described in Comparation with Symmetric Algorithm above

Process Analysis

The Process to make many public key into single key is relatively secure because it takes n+1 hash function in order to have real key for encryption. The process can't create the same key from each process because it implements n+1 hash function.

From the key aspect, Robert Gunawan's Algorithm produce equal amount of public and private key. In encryption, all key is needed to complete the process, but in decryption the amount of private key needed may not same as the amount of private key needed. That's because of Shamir Secret Sharing Technique implementation.

From the algorithm in encryption, Robert Gunawan's Algorithm use 1 symmetric algorithm to encrypt all data with generated key with n+1 (n means number of people involved) hash function. The algorithm is relatively secure because use n+2 encryption and from hash function, we can get a unique value to encrypt something.

VI. CONCLUSION

Robert Gunawan's Algorithm is a new secure data sharing algorithm. It use n+2 encryption with principal of asymmetric algorithm, using public key algorithm.

For the innovation, the author suggest to retest the key produce protocol because it has not been tested in real time.

VII. ACKNOWLEDGEMENT

Thanks to Jesus Christ, my Saviour; my mother, Maria Wika Artanti for the support, DR. Ir. Rinaldi Munir for the cryptographic lesson, and my friends, who help and support this paper.

VIII. REFERENCES

1. http://www.informatika.org/~rinaldi/Kriptografi/kriptografi.htm (accessed 6th May 2011)
2. www.wikipedia.org (search word: public key algorithm, Shamir secret sharing, hash function) (accessed 6th May 2011)

IX. STATEMENT

I hereby declare that I have written this paper are my own writing, not adaptation, or translation of the papers of others, and not plagiarism.

Bandung, 9th May 2011

Robert Gunawan