Algorithms Research 2012, 1(5): 31-42
DOI: 10.5923/j.algorithms.20120105.01

Cryptography: A Comparison of Public Key Systems

Tzvetalin S. Vassilev*, Andrew Twizell

Department of Computer Science and Mathematics, Nipissing University, North Bay, Ontario, P1B 8L7, Canada

Abstract Exchanging information electronically in a secure way become very important in the age of computers. Over fifty years of research and implementation were devoted to developing methods and systems for such a secure communication. These systems are commonly known as cryptosystems. Among the cryptosystems currently in use, the public key cryptosystems have a special place. These are cryptosystems which allow any user, completely unaware of the methods of encryption and decryption, i.e. of the intricate details of the system, to use them for variety of purposes in their daily life. The most common examples of this are digital signatures and electronic banking transactions. Here we provide an overview of different cryptosystems, including but not limited to an examination and comparison of five different influential public key cryptosystems. This examination includes an in-depth look at: the RSA algorithm, the Diffie-Hellman key exchange protocol, the elliptic curve cryptosystems, the ElGamal encryption method and the knapsack approach. We provide the necessary mathematical and number theoretic preliminaries. Further, we introduce each cryptosystem starting with the relevant definitions, theorems and properties. We discuss the encryption and the decryption process, the creation and exchange of the keys, present plenty of examples, as well as discussion of the implementation issues and the known strengths and weaknesses of each cryptosystem against cryptanalytic attacks.

Keywords Cryptography, Public Key Cryptosystems, Algorithms, Elliptic Curves, RSA, Knapsack

1. Introductory Mathematics

Cryptography is the practise of creating techniques that allow the secure communication among two parties in the presence of eavesdroppers. Cryptography is a very old science dating back hundreds of years where its origins began with the very basic transposition ciphers, which would rearrange letters in words to form something else. Today it has become much more scientific with the design of cryptographic algorithms designed to be infeasible for an attacker to decipher. The purpose of the paper is to discuss the use of public key cryptosystems, describe and compare five major types of these systems. The five systems that will be discussed include: the RSA Cryptosystem, the Diffie-Hellman Key Exchange Protocol, the ElGamal Encryption Method, the Knapsack approach and the use of Elliptic Curves in Cryptosystems. Before these methods are discussed in detail, some introductory mathematics must first be established.

1.1. Greatest Common Divisor

The first fundamentals of number theory that will be discussed are the concepts of greatest common divisor, divides and relatively prime. Consider the following definitions:

Definition 1.1.1: Let a and b be two integers. We say that a divides b if there exists an integer k such that $b = ak$. We denote a divides b as: $a|b$

Definition 1.1.2:

Let a and b be two integers. The greatest common divisor (gcd) of a and b, denoted by (a,b), satisfies the following two properties:

- $(a, b) \mid a$ and $(a, b) \mid b$
- If $d \mid a$ and $d \mid b$ then $d \mid (a, b)$

The concepts of divisibility and gcd are of the utmost importance when working with cryptosystems, specifically the ones mentioned in this paper.

Definition 1.1.3: Let a and b be two integers. a and b are said to be relatively prime if and only if the only positive integer that divides both is 1, that is the gcd(a, b) = 1

1.2. Congruence Modulo *n*

The idea of division with remainders is one of the key components of most public key cryptosystems. Consider the following definition:

Definition 1.2.1: Let a and b be two integers. We say that there is a remainder after division if there exist integers k and r such that $b = ak + r, 0 \leq r < a$ when r is the remainder.

This definition ties perfectly into the idea of congruence modulo n, as defined by the following definition.

Definition 1.2.2: Let a and b be two integers. a and b are said to be congruent modulo n. written: $a \equiv b \ (mod\ n)$,

* Corresponding author:
tzvetalv@nipissingu.ca (Tzvetalin S. Vassilev)

Published online at http://journal.sapub.org/algorithms

if they leave the same remainder on division by n. Equivalently, a is congruent to b, mod n if n divides $a - b$.

It can also be said that a and b belong to the same congruence class, mod n.

Example 1.2.1: Find three integers that are congruent to 5 (mod 13).

Solution:

By definition of congruence we need to find three integers a, b, c that are all congruent to 5 (mod 13).

That is:

$$a \equiv 5\ (mod\ 13)$$
$$b \equiv 5\ (mod\ 13)$$
$$c \equiv 5\ (mod\ 13)$$

By definition of congruence mod n, 13 $|(a - 5)$, Thus we can notice we have: $13k = a - 5$, thus $a = 13k - 5$.

Set $k = 0, 1, 2$. We get $a = 5, b = 18, c = 31$. Thus we have,

$$5\ (mod\ 13) \equiv 18\ (mod\ 13) \equiv 31\ (mod\ 13)$$

All these values belong to the same congruence class mod 13.

1.3. Extended Euclidean Algorithm

Using **Definition 1.2.2** Euclid developed an algorithm to find the greatest common divisor by repeatedly subtracting the smaller number from the larger number. More precisely, his algorithm goes as follows:

Suppose that $a > b$ and let $a_1 = a, b_1 = b$.

Then for each pair (a_i, b_i) we form the pair (a_{i+1}, b_{i+1}), where $a_{i+1} = max(b_i, a_i - b_i), b_{i+1} = min(b_i, a_i - b_i)$.

Since this algorithm produces smaller and smaller pairs, eventually it must end and we notice that we will get: $a_k = b_k$ in which the case we conclude that $gcd(a, b) = a_k = b_k$.

Arguably the most important consequence of the Euclidean algorithm is that we have the following proposition

Proposition 1.3.1: Let a and b be integer values we can see from the Euclidean Algorithm that we have:

$gcd(a, b) = ma + nb$ for some integers m and n.

Example 1.3.1: Find the $gcd(34, 19) = gcd\ (a, b)$ in the form $ma + nb$.

Solution: We must begin by using division with remainder, subtracting the appropriate multiple of the second number from the first to get the remainder at each step. We have:

$$(34, 19) = (a, b)$$
$$(19, 15) = (b, a - b)$$
$$(15, 4) = (a - b, b - (a - b)) = (a - b, -a + 2b)$$
$$(4, 3) = \left(-a + 2b, a - b - 3(-a + 2b)\right)$$
$$= (-a + 2b, 4a - 7b)$$
$$(3, 1) = (4a - 7b, -a + 2b - (4a - 7b))$$
$$= (4a - 7b, -5a + 9b)$$

We can see that we now have, $1 = -5a + 9b$.

We can try this out by substituting in our original values for a and b, we have $1 = -5(34) + 9\ (19) = 1$

The Euclidean algorithm is extremely important in practice and theory. It is useful in practice because it is unusually fast, you can get the gcd of a k-digit number in around k steps, which is much faster than any known algorithm for finding divisors of a k-digit number.

1.4. Fast Exponentiation

The next fundamental mathematical algorithm we must discuss is the process of computing large powers in a monoid G. This algorithm and its variants are central ingredients of many cryptographic protocols. We must begin by letting $g \in G$ and e be a positive integer. Let

$$e = \prod_{i=1}^{k} e_i 2^i$$

Be the binary expansion of e. Observe that the coefficients e_i are either 0 or 1. Therefore,

$$g^e = g^{\sum_{i=0}^{k} e_i 2^i} = \prod_{i=1}^{k} (g^{2i})^{e_i} = \prod_{0 \le i \le k, e_i = 1} g^{2^i}$$

From this formula, we obtain the following idea for computing g^e:

1. Compute the successive squares $g^{2^i}, 0 \le i \le k$.

2. Determine g^e as the product of those g^{2^i} for which $e_i = 1$.

Observe that $g^{2^i+1} = (g^{2^i})^2$.

Therefore, g^{2^i+1} can be computed from g^{2^i} by squaring only once. Here is an example of a situation where this method is very useful.

Example 1.4.1: Determine the value of 6^{73} (mod 100).

Solution: Consider

$$6^1\ (mod\ 100) \equiv 6\ (mod\ 100)$$
$$6^2\ (mod\ 100) \equiv 36\ (mod\ 100)$$
$$6^4\ (mod\ 100) \equiv 36^2\ (mod\ 100) \equiv 96\ (mod\ 100)$$
$$6^8\ (mod\ 100) \equiv 96^2\ (mod\ 100) \equiv 16\ (mod\ 100)$$
$$6^{16}\ (mod\ 100) \equiv 16^2\ (mod\ 100) \equiv 56\ (mod\ 100)$$
$$6^{32}\ (mod\ 100) \equiv 56^2\ (mod\ 100) \equiv 36\ (mod\ 100)$$
$$6^{64}\ (mod\ 100) \equiv 36^2\ (mod\ 100) \equiv 96\ (mod\ 100)$$
$$6^1 * 6^8 * 6^{64}\ (mod\ 100) \equiv 6 * 16 * 96\ (\ mod\ 100)$$
$$\equiv 96^2\ (mod\ 100) \equiv 16\ (mod\ 100)$$

Note that using fast exponentiation only 6 squares were calculated and then the product of three integers (mod 100), compared to computing $6 * 6 * 6 * ... * 6$ (with 73 calculations) then taking the modulus 100 of that value, this is a much easier computation.

2. RSA Cryptosystem

The first public key cryptosystem invented was developed and named after its creators Ron Rivest, Adi Shamir and Len Adleman[9]. The RSA cryptosystem is a relatively strong security, and what makes it rather impressive is that since the system has been implemented in 1978 it still, to this day has not been broken[11]. To break the system a user must be able to factor a very large composite number which is the product of two large prime numbers.

The RSA system is built on basic number theory, more specifically on (+, *) arithmetic modulo n, and on Fermat's little theorem as generalized by Euler. The whole system works due to three simple properties of integers, well known by theoreticians:

- It is difficult for a computer to factor a large number
- It is easy for a computer to construct large prime numbers
- It is easy for a computer to decide whether a given large number is prime

2.1. Generation of the Public Key

To generate the public key we must begin with a few simple rules. Let us generate p, and q to be two large distinct prime numbers. Now we must find the product of p and q, and let n be this product, that is: $n = pq$

Now determine an integer e such that,

$$1 < e < \varphi(n) = (p-1)(q-1)$$

Where e is relatively prime to $\varphi(n)$, that is $gcd\,(e,(p-1)(q-1)) = 1$. It can be noted that e will always be odd, since $\varphi(n)$ is always even, and if e was even then by definition of greatest common divisor, $gcd(e,\varphi(n)) \neq 1$. We must now compute an integer d, such that: $1 < d < \varphi(n)$ where $de \equiv 1\,(mod\,((p-1)(q-1)))$.

The existence of this integer d is known from the simple fact that $gcd\,(e,(p-1)(q-1)) = 1$. Using the extended Euclidean algorithm we can determine a value d for every public key (n,e) if the primes p and q are known. We call the value e the encryption exponent and our value d the decryption exponent.

Example 2.1.1: Develop values p, q and use them to calculate the value n, and find encryption and decryption exponents that work with these values.

Solution: Let our prime numbers be $p = 13, q = 17$. We can easily compute that $n = pq = (13)(17) = 221$.

It can easily be noticed that our $\varphi(n) = \varphi(221) = (13-1)(17-1) = 192$.

Next we must determine an integer e such that $1 < e < \varphi(n) = 192$, and e is relatively prime to $\varphi(n)$. We can use $e = 7$ for this. It can easily be seen that $gcd\,(e,\varphi(n)) = 1$ (which is the definition of relatively prime), and now we can use Euclid's algorithm to find d:

$$192 = 7 * 27 + 3$$
$$7 = 3 * 2 + 1$$

Working backwards, we can see that we have:

$$1 = 7 - 3 * 2 = 7 - (192 - 7 * 27) * 2$$
$$= 7 * 55 - 192 * 2 = 1$$

It can be noticed from this that we have: $7 * 55 \equiv 1\,(mod\;192)$, thus giving us $d = 55$.

2.2. Encrypting a Message Using the Public Key

The encrypting of an integer is very simple using the RSA cryptosystem. Let our message that we wish to encrypt be called m, note that any $m \in \{0,1,2,\ldots,n-1\}$ where m is relatively prime to n. To encode m using our public key (n,e), we must compute: $c \equiv m^e\;(mod\;n)$, where c is our desired ciphertext.

We are now able to safely send our encoded value c to the receiver who has access to the decryption key d. For smaller values of $c \equiv m^e\;(mod\;n)$ we can use the method discussed in Section 1.4 for fast exponentiation.

Example 2.2.1: Encrypt the simple integer $m = 126$ using the values computed in Example 2.1.1 so the message m is secure and cannot be decrypted without access to the decryption key d.

Solution: We know from Example 2.1.1, we have the following values: $p = 13, q = 17, n = 221$, where $\varphi(n) = \varphi(221) = (13-1)(17-1) = 192$. We chose our value $e = 7$ and calculated $d = 55$.

Using the given integer $m = 126$, we must use the public key $(221, 7)$ to calculate:

$$c \equiv m^e\;(mod\;n) = 126^7\;(mod\;221)$$

It can easily be seen that:

$$126^2 = 15876\;(mod\;221) \equiv 185\;(mod\;221)$$
$$126^4 = (15876)2 \equiv 185^2\;(mod\;221)$$
$$\equiv 191\;(mod\;221)$$

It can be seen that $126^7 = 126^4 * 126^2 * 126^1 \equiv 191 * 185 * 126\;(mod\;221) \equiv 165\;(mod\;221)$. It can be noted that our value for $c = 165$.

2.3. Decrypting a Message Using the Decryption Key

The decryption of RSA is based on the following theorem:

Theorem 2.3.1: Let (n,e) be a Public RSA key and d the corresponding private RSA key.

Then: $(m^e)^d\;mod\;n = m$ for any integer m with $0 \leq m < n$.

Proof: We know that $ed \equiv 1\;mod\;(p-1)(q-1)$ by the definition of d, thus there must exist an integer t such that $ed = 1 + (t(p-1)(q-1))$.

Therefore

$$(m^e)^d = m^{ed} = m^{1+t(p-1)(q-1)} = m(m^{(p-1)(q-1)})^t$$

It follows that: $(m^e)^d = m(m^{(p-1)})^{(q-1)t} \equiv m\;(mod\;p)$

If p is not a divisor of m, then this congruence follows from Fermat's little theorem (If $gcd\,(a,m) = 1$, then $a^{\varphi\,(m)} \equiv 1\;(mod\;m)$). Otherwise, the assertion is trivial because both sides of the congruence are 0 (mod p). Analogously, we see that $(m^e)^d \equiv m\;(mod\;q)$

Because p and q are distinct prime numbers, we obtain

$$(m^e)^d \equiv m\;(mod\,n)$$

The assertion follows from the fact that $0 \leq m < n$. This concludes the proof.

Corollary 2.3.2: When the message m is encrypted by using $c \equiv m^e\;(mod\;n)$ where (n,e) is the public key, it follows that the ciphertext c can be decrypted by the following equation: $m \equiv c^d\;(mod\;n)$

Thus using Corollary 2.3.2 we can now decrypt our ciphertext to find the original message m.

Example 2.3.1: Decrypt the cipher text calculated in Example 2.2.1 using the values computed in Example 2.1.1 so the message m is recovered properly.

Solution: We know from Examples 2.1.1 and 2.2.1 we have the following values: $p = 13, q = 17, n = 221$ where $\varphi(n) = \varphi(221) = (13-1)(17-1) = 192$. We chose our value $e = 7$ and calculated $d = 55, c = 165$.

Using Corollary 2.3.2, we can now decrypt our ciphertext $c = 165$. We have: $m \equiv c^d (mod\ n) \equiv 16555 (mod\ 221)$.

Using the fast exponentiation as described in Section 1.4, we can see that:

$$165^2 \equiv 27225\ (mod\ 221) \equiv 42\ (mod\ 221)$$
$$165^4 \equiv (42)^2\ (mod\ 221) \equiv 217\ (mod\ 221)$$
$$165^8 \equiv (217)^2\ (mod\ 221) \equiv 16\ (mod\ 221)$$
$$165^{16} \equiv (16)^2\ (mod\ 221) \equiv 35\ (mod\ 221)$$
$$165^{32} \equiv (35)^2\ (mod\ 221) \equiv 120\ (mod\ 221)$$

It can be noted that

$165^{55} = 165^{32} * 165^{16} * 165^4 * 165^2 * 165^1 \equiv 120 * 35 * 217 * 42 * 165\ (mod\ 221) \equiv$
$217 * 42 * 165\ (mod\ 221) \equiv 53 * 165\ (mod\ 221) \equiv$
$126\ (mod\ 221)$.

Thus we have decrypted our original message and found the desired result of $m = 126$.

2.4. Security of the RSA Cryptosystem

The RSA cryptosystem is a public key system, thus to consider this a legitimate cryptosystem we must first consider the security of the system itself. More specifically we must consider the security of the secret key d.

It can be noticed that for an attacker to compute the secret key directly, they will need to know the prime factorization of n, which gives them p and q. They use p and q with the public key to generate the value of d (by solving the congruence: $de \equiv 1\ mod\ ((p-1)(q-1))$), and decrypt any stolen message. The converse of this statement is also true, that it is possible to compute the values of p and q from the public key and the secret key. To show this, we must state the following lemma and theorems. First, we let

$S = max\ \{ t \in \boldsymbol{N} : 2t|(ed - 1)\}$ and $k = (ed - 1)/2^s$.

Theorem 2.4.1: Let $g \in G$ and $e \in \boldsymbol{Z}$. Then $g^e = 1$ if and only if e is divisible by the order of g in G.

Proof: Let $n = |g|$ (the order of g). If $e = kn$, then:

$$g^e = g^{kn} = (g^n)^k = 1^k = 1$$

Conversely, let $g^e = 1$ and $e = qn + r$ with $0 \leq r < n$. Then: $g^r = g^{e-qn} = g^e (g^n)^{-q} = 1$

Because n is the least positive integer with $g^n = 1$, and since $0 \leq r < n$, we have $r = 0$ and therefore $e = qn$. Hence, n is a divisor of e, as asserted. This concludes the proof.

Lemma 2.4.2: For all integers a that are relatively prime to n, the order of the residue class $a^k + n\boldsymbol{Z}$ in the group $(\boldsymbol{Z}/n\boldsymbol{Z})^*$ is in $\{2^i : 0 \leq i \leq s\}$.

Proof: Let a be an integer that is relatively prime to n. By Euler's theorem we have $a^{ed-1} \equiv 1\ (mod\ n)$. Since $ed - 1 = k2^s$, this implies $(a^k)^{2^s} \equiv 1\ (mod\ n)$. Hence, by Lagrange's theorem the order of $a^k + n\boldsymbol{Z}$ is a divisor of 2^s. This concludes the proof.

Theorem 2.4.3: Let a be an integer that is relatively prime to n. If the orders of the residue class $a^k + p\boldsymbol{Z}$ in $(\boldsymbol{Z}/p\boldsymbol{Z})^*$ and for $a^k + q\boldsymbol{Z}$ in $(\boldsymbol{Z}/q\boldsymbol{Z})^*$ are different, then $1 < gcd\ (a^{2^t k} - 1, n) < n$ for some $t \in \{0, 1, 2, \ldots, s-1\}$.

Proof: By Lemma 2.4.2, the order of $a^k (mod\ p)$ and $a^k (mod\ q)$ is in $\{2^i : 0 \leq i \leq s\}$. Without loss of generality, the order of $a^k (mod\ p)$ is greater than the order of $a^k (mod\ q)$. Let the order of $a^k (mod\ q)$ be 2^t. Then $t < s, a^{2^t k} \equiv 1\ (mod\ q)$ but $a^{2^t k} \neq 1\ (mod\ p)$. Therefore, $gcd\ (a^{2^t k} - 1, n) = q$. This concludes the proof.

Thus we can finally factor n, proceeding with the following algorithm:

1. Choose at random an integer a in the set $\{1, 2, \ldots, n-1\}$
2. Compute $g = gcd(a, n)$
3. If $g = 1$, then compute $g = gcd(a^{2^t k} - 1\ (mod\ n), n)$ for $t = s-1, s-2, \ldots$ until g > 1 or t = 0.
4. If $g > 1$, then $g = p$ or $g = q$. Hence, the factorization of n is found and the algorithm terminates. Otherwise, the algorithm was unsuccessful with the chosen base.

If the algorithm is not successful with the chosen a, then we run it again, therefore choosing a different base and starting over.

Example 2.4.1: Let's consider the following case:

Let our prime numbers be $p = 11, q = 23$. We can easily compute that $n = pq = (11)(23) = 253$.

It can easily be noticed that our $\varphi(n) = \varphi(253) = (11-1)(23-1) = 220$.

Next we must determine an integer e such that $1 < e < \varphi(n) = 220$, and e is relatively prime to $\varphi(n)$. We can use $e = 3$ for this. It can easily be seen that $gcd\ (e, \varphi(n)) = 1$ (which is the definition of relatively prime). We can use Euclid's algorithm to find d:

$$220 = 3 * 73 + 1$$
$$3 = 3 * 1 + 0$$

Working backwards, we can see that we have:

$$1 = 220 - 3 * 73 = 3 * 147 - 220 * 2$$

It can be noticed from this that we have: $3 * 147 \equiv 1\ (mod\ 220)$, thus giving us $d = 147$.

Consider a situation where an attacker would like to factor n to find the primes p and q. They must know $n = 253, e = 3$ and $d = 147$.

Solution: It can be easily noted that $ed = 441 - 1 = 440$. For the purpose of the algorithm we must select a base a in the set $\{1, \ldots, n-1\}$. For simplicity reasons, let us use $a = 2$. We now can compute $g = gcd\ (a, n) = gcd\ (2, 253) = 1$. Since our $g = 1$, we can move on to step 3, we must calculate $g = gcd\ (a^{2^t k} - 1\ (mod\ n), n)$ for $t = s-1, s-2, \ldots$ until $g > 1$ or $t = 0$.

Thus we have: $t = s - 1$, this implies that

$$2^t k = \frac{(ed-1) * 2^t}{2^s} = \frac{ed-1}{2} = \frac{440}{2} = 220.$$

But $gcd\ (2^{220} - 1, 253) = 253$.

So we try: $t = s - 2$, this implies that

$$2^t k = \frac{(ed-1) * 2^t}{2^s} = \frac{ed-1}{4} = \frac{440}{4} = 110.$$

But $gcd\ (2^{110} - 1, 253) = 253$.

So we try: $t = s - 3$, this implies that

$$2^t k = \frac{(ed-1)*2^t}{2^s} = \frac{ed-1}{8} = \frac{440}{8} = 55.$$

But $gcd\ (2^{55} - 1, 253) = 23$.

Taking $253/23 = 11$, thus the attacker has found our $p = 11$ and $q = 23$.

Now that we have considered theoretical uses to the RSA Cryptosystem, we will consider a more practical use for the system.

Example 2.4.2: A company approaches you to create a secure online ordering system. Your task is to create a system to securely transfer customer's credit card information.

Solution: It can be noted that every credit card number is 16 digits long, with 4 digits describing its expiry date, giving us 20 digits in total.

For the purpose of this example we will use prime numbers of greater than 25 digits, yielding $n = pq$ of roughly 50 digits in length. Let:

$$p = 12345679801994567990089459$$

and

$$q = 8369567977777368712343087$$

This gives

$$n = pq = 1033280063346665821884785640073336248556222630219933$$

And

$$\varphi(n) = (p-1)(q-1)$$
$$= 10332800633466658218847854329208 5845083685927787388$$

Next we must determine an integer e such that $1 < e < \varphi(n)$, and e is relatively prime to $\varphi(n)$. We can use $e = 115670849$ for this. It can be seen that $gcd\ (e, \varphi(n)) = 1$ (which is the definition of relatively prime).

We now must determine an integer d such that $1 < d < \varphi(n)$ such that $de \equiv 1\ (mod\ ((p-1)(q-1))$. Our values are much too large to use the Euclidean Algorithmor any sort of trial an error method, although it can be calculated that our d would be:

$$d = 34113931743910925784483561065442183977516731202177$$

This cryptosystem is constrained to only send messages m that are relatively prime to n. Although, the only divisors of n are at least 25 digits long, and as mentioned above credit card numbers will only be of length 16 digits + 4 digits for the expiry date.

Consider a customer with a credit card number of 4540 3204 4567 8231 1002 with an expiry date of 10/02. This gives us an m of:

$$m = 454032044567823110021002$$

Thus we can calculate (encrypt) the message as follows:

$$m^e \equiv a$$
$$a \equiv 49329085221791275793017511397 395566847998886183308 \quad (mod\ n)$$

We can see that this message has been safely transmitted without the worry of being broken (unless someone knows the secret key, d)

Upon receiving the encrypted message, we can easily calculate the original message using the following equivalency:

$$a^d \equiv m\ (mod\ n)$$
$$\equiv 454032044567823110021002 \quad (mod\ n)$$

Note: Although the values of p and q were seemingly very large, it should be considered that a computer could easily factor n of only roughly 50 digits. Thus leaving the encrypted message unprotected.

3. Diffie-Hellman Key Exchange

The Diffie-Hellman key exchange was first established in 1976 during a collaboration between its creators Whitfield Diffie and Martin Hellman[12]. It was the first practical method for establishing a shared secret over an unprotected communications channel. It was considered the first major milestone in public key cryptography[13]. Before we can begin exploring the Diffie-Hellman key exchange we must first look at discrete logarithms:

3.1. Discrete Logarithms

Definition 3.1.1: Let p be a prime number, we know that the group $(\boldsymbol{Z}/p\boldsymbol{Z})^*$ is cyclic of order $p-1$. Let g be primitive root $mod\ p$. (g is a generator of the multiplicative group of residues $(\boldsymbol{Z}/p\boldsymbol{Z})^*$). That is:

Let A be the set $A = \{1, 2, \dots, p-1\}$. We know that for all $a \in \{0,1,2, \dots, p-2\}$ with $A \equiv g^a\ (mod\ p)$.

The exponent a is referred to as the discrete logarithm of A to the base g. It can be written as: $a = log_g A$. Since the computation of discrete logarithms is considered to be difficult no efficient algorithm for solving the problem is known.

Example 3.1.1: Choose a prime number and show one of its primitive roots.

Solution: Let $p = 13$. A primitive root modulo 13 is 6. Below we will show that 6 is primitive root of 13 by computing the discrete logarithms of all integers in $A = \{1, 2, \dots, 13\}$.

We can see that we have:

Table 3.1.1. Discrete logarithms base 6 in $(\boldsymbol{Z}/13\boldsymbol{Z})^*$

A	1	2	3	4	5	6	7	8	9	10	11	12
$log_6 A$	0	5	8	10	9	1	7	3	4	2	11	6

It can be noticed that discrete logarithms can be defined in arbitrary cyclic groups. If we let G be a cyclic group of order n with generator g, and let A be a group element. Then there is an exponent $a \in \{0,1,2, \dots, n-1\}$ with $A = g^a$.

3.2. The Diffie Hellman Key Exchange

The Diffie-Hellman key exchange is used for a specific reason, two people let's say Alice and Bob would like to agree on a secret key, but their only means of communication is over an unsecured channel.

The first step is for Alice and Bob to agree on a large prime number p and an integer g with $2 \leq g \leq p-2$ such that the order of $g \ (mod\ p)$ is sufficiently large. The prime p and the primitive root g can be publicly known. Thus using the unsecured channel won't be a problem for Alice and Bob.

The next step of the Diffie-Hellman Key exchange, Alice choose an integer $a \in \{0,1,\dots,p-2\}$ and computes:

$$A = g^a \ (mod\ p)$$

This result A is sent to Bob, but Alice keeps the exponent a secret.

Bob follows the same step choosing an integer $b \in \{0,1,\dots,p-2\}$ and computes:

$$B = g^b \ (mod\ p)$$

This result B is sent to Alice, keeping his exponent b secret. For both to obtain the secret key they compute the following:

Alice: $B^a \ (mod\ p) = g^{ab} \ (mod\ p)$

Bob: $A^b \ (mod\ p) = g^{ab} \ (mod\ p)$

Thus, the secret key is known by both parties, but no one listening on the unsecured network can compute what Bob and Alice chose for their secret key.

Secret key: $K = g^{ab} \ (mod\ p)$

3.3. The Selection of p and q

It can be noted that the selection of p and g should be chosen so that the computation of the integer g has an order (mod p) that is sufficiently high. That is:

By definition of order, let n be the smallest positive integer such that $g^n = e \ (mod\ p)$, where e is the identity element. We want the integer n to be sufficiently large, so an attacker cannot easily compute it.

As mentioned above, one possibility is to choose g as a primitive root $(mod\ p)$. However to compute a primitive root of p, you must find the prime factorization of *p-1*. Since p should be a large prime number, finding this prime factorization should be very difficult. Which leads to the fact that selecting a primitive root of a large prime p, is also in fact very difficult. Selecting a prime p of a special form that is a prime p where $(p-1)\ /\ 2$ is also a prime number. This makes it easier to find a primitive root g of p.

Example 3.3.1: Show how two people can exchange a secret key K over an unsecured channel.

Solution: Consider Alice and Bob want to share a message over an open channel, they agree on a prime number, say $p = 13$ and a primitive root (or base) say $g = 6$. (For the purpose of this example we will use small values for p and g.)

Alice chooses $a = 5$ to be her exponent and computes $A = g^a \ (mod\ p) = 6^5 \ (mod\ 13) = 7776 \ (mod\ 13) = 2 \ (mod\ 13)$. Alice then sends this value $A = 2$ to Bob.

Bob chooses $b = 3$ to be his exponent and computes $B = g^b \ (mod\ p) = 6^3 \ (mod\ 13) = 216 \ (mod\ 13) = 8 \ (mod\ 13)$. Bob then sends this value $B = 8$ to Alice.

Once both parties receive their values, they compute their secret key:

Alice:
$B^a \ (mod\ p) = 8^5 \ (mod\ 13) = 32768 \ (mod\ 13) = 8 \ (mod\ 13)$

Bob: $A^b \ (mod\ p) = 2^3 \ (mod\ 13) = 8 \ (mod\ 13)$

Thus the secret key is 8.

3.4. The Diffie Hellman Problem and the Discrete Logarithm Problem

Definition 3.4.1: Let G be a finite cyclic group generated by an element g. The problem of computing g^{ab} from g^a and g^b where a and b are the secret values Alice/Bob choose. This is called the Diffie–Hellman Problem (DH problem) with respect to g.

Definition 3.4.2: Let G be a finite cyclic group generated by an element g. The problem of computing from $a \in G$ a number s such that $g^s = a$ is called the discrete logarithm problem (DL problem) with respect to g.

It can be noticed that if someone can solve the discrete logarithm problem for p they can immediately solve the Diffie–Hellman problem. The converse is not known. That is, it is thought to be possible (highly unlikely) that someone could invent a way to solve the Diffie–Hellman problem without being able to find discrete logarithms.

To use the discrete logarithm problem to solve the DH problem, the attacked can determine the discrete logarithm b of B to the base g and compute the key $K = A^b$.

Definition 3.4.3: A Diffie–Hellman oracle (DH oracle for short) for a group G with respect to a given generator g takes as inputs two elements $a, b \in G$ (where $a = g^u$ and $b = g^v$) and returns (without computational cost) the element g^{uv}.

It can be shown that under a plausible but unproven number theoretic assumption, for every finite cyclic group whose order is not divided by a multiple large prime factor there exists a polynomial-time algorithm for computing discrete logarithms and that makes calls to a DH oracle for this group.

Definition 3.4.4: For $\varepsilon > 0$, an ε-DH-oracle is a probabilistic oracle which returns for an input (g^u, g^v) the correct answer g^{uv} with probability at least ε if the input is uniformly distributed over $G \times G$. The offset of the oracle's answer is g^{uv+t} to the input (g^u, g^v) is defined as $t \ (mod\ |G|)$. A translation – invariant ε-DH-oracle is an ε-DH-oracle whose offset distribution is the same for every (g^u, g^v).

4. Elliptic Curve Cryptosystems

The use of elliptic curves in cryptography was first proposed by Neal Koblitz[17] and Victor Miller[18] in 1985. Elliptic curves can be defined over any field, but for the purpose of cryptography only finite fields are of interest. The size of the elliptic curve determines the difficulty of an attacker discovering the user's secret key.

Advantages of using an elliptic curve public key cryptosystem compared to another public key system, such as the RSA Algorithm, is that elliptic curves are much more efficient then RSA that is the calculations are much smaller[19]. The idea that RSA may one day become insecure, or at least the key sizes needed to safely secure a message become too large, it is believed that the elliptic curve cryptosystem may prove to be a very good alternative. Although there are disadvantages to using an elliptic curve cryptosystem, mainly group operations on elliptic curves are more complicated than group operations in prime fields.

4.1. Elliptic Curves

An introduction to elliptic curves is necessary before the discussion of elliptic curves use in public key cryptosystems. There is extensive literature on elliptic curves as they arise naturally in many branches of mathematics and are closely linked with the theory of elliptic functions, where of course they get their name. Consider the following definitions:

Definition 4.1.1: A ring R is a set with two binary operations, addition (denoted by $a + b$) and multiplication (denoted by ab), such that for all a, b, c in R:

1. $a + b = b + a$
2. $(a + b) + c = a + (b + c)$
3. There is an additive identity 0. That is, there is an element 0 in R such that $a + 0 = a$ for all a in R.
4. There is an element $-a$ in R such that $a + (-a) = 0$.
5. $a(bc) = (ab)c$
6. $a(b + c) = ab + ac$ and $(b + c)a = ba + ca$.

A ring is an Abelian group under addition, also having associative multiplication that is left and right distributive over addition. When multiplication is commutative we say that the ring is commutative. When a ring has an identity under multiplication we say that the ring has unity. Unity is a nonzero element that is an identity under multiplication.

Definition 4.1.2: A field $\boldsymbol{F}$ is a commutative ring with unity in which every nonzero element is a unit. A unit of a ring is a nonzero element that has a multiplicative inverse.

Definition 4.1.3: Let E be an elliptic curve over a field $\boldsymbol{F}$, E is a curve that is given by an equation of the form:

$$Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6, a_i \in \boldsymbol{F}.$$

We let $E(\boldsymbol{F})$ denote the set of points $(x, y) \in \boldsymbol{F}^2$ that satisfy this equation, along with a point of infinity, denoted by O. For the purpose of this paper, only elliptic curves over prime fields will be considered.

Definition 4.1.4: Let p be a prime number, $p > 3$ and let $a, b \in GF(p)$. Consider the equation:

$$y^2z = x^3 + axz^2 + bz^3$$

Its discriminant is:

$$\Delta = -16\ (4a^3 + 27b^2)$$

It can be assumed that the discriminant is non-zero.

If $(x, y, z) \in GF(p)^3$ is a solution of this equation, then for any $c \in GF(p), c(x, y, z)$ is also a solution. Two solutions (x, y, z) and (x', y', z') are called equivalent if there is a non-zero $c \in GF(p)$ with $(x, y, z) = c(x', y', z')$.The equivalence class of (x, y, z) is denoted by $(x{:}\,y{:}\,z)$. The elliptic curve $E(p; a, b)$ is the set of all equivalence classes of solution of $y^2z = x^3 + axz^2 + bz^3$. Each point of the curve is an element of this set.

Example 4.1.1: Consider the field $GF(11)$. Over this field, consider the equation $y^2 = x^3 + x + 6$. Find the set of all solutions of this curve.

Solution: We can see that we have $a = 1, b = 6$ and $z = 1$. The discriminant is $\Delta = -16(4a^3 + 27b^2) = -16\ (4 + (27 * 62)) = 4$. Thus we can see that $y^2 = x^3 + x + 6$ defines an elliptic curve over $GF(11)$. It is:

$$E(11;\ 1,6) = \{O, (2,4), (2,7), (3,5), (3,6), (5,2), (5,9), (7,2), (7,9), (8,3), (8,8), (10,2), (10,9)\}.$$

4.2. Elliptic Curves in Cryptography

Elliptic Curve Diffie-Hellman (ECDH) is a key agreement protocol that allows two parties, each having an elliptic curve public-private key pair, to establish a shared secret over insecure channel. This shared secret key may be directly used as a key, or better yet, to derive another key which can be used to encrypt a message using a symmetric key cipher.

4.3. Key Establishment Protocol

Suppose Alice wants to establish a shared key with Bob, but the only channel available for them may be unsecure. Initially, the domain parameters (that is, (p, a, b, G, n, h) in the prime case or $(m, f(x), a, b, G, n, h)$ in the binary case) must be agreed upon. Also, each party must have a key pair suitable for elliptic curve cryptography, consisting of a private key d (a randomly selected integer in the interval$[1, n - 1]$) and a public key Q (where $Q = dG$). Let Alice's key pair be (d_A, Q_A) and Bob's key pair be (d_B, Q_B). Each party must have the other party's public key (an exchange must occur).

Alice computes

$$(x_k, y_k) = d_AQ_B$$

Bob computes

$$(x_k, y_k) = d_BQ_A$$

The shared key is x_k (the x coordinate of the point). The number calculated by both parties is equal, because

$$d_AQ_B = d_Ad_BG = d_Ad_BG = d_BQ_A$$

The protocol is secure because nothing is disclosed (except for the public keys, which are not secret), and no party can derive the private key of the other unless it can solve the Elliptic Curve discrete logarithm Problem.

4.4. Message Transmission

It is not hard to modify the Elliptic Curve Diffie-Hellman key agreement protocol for the purpose of message transmission by using the ElGamal encryption method. Suppose that the set of message unit has been imbedded in E in some agreed upon way and Bob wants to send Alice a message $M \in E$.

Using the above method to exchange keys, d_AQ_B, d_BQ_A have already been exchanged. Bob now chooses another secret random integer, call this l, and sends Alice the pair of points $(lQ_B, M + l(d_AQ_B))$. This is the encryption method, to decrypt this message Alice multiplies the first point in the pair by her secret key d_A and then subtracts the result from the second point in the pair.

4.5. The Elliptic Curve Discrete Logarithm Problem

Referring to **Definition 3.4.2** the discrete logarithm problem is the problem of computing from $a \in G$, a number s such that $g^s = a$ with respect to g provided that such an integer exists. This in the case of $G = E$, the elliptic curve discrete logarithm problem to the base $Q \in E$ is the problem. Given $P \in E$, of finding an integer x such that $P = xQ$, if such x exists.

The belief is that in practice it takes about the same amount of time to factor a well-chosen finite field as the factorization of an integer of approximately the same size. However, as in the case of factorization of an integer, there are many subexponential time algorithms to factor a well-chosen finite field. In the case of elliptic curves (except for supersingular ones,) the only methods available for finding discrete logs on $E/\boldsymbol{F}_q$ are the methods that apply to arbitrary groups. All of the algorithms have running time of the form $exp(O(ln\ q))$, provided that E is divisible by a large prime (a prime with an order of magnitude not much less than q)

That is:

Let E be elliptic curve over finite field $\boldsymbol{F}_q$

1) Take point $P \in E(\boldsymbol{F}q)\backslash\{O\}$ and compute $Q = d * P$

2) Elliptic Curve Discrete Logarithm Problem: given P and Q, compute d

Much work has been put into the solutions of both the DLP and the ECDLP. There are different attempts at solving both, but none have been conclusively shown to solve all cases of these problems. It is a field of mathematics with extensive work being covered[7]. Recently, technology-based attacks against smart cards raised concerns about the general security of this method[20]. One aspect of the elliptic curve encryption that revived the interest in it is that elliptic curve based cryptography is suitable for implementation on a quantum computer[21].

5. The ElGamal Encryption Method

The ElGamal encryption method[14] is closely connected to the Diffie Hellman key exchange protocol because the users can exchange a private key over an unsecured channel and then use this key to encrypt a message and send to the other. The security of this cryptosystem is based solely on the difficulty of solving the Diffie-Hellman problem[6].

5.1. Encrypting a Message Using the ElGamal Encryption Method

The plaintext space is the set $\{0,1,\ldots,p-1\}$. To encrypt a plaintext m, Bob gets the authentic public key (p,g,A) of Alice. He chooses a random integer $b \in \{1,\ldots,p-2\}$ and computes:

$$B = g^b\ (mod\ p)$$

The number B is Bob's key part from the Diffie-Hellman system. Bob determines:

$$c = A^b m\ (mod\ p)$$

In other words, Bob encrypts the message m by multiplying it by the Diffie-Hellman key. The complete ElGamal ciphertext is the pair (B,c).

5.2. Decrypting a Message using the ElGamal Encryption Method

Alice has obtained the ciphertext (B,c). She knows her secret key a. To obtain the plaintext m, she divides c by the Diffie-Hellman key $B^a\ (mod\ p)$. In order to avoid inversions $(mod\ p)$, she determines the exponent $x = p-1-a$. Since $1 \le a \le p-2$. We have $1 \le x \le p-2$. Then she computes $m = B^x c\ (mod\ p)$. That is, in fact, the original plaintext, as the following computation shows:

$$B^x c \equiv g^{b(p-1-a)} A^b m \equiv (g^{p-1})^b (g^a)^{-b} A^b m \equiv A^{-b} A^b m \equiv m\ (mod\ p)$$

Example 3.5.1: Show how a use can encrypt a message and send it over an unsecured channel using the ElGamal encryption method.

Solution: Alice chooses $p = 23, g = 7$, and $a = 3$ and computes $A = g^a (mod\ p) = 21$. Her public key is $(p = 23, g = 7, A = 21)$. Her secret key is $a = 3$.

Bob encrypts $m = 7$. He chooses $b = 6$ and computes $B = g^b\ (mod\ p) = 4$ and $c = A^b m\ (mod\ p) = 14$. The ciphertext is $(B,c) = (4, 14)$.

Alice recovers m by computing $B^{p-1-3} c\ (mod\ p) = 7 = m$.

Example 3.5.2: Show how a use can encrypt a message and send it over an unsecured channel using the ElGamal encryption method

Solution: As in Example 3.5.1, the public key of Alice is $(p = 23, g = 7, A = 21)$. Her secret key is $a = 3$.

As a precomputation, Bob chooses $b = 6$ and computes $B = g^b\ (mod\ p) = 21$ and $K = A^b\ (mod\ p) = 18$. Then he simply computes $c = K * m(mod\ 23) = 11$. The cipher text is $(B,c) = (4,14)$. Again Alice recovers the plaintext by computing $B^{p-1-3} c\ (mod\ p) = 7 = m$.

The ciphertext is twice as long as the plaintext. This is called message expansion and is a disadvantage of this cryptosystem. On the other hand, the ElGamal system is a randomized cryptosystem, which can be regarded as an advantage.

6. The Knapsack Approach

In this section the idea of cryptosystems based on the knapsack problem will be discussed. It was introduced by Merkle and Hellman in 1978[15]. The knapsack problem is a very interesting problem found in combinatorial optimization. It has various uses from the perspective of computer science. Studies have been done into the pseudo-polynomial time algorithm using dynamic programming, fully polynomial-time approximation scheme and of course in public key cryptosystems. This section will begin with a comprehensive introduction to the problem itself, and then a discussion of attempts at public key systems using this approach.

6.1. The Knapsack Problem

Given a set of items, each with a weight and a value, determine the count of each item to include in a collection so that the total weight is less than or equal to the given limit and the total value is as large as possible. In other words, you are given a set of positive integers $a_1, a_2, \ldots, a_n$ and an integer S, the knapsack problem asks which of these integers, if any, add together to give S. That is, consider the set of values $x_1, x_2, \ldots, x_n$, each either 0 or 1, such that:

$$S = a_1x_1 + a_2x_2 + \ldots + a_nx_n.$$

Considering the following example will give a better idea of how this works.

Example 6.1.1: Show all the subsets of the set $(a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8) = (2, 5, 6, 9, 12, 13, 17, 18)$ that add up to $S = 23$.

Solution: By inspection we can see that there are four subsets that add up to 23. They can be seen as:

$$23 = 6 + 17$$
$$23 = 5 + 18$$
$$23 = 2 + 9 + 12$$
$$23 = 5 + 6 + 12$$

Equivalently, there are exactly four solutions to the equation $2x_1 + 5x_2 + 6x_3 + 9x_4 + 12x_5 + 13x_6 + 17x_7 + 18x_8 = 23$ with $x_i = 0$ or 1 for $i = \{1, 2, 3, 4, 5, 6, 7, 8\}$. These solutions are:

$$x_3 = x_7 = 1, x_1 = x_2 = x_4 = x_5 = x_6 = x_8 = 0,$$
$$x_2 = x_8 = 1, x_1 = x_3 = x_4 = x_5 = x_6 = x_7 = 0$$
$$x_1 = x_4 = x_5 = 1, x_2 = x_3 = x_6 = x_7 = x_8 = 0$$
$$x_2 = x_3 = x_5 = 1, x_1 = x_4 = x_6 = x_7 = x_8 = 0$$

We can see that to verify that the equation $S = a_1x_1 + a_2x_2 + \ldots + a_nx_n$ holds when $x_i = 0$ or 1 it takes at most n additions, which is much less then to search for solutions by trial and error which could take up to 2^n attempts.

Definition 6.1.1: If a sequence of integers $a_1, a_2, \ldots, a_n$ is such that the sum of the first $j-1$ of these integers is always less than the jth integer, that is:

$$\sum_{i=1}^{j-1} a_i < a_j, \qquad j = 2, 3, \ldots, n$$

This sequence is called **super-increasing**.

Example 6.1.2: Is the sequence 2,4,9,17,35,71 super-increasing?

Solution: We can see that:

$$4 > 2$$
$$9 > 4 + 2 = 6$$
$$17 > 9 + 4 + 2 = 15$$
$$35 > 17 + 9 + 4 + 2 = 32$$
$$71 > 35 + 17 + 9 + 4 + 2 = 67$$

Thus, this sequence is super-increasing, as we wanted to show.

Working with super-increasing sequences makes the knapsack problem much easier to solve, considering the following example will show this.

Example 6.1.3: Is it possible to get a sum of 48, using the above super-increasing sequence: 2,4,9,17,35,71.

Solution: Notice that we have $71 > 48$, so we cannot include 71 in any of our sums. Also notice that the elements which are less than 35, have a sum less than 35 when all added together:

$$35 > 17 + 9 + 4 + 2$$

Thus 35 must be used in our sum; we can notice that we now have:

$$48 - 35 = 13$$

Again $17 > 13$, which means it cannot be in our sum, we can easily notice among the final three integers that if we choose $9 + 4$, we get the required sum. That is:

$$48 = 35 + 9 + 4$$

This is what we wanted to show.

It can be noticed that in general to solve the knapsack problem of a super-increasing sequence $a_1, a_2, \ldots, a_n$, that is to find the values of $x_1, x_2, \ldots, x_n$ with $S = a_1x_1 + a_2x_2 + \ldots + a_nx_n$ and $x_i = 0$ or 1 for $i = \{1, 2, \ldots, n\}$ when S is given, we use the following algorithm.

First we find x_n by noting that:

$$x_n = \begin{cases} 1 & if\ S \geq a_n \\ 0 & if\ S \geq a_n \end{cases}$$

Then we find $x_{n-1}, x_{n-2}, \ldots, x_1$, in succession, using the equations

$$x_j = \begin{cases} 1 & if\ S - \sum_{i=j+1}^{n} x_i a_i \geq a_j \\ 0 & if\ S - \sum_{i=j+1}^{n} x_i a_i < a_j \end{cases}$$

for $j = n - 1, n - 2, \ldots, 1$.

Using this algorithm, knapsack problems based on super increasing sequences can be solved extremely quickly. The next section will discuss a cryptosystems based on this observation.

6.2. Cryptosystems Based on the Knapsack Problem

The idea of using the knapsack problem as a cryptosystem was first thought up by Merkle and Hellman, and we initially considered a good choice for a public key cryptosystem. This idea has since then been broken[16], and the knapsack approach is now considered to be an infeasible method. The ciphers we will discuss in this section will be based on transformed super-increasing sequences.

To be specific, let $a_1, a_2, \ldots, a_n$ be super-increasing and let m be a positive integer with $m > 2a_n$. Let w be an integer relatively prime to m with inverse $\overline{w}$ modulo m. We can now form the sequence $b_1, b_2, \ldots, b_n$ where $b_j \equiv wa_j\ (mod\ m)$ and $0 \leq bj < m$. We cannot use this special technique to solve a knapsack problem of the type $= \sum_{i=1}^{n} b_i x_i$, where S is a positive integer, because the sequence $b_1, b_2, \ldots, b_n$ is not super-increasing in general. However, when $\overline{w}$ is known, we can find:

$$\overline{w}S = \sum_{i=1}^{n} \overline{w} b_i x_i \equiv \sum_{i=1}^{n} a_i x_i\ (mod\ m)$$

Because $\overline{w}b_j \equiv a_j\ (mod\ m)$ we see that:

$$S_0 = \sum_{i=1}^{n} a_i x_i$$

Where S_0 is the least positive reside of $\overline{wS}$ modulo m. We can easily solve the equation because $a_1, a_2, \ldots, a_n$ is super-increasing. This solves the knapsack problem:

$$S = \sum_{i=1}^{n} b_i x_i$$

Because $b_j \equiv wa_j \ (mod\ m)$ and $0 \leq bj < m$. This procedure can be shown with an example.

Example 6.2.1: Use the above method to show the super-increasing sequence

$$(a_1, a_2, a_3, a_4, a_5) = (2, 7, 11, 23, 46)$$

can be transformed using $b_j \equiv 53a_j \ (mod\ 77)$, for $j = 1, 2, 3, 4, 5$. When

$$2x_1 + 7x_2 + 11x_3 + 23x_4 + 46x_5 = 76$$

Solution: We can form the sequence $b_j \equiv 53a_j \ (\text{mod } 77)$, for $j = 1, 2, 3, 4, 5$. That is

$$(b_1, b_2, b_3, b_4, b_5) = (53 * 2(mod\ 77), 53 * 7\ (mod\ 77),$$
$$53 * 11\ (mod\ 77), 53 * 23\ (mod\ 77), 53 * 46\ (mod\ 77))$$
$$= (29, 63, 44, 64, 51)$$

That is,

$$29x_1 + 63x_2 + 44x_3 + 64x_4 + 51x_5 = 24(mod\ 77)$$

We can check to see if this transformation was successful by solving the original sequence, which is

$$2x_1 + 7x_2 + 11x_3 + 23x_4 + 46x_5 = 76$$

This sequence is super increasing which means we can follow the above procedure in Example 6.1.3.

$$76 - 46 = 30$$
$$30 - 23 = 7$$
$$7 - 7 = 0$$

Thus, we have a solution of $x_5 = x_4 = x_2 = 1$ and $x_3 = x_1 = 0$. We can check this solution with our above transformed sequence $(b_1, b_2, b_3, b_4, b_5)$. We have

$$29x_1 + 63x_2 + 44x_3 + 64x_4 + 51x_5 = 24(mod\ 77)$$
$$51 + 64 + 63 = 178(mod\ 77) \equiv 24(mod\ 77)$$

Which is what we wanted to show, thus our transformation was successful in making a super-increasing sequence into a sequence that is not super-increasing.

The cryptosystem based on the knapsack problem invented by Merkle and Hellman works as follows:

Each individual chooses a super-increasing sequence of positive integers of a specified length, say, N (for example, $a_1, a_2, \ldots, a_N$) as well as a modulus m with $m > 2a_N$ and a multiplier w with $(m, w) = 1$. The transformed sequence $b_1, b_2, \ldots, b_N$ is made public.

To send a message P to an individual, the message is first translated into a string of zeros and ones using the binary equivalents of letters given in Table 6.2.1 below. This string of zeros and ones is split into segments of length N; if N does not divide the length, simply fill out the last block with ones.

For each of the blocks we have, a sum is computed using the sequence $b_1, b_2, \ldots, b_N$, for example, when we have the block $x_1 x_2 \ldots x_N$:

$$S = b_1 x_1 + b_2 x_2 + \ldots + b_N x_N$$

The sum generated by each block form a ciphertext message. Since $b_1, b_2, \ldots, b_N$ is not a super-increasing sequence, it is much harder to solve the knapsack cipher, without knowledge of m and w. When someone knows m and w, the knapsack problem can be transformed into a super-increasing sequence, which makes for a much easier knapsack problem. For someone with knowledge of m and w, we get:

$$\overline{w}S = \overline{w}b_1 x_1 + \overline{w}b_2 x_2 + \ldots + \overline{w}b_N x_N \equiv a_1 x_1 + a_2 x_2 + \ldots + a_N x_N \ (mod\ m)$$

where $\overline{w}b_j \equiv a_j \ (mod\ m)$, where $\overline{w}$ is an inverse of w modulo m, so that:

$$S_0 = a_1 x_1 + a_2 x_2 + \ldots + a_N x_N$$

Table 6.2.1. The binary equivalents of letters

Letter	Binary Equivalent	Letter	Binary Equivalent
A	00000	N	01101
B	00001	O	01110
C	00010	P	01111
D	00011	Q	10000
E	00100	R	10001
F	00101	S	10010
G	00110	T	10011
H	00111	U	10100
I	01000	V	10101
J	01001	W	10110
K	01010	X	10111
L	01011	Y	11000
M	01100	Z	11001

Example 6.2.2: Consider the super-increasing sequence $(a_1, a_2, a_3, a_4, a_5) = (2, 15, 19, 41, 83)$. Let our $m = 211$, so that $m > 2a_5$, and $w = 101$, thus we can see that we have $gcd(m, w) = 1$. Encrypt the message KNAPSACK and show the decryption process.

Solution: We begin by transforming our super-increasing sequence $(a_1, a_2, a_3, a_4, a_5) = (2, 15, 19, 41, 83)$ into a sequence that is not super-increasing, we note:

$$(b_1, b_2, b_3, b_4, b_5) = (wa_1(mod\ m), wa_2(mod\ m),$$
$$wa_3(mod\ m), wa_4(mod\ m), wa_5(mod\ m))$$
$$(b_1, b_2, b_3, b_4, b_5) = (101 * 2(mod\ 211),$$
$$101 * 15(mod\ 211), 101 * 19(mod\ 211),$$
$$101 * 41\ (mod\ 211), 101 * 83\ (mod\ 211))$$
$$(b_1, b_2, b_3, b_4, b_5) = (202, 38, 20, 132, 154)$$

To translate this message, we must first translate the letters of the message into their five-digit binary equivalents and put them into blocks of length N, this sequence is of length 5, so the blocks will also be of length 5.

We now have:

01010 01101 00000 01111 10010 00000 00010 01010

For each block of 5 binary digits, we form a sum by adding together the appropriate terms of the sequence in the slots corresponding to the positions of the block containing a digit equal to 1. This gives us:

170 212 0 344 334 0 132 170

This string of 8 sums is our found ciphertext. We can see that this example is very simplistic, although the decrypted message is protected from basic attacks.

To decrypt and arrive back at our original message we must find the least positive residue modulo 211 of 117 times each sum. We can calculate the inverse of 101 modulo 211 to be 117. Once we finish this we can solve the corresponding easy knapsack problem with respect to the original super-increasing sequence

$$(a_1, a_2, a_3, a_4, a_5) = (2, 15, 19, 41, 83)$$

We then have:

$$170 * 117 = 19890\,(mod\ 211) \equiv 56(mod\ 211)$$

We can see that $56 = 15 + 41$, which corresponds to the binary representation of 01010.

$$212 * 117 = 24804\,(mod\ 211) \equiv 117(mod\ 211)$$

We can see that $117 = 15 + 19 + 83$, which corresponds to the binary representation of 01101.

$$0 * 117 = 0(mod\ 211)$$

We can see that we have a corresponding representation of 00000.

$$344 * 117 = 40248\,(mod\ 211) \equiv 158(mod\ 211)$$

We can see that $158 = 15 + 19 + 41 + 83$, which corresponds to the binary representation of 01111.

$$334 * 117 = 39078\,(mod\ 211) \equiv 43(mod\ 211)$$

We can see that $43 = 2 + 41$, which corresponds to the binary representation of 10010.

$$0 * 117 = 0(mod\ 211)$$

We can see that we have a corresponding representation of 00000.

$$132 * 117 = 15444\,(mod\ 211) \equiv 41(mod\ 211)$$

We can see that $41 = 41$, which corresponds to the binary representation of 00010.

$$170 * 117 = 19890\,(mod\ 211) \equiv 56(mod\ 211)$$

We can see that $56 = 15 + 41$, which corresponds to the binary representation of 01010.

Thus our decrypted message comes out to be:

01010 01101 00000 01111 10010
00000 00010 01010
K N A P S
A C K

This is our original message and we have properly encrypted and decrypted the message.

Knapsack ciphers were originally thought to be excellent candidates as an alternate form of public key encryption instead of the standard RSA approach. However, in 1982 Shamir (one of the inventors of RSA) showed they do not provide satisfactory security for encryption of messages. The reason being, he had found an efficient algorithm for solving knapsack problems involving sequence $b_1, b_2, \ldots, b_n$ with $b_j \equiv wa_j\ (mod\ m)$, where m and w are relatively prime positive integers and $a_1, a_2, \ldots, a_n$ is a super-increasing sequence. This is why knapsack ciphers are not used in modern day cryptography.

7. Conclusions

In this paper we have described and compared five of the major public key cryptosystems developed in the last forty years. Although they all share some characteristics, they are based on a variety of hard mathematical problems: the factorization of a number into primes, the one way trapdoor functions, the elliptic curve arithmetic, and the discrete logarithms in finite fields. As the researchers develop cryptosystems, they have to argue that they are hard to break and therefore useful. At the same time they have to study the ways to break them, serving as further reassurance of the strength of the cryptosystems. The importance of the public key cryptosystems lies in their accessibility, and thus their practicality.

Most authors consider the RSA systems to be the most secure. It is well studied and implemented and widely used. It is widely believed that even if the factorization problem is solvable in polynomial time, which is currently not known, increasing the size of the primes used in the key generation will keep the method secure for years to come.

The Diffie-Hellmann key exchange and the ElGamal encryption method based on it are a very useful in understanding the basics of exponentiation based cryptography. It is further important for the study of the strengths and weaknesses of any cryptosystem based on exponentiation, and thus valuable cryptanalytic tool. The discrete logarithm problem became one of the most studied problems due to this.

Elliptic curve based cryptography has one very important feature that appeals to the manufacturers of security protected electronic equipment – quicker calculations compared to the other major cryptographic algorithms. However, both theoretically and from the point of view of its hardware implementation it is less secure than RSA. Another appealing feature of the elliptic curve cryptography is its applicability under the quantum computing paradigm, which spurs further research into this approach.

The knapsack approach had important role in the development of the idea of a cryptosystem based on one way trapdoor function. Today, we know that the knapsack cipher is not secure enough for practical applications, due to its polynomial time breakability.

ACKNOWLEDGEMENTS

Dr. Vassilev's research is supported by NSERC Discovery Grant. Andrew Twizell would like to acknowledge the help and support of Dr. Vassilev during the work on this paper.

REFERENCES

[1] Neal Koblitz. Algebraic Aspects of Cryptography. Berlin, Springer, 1999.

[2] Johannes A. Buchmann. Introduction to Cryptography. Springer, New York, 2004.

[3] Joseph A. Gallian. Contemporary Abstract Algebra. Brooks/Cole, Australia, 2010.

[4] John Stillwell. Elements of Number Theory. Springer, New York, 2003.

[5] Kenneth H. Rosen. Elementary Number Theory and Its Applications. Pearson/Addison Wesley, Boston, 2011.

[6] Ueli M. Maurer and Stefan Wolf. The Diffie-Hellman Protocol. Designs, Codes and Cryptography 19: 147–171, 2000.

[7] Neal Koblitz, Alfred Menezes and Scott Vanstone. The State of Elliptic Curve Cryptography. Designs, Codes and Cryptography 19: 173–193, 2000.

[8] Christiane Rousseau and Yvan Saint-Aubin. Mathematics and Technology. Springer, New York, 2008.

[9] Ronald Rivest, Adi Shamir and Leonard Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM 21(2): 120–126, 1978.

[10] Alfred Menezes, Paul C. van Oorschot and Scott A. Vanstone. Handbook of Applied Cryptography. CRC Press, 1996.

[11] Dan Boneh. Twenty Years of attacks on the RSA Cryptosystem. Notices of the American Mathematical Society 46 (2): 203–213, 1999.

[12] Withfield Diffie and Martin E. Hellman. New Directions in Cryptography. IEEE Transactions on Information Theory, vol. IT-22: 644–654, 1976.

[13] Martin E. Hellman. An Overview of Public Key Cryptography. IEEE Communications Magazine, 42–49, 2002.

[14] Taher ElGamal. A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. IEEE Transactions on Information Theory 31 (4): 469–472, 1985.

[15] Ralph Merkle and Martin E. Hellman. Hiding information and signatures in trapdoor knapsacks. IEEE Transactions on Information Theory 24 (5): 525–530, 1978.

[16] Adi Shamir. A polynomial-time algorithm for breaking the basic Merkle - Hellman cryptosystem. IEEE Transactions on Information Theory 30 (5): 699–704, 1984.

[17] Neal Koblitz. Elliptic curve cryptosystems. Mathematics of Computation 48 (177): 203–209, 1987.

[18] Victor S. Miller. Use of elliptic curves in cryptography. CRYPTO 85: 417–426, 1985.

[19] Y. Hitchcock, E. Dawson, A. Clark and P. Montague. Implementing an efficient elliptic curve cryptosystem over GF(p) on a smart card. ANZIAM Journal 44, 2002.

[20] Ingrid Biehl, Bernd Meyer and Volker Müller. Differential Fault Attacks on Elliptic Curve Cryptosystems. Advances in Cryptology – CRYPTO 2000. Lecture Notes in Computer Science 1880: 131–146, 2000.

[21] Michael A.Nielsen and Isaac L.Chuang. Quantum Computation and Quantum Information. 2002