Spring 2018

THE RSA CRYPTOSYSTEM

Rodrigo Iglesias
ri16@zips.uakron.edu

THE RSA CRYPTOSYSTEM

RODRIGO IGLESIAS

25th of April 2018

RESEARCH REPORT

TABLE OF CONTENTS

LIST OF TABLES & FIGURES

INTRODUCTION

This paper intends to present an overview of the RSA cryptosystem. Cryptosystems are mathematical algorithms that disguise information so that only the people for whom the information is intended can read it. The invention of the RSA cryptosystem in 1977 was a significant event in the history of cryptosystems. We will describe in detail how the RSA cryptosystem works and then illustrate the process with a realistic example using fictional characters. In addition, we will discuss how cryptosystems worked prior to the invention of RSA and the advantage of using RSA over any of the previous cryptosystems. This will help us understand the significance of the invention of the RSA in the world of security. We will also explain how RSA was created and who the inventors are.

The RSA algorithm has become the standard cryptosystem for industrial-strength encryption, especially for data sent over the Internet. At the present time, RSA is one of the most convenient, widely used and strongest cryptosystems available.

CRYPTOSYSTEMS

In order to understand how the RSA cryptosystem works, first it is necessary to understand how cryptosystems work. First of all, what is a cryptosystem? Suppose that a certain person wishes to send a message to another person, but does not want anyone else other than this second person to know the information in the message. A cryptosystem is a method which puts a disguise on the message, so that only the second person is able to remove the disguise and therefore read the message. The message that does not have a disguise on it is called the plaintext, and the disguised form of the message is called the ciphertext. The process in which the plaintext is converted into the ciphertext is called the encryption process. Likewise, the process in which the ciphertext is turned back to plaintext is called the decryption process.

For a better understanding we will use the following fictional characters: Alice, Bob, and Eve. Now imagine the situation where Bob desires to send a message to Alice, but he doesn't want anyone other than Alice to read the message. Like in the real world there may be someone interested in eavesdropping and reading the content of the message. Let's call this person Eve. Bob decides to use a certain cryptosystem which will put a disguise on his message. Thereby, even if Eve intercepts Bob's message she should not be able to read it. As we just said, the encryption is the process in which a disguise is put on the plaintext converting it into ciphertext. Now, imagine the situation where more people start to use the same cryptosystem. But, what happens if Bob and all these people go through the exact same encryption process every time one of them needs to send a message? If all of them do that, all the messages they send will be disguised the same way. Therefore, if Eve manages to intercept the messages and somehow works out a way to reverse that particular process of

encryption, then she will be able to read the content of every message encrypted with that particular cryptosystem. In other words, if someone finds out how to reverse the encryption process, then the corresponding cryptosystem is no longer helpful. Hence, in order to avoid this hypothetical case, the process of encrypting a message involves the use of a piece of specific data (generally a number), called the encryption key, which will produce different forms of disguised messages when varied. Therefore, if Bob and someone else happen to use the same cryptosystem they will input different encryption keys in their encryption process and thus generate different disguised forms for their messages. Likewise, a piece of specific information, called the decryption key, is also needed in the process of decryption, this key will be input in the corresponding decryption process to remove the disguise created with a certain encryption key. Therefore, Bob will use an encryption key during the process of encryption, and Alice will use the corresponding decryption key during the decryption process. Note that Alice can never let Eve know what her decryption key is. Because if she does so, Eve will only have to intercept Bob's messages to be able to read them.

MATHEMATICAL BACKGROUND AND DEFINITIONS

We will introduce some mathematical definitions and formulas that will help us understand the RSA encryption and decryption processes.

We are going to begin with a basic definition: Assume that d and n are two positive integers. What does it mean to say that d is a divisor of n? It means that there exists an integer k such that n = (d)(k). For instance, 5 is a divisor of 35 because there exists an integer k such that 35 = (5)(k), namely k = 7. Indeed, 1, 5, 7, and 35 are all the divisors of 35.

Using the previous definition, we are going to define the greatest common divisor of a pair of integers a and b. The greatest common divisor of a and b, denoted g.c.d. (a, b), is the largest positive integer that divides both a and b. For example the greatest common divisor of 35 and 45 is 5, One way to see that is by listing all possible divisors of each number. The divisors of 35 are 1, 5, 7, and 35 (as we saw before), and the divisors of 45 are 1, 3, 5, 9, 15, and 45. Note that 1 and 5 are the only two common divisors, but 5 > 1; thus, g.c.d. (35,45) = 5. Nonetheless, RSA deals with much larger numbers; therefore, finding all the divisors of every pair of integers is not very efficient. RSA uses the Euclidean Algorithm to find the greatest common divisor of two numbers, which is much more efficient.

A positive integer greater than 1 is said to be prime if its only positive divisors are 1 and itself. For example 31 is a prime number, but 35 is not.

A pair of integers a and b are said to be relatively prime if the greatest common divisor of a and b is 1. For instance, the only divisors of 33 are 1, 3, 11, and 33; thus, 33 and 35 are relatively prime since g.c.d. (33, 35) = 1.

If n is a positive integer, then the totient of n, denoted $\phi(n)$, is the number of integers m such that $1 \leq m \leq n$, and such that m and n are relatively prime. There is a well-known formula, that we shall not include in this paper, to calculate the totient $\phi(n)$ of any positive integer n. For our purposes we only need to know that if $n = (p)(q)$, for distinct primes p and q, then $\phi(n) = (p - 1)(q - 1)$.

Later in section How RSA Works, we are going to illustrate the Extended Euclidean Algorithm, which computes integers x and y such that: $ax + by = g.c.d.(a, b)$, for a given pair of integers a and b. This algorithm is very quick and efficient. It can be done within seconds with a computer even when the integers a and b have many digits.

Given two positive integers n and k, the Division Algorithm produces a pair of positive integers q and r that are unique such that $n = (k)(q) + r$, for $0 \leq r < k$. The integer q is the quotient when n is divided by k, and the integer r is the remainder.

Lastly, let us define modular arithmetic. Modular arithmetic is a system of arithmetic for integers, which considers the remainder. In other words, given two positive integers N and x, such that $N > 0$, the number x mod N is the equivalent of asking for the remainder when x is divided by N.

Two integers a and b are said to be congruent modulo N if a and b have the same remainder when they are divided by N. In such a case, we say that $a \equiv b \pmod{N}$. Indeed, we use modular arithmetic very often in a regular day, specially when we talk about time. Let us say that it is 11 o'clock and that a basketball game starts in 7 hours. Then, we would say that the game starts at 6 o'clock, not at 18 o'clock. We mentally calculate 11+7 in (mod 12). $11 + 7 = 18 \equiv 6 \pmod{12}$.

Given the integers n and m, such that m is greater than zero and n is relatively prime to m, a modular multiplicative inverse of n with respect to modulus m is an integer a, denoted n^{-1}, such that: $n * a \equiv 1 \pmod{m}$.

For instance, a multiplicative inverse of 5 modulo 11 is 9 because (5)(9) = 45, and 45 = (4)(11) + 1 ≡ 1 (mod 11).

Allow us to also mention Euler's Totient Theorem now that we know about modular arithmetic, totient functions and greatest common divisor. Euler's Totient Theorem is a generalization of Fermat's Little Theorem. It states that if an integer n is relatively prime to an integer a, then a^ϕ(n) ≡ 1 (mod n).

HOW RSA WORKS

In this section we are going to describe in detail and illustrate the processes of encryption and decryption of the RSA cryptosystem.

Suppose that we have a community of people and we would like every member to have a convenient way to send a message to any other member in such a way that only the intended recipient should be able to read the content of the message. The RSA cryptosystem offers us a way to achieve that. Thus, our community is going to use the RSA cryptosystem. The first step for setting up the RSA cryptosystem is that each person in the community creates his/her own encryption key and decryption key. The encryption key is created by selecting a pair of distinct large primes and multiplying them together to get a number. Let us call this number n; thus, $n = (p)(q)$, for p and q primes. Since the two prime factors of n are known, it is easy to calculate the totient $\phi(n)$. As we explained before if $n = (p)(q)$, then $\phi(n) = (p-1)(q-1)$. After that, each person picks a prime e which satisfies $g.c.d.(\phi(n), e) = 1$. Then, the encryption key (n, e) is created. It consists of a large integer n, and an exponent, e.

In RSA, the encryption key is also known as the public key because it is made public. Imagine that Alice already created her own encryption key, then she made it public. Thereby, if Bob wants to send a message to Alice, he can use Alice's encryption key to put a disguise on the message that he is sending to Alice since he (like everyone else) has access to Alice's encryption key. Anyone who wishes to send a message to Alice (Bob in this case) will use Alice's encryption key to encrypt his/her message.

After the encryption keys have been created, each member still needs to find his/her corresponding decryption key, which as we stated before, can be used to remove the disguise from messages that have been encrypted using the corresponding encryption key. In

RSA, the decryption key is also known as the private key because it has to be kept private or secret. The decryption key is created by finding the private exponent d, which is calculated using the Extended Euclidean Algorithm to find the multiplicative inverse of e with respect to $\phi(n) = (p-1)(q-1)$. Thus, the decryption key is mathematically related to the encryption key, but it is still different.

Therefore, once everyone in the community has generated his/her own encryption key and decryption key, all members in the community publish their encryption keys. Thereby, from now on if any member from the community wishes to send a message to any other member, he or she can put a disguise on the message using the corresponding encryption key since every member's encryption key is now available.

Now, suppose that Bob is in the community and wishes to send a message to Alice who is also in the community. Just like before, Eve will be a person interested in reading the message that Bob is sending to Alice. Like any other member in that community, Alice and Bob have each already generated their own encryption and decryption keys, and published the encryption keys. Then, Bob obtains Alice's encryption key which is available to everyone (including Eve), encrypts his plaintext using that key, and sends the resulting ciphertext to Alice. Note that after Bob encrypts his message using Alice's encryption key, neither he, nor Eve, nor anyone would be able to reverse the process and decrypt the message using that encryption key. Only Alice, who has the corresponding (secret) decryption key, is going to be able to decrypt the message that she received from Bob.

Thus, although Eve has access to see Alice's encryption key and even in case that she could somehow intercept the ciphertext message that is being sent from Bob to Alice, she

still will not be able to read Bob's message because it has a disguise on and she does not have enough information to remove it.

We now present an example to illustrate how RSA is used. We will use "very small" prime numbers so it is easier to understand. Remember that Alice, like any other member of the community, has already generated and published her encryption key. Let us say that she picked the following primes for her encryption key: p = 61 and q = 53; hence, n = (p)(q) = 3233. Then, Alice calculated the totient namely $\phi(n) = (p-1)\ (q-1) = (60)\ (52) = 3120$. Thus, $\underline{\phi(n) = 3120}$.

After that, Alice must choose a positive integer e such that g.c.d. (3120, e) =1. Suppose that then Alice chose $\underline{e = 7}$.

Alice then calculates the multiplicative inverse of e modulo $\phi(n)$, which is denoted by d. This means that d is an integer such that $(e)(d) \equiv 1 \pmod{\phi(n)}$. Note that e = 7 and $\phi(n)$ = 3120. In order to find d Alice uses the Extended Euclidean Algorithm:

The first step of the Extended Euclidean Algorithm is to use the Division Algorithm with $n_1 = \phi(n)$ and $k_1 = e$ to produce the unique integer values q_1 and r_1, such that:

$3120 = (7)(q_1) + r_1$, for $0 \leq r_1 < 7$.

By dividing 3120 by 7 we get the quotient, $q_1 = 445$, with a remainder, $r_1 = 5$. Thus, we obtain the equation:

$$3120 = (7\ )(445) + 5.$$

We replicate the process; however, now our $n_2 = k_1 = 7$ and $k_2 = r_1 = 5$. We get the quotient, $q_2 = 1$, with a remainder, $r_2 = 2$. Thus, we obtain the equation:

$$7 = (5)(1) + 2.$$

We replicate the process over and over by substituting n_i for k_{i-1} and k_i for r_{i-1} every time until we get remainder $r_n = 1$. In this case we only need another division to obtain $r_3 = 1$. Since,

$$5 = (2)(2) + 1.$$

Now we are going to used the previous centered equations to find two integers x and y such that $1 = (3120)(x) + (7)(y)$. Note that: $1 = (3120)(x) + (7)(y) \equiv (7)(y) \pmod{3120}$. Thus, y is a multiplicative inverse of 7 with respect to modulus 3120, which is what we need for our decryption exponent d.

Therefore, from the third centered equation, we have that: $5 = (2)(2) + 1$.

We may rewrite this equation as: $1 = 5 – (2)(2)$. (1)

From the second centered equation, we have that: $7 = (5)(1) + 2$.

We may rewrite the previous equation as: $2 = 7 – (5)(1)$. (2)

By using equation (2) to substitute for 2 in equation (1) we obtain: $1 = 5 – (2)(7 – (5)(1))$.

We may rewrite the previous equation as: $1 = (-2)(7) + (3)(5)$. (3)

Also, from the first centered equation, we have that: $3120 = (7)(445) + 5$.

We may rewrite the previous equation as: $5 = 3120 – (7)(445)$. (4)

By using equation (4) to substitute for 5 in equation (3) we obtain: $1 = (-2)(7) + (3)(3120 – (7)(445))$.

We may rewrite the previous equation as: $1 = (3)(3120) – ((445)(3) + 2)(7) = (3)(\underline{3120}) + (– 1337)(\underline{7})$.

Thus, by the definition of modular arithmetic we have that: $1 = (3)(3120) + (-1337)(7) \equiv (-1337)(7) \pmod{3120}$.

Thus, y = -1337 (mod 3120). To make it easier to understand we are going to avoid using a negative value for our exponent d; thus, we pick a positive multiplicative inverse for Alice's decryption exponent d. Recall that, d ≡ y (mod 3120).

Thus, since -1337 ≡ – 1337 + 3120 = 1783 (mod 3120), we obtain d = 1783.

We can check if d = 1783 works for Alice 's decryption exponent d by computing (d) (e) (mod n): (1783)(7) = 12481 = 1 + (3120)(4) ≡ 1 (mod 3120).

Thus, d = 1783 works.

Then Alice, who has already published her encryption key (n, e) = (3233, 7), needs to keep her decryption key (n, d) = (3233, 1783) secret. Note that even though Alice publishes n, she also needs to keep the prime factors p and q secret. Because if Eve finds what prime factors Alice used to obtain n, she will be able to calculate $\phi(n)$ and thus reproduce the process we just did to calculate the decryption exponent d. That will enable her to remove the disguise from the messages sent to Alice.

Let us go back to the situation where Bob wanted to send a message to Alice. Then, Bob obtains Alice's encryption key (n, e) which is (3233, 7). Let us say that Bob's message is "HI".

When using a cryptosystem to put a disguise on a message, all characters from the message must be turned into numbers using a substitution table. (This conversion is generally done with the ASCII table, which stands for American Standard Code for Information Interchange). This is because the encryption and decryption algorithms only work on numbers.

To make it simple, we will use the following table to convert each letter into a two digit number, and then we will put the numbers together. Therefore, the message will be turned into a string of numbers, which will be Bob's plaintext:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

TABLE 1

Hence, Bob's message, "HI", will be turn into "0708". Thus, our plaintext now is P = 0708. After turning the original message into a string of numbers, Bob is now able to encrypt the plaintext with Alice's encryption key (3233, 7) to obtain the ciphertext:

Thus, (P)^e = (0708)^7 ≡ 663 (mod 3233) = C. (5)

Therefore, Bob obtains the following ciphertext: C = 663.

Then, Bob sends Alice the disguised form of the message or ciphertext (C = 663), which only Alice can turn back to the original plaintext using her secret decryption key (n, d) which is (3233, 1783). Let us check if the decryption key works by computing C^d:

From equation (5) we have that: (C)^d ≡ ((P)^e)^d (mod n)

We may rewrite the previous equation as: C^d ≡ P^(ed) (mod n)

Recall that: ed ≡ 1 (mod ϕ(n)). Therefore, ed = 1 + (ϕ(n))(k), for some positive integer k.

Hence, we have that: C^d ≡ P^(ed) = P^(1 + (ϕ(n))(k)) = [P^1] [P^(ϕ(n) k)] (mod n)

We may rewrite the previous equation as: C^d ≡ [P^1] [(P^ϕ(n))^k] (mod n)

By Euler's Totient Theorem, P^ϕ(n) ≡ 1 (mod n), since g.c.d.(P, n) = 1.

Thus, we have that: [P^1] [(P^ϕ(n))^k] ≡ [P^1] [(1)^k] (mod n)

Since k is a positive integer, we have that 1^k = 1.

Therefore, we finally obtain that: C^d ≡ [P^1] [(1)^k] = (P)(1) = P (mod n).

Thus, Alice decrypts the message with her decryption key (n, d) and recovers the plaintext P which is equal to 0708. Then, she uses the same table to turn the plaintext back into Bob's message, and she obtains: 0708 = "HI".

If Eve was able to factor n by finding the primes p and q such that n = (p)(q), then Eve would know the value $\phi(n) = (p-1)(q-1)$, which she could use (along with the value of e which is public) to calculate the decryption exponent d and thereby be able to decrypt all messages sent to Alice. However, factoring n is computationally difficult, which means that this path may not be available for Eve. We shall discuss this issue later in this paper.

FEATURES AND HISTORY OF RSA

In this section we will discuss how cryptosystems worked prior to the invention of RSA. Then we will describe some of the inconveniences of using pre-RSA cryptosystems. These inconveniences served as a motivation for the invention of RSA and a new type of cryptosystems that changed the world. We will also describe how RSA was created, who the authors are, and some features that make the RSA cryptosystem different from all other cryptosystem that existed before RSA.

Before the RSA cryptosystem was created, cryptosystems had a completely different structure from the modern cryptosystems, and thus worked differently. All these previous cryptosystems are examples of what are called symmetric cryptosystems. In a symmetric cryptosystem knowing what the encryption key is and how to encrypt allows you to easily or quickly find the decryption key. For that reason, when using a symmetric cryptosystem both encryption and decryption keys have to be kept secret.

In order to understand this new concept, we are going to illustrate how symmetric cryptosystems work by presenting a very simple example. Allow us to describe the Caesar cryptosystem, also known as the shift cryptosystem. This cryptosystem was named after Julius Caesar, who used it in his correspondence over 2 thousand years ago (this might give a small idea of the significance of cryptosystems in human history). Now, imagine that Bob wants to send a message to Alice and again he does not want Eve (or anyone other than Alice) to know the content of the message. Then, Bob meets with Alice in a secret place and whispers in her ear that he is going to use the Caesar cryptosystem and the number 3 for the encryption key. Thus, only Alice and Bob have knowledge of the encryption key. Now

imagine that Bob wants to send the following message: “S E C R E T”. Assume that Bob uses the same table that we used before to convert his message into numbers.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

TABLE 2

Therefore, Bob obtains the following string of numbers (which is going to be his plaintext): “18 04 02 17 04 19”. Bob now uses the Caesar cryptosystem to turn his plaintext into ciphertext.

Let Plaintext = P = P_1 P_2 P_3 … P_n.

Then, ciphertext = C = C_1 C_2 C_3 … C_n. Where, C_i = (P_i + “encryption key”) mod 26, for i = 1, 2, 3, …, n (This is known as the encryption process in the Caesar cryptosystem). In this case we are using 26 because there are 26 letters in the English alphabet. We use modulo 26 so that when we convert our plaintext into ciphertext we obtain a number from 0 to 25 which we can use to convert back to letters using the same conversion table. (This is known as the encryption process in the Caesar cryptosystem).

Similarly, P_i = (C_i + “decryption key”) mod m, for i = 1, 2, 3, …, n (This is known as the encryption process in the Caesar cryptosystem).

Note that in this case the decryption key is equal to the encryption key times negative one.

Thus, as we said above knowing the encryption key allows us to easily find the decryption key.

For instance if P_j = 25, then C_j = 25 + 3 ≡ 2 (mod 26).

Similarly, if C_j = 2, then P_j = 2 – 3 = – 1 ≡ 25 (mod 26).

Therefore, Bob applies the Caesar cryptosystem and obtains the following ciphertext: “21 07 05 20 07 22” (= V H F U H W). Then, Bob sends this ciphertext to Alice.

Alice, who knows that Bob used the Caesar cryptosystem and 3 for his encryption key can easily find the decryption key. Hence, she can reverse the process of encryption and thus obtain Bob’s original plaintext by simply imputing the decryption key (– 3 in this case) with Bob’s ciphertext into the decryption process.

As we just saw with the Caesar cryptosystem, knowing the encryption key of a symmetric cryptosystem is enough information to easily or quickly calculate the decryption key. One of the issues with that characteristic is that every time a certain person wants to use this kind of cryptosystem to put a disguise on a message he/she is going to need to agree secretly on an encryption key with the person whom the message is being sent to. Communicating with someone via phone call, messages, or any other channel of communication is not secure enough since there might be someone else intercepting the messages or listening to their conversation. Recall that learning about the encryption key in symmetric cryptosystems entails finding the decryption key and thus breaking the cryptosystem. Thus, most of the times, the only truly safe way to agree on a secret key was that the sender and receiver meet in person.

Now, let us imagine that we are the owners of a big bank, and we decide to use a symmetric cryptosystem to put a disguise on every message that is being sent to each of our clients. Thereby, every client can feel reassured that no one else will be able read his/her personal bank information. However, as we said we can not simply send the encryption keys in messages to our clients because there might be someone intercepting the messages. Thus, our only option to keep our clients’ privacy safe is to meet in person with each of them

individually and hand him/her the encryption key. The problem is that if our bank is really big, it causes this process to be extremely expensive. Likewise, we will also have to keep track of every single client's encryption key which implies more cost. This issue of having to meet in person to agree on a key is known as the key distribution problem.

Whitfield Diffie and Martin Hellman, two Stanford University researchers in cryptography, together with Ralph Merkle, a computer scientist, worked for years trying to find a way to fix the problem of key distribution. After years of hard work, they published a system in 1976 that allows two people two agree on a key without having to meet in person. This system uses modular arithmetic functions and allows the two people to agree on an encryption key, while making sure that even if someone were to be listening to their conversation, he/she still will not have enough information to find out about the encryption key. The only requirement with this system is that the two people that are agreeing on a key have to be available at the same time.

Suppose that we are back to the situation where we are the owners of the big bank. In order to save money and time we decide to use the system to establish an encryption key for each client. Combining the symmetric cryptosystem with the system to agree on keys is a big improvement for our bank. However, we still have to keep track of all keys of our clients and also there is another inconvenience:

Let us now imagine that we make a new client from the Middle East and that our bank is located in Ohio. Imagine that we need to contact him urgently about some private information. Hence, first we need to agree on an encryption key with him. Nonetheless, we can not use the system because he is in the Middle East and they have a different time zone. We want the message to be received in a timely manner. However, by the time that he may

be available it might be already late. Therefore, our new system is very useful, but is not very convenient.

The two Stanford University researchers, Whitfield Diffie and Martin Hellman, became aware of that and tried to find a new solution. They worked for years in a new idea that could represent a very important innovation with regard to cryptosystems. They thought about a completely new kind of cryptosystem, one that would overcome all the disadvantages that the symmetric cryptosystems had even with the aid of the key establishment system. This new kind of cryptosystem is called an asymmetric cryptosystem. In asymmetric cryptosystems knowing the encryption key does not easily lead to knowledge of the decryption key. Indeed, it would require an unreasonable amount of time and/or effort to calculate the decryption key based solely on the knowledge of the encryption key. They suggested that, although this had never been done before, there might be a way to encrypt messages using an encryption key that could be made public without affecting the strength of the cryptosystem. Then, the decryption would be done using a decryption key (a secret or private key) which only the people for whom the messages are intended would have access to.

Let us go back to the example of Alice, Bob, and Eve. Just as before, assume that Bob wants to send a message to Alice, but does not want Eve to know the content of it. Bob decides to use an asymmetric cryptosystem to put a disguise on his message. Then, he just needs to obtain Alice's public key which has been published before and so is available to everyone (including Eve), encrypt the message using that key, and send it to Alice. After this, Alice who has the secret decryption key is the only person that is going to be able to

decrypt Bob's message. Thus, Eve can not decrypt the encrypted message if the cryptosystem is strong enough.

This was a powerful new idea. With this kind of cryptosystem the whole problem of the key distribution was avoided since the encryption keys do not enable anyone to find the decryption keys, so they can be made public. Thus, there is no need to meet secretly in person to avoid eavesdropping. Because of that reason, everyone using this kind of cryptosystem could publish his/her encryption keys, and thus there will be no need for big companies (like our bank) to establish an individual encryption key with each client and to keep track of all them.

ASYMMETRIC KEY ENCRYPTION

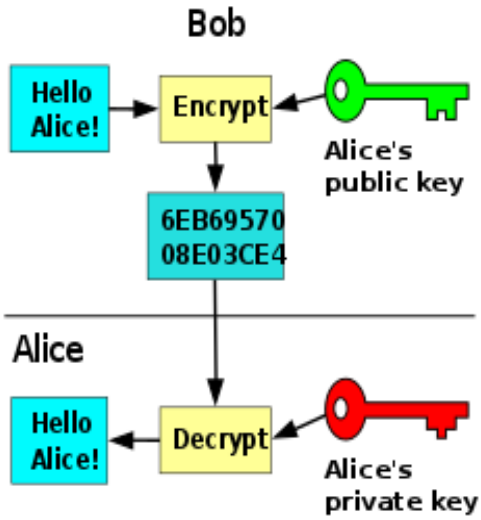

FIGURE 1

VS

SYMMETRIC KEY ENCRYPTION

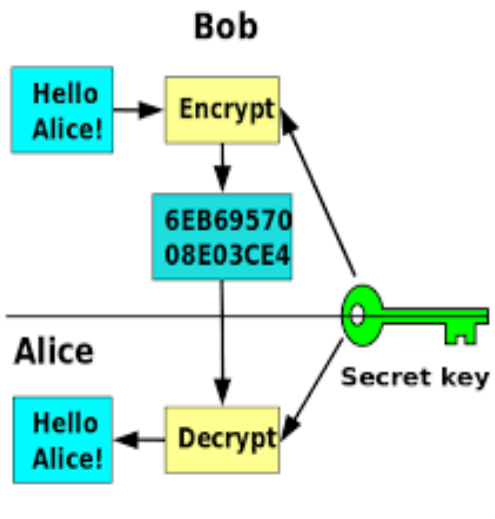

FIGURE 2

For a while the idea worked perfectly fine in theory, but seemed unfeasible in practice. Since the cryptosystem needed to be secure enough, the challenge was to find a suitable mathematical function that would make the encryption process simple for anyone, while

making exceedingly difficult for anyone to reverse the encryption process even with knowledge of the encryption key. Recall that in other to be useful, in any cryptosystem having knowledge of a decryption key should be enough to easily or quickly reverse the encryption process. There are some functions in mathematics that fit this prior description; they are called one-way functions. These are functions that are “easy” to do (encrypt); nonetheless, extremely difficult to undo (decrypt).

Modular arithmetic is an area of mathematics rich in one-way functions. A good example to see how modular arithmetic functions fit as one-way function is the following: Let us say we have the following function: $y = 55^x \pmod{111,111,111}$. We could pick different values for x and get the corresponding values of y fairly easy. For instance, let $x = 5$, then the result of the function is $y = 55^5 = 503,284,375 \equiv 58,839,931 \pmod{111,111,111}$.

However, if we just say that the result of 55^x is for example $y = 88888 \pmod{111,111,111}$, it will be extremely difficult to find the value of x. It would require an unreasonable amount of computer time for this calculation.

Thus, as we just saw in this modular arithmetic function, the first direction (finding y for a value of x) was significantly simple to do, but it was extremely difficult to reverse (finding x for a value of y). Thus, the function we described above is an example of a one way function.

Whitfield Diffie and Martin Hellman worked for years trying to find a one way function that would fulfill the requirements for the asymmetric cryptosystem. However, they could not find such a function. In 1976 they published a scientific paper called “New Directions in Cryptography,” for the purpose of calling on other researchers around the

world to look for a function that would make this possible. Since such a function had potential to make an important change in the world of cryptosystems; many mathematicians and scientists from all over the world tried very hard to find a suitable function.

For a short period of time it seemed that Diffie and Hellman's idea worked only in theory. No one seemed to be able to find a one-way function that worked well in practice. Nonetheless, this would all change in 1977:

A group of three researchers of the MIT Laboratory of Computer Science was very interested in finding the function that satisfies the requirements of an asymmetric cryptosystem and willing to solve the problem. Their names are Ronald Rivest, Adi Shamir and Leonard Adleman. The three of them together formed a perfect team. Rivest is a computer scientist with a fantastic ability to assimilate new ideas and apply them in unlikely places. He was always bringing innovative ideas and coming up with all sorts of new candidates for the function. Shamir, also a computer scientist, had a lightning intellect and an ability to see through the debris and focus on the core of the problem. He often came up with ideas for formulating an asymmetric cipher as well. Adleman, a mathematician with tremendous stamina, rigor and patience, was mainly in charge of spotting the flaws in the ideas of Rivest and Shamir, making sure that they do not waste time following false leads. Rivest and Shamir spent almost a year coming up with new ideas for the system, and Adleman spent a year shooting them down.

One night in April 1977, Rivest, unable to sleep, started to read a mathematics textbook. Suddenly, he came up with a new idea, he worked on that for the entire night. The morning after he showed it to Adleman, who as usual went through his process trying to tear it apart; however, surprisingly, he did not find any mistakes. They named it the RSA

cryptosystem (Rivest, Shamir, Adleman). Before we continue with Rivest's idea, allow us to review the requirements that Diffie and Hellman set:

- Alice needs to have an encryption key which could be made public without compromising the security of the decryption key. Then, Alice would publish the key so that Bob (or anyone) can use it to encrypt his messages and send them to her. Since the process of encryption corresponds to a one-way function it would take an unreasonable amount of time and/or effort to anyone (like Eve) to undo the process of encryption and/or find the decryption key.
- Alice must have a secret decryption key (different than the encryption key) that can reverse the process. That way, Alice can read messages encrypted with her encryption key.

As we just saw (in section How RSA Works), RSA perfectly fulfill these requirements. However, one may think that this cryptosystem is not very safe since everyone has access to the encryption key and thus to the number n. Recall that in RSA the decryption key is calculated by finding the multiplicative inverse of the public exponent e modulo the totient $\phi(n)$, and $\phi(n) = (p - 1)(q - 1)$. Therefore, basically, if a person wants to read a message encrypted with RSA, all he/she needs to do is find the two prime factors of the number n. Because, as we just said, knowing the two prime factors enables to easily calculate the private decryption key. We can quickly find the prime factors when n is relatively small. For instance, it is not hard to say that the factor primes of 55 are 5 and 11. Nonetheless, factoring and multiplying large primes is also a one way function. Multiplying very large prime numbers is relatively fast (or easy) for a big computer. However, if the two

primes are large enough (hundreds of digits long) it would take an enormous amount of time for any computer to factor n. For example, for important banking transactions n tends to be at least 10^308. The combined efforts of a hundred million personal computers would take more than a thousand years to find the two prime factors; and thus, crack such a cryptosystem. The reality is that, nowadays, with sufficiently large values of p and q, RSA is invulnerable.

This is also one of the best features about RSA, its strength depends on how large the public key (encryption key) is. In other words, the larger the pair of prime factors are, the longer it takes to break it because it implies that more computations are needed to be done. For instance, back in the 80's when RSA was starting to become popular, a public key n of 50 digits was considered secure enough because computers were not as fast as they are now. Hence, the amount of time that those computers needed to factor n of 50 digits was unreasonable. However, as technology evolves, computers become more efficient and thus they can do more computation in less amount of time. For example, nowadays almost any modern personal computer can factor n of 50 digits in a reasonable amount of time. That is the reason why nowadays the public key n tends to have many more digits (hundreds of digits long).

Even though technology keeps evolving, and thus computers become faster and faster, a regular computer will never be able to break the RSA cryptosystem. No matter how fast a computer is, we can always pick larger primes for our encryption key n (there are infinitely many prime numbers).

REFERENCES

"Diffie-Hellman Key exchange protocol." *Key exchange protocol*, diamond.boisestate.edu/~liljanab/MATH307/course_materials/DH.pdf.

Diffie, W., and M. Hellman. "New directions in cryptography." *IEEE Transactions on Information Theory*, vol. 22, no. 6, 1976, pp. 644–654., doi:10.1109/tit.1976.1055638.

Garrett, Paul B. *Making, breaking codes: an introduction to cryptology*. Pearson Education, 2011.

"Modular Arithmetic." *Brilliant Math & Science Wiki*, brilliant.org/wiki/modular-arithmetic/.

Rouse, Margaret. "What Is RSA Algorithm (Rivest-Shamir-Adleman)? - Definition from WhatIs.com." *SearchSecurity*, searchsecurity.techtarget.com/definition/RSA.

Singh, Simon. *The code book: the science of secrecy from ancient Egypt to quantum cryptography*. Distributed by Paw Prints / Baker & Taylor, 2009.

FIGURES:

(Figure 1) "Public Key Encryption." WIKIPEDIA, 7 Aug. 2006, it.wikipedia.org/wiki/File:Public_key_encryption.svg.

(Figure 2) "Symmetric Key Encryption." WIKIPEDIA COMMONS, 26 May 2014, commons.wikimedia.org/wiki/File:Symmetric_key_encryption.svg.