Practical Applications of Modular Arithmetic to Public Key Cryptography

Ohwadua, Emmanuel Obaro

Matric. Number: 029041042

Submitted as part of the requirements for the award of the MSc in Mathematics at University of Lagos, Nigeria.

October 2009

Certification

This research project was prepared and submitted by OHWADUA, Emmanuel Obaro of the Department of Mathematics, Faculty of Science, University of Lagos under the supervision of Dr. Olaleru, J.O., of the Department of Mathematics, University of Lagos, Akoka, Nigeria.

Signature:…… Date:…….....

Dr. Olaleru, J.O.

Supervisor

Signature:……. Date:…….....

Head of Department

Signature:…….. Date:…….....

External Examiner

Acknowledgement

My special thanks go to Dr. J.O. Olaleru – my project supervisor, for his understanding and assistance to me while carrying out this research work. I wish to also express my appreciation to other academic and non-academic staff of mathematics department for their support and cooperation throughout the entire programme.

My gratitude also go to others whose various contributions, one way or the other made it possible for the successful completion of this programme.

E. Obaro Ohwadua
October 2009

Table of Contents

List of Abbreviations

RSA – Rivest, Shamar, Adleman

DSA – Digital Signature Algorithm

DSS – Digital Signature Standard

Abstract

Practical Applications of Modular Arithmetic to Public Key Cryptography

by

Emmanuel Obaro Ohwadua

The Greek word for secret codes – cryptography refers to the science of encrypting and decrypting information by using mathematics to conceal the secret text – better refers to as the plaintext. Only the parties who know the pattern or are capable of breaking the code can discover the true contents of the coded text. A coded text is known as the ciphertext and the encryption algorithm or mathematical system used in enciphering the plaintext is known as the cipher. As privacy became more and more of a necessity in the digital age, partly due to the challenge to protect sensitive data on the Web and many other applications like it, the cipher algorithms became increasingly complex in order to counter the activities of cybercriminals sprawling on the internet. This complexity was obtained through the applications of advanced mathematical techniques such as discrete logarithms and large integer factorisation problems which is the subject of this research project.

We commenced this research project by giving a brief introduction to the science of cryptography in chapter 1. We equally elaborated on objectives, scope and methods of our research. In chapter 2, we dealt with the review of related mathematical topics in number theory such as groups and modular arithmetic that are relevant in the cryptographic algorithms such as RSA, Diffie-Hellman, Elgamal and DSA which we covered in chapter 4. Our chapter 3 focused on the literature review of the various cryptosystems mentioned in the foregoing in addition to elliptic curves and quantum cryptography. In each cryptographic algorithms discussed in chapter 4, we gave elaborate description of the encryption and decryption algorithms as the case may be, and the key generation techniques were illustrated with elaborate examples by applying the knowledge of the mathematical concepts discussed in chapter 2.

We concluded this research work by recommending areas for further research work for readers interested in elliptic curve and quantum cryptography.

CHAPTER 1

INTRODUCTION

Number theory where modular arithmetic is “born” may be one of the “purest” branches of mathematics, but it has turned out to be one of the most useful when it comes to cryptography – an area of study that has attracted intensive research from the 1970s that is now being applied worldwide. Cryptography has a long and fascinating history. Cryptography is the study of mathematical techniques that is used to protect information in digital media either stored or transmitted, from unauthorized access that would prevent and detect cheating and other malicious activities (1).

Cryptography can be traced from its initial and limited use by the Egyptians some 4000 years ago, to the twentieth century where it played a crucial role in the outcome of both world wars (2). The predominant practitioners of the art were those associated with the military, the diplomatic service and government in general. Cryptography was used as a tool to protect national secrets and strategies. However, the proliferation of computers and communications systems in the 1960s brought with it a demand from the private sector for means to protect information in digital form and to provide security services. Over the years, various standards and infrastructures involving cryptography are being put in place. Security products are being developed to address the security needs of an information intensive society. The subject of this

research project is to discuss the significant role of modular arithmetic in shaping major developments in modern cryptography.

1.1 Information security and cryptography

The concept of information will be taken to be an understood quantity. To introduce cryptography, an understanding of issues related to information security in general is necessary. Information security manifests itself in many ways according to the situation and requirement. Regardless of who is involved, to one degree or another, all parties to a transaction must have confidence that certain objectives associated with information security have been met. The overall objective may be plausibly described as the secrecy of the information either stored or transmitted so that only authorised users or persons can have access to the information while unauthorised users or persons are denied access. Over the centuries, an elaborate set of protocols and mechanisms has been created to deal with information security issues when the information is conveyed by physical documents (3). Often the objectives of information security cannot solely be achieved through mathematical algorithms and protocols alone, but require procedural techniques and abidance of laws to achieve the desired result. For example, privacy of letters is provided by sealed envelopes delivered by an accepted mail service. The physical security of the envelope is, for practical necessity, limited and so laws are enacted which make it a criminal offense to open mail for which one is not authorized. It is sometimes the case that security is achieved not through the information itself but through the physical document

recording it. For example, paper currency requires special inks and material to prevent counterfeiting.

One of the fundamental tools used in information security is the signature. It is a building block for many other services such as identification and witnessing. Having learned the basics in writing, an individual is taught how to produce a handwritten signature for the purpose of identification. At contract age, the signature evolves to take on a very integral part of the person's identity. This signature is intended to be unique to the individual and serve as a means to identify, authorize, and validate. With electronic information the concept of a signature needs to be redressed; it cannot simply be something unique to the signer and independent of the information signed. Electronic replication of it is so simple that appending a signature to a document not signed by the originator of the signature is almost a triviality. Analogues of the "paper protocols" currently in use are required. Hopefully these new electronic based protocols are at least as good as those they replace. There is a unique opportunity for society to introduce new and more efficient ways of ensuring information security. Much can be learned from the evolution of the paper based system, mimicking those aspects which have served us well and removing the inefficiencies. Achieving information security in an electronic society requires a vast array of technical and legal skills. There is, however, no guarantee that all of the information security objectives deemed necessary can be adequately met. The technical means is nonetheless provided through cryptography.

Cryptography, over the ages, has been an art practised by many who have devised ad hoc techniques to meet some of the information security requirements (4). The last twenty-five years have been a period of transition as the discipline moved from an art to a science (5). Before the

advent of modern cryptography, modular arithmetic could lay claim to being one of the purest – most application-free areas of mathematics. However, with the advent of modern cryptography, applications of modular arithmetic and similar applications in polynomial fields and elliptic curves renewed interest on a number of mathematical fronts (6). In the coming chapters, we shall give a lucid discussion on the application of modular arithmetic to cryptography, while interested readers on other mathematical applications in polynomial fields and elliptic curves can consult advance text in cryptography.

1.2 Cryptographic Keys

Cryptographic keys are central to cryptographic operations. Many cryptographic schemes consist of pairs of operations, such as encryption and decryption, or signing and verification. In cryptography, encryption is the process of transforming information (referred to as plaintext) using an algorithm (called cipher) to make it unreadable to anyone except those possessing special knowledge, usually referred to as a key (7). In many contexts as we shall see below, the word encryption also implicitly refers to the reverse process, decryption (e.g. “key for encryption” can typically also perform decryption), to make the encrypted information readable again (i.e. to make it unencrypted). A key is a piece of variable data that is fed as input into a cryptographic algorithm to perform one such operation. In a well-designed cryptographic scheme, the security of the scheme depends only on the security of the keys used.

Cryptographic keys can be classified based on their usage within a cryptographic scheme, as Symmetric Keys or Asymmetric Keys. A symmetric key is a single key that is used for both

operations in a cryptographic scheme (for example, to both encrypt and to decrypt your data). Usually, the security of the scheme depends on ensuring that the key is only known to the authorized participants. Asymmetric keys, on the other hand, are used in cryptographic schemes where different keys are needed for each operation. Common examples of such schemes are those using public/private key pairs, where the security of the scheme depends on ensuring that the private key is only known to one party. For example, public/private key encryption systems use two keys, a public key that anyone can use to encrypt data and a private key that only the authorized recipient possesses and that can be used to decrypt the data. Cryptographic keys shall be discussed in detail in subsequent chapters.

1.3 An Overview of Modular Arithmetic

Modular arithmetic is a branch of Number Theory that is concerned with the arithmetic of congruences, sometimes known informally as "clock arithmetic" (8). In modular arithmetic, numbers "wrap around" upon reaching a given fixed quantity, which is known as the *modulus* (which would be 12 in the case of hours on a clock, or 60 in the case of minutes or seconds on a clock). For example, a familiar use of modular arithmetic is its use in the 12-hour clock, in which the day is divided into two 12 hour periods. If the time is 8:00 now, then 8 hours later it will be 4:00. Usual addition would suggest that the later time should be 8 + 8 = 16, but this is not the answer because clock time "wraps around" every 12 hours; there is no "16 o'clock". Likewise, if the clock starts at 12:00 (noon) and 20 hours elapse, then the time will be 8:00 the next day, rather than 32:00. Since the hour number starts over when it reaches 12, this is arithmetic *modulo* 12. It is this “wrap around” property of the modular arithmetic that has given it

the enviable attention in the area of cryptographic research which is the subject of this research project.

1.4 Objectives

Our goal is to demonstrate the applicability of Modular Arithmetic to Public Key Cryptosystems in the computation of cryptographic keys used for securing information and for digital signature scheme.

1.5 Motivation

We are motivated by the increase in research activities in this field and developments that have taken place in the last 25 years. It is therefore necessary to inform wherever possible would-be mathematicians or students of mathematics of this important area of research and the impact it has made in the security of information either stored or transmitted in the digital age.

1.6 Methods of Research

Review of related literatures and the computation of public and private key pairs for some selected public key cryptosystems using modular arithmetic.

1.7 Scope of Research

Our scope shall include treatment of background subject – Groups and its properties; discussion of Modular Arithmetic, Modular Exponentiation and Euclidean Algorithms. We shall give an overview of Cryptography, followed by an introduction to Public Key Cryptography/ Cryptosystems. This shall be followed by the computation of private and public key pairs of some selected public key cryptosystems such as RSA, Diffie-Hellman and Elgamal signature scheme.

1.8 Chapter Summary

This chapter is essentially an introduction to the research topic with a peep into cryptography – how it all started, some key terminologies explained and an overview of modular arithmetic. We equally elaborated our research objective, methods of research and finally, motivation and scope of our research. The next chapter shall be devoted to the treatment of relevant areas in Number Theory – groups and modular arithmetic and their properties required for public-key cryptosystems.

CHAPTER 2

MODULAR ARITHMETIC

The mathematical basis for the algorithms used in public key cryptosystems (RSA, Diffie–Hellman, Elgamal, elliptic curves, etc.) relies on some moderately deep concepts and results from number theory which is covered comprehensively in any advanced book on Number Theory, however interested readers can consult (8). Underlying the deep results in number theory, however, are some fundamentals that are almost, but perhaps not quite, obvious. Our purpose in this chapter is to present those fundamental parts of the theory of numbers necessary for an understanding of the computations involved in generating cryptographic keys in selected public key cryptosystem such as RSA, Diffie-Hellman, Elgamal and Digital signature scheme, while advanced number theory necessary for elliptic curves etc. shall not be covered . In our discussion in this chapter, most related elementary theorems shall be stated without proof, but examples shall be given whenever necessary for a proper understanding. We shall begin our discussion with a look at the ordinary integers, though elementary at the level of this research work, however central to any work in number theory is the nature of arithmetic (addition, multiplication, and the like) modulo prime numbers.

2.1 Overview of Group Theory

We'll commence our discussion by examining a question of the form: What is the solution of the equation (9):

$$4x = 3 \qquad 2.1.1$$

The answer depends on what values we allow x to be. If we are doing all our arithmetic using the integers then there is no solution – there is no integer that gives 3 upon being multiplied by 4. On the other hand if we are doing our arithmetic in $\mathbb{Z}/5$ (integers mod 5) then x = 2 is a solution. If we are using the more usual rational number system $\mathbb{Q}$, then x = ¾ is a solution.

We can gain insight into all such questions by considering the general equation

$$\alpha \bullet x = b \qquad 2.1.2$$

and then bringing up the question of solutions. Well, what values are α and b? To what class of objects is x allowed to belong? What is the operation symbolized by the dot (•)?

Group theory is concerned with systems in which (2.1.2) always has a unique solution. The theory does not concern itself with what α and b actually are nor with what the operation symbolized by "•" actually is. Group theory requires only that a mathematical system obey a few simple rules. The theory then seeks to find out properties common to all systems that obey these few rules.

The axioms (basic rules) for a group are:

1. **Closure:** If a and b are in the group then a • b is also in the group.
2. **Associativity:** If a, b and c are in the group then (a • b) • c = a • (b • c).
3. **Identity:** There is an element e of the group such that for any element a of the group, a • e = e • a = a.
4. **Inverse:** For any element a of the group there is an element a^{-1} such that $a \cdot a^{-1} = e$ and $a^{-1} \cdot a = e$

Any mathematical system that obeys those four rules is a group. The study of systems that obey these four rules is the basis of GROUP THEORY.

2.1.1 Cyclic Groups and Subgroups

Let's start with the number 1. We'll allow ourselves to add or subtract the number 1 to get to new numbers. So, to get to 13 simply add 1 12 times. To get to -42 simply subtract 1 43 times. The fact that the integers can be "built" by adding and subtracting 1 means that the additive group of integers is a *cyclic group* (8).

Thus, groups that can be generated in their entirety from one member are called cyclic groups. For infinite groups, we have to clarify what we mean by "generated from." For example in the additive group of Integers starting with 1 and adding it over and over to itself will never get a negative number nor the identity zero. Hence for a cyclic group

we have the definition that all the elements may be generated from a single element together with its inverse. For finite cyclic groups the addition of "together with its inverse" is not needed.

If we select some element α from a group G then we can consider the subset of all elements of G that are powers of α. This subset forms a subgroup of G and is called the *cyclic subgroup* generated by α (8). It forms a subgroup since it is

1. It is closed – if you multiply powers of α you end up with powers of α.
2. Has the identity – $\alpha \bullet \alpha^{-1} = \alpha^0 = e$
3. Has inverses – the inverse of any product of α's is a similar product of α^{-1}'s.

A few facts about cyclic groups and cyclic subgroups:

1. Cyclic groups are Abelian (commutative group).
2. All groups of prime order are cyclic.
3. The subgroup of a group G generated by α is the intersection of all subgroups of G containing α.
4. All infinite cyclic groups look like the additive group of integers.

2.2 The Number System

We have just seen in 1.3 two examples of modular arithmetic – mod 7 for days of the week, and mod 12 for hours of the clock. Another common example is mod 24 for

hours of in a day. We can actually do modular arithmetic using any number. When working modulo N (otherwise referred to as mod N), where N is some positive number, the numbers that we work with are the numbers from 0 to N – 1 inclusive. That is: 0, 1, 2, 3, 4 .... , N – 1.

In other words, instead of having numbers that go on and on forever, when working modulo N we only have N different numbers to work with. These are the only numbers available to us. For example:

mod 5 uses the numbers 0, 1, 2, 3, and 4 only

mod 12 uses the numbers 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, and 11 only

mod 125 uses the numbers 0, 1, 2, 3, ... , 121, 122, 123, and 124 only

When working mod N we often refer to the number N as the modulus.

2.2.1 Modular Addition

As we have just seen, when working modulo N we only have N different numbers to use: 0, 1, 2, 3, ... , N – 1. However, we can add two numbers modulo N and end up N? For example, if we are working in modulo 8, we would like to think that 3 + 5 = 8, but in mod 8 arithmetic, there is no such number as 8 – there are only numbers 0, 1, 2, 3, 4, 5, 6 and 7. In other words, after we have reached 7, we start to count again from 0. We write this as follows:

$$3 + 5 = 0 \bmod 8.$$

Also,

$$3113 = 8 * 359 + 241.$$

Since adding these eight multiples of 359 makes no difference, we can just 'forget about them' and thus 3113 is the same as 241 modulo 359. In other words:

$$3113 \equiv 241 \bmod 359.$$

Thus, when working modulo N, adding or subtracting multiples of N does not change a number.

Similarly, for negative numbers modulo another number, we have for example,

$$-17 \bmod 10 = -2 * 10 + 3$$

Since subtracting these two multiples of 10 makes no difference, we can just 'forget about them' and thus -17 is the same as 3 modulo 10. In other words:

$$-17 \equiv 3 \bmod 10.$$

Another example is $-61 = -13 * 5 + 4$.

Thus, $-61 \equiv 4 \bmod 5$

2.2.2 Divisibility

Most of elementary number theory concerns the integers $\mathbb{Z}$ = {. . . , -4, -3, -2, -1, 0, 1, 2, 3, 4 . . .}. Occasionally, we refer to the rational numbers $\mathbb{Q}$ = {a/b : a, b, $\in \mathbb{Z}$}. If a, b $\in \mathbb{Z}$, and b ≠ 0, then the integer part of a/b, written [a/b], also referred to as the floor of a/b and written $\lfloor a/b \rfloor$, is the integer c such that c ≤ a/b < c + 1 (10). The ceiling of a/b, written $\lceil a/b \rceil$, is the integer c such that c − 1 < a/b ≤ c.

We present these definitions to clarify the notation for negative values. The integer part function "truncates toward negative infinity" in that, for example,

$$[(-5)/4] = [5/(-4)] = -2.$$

An integer b is divisible by an integer a if there exists an integer c such that b = ac. We say that a is a divisor of b.

Notes:

- We write a|b or a∤b, read "a divides b" or "a does not divide b", according as b is or is not divisible by a.
- We note that a|a always holds. If a|b and 0 < a < b, then we say that a is a proper divisor of b.
- We note also that a|0 holds for all integers a but that 0|b never holds for any non-zero integer b. The only instance in which a|0 would make sense would be in the expression 0|0. By convention, we specifically exclude this relatively useless special

case because permitting it would require the addition of extra qualifications on a large number of theorems.

Theorem 2.1. *The following hold for all integers:*

1. *If $a|b$, then $a|bc$ for any integer c.*
2. *If $a|b$ and $b|c$, then $a|c$.*
3. *If $a|b$, then $ac|bc$ for any integer c.*
4. *If $a|b$ and $b|a$, then $a = \pm b$.*
5. *If $a|b$ and $a|c$, then $a|(bx + cy)$ for any integers x and y.*
6. *If $a|b$, $a > 0$, and $b > 0$, then $a \leq b$.*

One of the basic results of school arithmetic is that division of two integers yields a remainder that is smaller than the divisor. This is formalized in the following theorem.

Theorem 2.2. *(The division algorithm). If a and $b > 0$ are integers, then there exists a unique pair of integers q and r such that $a = bq + r$, and $0 \leq r < b$;*

where the integer b can be regarded as the divisor, q as the quotient and r – the remainder (10).

Example. Just as was the case above, in our usual decimal arithmetic we work in modulo 10; now suppose a = 57, and b = 5, then

$$57 = 5 * 11 + 2, \text{ for } 0 \leq 2 < 5$$

Theorem 2.3. *The gcd of integers a and b, $g = gcd(a, b)$, is the least positive value of $ax + by$ as x and y range over all integers* (10).

The greatest common divisor (usually abbreviated to *gcd*) of two numbers is the largest whole number that divides neatly into both numbers without leaving a remainder. In other words, the *gcd* of two numbers is the largest number that is a divisor of both numbers. Let's look at some examples:

- the *gcd* of 14 and 21 is 7, since 7 is the largest divisor of both 14 and 21.
- the *gcd* of 35 and 70 is 35, since 35 is the largest divisor of both 35 and 70.
- the *gcd* of 1415 and 1500 is 5, since 5 is the largest divisor of both 1415 and 1500.

We normally abbreviate the writing of the *gcd* of two numbers using the following notation: gcd(1415,1500) = 5

2.3 The Euclidean Algorithm

The Euclidean algorithm, also called Euclid's algorithm, is an algorithm for finding the greatest common divisor of two numbers ɑ and b. It is the key to 'dividing' in modular arithmetic in a simple mathematical approach. The process can be explained in the following two theorems:

Theorem 2.4. *(The Euclidean algorithm). If, given integers ɑ and b, we make a repeated application of the division algorithm:*

$$b = r_0,$$

$$a = bq_1 + r_1,\ 0 < r_1 < b$$

$$b = r_1q_2 + r_2,\ 0 < r_2 < r_1 \qquad (2.3.1)$$

$$r_1 = r_2q_3 + r_3,\ 0 < r_3 < r_2$$

. . .

then the process must terminate with $r_{j+1} = 0$ for some j and we have that $r_j = gcd(a, b)$ (10).

Theorem 2.5. *(The extended Euclidean algorithm). The values x_0 and y_0 such that $gcd(a, b) = ax_0 + by_0$ can be obtained by eliminating the r_i from Eq. (2.3) above* (10).

Example. Suppose a = 261 and b = 48, we can apply the Euclidean algorithm to the pair as follows:

$$261 = 48 \times 5 + 21,$$

$$48 = 21 \times 2 + 6,$$

$$21 = 6 \times 3 + 3,$$

$$6 = 3 \times 2 + 0,$$

so the sequence q_1, q_2, q_3, q_4 is 5, 2, 3, 2.

2.3.1 Prime Numbers and Coprime

We say that an integer p > 1 is a prime number if there are no positive divisors of p other than 1 and p. If p is not prime, then it is a composite number.

Theorem 2.6. *The following hold* (11):

1. *Every positive integer has a prime divisor.*
2. *If, for a prime p, we have p|ab, then either p|a or p|b.*
3. *If, for a prime p, we have* $p|\prod_{i=1}^{n} ai$, *then* $p|a_i$ *for some i.*

For example, 19 is prime because 19 and 1 are the only numbers that divide into 19. Other examples:

27 is prime

18 is not prime because 2, 3, 6 and 9 are also divisors

2 is prime no other even number is prime because 2 is a divisor

We say that two numbers are **coprime** if their greatest common divisor is 1. In other words, two numbers are coprime if the only divisor that they have in common is the number 1. Or using *gcd* notation, two numbers X and Y are coprime if gcd(X,Y) = 1.

For example,

- 42 and 55 are coprime, since no number other than 1 divides evenly into both 42 and 55.

- 101 and 283 are coprime, since no number other than 1 divides evenly into both 101 and 283.

2.3.2 Multiplicative Inverse

The multiplicative inverse of a chosen number is the number that you multiply the chosen number by to get 1. In other words, the multiplicative inverse of 5 is 1/5. It can be written thus,

$$57^{-1} = 1/57.$$

Now let us consider the multiplicative inverse of one number modulo another, sometimes referred to as a **modular inverse** (11). Once we start to compute modular inverses we will discover that they do not behave much like multiplicative inverses in the 'real numbers'. For example:

- many numbers (other than 0) do not have a multiplicative inverse modulo another number;
- there exist numbers (other than 1) that are their own multiplicative inverse modulo another number.

Let us consider the multiplicative inverse of 2 modulo 7; in other words, what numbers do you multiply by 2 to get the result 1 mod 7? One rather crude way is simply to try out all of the numbers mod 7 until we find out if there is an answer:

$2 * 0 \equiv 0 \bmod 7$

$2 * 1 \equiv 2 \bmod 7$

$2 * 2 \equiv 4 \bmod 7$

$2 * 3 \equiv 6 \bmod 7$

$2 * 4 \equiv 1 \bmod 7$

$2 * 5 \equiv 3 \bmod 7$

$2 * 6 \equiv 5 \bmod 7$

So $2^{-1} \equiv 4 \bmod 7$

Note: A number has a multiplicative inverse modulo another number when the two numbers are coprime.

For example,

- gcd(2,7) = 1, which means 2 and 7 are coprime, and thus $2^{-1} \bmod 7$ exists.
- gcd(5,17) = 1, which means 5 and 17 are coprime, and thus $5^{-1} \bmod 17$ exists.
- gcd(6,10) = 2, which means 6 and 10 are not coprime, and thus $6^{-1} \bmod 10$ does not exist.

However, in order to find modular inverses to set up key pairs for public key cipher system, we need to work with modulus numbers that are very large. Thus, the idea of exhaustively trying out all the possible numbers less than our modulus is not a good one, as it might take a ridiculous amount of time to perform all the necessary

calculations. A simple and more efficient technique for finding the multiplicative inverse of one number modulo another number is to use Euclid algorithm. The Euclidean Algorithm is simply a recipe for calculating modular inverses pairs of numbers for which there does exist a multiplicative inverse of one modulo the other – thus, our numbers must be coprime (12).

Suppose we wish to find the multiplicative inverse of A modulo N (where A and N are numbers and A is less than N). The Euclidean Algorithm stated in theorems 2.4 and 2.5 can be broken down into three steps:

Step 1: Repeated division

The first step involves conducting a series of divisions that start with A and N, and end up with us obtaining remainder 1.

Step 2: Backward substitution

The second step involves using the divisions we conducted in Step 1 to produce an equation of the form:

$$X * A + Y * N = 1$$

where X and Y are numbers that will be determined by Step 1 equations.

Step 3: Reduction modulo N

We now reduce modulo N by adding (or subtracting) multiples of N – this does not change a number modulo N, the Y * N bit of the above equation vanishes and we are left with

$$X * A \equiv 1 \bmod N$$

So X is the multiplicative inverse of A modulo N.

Example: We illustrate the steps of the Euclidean Algorithm by using it to find 9^{-1} mod 31.

Step 1: Repeated division

$$31 = 3 * 9 + 4 \qquad 2.2.1(a)$$

$$9 = 2 * 4 + 1 \qquad 2.2.2(a)$$

From 2.2.1(a) and 2.2.2(a), we have

$$4 = 31 - 3 * 9 \qquad 2.2.1(b)$$

and, $$1 = 9 - 2 * 4 \qquad 2.2.2(b)$$

Step 2: Backward substitution

The aim here is to take the 'B' equations that we established in Step 1 and use them in reverse order to produce an equation of the form:

$$X * 9 + Y * 31 = 1$$

where X and Y are numbers that will be determined by equations (2.3.1b) and (2.3.2b).

Thus,

$1 = 9 – 2 * 4$ by (2.3.2b)

$1 = 9 – 2 * (31 – 3 * 9)$ using (2.3.1b)

$1 = 9 – 2 * 31 + 6 * 9$

$1 = (9 + 6 * 9) – 2 * 31$

or $1 = 7 * 9 – 2 * 31$

which takes the form of equation:

$X * 9 – Y * 31 = 1$

where X = 7 and Y = -2 respectively.

Step 3: Reduction modulo N

In Step 2, we ended up with the expression:

$7 * 9 – 2 * 31 = 1$

Our goal is to obtain 9^{-1} mod 31. In other words we are looking for the number that you multiply 9 by to get 1 mod 31.

From the above equation we see that

$7 * 9 – 2 * 31 \equiv 1 \bmod 31.$

Now recall that adding or subtracting multiples of 31 makes no difference to a number modulo 31. So,

$$7 * 9 \equiv 1 \bmod 31.$$

Thus, $9^{-1} = 7 \bmod 31.$

The inverse of 9 modulo 31 is 7. We can easily check that we correct:

$$7 * 9 = 63 \equiv 1 \bmod 31$$

Further example: We shall illustrate the steps of the Euclidean Algorithm by using it to find $42^{-1} \bmod 47$.

Step 1: Repeated division

$$47 = 1 * 42 + 5 \qquad 1a$$

$$42 = 8 * 5 + 2 \qquad 2a$$

$$5 = 2 * 2 + 1 \qquad 3a$$

From 1a, 2a and 3a, we have respectively

$$5 = 47 - 1 * 42 \qquad 1b$$

$$2 = 42 - 8 * 5 \qquad 2b$$

$$1 = 5 - 2 * 2 \qquad 3b$$

Step 2: Backward substitution

$$1 = 5 - 2 * 2 \qquad \text{by (3b)}$$

$$1 = 5 - 2 * (42 - 8 * 5)$$

$$1 = 5 - 2 * 42 + 16 * 5$$

$$1 = 17 * 5 – 2 * 42$$

$$1 = 17 * (47 – 1 * 42) – 2 * 42$$

$$1 = 17 * 47 – 17 * 42 – 2 * 42$$

$$1 = 17 * 47 – 19 * 42$$

or, $$-19 * 42 + 17 * 47 = 1$$

where X = -19 and Y = 17. To check whether we are correct, we have

$$19 * 42 + 17 * 47 = -798 + 799 = 1$$

We are correct!

Step 3: Reduction modulo N

$$-19 * 42 + 17 * 47 = 1 \bmod 47$$

or, $$-19 * 42 \equiv 1 \bmod 47$$

Now of course there is no such number as –19 mod 47, but we do know that

$$–19 \equiv –19 + 47 \equiv 28 \bmod 47.$$

So we have,

$$28 * 42 \equiv 1 \bmod 47.$$

Thus, $42^{-1} \equiv 28 \bmod 47.$

We can easily check that we are correct:

$$28 * 42 = 1176 \equiv 1 \bmod 47.$$

2.4 Modular Exponentiation

Modular exponentiation is a type of exponentiation performed over a modulus. Modular exponentiation involves calculating the remainder when dividing by a positive integer m (called the modulus) a positive integer b (called the base) raised to the e-th power (e is called the exponent). In other words, suppose we have base b, exponent e, and modulus m, one wishes to calculate c such that (8):

$$c \equiv b^e \bmod m$$

While small examples are easy enough to do, in general the type of calculation that is needed in public key cipher system such as RSA, for example, potentially involves very large numbers. This requires the multiplication of large numbers by itself many times which can be considered very inefficient. Hence, a better approach is developed to speed up the process. In this section we shall present one such process that would enable us to perform very large modular exponentiations fairly easily.

Example: We shall begin by examining problems involving small numbers such as: $5^7 \bmod 21$

Thus,

$$5^7 = 5 * 5 * 5 * 5 * 5 * 5 * 5$$

$$= 78125$$

$$= 3720 * 21 + 5$$

$\equiv 5 \bmod 21$

Example: Let us look at $4^{11} \bmod 26$.

Thus,

$$4^{11} = 4 * 4 * 4 * 4 * 4 * 4 * 4 * 4 * 4 * 4 * 4$$

$$= 4194304$$

$$= 161319 * 26 + 10$$

$$\equiv 10 \bmod 26$$

Observe that this is a slightly tedious computation than the previous example.

Further example: Suppose we have $13^{22} \bmod 65$.

Then,

$$3^{22} = 3 * 13 * 13 * 13 * 13 * ...* 13 * 13 \text{ (22-times)}$$

$$= 3211838877954855105157369$$

$$= 49412905814690078540882 * 65 + 39$$

$$\equiv 39 \bmod 65$$

Observe that the calculations necessary to get from the first line to the second line and again from the second line to the third line involve very large numbers despite that fact that we only started with the relatively small numbers 13 and 22.

The time and computer power required to perform this type of calculation increases dramatically as the numbers involved start to rise. Clearly a more efficient method of exponentiation is needed other than brute force multiplication of the type that we have just tried in these examples. The algorithm is called **Repeated Squares**.

2.4.1 Repeated Squares Algorithm

There are three steps involved in the Repeated Squares Algorithm. The Algorithm can be best explained by means of an example. For clarity, we shall use a slightly different notation for multiplication in our description. Thus we shall write (X)(Y) to mean X * Y or "X multiplied by Y".

Example: We shall consider our earlier example given above: $5^7 \bmod 21$.

Step 1: Write the exponent as a sum of powers of 2.

We first write the exponent 7 in binary: $7_{10} = 111_2$

Recall that binary just defines a sum of powers of 2, so: $7 = 2^2 + 2^1 + 2^0$

Thus, $7 = 4 + 2 + 1$

We can now re-write the problem as:

$$5^7 \bmod 21 \equiv 5^{4+2+1} \bmod 21$$

$$\equiv (5^4)(5^2)(5^1) \bmod 21$$

Step 2: Construct a table of repeated squares.

From Step 1, we see that if we calculate each of 5^4, 5^2 and 5^1 modulo 21, then we can multiply them together to get the solution to the problem. We thus construct this table of "repeated squares". This is a faster process because in each case we use the previous calculation to perform the next one and then reduce modulo 21:

$$5^1 \equiv 5 \bmod 21$$

$$5^2 = (5)^2 = 25 \equiv 4 \bmod 21$$

$$5^4 = (5^2)^2 \equiv (4)^2 \equiv 16 \bmod 21$$

Observe that each line is the square of the previous line (reduced mod 21).

Step 3: Combine the results of Steps 1 and 2.

$$5^7 \bmod 21 \equiv (5^4)(5^2)(5^1) \bmod 21$$

$$\equiv (16)(4)(5) \bmod 21$$

$$\equiv 320 \bmod 21$$

$$\equiv 5 \bmod 21$$

This agrees with our previous answer.

Further example: Let us re-calculate our previous example, $13^{22} \bmod 65$ with this approach to appreciate the difference.

Step 1: Write the exponent as a sum of powers of 2.

We first write the exponent 22 in binary: $22_{10} = 10110_2$

Recall that binary just defines a sum of powers of 2, so: $22 = 2^4 + 2^2 + 2^1$

Thus, $22 = 16 + 4 + 2$

We can now re-write the question as:

$$13^{22} \bmod 65 \equiv 13^{16+4+2} \bmod 65$$

$$\equiv (13^{16})(13^4)(13^2) \bmod 65$$

Step 2: Construct a table of repeated squares.

$$13^1 = 13 \bmod 65$$

$$13^2 = (13)^2 = 169 \equiv 39 \bmod 65$$

$$13^4 = (13^2)^2 \equiv (39)^2 = 1521 \equiv 26 \bmod 65$$

$$13^8 = (13^4)^2 \equiv (26)^2 = 676 \equiv 26 \bmod 65$$

$$13^{16} = (13^8)^2 \equiv (26)^2 = 676 \equiv 26 \bmod 65$$

Step 3: Combine the results of Steps 1 and 2.

$$13^{22} \bmod 65 \equiv (13^{16})(13^4)(13^2) \bmod 65$$

$$\equiv (26)(26)(39) \bmod 65$$

$$\equiv 26364 \bmod 65$$

$$\equiv 39 \bmod 65$$

which agrees with our previous answer.

2.5 Chapter Summary

In this chapter, we covered extensively all relevant areas in modular arithmetic that is required to computer cryptographic keys in public key cipher system such as RSA, Diffie-Hellman, and Elgamal signature scheme. The topics include basic group theory, and introduction to modular arithmetic such as modular addition, subtraction and multiplication. Others include Euclidean Algorithm and Modular exponentiation. In chapter 3, we shall discuss Public Key cryptosystems with emphasis on RSA, Diffie-Hellman and Elgamal.

CHAPTER 3

PUBLIC KEY (ASYMMETRIC) CRYPTOGRAPHY

Even though cryptography has existed for thousands of years, its development was slow and unyielding. The need to communicate secretly had never been considerably threatened until the invention of computers and high-tech machines. As the need for privacy heightened, encryption techniques became further advanced. Encryption is the procedure of rendering a message into a concealed form so that it is decipherable exclusively by a particular recipient or recipients (13). The message in its original state is known as a plaintext (or cleartext); in its encrypted form, it is known as a ciphertext. Historically, the aim of encryption has been to enable two parties to exchange messages confidentially, even in the presence of an eavesdropper capable of intercepting most or all of their communications. The use of encryption has been confined chiefly to diplomatic and military circles in the past, but its scope in everyday life has broadened enormously in recent years. Thanks to the rise of the Internet. Active users of the Internet employ encryption on a regular basis. For instance, when accepting credit card information or processing other financial transactions on the internet, most web servers initiate encryption sessions with clients. In most browsers, the appearance of an icon representing a closed padlock on the bottom of the screen

indicates the use of encryption. By clicking on this padlock, a user can learn detailed information about the encryption session. Encryption also plays an important role in most important industrial communications systems, such as networks used for banking transactions. It is equally being used extensively by individuals and institutions for securing stored information on a computer system or on transit such as encrypted email messages.

3.1 Background of Cryptography

Cryptography dates back as far as 400 BC, during the times of Julius Caesar and the Roman Empire. To communicate between the Roman armies at different locations, Caesar used a simple substitution code involving the alphabet (Hebert) (3). This type of code shifts the letters of the alphabet so that each letter must be substituted for another one. This technique is easiest when one knows the number of shifts to the right or the left in the alphabet.

A typical Caesar cipher system with shift 5 to the right is displayed below:

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

V W X Y Z A B C D E F G H I J K L M N O P Q R S T U

Find each letter from a plaintext on the top row and replace it with the letter directly below it. This will produce a ciphertext. For example, an encrypted code that

says, “Meet me in the library” will look like the following: HZZOHZDIOCZGDWMVMT. To decipher the code, look for each letter from the ciphertext on the bottom row and replace it with the letter directly above it. The only concern with this technique is that it can easily be solved because there are only 25 other variations of the message. One only needs to write out all 26 and find the one decrypted message that makes sense.

A more advanced kind of encryption technique is called the Vigenere Cipher (13). This cipher uses the same basic principles of substitution similar to Caesar’s method. In this method, there must be a key word to use. Primarily, this is agreed on beforehand by both communicating parties. For this example, the key word shall be “MILK” and the message shall be, “See you soon.” We begin by writing the word “MILK” across the top of the message:

M I L K M I L K M I

S E E Y O U S O O N

To encrypt this message, we must use a chart called the Vigenere table. The rows and columns are labelled off by the letters of the alphabet. It should look something like this:

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
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	A
C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G

I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	**E**	F	G	H	I	J	K	L
N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	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

Table 1: The Vigenere Square

Using the two letters in the chart of the secret message and a key word, we can find the intersecting letter in the Vegenere chart. Thus, examining the chart, “M” and “S” intersect at point “E.” The encrypted message would look like the following: “EMPIACDYAV.” The advantage to this technique is that it provides additional variations of decrypted messages. Subsequently, an attacker will have a more strenuous time decoding the message without the key word.

The Caesar cipher and the Vigenere cipher are the simplest encryption techniques. Other great advancements in this science occurred especially during the

years of the World Wars. During the 1930's through the 1940's, the Nazi Germans used an electronic ciphering system, known as Enigma. The Enigma system of Second World War was like a "combination lock with more than 1023 possible combinations" (14). After the "breaking" of an earlier version of Enigma by Marian Rejewski in 1932, the Nazis strengthened their system and decided to change the combination daily, making it more difficult for Enigma to be broken (15). However, a team of British cryptographers led by Alan Turing discovered a way to eliminate many of the incorrect combinations. This discovery narrowed down the 1023 possibilities significantly so that only a handful remained. Other major symmetric ciphers that are being used today are Data Encryption Standard (DES) – earliest version which was developed in 1976, and Advanced Encryption Standard (AES) which came into existence in 2000 in a bid to replace DES because of inherent vulnerability discovered in the design in the 1990s.

However, there is one major flaw in both of these ciphering methods – the distribution of the key. How is it possible to distribute the key so that any attacker can not discover it? This dilemma led to the discovery of the Diffie-Hellman key exchange in the 1970's (16). This ingenious method incorporates mathematics so that a party could distribute the key to another party without the risk of an intruder receiving the information.

3.2 Public Key Cipher System

The general subject matter of modern cryptography as we have mentioned above is communication over a problematic channel. The channel may be insecure in the sense that any communication may be overheard, or insecure in the sense that communications may be arbitrarily altered before reaching their destination, and, in addition, the participants in the communication may not be trustworthy. They may repudiate important warrants at a later time, thus assurances of various sorts would be valuable. Thus, public-key cryptography is the verification and securing of communications over channels that are not secure and cannot be assumed to transmit information reliably and between participants who do not necessarily trust one another.

In traditional cryptography however, such as was available prior to the 1970's, the encryption and decryption operations are performed with the same key. This means that the party encrypting the data and the party decrypting it need to share the same decryption key. Establishing a shared key between the parties is an interesting challenge. If two parties already share a secret key, they could easily distribute new keys to each other by encrypting them with prior keys. But if they don't already share a secret key, how do they establish the first one?

This line of thinking – in pre-Web terminology – prompted two Stanford University researchers, Whitfield Diffie and Martin Hellman, to write a landmark paper, *"New Directions in Cryptography,"* in 1976 (17). The paper suggested that perhaps encryption

and decryption could be done with a pair of different keys rather than with the same key. The decryption key would still have to be kept secret, but the encryption key could be made public without compromising the security of the decryption key. This concept was called public key cryptography because of the fact that the encryption key could be made known to anyone.

Diffie and Hellman also introduced the concept of digital signatures (18). Parties who share a secret key can easily verify the data they have exchanged whether it has not been modified by performing an authentication operation using the key. Unlike for encryption and decryption using public key and private key respectively, in digital signatures, the private key is used to generate the signature i.e., for signing while the public key is used for verification of the signature by the receiving party. Digital signature has similar properties to handwritten signatures – any party can verify the signature provided they have the public key, but only one party with the private key can generate it. In fact, the assurances are stronger than for ordinary signatures, because the signature is dependent on the data itself, unlike ordinary signatures which can potentially be cut-and-pasted among different documents and yet still appear to be valid.

Diffie and Hellman introduced a specific method based on number theory for establishing secret keys by parties who don't previously share a secret. The method is called Diffie-Hellman key agreement (19). The security of the method is related to a

longstanding problem in number theory, discrete logarithms which we shall discuss in chapter 4. The pioneering paper by Diffie and Hellman (17) introduced a new approach to cryptography and, in effect, challenged cryptographers to come up with a cryptographic algorithm that met the requirements for public-key systems. One of the first of the responses to the challenge was developed in 1977 by Ron Rivest, Adi Shamir, and Len Adleman at MIT and first published in 1978 (20). The Rivest-Shamir-Adleman (RSA) scheme based on integer factorization has since that time reigned supreme and is perhaps the most widely accepted and implemented general-purpose approach to public key encryption.

Other forms of public key cryptosystems that have been developed since the 1970's are ElGamal and most recent research areas such as elliptic curve and Quantum cryptography. The ElGamal public key cryptosystem was created by Taher ElGamal (1985), like the Diffie–Hellman key exchange, it is based on the discrete logarithm problem. But, unlike Diffie–Hellman key exchange, it can be used for encryption. There is a form of the ElGamal algorithm that can equally be used for digital signature and is the base for the U.S. government's Digital Signature Standard (DSS) in 1984.

3.2.1 Overview of Elliptic Curve and Quantum Cryptography

Elliptic curves (elliptic curve discrete logarithm problem) were first introduced into cryptography by Miller (1986) and Koblitz (1987) (21). In recent years they have gained widespread interest because of their shorter key size compared to systems such as RSA, their superior efficiency in certain situations, and the smaller bandwidth that they require. These advantages are becoming more compelling as time progresses because of the increased use of smaller and smaller mobile computing devices with associated bandwidth and computational constraints.

Quantum cryptography on the other hand, is the name under which are commonly known those techniques that make use of the laws of quantum mechanics to prevent the unauthorized access to secret information. The most celebrated of such protocols, is quantum key distribution, undoubtedly the most successful quantum information processing protocol from the technological viewpoint. Quantum key distribution is a scheme that allows two remote parties to share a common secret random string of bits even in the presence of an adverse party who tries to eavesdrop. It solves the problem of secure distribution of private cryptographic keys by allowing the detection of any eavesdropping. Quantum key distribution is historically the first quantum information processing protocol. Some of the ideas behind it were implicitly suggested by Stephen Wiesner, who proposed quantum tokens which cannot be forged in his paper – “Conjugate Coding”, which unfortunately took more than ten years to be published (22).

Elliptic curve and Quantum cryptography are current areas of increasing research in modern cryptography.

3.3 Chapter Summary

This chapter covers a brief background of cryptography – from symmetric cryptography to asymmetric cryptography. In the symmetric cryptography we discussed major examples of cipher system such as Caesar cipher, Vegenere cipher and the Enigma machine used in the Second World War by the Germans. The major development of asymmetric cipher system in the 1970's till date was discussed with emphasis on Diffie-Hellman, RSA and Elgamal cipher systems. Current developments in cryptography such as elliptic curve and quantum cryptography were also highlighted. In the next chapter, we shall be discussing the computation of public keys for selected public key cipher system using modular arithmetic that we treated in chapter 2.

CHAPTER 4

ASYMMETRIC KEY GENERATION

Public-key cryptography finds its strongest application when parties who have no prior relationship (and therefore no opportunity to establish shared secret keys) want to exchange sensitive data with each other. Anyone who wants to be a receiver needs to "publish" an encryption key, which is known as the public key, while at the same time generate a unique and secret corresponding decryption key known only to him and known as the private key. Thus anyone with the public key can encrypt a message but the message can only be decrypted by the owner or anyone with the knowledge of the private key. Because the encryption and decryption algorithms for asymmetric ciphers are considerably slower than those for symmetric ciphers, in practice the asymmetric ciphers are used to securely exchange a session key for a symmetric cipher to be used for the actual communication. That is, the only plaintext encrypted with the symmetric cipher is the key for a symmetric cipher, and then the faster-running symmetric cipher is used for encryption of the actual message.

Throughout the 1980s, however, most of the applications that needed to protect data had centralized control. Banking networks and pay-TV systems are typical examples where secret keys could generally be pre-established by a central authority.

Applications that didn't have centralized control – like e-mail – were meanwhile growing without much attention to security. Equally important was the fact that the mathematical operations in public-key cryptography required considerable computational resources relative to computer performance at the time. As a result, public-key cryptography was a slow sell through that first full decade.

With the advent of the World Wide Web in the 1990s, however, the situation changed. Computer performance had by then advanced to the point that the time for the encryption and decryption operations was no longer an issue. Meanwhile, the "killer application" of online purchasing had exactly the characteristics that required public-key cryptography. The Web inherently didn't have central control for security – any merchant could go online without any prior security relationships with anyone else. Thus, there was clearly a need to protect sensitive data – many consumers would not shop online if there were a risk that credit card numbers and order information might be intercepted by an eavesdropper. Public-key cryptography caught on rapidly as a result.

The generation of session keys for messages to be exchanged securely between parties requires the application of mathematics – the modular arithmetic already discussed in chapter 2 will be the base for the key generation algorithms that we shall deal with in subsequent sections in this chapter. Meanwhile, for the purpose of this research work, we shall limit our implementation to three asymmetric cipher systems – RSA, Diffie-Hellman, and ElGamal algorithms.

4.1 RSA Public-key Algorithm

The RSA public key encryption algorithm was the first practical implementation of public key encryption discovered and it remains perhaps the most widely used public key encryption algorithm today (23). RSA public-key algorithm depends on the difficulty of factoring large integers. There are four basic steps involved in setting up and using an RSA public and private key pair.

Step 1: Choose two large primes

Two prime numbers **p** and **q** are chosen by some means.

Step 2: Compute the RSA modulus

This step involves multiplying **p** and **q** together. The resulting number

$$n = p * q$$

is part of the public key. This number **n** will be the modulus that we use when we compute RSA encryptions and decryptions using modular arithmetic.

Step 3: Choose the rest of the public key

A number **e** is selected that is smaller than $(p - 1) * (q - 1)$ and has the important property that

e is coprime to $(p - 1) * (q - 1)$.

Thus, the number e is greater than 1 and less than (p-1)(q-1), and must have the precise mathematical property that there must be no numbers that divide neatly into e and into (p-1)(q-1) except for i.e., gcd[e,(p-1)(q-1)] = 1.

The two numbers **n** and **e** i.e., (n,e) pair together form the public key and can be freely published or distributed to anyone who you want to be able to send encrypted messages to you.

Step 4: Compute the private key d from p, q and e

The private key d is the inverse of e modulo (p – 1) * (q – 1). In other words:

$$d \equiv e^{-1} \bmod (p - 1) * (q - 1).$$

This number d is calculated from e and (p – 1) * (q – 1) using the Euclidean Algorithm (2.3). The private key d must be kept secret by the holder of the key pair and nobody else should learn its value. Thus, an RSA key pair, has (n,e) as the public key and d as the private key.

Let us consider an example of the setup of an RSA key pair.

Example: We shall choose two small prime numbers for ease of computation, p = 29 and q = 41.

Step 1: Our two primes are p = 29 and q = 41

Step 2: Compute the RSA modulus

This step involves multiplying 29 and 41 together. The resulting number

$$n = 29 * 41 = 1189$$

is part of the public key.

Step 3: Choose the rest of the public key

A number e is chosen that is greater than 1 and smaller than (29 – 1) * (41 – 1) = 28 * 40 = 1120 and has the important property that e is coprime to 1120 i.e.

$$\gcd(e,1120) = 1.$$

Let us try and find one that is suitable.

Consider e = 7, is not a valid value for e because 7 and 1120 are not coprime (7 is a common divisor).

Consider e = 8, is not a valid value for e because 8 and 1120 are not coprime (2 is a common divisor).

Consider e = 9, no number other than 1 divides evenly into both 9 and 1120. Thus 9 is coprime to 1120 i.e.

$$\gcd(9,1120) = 1,$$

so we may use e = 9.

Thus, the two numbers 1189 and 9 i.e. (1189,9) together form the public key and can be freely distributed to anyone who you want to be able to send encrypted messages to you.

Step 4: Compute the private key

The private key d is the inverse of 9 mod 1120. In other words:

$$d \equiv 9^{-1} \bmod 1120.$$

This number d is calculated from 9 and 1120 using the Euclidean Algorithm (2.3):

Step I:

$$1120 = 124(9) + 4 \quad \Rightarrow \quad 4 = 1120 - 124(9)$$

$$9 = 2(4) + 1 \quad \Rightarrow \quad 1 = 9 - 2(4)$$

Step II: Backward substitution

$$1 = 9 - 2(4)$$

$$1 = 9 - 2(1120 - 124(9))$$

$$1 = 9 - 2(1120) + 248(9)$$

$$1 = 249(9) - 2(1120)$$

Step III: Reduce this mod 1120 to get

$$1 \equiv 249(9) \bmod 1120$$

So $9^{-1} \equiv 249 \bmod 1120$

Thus our private key is d = 249. The private key 249 must be kept secret by the holder of the key pair and nobody else should know its value.

4.1.1 RSA Encryption and Decryption

RSA Encryption:

To encrypt a plaintext M using an RSA public key, we simply represent the plaintext as a number between 0 and N – 1 and then compute the ciphertext C as:

$$C = M^e \bmod N.$$

RSA Decryption:

To decrypt a ciphertext C using an RSA public key we simply compute the plaintext M as:

$$M = C^d \bmod N.$$

Note that both RSA encryption and RSA decryption involve modular exponentiation (2.4).

Example: Let us consider an example of RSA encryption and decryption using the key pair established in our previous example where public key is given as (1189, 9) and private key as 249.

a) Encryption:

Suppose someone wants to encrypt the plaintext M = 19. We thus have to calculate the ciphertext

$$C \equiv 19^9 \bmod 1189.$$

This is most efficiently calculated using the Repeated Squares Algorithm (2.4.1):

Step I:

$$C \equiv 19^{8+1} \bmod 1189$$

$$C \equiv (19^{8})(19^{1}) \bmod 1189$$

Step II:

$$19^{1} \equiv 19 \bmod 1189$$

$$19^{2} \equiv 19^{2} \equiv 361 \bmod 1189$$

$$19^{4} = (19^{2})^{2} \equiv (361)^{2} = 130321 \equiv 720 \bmod 1189$$

$$19^{8} = (19^{4})^{2} \equiv (720)^{2} = 518400 \equiv 1185 \bmod 1189$$

Step III:

$$C \equiv (19^{8})(19^{1}) \bmod 1189$$

$$\equiv (1185)(19) \bmod 1189$$

$$\equiv 22515 \bmod 1189$$

$$\equiv 1113 \bmod 1189$$

So the ciphertext C is 1113.

b) Decryption

Suppose we now receive this ciphertext C = 1113. To decrypt it we have to calculate:

$$M \equiv 1113^{249} \bmod 1189$$

This is most efficiently calculated using the Repeated Squares Algorithm (2.4.1):

Step I:

$M \equiv 1113^{249} \bmod 1189$

$M \equiv 1113^{128+64+32+16+8+1} \bmod 1189$

$M \equiv (1113^{128})(1113^{64})(1113^{32})(1113^{16})(1113^{8})(1113^{1}) \bmod 1189$

Step II:

$1113^{1} \equiv 1113 \bmod 1189$

$1113^{2} \equiv 1238769 \equiv 1020 \bmod 1189$

$1113^{4} = (1113^{2})^{2} \equiv (1020)^{2} = 1040400 \equiv 25 \bmod 1189$

$1113^{8} = (1113^{4})^{2} \equiv (25)^{2} \equiv 625 \bmod 1189$

$1113^{16} = (1113^{8})^{2} \equiv (625)^{2} = 390625 \equiv 633 \bmod 1189$

$1113^{32} = (1113^{16})^{2} \equiv (633)^{2} = 400689 \equiv 1185 \bmod 1189$

$1113^{64} = (1113^{32})^{2} \equiv (1185)^{2} = 1404225 \equiv 16 \bmod 1189$

$1113^{128} = (1113^{64})^{2} \equiv (16)^{2} \equiv 256 \bmod 1189$

Step III:

$M \equiv (1113^{128})(1113^{64})(1113^{32})(1113^{16})(1113^{8})(1113^{1}) \bmod 1189$

$\equiv (256)(16)(1185)(633)(625)(1113) \bmod 1189$

$\equiv 2137259174400000 \bmod 1189$

≡ 19 mod 1189

So the plaintext M is 19.

This corresponds with what we originally encrypted and so the decryption has been successful.

4.2 Diffie-Hellman Key Exchange Algorithm

Diffie–Hellman key exchange (D–H) is a cryptographic protocol that allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel (24). This key can then be used to encrypt subsequent communications using a symmetric key cipher. The simplest, and original, implementation of the protocol uses the multiplicative group of integers modulo p, where p is prime and g is primitive root mod p. In modular arithmetic, a primitive root modulo n is any number g with the property that any number coprime to n is congruent to a power of g (mod n). That is, if g is a primitive root (mod n), then for every integer α that has gcd(α, n) = 1, there is an integer *k* such that $g^k \equiv \alpha \pmod{n}$, where *k* is called the index of *α*. That is, g is a generator of the multiplicative group of integers modulo n.

Thus in D-H set up, the system parameters required are a prime p and a generator g known already to both parties. Suppose we have the two parties as A and B, the algorithm is illustrated in the following steps:

Step 1: Computation of public key

Party A selects a secret key α and computes a public key:

$$y = g^{\alpha} \bmod p$$

Similarly, party B selects a secret key b and computes a public key:

$$y = g^{b} \bmod p$$

Step 2: Key Exchange

Party A sends g^{α} to B while party B sends g^{b} to A.

Thus both parties A and B can now compute

$$(g^{\alpha})^{b} = (g^{b})^{\alpha} = g^{\alpha b} \bmod p$$

$g^{\alpha b} \bmod p$ is now the shared secret between both parties.

Example: Suppose we have a generator g = 23 with a prime p = 517.

Step 1: Computation of public key

Suppose A chooses a secret key α = 8, then

$$23^{8} \equiv 23^{4+2+1} \bmod 517$$

So, $23^{1} \equiv 23 \bmod 517$

$$23^{2} \equiv 529 \bmod 517 = 12 \bmod 517$$

$$23^{4} = (23^{2})^{2} = 12^{2} \equiv 144 \bmod 517$$

$$23^{8} = ((23^{2})^{2})^{2} \equiv 144^{2} \bmod 517$$

$\equiv 20736 \bmod 517$

$\equiv 56 \bmod 517$

So 56 is a public key.

Similarly, B computes $23^{14} \bmod 517$:

$23^{14} = 23^{8+4+2} = (23^8)(23^4)(23^2) \bmod 517$

Thus, $23^2 = 12 \bmod 517$

$23^4 = (23^2)^2 = 12^2 \equiv 144 \bmod 517$

$23^8 = (23^4)^2 = 144^2 \equiv 20736 \bmod 517 = 56 \bmod 517$

and $23^{14} \bmod 517 = (56)(144)(12) \bmod 517 = 96768 \bmod 517$

$= 89 \bmod 517.$

So, 89 is a public key.

Step 2: Key exchange

Party A sends 56 to B while party B sends 89 to A.

A now computes

$89^8 \bmod 517 = 89^{4+2+1} \bmod 517$

Thus, $89^1 \equiv 89 \bmod 517$

$89^2 \equiv 7921 \bmod 517 = 166 \bmod 517$

$89^4 = 166^2 \equiv 27556 \bmod 517 = 155 \bmod 517$

$89^8 = 155^2 \equiv 24025 \bmod 517 = 243 \bmod 517$

So, $89^8 \bmod 517 = 243 \bmod 517$

Similarly, B computes

$56^{14} = 56^{8+4+2} \equiv (56^8)(56^4)(56^2) \bmod 517$

where $56^1 \equiv 56 \bmod 517$

$56^2 \equiv 3136 \bmod 517 = 34 \bmod 517$

$56^4 = 34^2 \equiv 1156 \bmod 517 = 122 \bmod 517$

$56^8 = 122^2 \equiv 14884 \bmod 517 = 408 \bmod 517$

Thus, $(56^8)(56^4)(56^2) \bmod 517 = (408)(122)(34) \bmod 517$

$= 1692384 \bmod 517$

$= 243 \bmod 517$

or $56^{14} \bmod 517 = 243 \bmod 517$

which corresponds to our earlier result for A, thus both A and B obtains 243 as their secret key. Both A and B can now exchange information using this secret key as their session key with symmetric cipher.

4.3 ElGamal Public Key System

The ElGamal cipher system is a public-key cryptosystem based on the discrete logarithm problem. It consists of both encryption and signature algorithms. The encryption algorithm is similar in nature to the Diffie-Hellman key agreement protocol discussed above. It was described by Taher Elgamal in 1985. ElGamal encryption consists of three components – the key generator, the encryption algorithm, and the decryption algorithm (25). The system parameters consist of a prime p and an integer g, whose powers modulo p generate a large number of elements, as in Diffie-Hellman.

Setting up ElGamal

Let p be a large prime and select a special number g such that g must be a primitive element modulo p (2.1.1). Next choose a private key x, where $1 \leq x \leq p-2$. Then compute public key y from x, p and g as

$$y = g^x \bmod p.$$

ElGamal encryption

The first task is to represent the plaintext as a series of numbers modulo p. Then

1. Generate a random number k, such that $1 \leq k \leq p-2$.
2. Compute two values C_1 and C_2, where

$$C_1 = g^k \bmod p \text{ and } C_2 = My^k \bmod p$$

3. Send the ciphertext C, which consists of the two separate values C_1 and C_2 to the recipient.

ElGamal decryption

1. To decrypt, the recipient uses its private key x to transform C1 into something more useful:

$$C_1^x = (g^k)^x \bmod p$$

$$= (g^x)^k = (y)^k = y^k \bmod p \qquad (1)$$

2. Divide C_2 by (1) to get M as follows:

$$C_2/y^k = (My^k)/y^k = M \bmod p$$

which gives the plaintext M.

Example: Suppose we wish to send the message, M = 10 to B, and suppose B has chosen a secret key x = 6, with primitive element g = 11 and p =23.

This means that B's public key is

$$y = g^x \bmod p = 11^6 \bmod 23$$

$$= (11^3)^2 \bmod 23 \equiv 9 \bmod 23 = 9$$

Thus, 9 is the public key, and 6 is the private key.

To encrypt M = 10, we choose a random value, k = 3; $1 \le k \le p-2$.

Compute $C_1 = g^k \bmod p = 11^3 \bmod 23 = 20$

and $C_2 = My^k \bmod p = 10(9^3) \bmod 23$

$= 10(16) \bmod 23 = 160 \bmod 23 = 22$

Thus, ciphertext C = $(C_1, C_2) = (20, 22)$

To decrypt C = (20, 22)

1. Compute $C_1^x = 20^6 = 16 \bmod 23$
2. Compute $C_2/y^k = 22/16 = 22(16^{-1}) \bmod 23 = 22(13) \bmod 23$

$= 286 \bmod 23 = 10$

Hence, plaintext M = 10.

4.4 Digital Signature Algorithm (DSA)

Digital Signature Standard is a United States Federal Information Processing Standards Publications (FIPS PUBS) issued by the National Institute of Standards and Technology (NIST). The Standard is based on ElGamal public-key system and it specifies a Digital Signature Algorithm (DSA) appropriate for applications requiring a digital rather than written signature. The DSA digital signature is a pair of large numbers represented in a computer as strings of binary digits (26). The digital signature is computed using a set of rules (i.e., the DSA) and a set of parameters such that the identity of the signatory and integrity of the data can be verified. The DSA provides the capability to generate and verify signatures. Signature generation makes use of a

private key to generate a digital signature, while signature verification makes use of a public key which corresponds to, but is not the same as, the private key. Each user possesses a private and public key pair and the public keys are assumed to be known to the public in general. Private keys are never shared and it is known to the owner only. Anyone can verify the signature of a user by employing that user's public key, while signature generation can be performed only by the possessor of the user's private key.

The European Community Directive on electronic signatures refers to the concept of an electronic signature as (27):

> Data in electronic form attached to, or logically connected with, other electronic data and which serves as a method of authentication.

The European Community Directive on electronic signatures also refers to the concept of an advanced electronic signature as an electronic signature that is:

1. Uniquely linked to the signatory
2. Capable of identifying the signatory
3. Created using means under the sole control of the signatory
4. Linked to data to which it relates in such a way that subsequent a change in the data is detectable.

DSA set up

The DSA makes use of the following parameters:

1. p is a prime modulus.
2. q is a prime divisor of p – 1.
3. $g = h^{(p-1)/q} \bmod p$, where h is any integer with $1 < h < p - 1$ such that

   $h^{(p-1)/q} \bmod p > 1$ (g has order q mod p)
4. x is a randomly or pseudo-randomly generated integer with $0 < x < q$
5. Compute public key,

   $$y = g^x \bmod p$$
6. k = a randomly or pseudo-randomly generated integer with $0 < k < q$

The integers p, q, and g can be public and can be common to a group of users. A user's private and public keys are x and y, respectively. Parameters x and k are used for signature generation only, and must be kept secret while parameter k must be regenerated for each signature.

Signing with DSA

To sign message m, the following steps are involved:

1. Hash message m to give h(m); $1 \leq h(m) \leq q-1$
2. Generate random secret k; $1 \leq k \leq q-1$
3. Compute $r = (g^k \bmod p) \bmod q$
4. Compute $k^{-1} \bmod q$
5. Compute $s = k^{-1}\{h(m) + xr\} \bmod q$
6. Signature on message, m is (r,s)

7. The signed message is {m, (r,s)}

Note: h(m) is pronounced as hash of message, m and it is process whereby the message is reduced to a numerical value.

DSA Signature Verification

To verify that (r,s) is a signature for message, m:

1. Check that $1 \leq r \leq q-1$ and $1 \leq s \leq q-1$
2. Compute $w = s^{-1} \bmod q$
3. Compute $u_1 = wh(m) \bmod q$ and $u_2 = rw \bmod q$
4. Accept signature if

$$g^{u1}y^{u2} \bmod p) \bmod q = r$$

Example: Let the parameters be p = 29, q = 7 with g = 16

a) To Sign:

1. Choose a random secret key, x = 3 (say)
2. Compute public key, $y = g^x = 16^3 = 7 \bmod 29$
3. Suppose the message to be signed gives h(m) = 5
4. Generate a random k = 6 (say)
5. Compute $r = (g^k \bmod p) \bmod q = (16^6 \bmod 29) \bmod 7$

$$= 20 \bmod 7 = 6.$$

6. Compute $k^{-1} \bmod 7 = 6^{-1} \bmod 7 = 6$

7. Compute $s = k^{-1}\{h(m) + xr\} \bmod q = 6\{5 + (3*6)\} \bmod 7 = 5$.

b) **To Verify:** the verifier wishes to verify that (6,5) is the correct signature on message, m.

1. Recover the public key, y = 7.
2. Compute h(m) which we assume to be 5.
3. We know that (r,s) = (6,5)
4. Compute $s^{-1} \bmod q = 5^{-1} = 3 \bmod 7$.
5. Compute $u_1 = 3(5) = 1 \bmod 7$ and $u_2 = 6(3) = 4 \bmod 7$
6. Compute $g^{u1}y^{u2} \bmod p) \bmod q = ((16^1*7^4) \bmod 29) \bmod 7$

$= 20 \bmod 7 = 6 = r$

Thus, the signature is verified as r = 6.

4.5 Chapter Summary

The cryptographic algorithms for the selected public-key cryptosystems were discussed and implemented using the knowledge we gained in modular arithmetic in chapter 2. The algorithms considered are RSA, Diffie-Hellman, ElGamal and Digital Signature Algorithm (DSA). In RSA and ElGamal we discussed their algorithm set up, encryption, decryption and examples to demonstrate the practicability. In Diffie-Hellman, we covered the algorithm set up, and key exchange algorithm with practical example.

Finally, Digital Signature Algorithm was discussed with an illustration of the set up with signature signing and verification algorithms. This is equally followed by a practical demonstration of the use of the Digital Signature Standard. The next chapter will be devoted to conclusion and recommendations for future reading.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

Even though the cryptography has existed for thousands of years, its development was slow and unyielding and highly classified, and it was restricted to government use particularly the military. The need to communicate secretly had never been considerably threatened until the invention of computers and high-tech machines. In today's society, one can discover all the secrets of another with the flick of one finger. The restriction on the practice, development and use of cryptography was however relaxed by governments due to the emergence of electronic commerce – this made it possible for companies and research institutions to invest in its development. There is no doubt why cryptography flourished during the twentieth century. With the emergence of public-key cryptography, cryptography now incorporates mathematics as part of the shield for protection of privacy and as the world's demand for secrecy increases, the need for cryptography and mathematics will continue to grow and progress.

In this research work, we have identified a major area in number theory – modular mathematics that has contributed immensely to the development and study of public key cryptography. Cryptographic keys are the main drivers of public key cryptosystem and the generation of these keys – private and public keys is carried out by the

application of modular mathematics. We discussed modular mathematics and the relevant properties required for the understanding of the mathematics behind the public key generation algorithms and equally gave a brief history and developments that have taken place since the doors of cryptography was open to the public. A major deviation from the past has been the development of public key cryptography that was pioneered by the work of Diffie and Hellman known as Diffie-Hellman key exchange algorithm based on discrete logarithm. Other works that followed which we also covered are RSA, ElGamal, Digital Signature Algorithm based on ElGamal that forms the basis for the Digital Signature Standard (DSS) made by the US government. The area of increasing research in modern cryptography – elliptic curve and quantum cryptography were highlighted and we hope that interested readers would consult a good text on these subjects for further reading.

We believe that our illustration and practical examples in the public and private key generation and usage for encryption and decryption or key exchange would be found interesting to those studying mathematics at undergraduate and graduate levels, and would further stimulate interest in those pure areas of mathematics.

5.1 Recommendations

As we have mentioned in our last paragraph above, the current areas of increasing research in modern cryptography is elliptic curve and quantum cryptography. Elliptic curve is based on discrete logarithm that relies on the elliptic curve discrete logarithm problem for the security of online communication – this is a very different and currently much harder problem. We hope our studies here will stimulate interested readers and students to venture into the mathematical application and contribution of elliptic curves to the further development of secure electronic commerce, and online communication in general.

Bibliography

1. **Menezes, Alfred J., Van Oorschot, Paul C. and Vanstone, Scott A.** *Handbook of Applied Cryptography.* Florida : CRC Press LLC, 1997.
2. **Kahn, David.** *The Code Breakers.* New York : Scribner, 1996.
3. **Singh, Simon.** *The Code Book: The secret history of codes and code-breaking.* London : Fourth Estate, 1999.
4. **Levy, Stephen.** *Crypto.* New York : Pengiun Books, 2001.
5. **Bidgoli, Hossein (ed.).** *Handbook of Information Security.* New Jersey : John Wiley & Sons, 2006.
6. **Schneier, Bruce.** *Applied Cryptography.* New York : John Wiley & Sons, 1996.
7. **Stallings, William.** *Cryptography and Network Security Principles and Practices.* Fourth. New Jersey : Prentice Hall, 2005.
8. **Garrett, Paul.** *Making, Breaking Codes: An Introduction to Cryptology.* New Jersey : Prentice Hall, 2001.
9. **Hard, G. H. and Wright, E. M.** *An introduction to the theory of numbers .* fifth edition. Oxford, UK : Oxford University Press, 1980.
10. **Koblitz, N.** *A course in number theory and cryptography.* New York : Springer-Verlag, 1987.
11. **Rosen, K. H.** *Elementary number theory and its applications.* fourth edition. Reading : Addison-Wesley, 2000.
12. **Niven, I., Zuckerman, H. S. and Montgomery, H. L.** *An introduction to the theory of numbers.* fifth edition. New York : John Wiley & Sons, 1991.
13. **Piper, Fred and Murphy, Sean.** *Cryptography: A very short introduction.* New York : Oxford University Press, 2002.
14. The Code War. *Beyond Discovery.* [Online] 2003. [Cited: 5 December 2009.] http://www.beyonddiscovery.org/content/view.article.asp?a=3420..
15. **Harris, Robert.** *Enigma.* London : Arrow, 1996.

16. Primes, modular arithmetic & public-key cryptography. *Cornell.* [Online] 2004. [Cited: 6 December 2009.] http://www.math.cornell.edu/~mec/modules/cryptography/diffiehellman/diffiehellman.html.

17. New Directions in Cryptography. *Purdue.* [Online] 1976. [Cited: 13 December 2009.] http://www.cs.purdue.edu/homes/ninghui/courses/Fall04/lectures/diffie-hellman.pdf.

18. **Oppliger, Rolf.** *Internet and internet security.* 2nd edition. Norwood, MA : Artech House, 2002.

19. **Stallings, William.** *Network security essentials: Applications and standards.* 3rd edition. New Jersey : Pearson Prentice Hall, 2007.

20. **Rivest, R.L., Shamir, A. and Adleman, L.** A method for obtaining digital signatures and public-key cryptosystems. *MIT.* [Online] 1978. [Cited: 15 December 2009.] http://people.csail.mit.edu/rivest/Rsapaper.pdf.

21. Elliptic Curve Cryptography. *rhul.* [Online] 11 June 2009. [Cited: 15 December 2009.] http://www.isg.rhul.ac.uk/~sdg/ecc.html.

22. **Brassard, Gilles.** A bibliography of quantum cryptography. *Mcgill.* [Online] 1996. [Cited: 17 December 2009.] htt:p://www.cs.mcgill.ca/~crepeau/CRYPTO/Biblio-QC.html.

23. RSA cryptosystem and its applications. *Electronic procedings of the ICTCM.* [Online] 2006. [Cited: 10 January 2010.] http://archives.math.utk.edu/ICTCM/i/17/C052.html.

24. What is Diffie-Hellman? *RSA Laboratories.* [Online] [Cited: 11 January 2010.] http://www.rsa.com/rsalabs/node.asp?id=2248.

25. A public key cryptosystem and a signature scheme based on discrete logarithms. *MIT.* [Online] [Cited: 15 January 2010.] http://www.groups.csail.mit.edu/cis/crypto/classes/6.857/papers/elgamal.pdf.

26. Fact sheet on digital signature standard. *NIST.* [Online] May 1994. [Cited: 16 January 2010.] http://www.nist.gov/public_affairs/releases/digsigst.htm.

27. The european commission's directive on electronic signatures: Technological "favouritism" towards digital signatures. [Online] [Cited: 16 January 2010.] http://www.bc.edu/bc_org/avp/law/lwsch/journals/bciclr/24_1/04_FMS.htm.

28. **Harris, Robert.** *Enigma.* London : Arrow, 1996.