SOME CRYPTOGRAPHIC ALGORITHMS

A THESIS SUBMITTED TO THE

NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA

IN THE PARTIAL FULFILMENT

FOR THE DEGREE OF

MASTER OF SCIENCE IN MATHEMATICS

BY

SOMYASHREE SATPATHY

UNDER THE SUPERVISION OF

Dr. DIVYA SINGH

DEPARTMENT OF MATHEMATICS

NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA

MAY, 2014

Certificate

This is to certify that the project report entitled "SOME CRYPTOGRAPHIC ALGORITHMS" submitted by Ms. Somyashree Satpathy to the National Institute of Technology Rourkela, Odisha for the partial fulfilment of requirements for the degree of Master of Science in Mathematics is a bonafide record of review work carried out by her under my supervision and guidance. The contents of this project report, in my knowledge, have not been submitted to any other institute or university for the award of any degree or diploma.

May 12, 2014

Dr. Divya Singh
Assistant Professor
Department of Mathematics
National Institute of Technology Rourkela
Odisha-769008

Preface

In the present thesis consisting of two chapters first we have given a brief review of some important number theoretic concepts and results. Then we have discussed S-DES and DES algorithms for Secret key cryptography, RSA and DSA algorithms for Public key cryptography and at last a brief introduction to elliptic curves and their use in Cryptography.

Rourkela — Somyashree Satpathy

May 12, 2014

Acknowledgements

It is my pleasure to thank to the people, for whom this dissertation is possible. I specially like to thank my guide Dr. Divya Singh, for her guidance and encouragement during the course of study and preparation of the final manuscript of this project. I would also like to thank the the H.O.D. and the other faculty members of Department of Mathematics for their co-operation. I heartily thank to my friends, who helped me during the preparation of this project. I thank the Director, National Institute of Technology Rourkela, for providing the facilities to pursue my postgraduate degree I thank all my classmates and friends for making my stay memorable at National Institute of Technology Rourkela.

Finally, I also thank my family and specially to my parents for their constant inspiration.

Contents

1. Basics of Number Theory

1.1 Prime numbers and their properties

1.2 Theory of Congruence

1.3 Fermat's theorem and related results

1.4 Number theoretic Functions

1.5 Euler's generalization of Fermat's Theorem

2. Cryptographic Algorithms

2.1 Introduction

2.2 Secret Key Cryptography

2.3 Public Key Cryptography

2.4 Elliptic Curve Cryptography

Chapter 1

Basics of Number Theory

1.1. Prime numbers and their properties

Definition 1.1.1. An integer $p > 1$ is called a *prime number* or simply *prime*, if its only positive divisors are 1 and p. An integer greater than 1 that is not prime is termed as *composite*.

Theorem 1.1.2 [3]. *If p is a prime and $p|ab$, then $p|a$, or $p|b$.*

Proof: If $p|a$, then there is nothing to do. So let's assume that p does not divide a. Because the only positive divisors of p are 1 and p itself; this implies that $gcd(p, a) = 1$. Then there exist integers m, n such that $mp + na = 1$. Thus, $mpb + nab = b$. Now $p|mpb$ and, by our assumption $p|nab$, consequently $p|(mpb + nab)$, or $p|b$.

Corollary 1.1.3 [3]. *If p is a prime and $p|a_1a_2 \cdots a_n$, then $p|a_k$, for some k, where $1 \leq k \leq n$.*

Corollary 1.1.4 [3]. *If $p, q_1, q_2, \cdots, q_n$ all are primes and $p|q_1q_2 \cdots q_n$, then $p = q_k$, for some k, where $1 \leq k \leq n$.*

Theorem 1.1.5 (Fundamental Theorem of Arithmetic)[3]. *Every positive integer $n > 1$ can be expressed as a product of primes; this representation is unique, apart from the order in which the factors occur.*

Proof: Either n is a prime or a composite. In case of prime, there is nothing more to prove. If n is composite, then there exists an integer d satisfying $d|n$ and $1 < d < n$. Among all such integers d, choose p_1, to be the smallest. This is possible by the well-ordering principle. Then p_1 must be a prime number. Otherwise it too have a divisor q with $1 < q < p_1$; but then $q|p_1$ and $p_1|n$ which imply that $q|n$, which contradicts the choice of p_1 as the smallest positive divisor, not equal to 1, of n.

Therefore we may write $n = p_1n_1$, where p_1 is a prime number and $1 < n_1 < n$. If n_1 happens to be a prime, then we have our representation. In the contrary case, the argument is repeated to produce a second prime number p_2 such that $n_1 = p_2n_2$; i.e.

$$n = p_1p_2n_2; \quad 1 < n_2 < n_1$$

If n_2 is a prime, then it is not necessary to go further. Otherwise, write $n_2 = p_3 n_3$, with p_3 a prime.

$$n = p_1 p_2 p_3 n_3; \quad 1 < n_3 < n_2$$

The decreasing sequence

$$n > n_1 > n_2 > \cdots > 1$$

can not continue indefinitely, so that after a finite number of steps n_{k-1} is a prime, say, p_k. This leads to the prime factorization.

$$n = p_1 p_2 \cdots p_k$$

For the uniqueness of prime factorization, let's suppose that the integer n can be represented as a product of primes in two ways, say,

$$n = p_1 p_2 \ldots p_r \quad q_1 q_2 \ldots q_s \text{ where}$$

where $r \leq s$ and p_i's and q_i's are all primes, written in increasing magnitude so that

$$p_1 \leq p_2 \leq \cdots \leq p_r, \qquad q_1 \leq q_2 \leq \cdots \leq q_s$$

Because $p_1 \mid q_1 q_2 \ldots q_s$. From the above corollary we know that $p_1 = q_k$, for some k but then $p_1 \geq q_1$. Similar reason gives $q_1 \geq p_1$ which together gives $p_1 = q_1$. We may cancel the common factors and obtain

$$p_2 p_3 \ldots p_r = q_2 q_3 \ldots q_s$$

Now repeat the process to get $p_2 = q_2$, and

$$p_3 p_4 \ldots p_r = q_3 q_4 \ldots q_s.$$

Continuing in this fashion, if the inequality $r < s$ were to hold, we would get that

$$1 = q_{r+1} q_{r+2} \ldots q_s$$

which is not possible as each $q_j > 1$. Hence $r = s$, and $p_1 = q_1$, $p_2 = q_2, \ldots, p_r = q_r$, making the two factorization of n identical.

1.2. Theory of Congruence

Definition 1.2.1.[3] Let n be a fixed positive integer. Two integers a *and* b *are said to be congruent modulo* n, written as

$$a \equiv b(\text{mod } n)$$

if n divides the difference $a - b$; i.e. $a - b = kn$, for some integer k.

Theorem 1.2.2.[3] *Let* $n > 1$ *be fixed and* a, b, c, d *be arbitrary integers. Then the following properties hold:*

(i) $a \equiv a \pmod{n}$.

(ii) If $a \equiv b \pmod{n}$, then $b \equiv a \pmod{n}$.

(iii) If $a \equiv b \pmod{n}$ and $b \equiv c \pmod{n}$, then $a \equiv c \pmod{n}$.

(iv) If $a \equiv b \pmod{n}$ and $c \equiv d \pmod{n}$, then $a + c \equiv b + d \pmod{n}$ and $ac \equiv bd \pmod{n}$.

(v) If $a \equiv b \pmod{n}$, then $a + c \equiv b + c \pmod{n}$ and $ac \equiv bc \pmod{n}$.

(vi) If $a \equiv b \pmod{n}$, then $a^k \equiv b^k \pmod{n}$, for any positive integer k.

Theorem 1.2.3.[3] *If* $ca \equiv cb(mod\, n)$, *then* $a \equiv b(mod\, n/d)$, *where* $d = gcd(c, n)$.

Proof: By hypothesis,we can write

$$c(a - b) = ca - cb = kn$$

for some integer k. Knowing that $gcd(c, n) = d$, there exist relatively prime integers r and s satisfying $c = dr$ and $n = ds$. Putting these two values in the above equation, we get

$$r(a - b) = ks.$$

Hence, $s|r(a-b)$ and $gcd(r, s) = 1$. Euclid's lemma [*If* $a|bc$, *with* $gcd(a, b)$=1, *then* $a|c$] yields $s|(a - b)$ which implies that $a \equiv b(\text{mod } s)$. In other words $a \equiv b(\text{mod } n/d)$.

Corollary 1.2.4 [3]. *If* $ca \equiv cb(mod\, n)$ *and* $gcd(c, n) = 1$, *then* $a \equiv b(mod\, n)$.

Corollary 1.2.5 [3]. *If* $ca \equiv cb(mod\, p)$ *and* $p \nmid c$, *where* p *is a prime number, then* $a \equiv b(mod\, p)$.

Theorem 1.2.6.[3] *Let* $P(x) = \sum_{k=0}^{m} c_k x^k$ *be a polynomial function of* x *with integral coefficients* c_k. *If* $a \equiv b(mod\, n)$, *then* $P(a) \equiv P(b)(mod\, n)$.

Proof: As $a \equiv b(\text{mod } n)$, we can write $a^k \equiv b^k(\text{mod } n)$ for $k = 0, 1, \cdots, m$. Therefore $c_k a^k \equiv c_k b^k(\text{mod } n)$, for all such k. Adding these $m + 1$ congruences, we conclude that

$$\sum_{k=0}^{m} c_k a^k \equiv \sum_{k=0}^{m} c_k b^k(\text{mod } n)$$

or, $P(a) \equiv P(b) (\text{mod } n)$.

Corollary 1.2.7 [3]. *If a is a solution of $P(x) \equiv 0 (mod\, n))$ and $a \equiv b (mod\, n)$, then b is also a solution.*

Theorem 1.2.8.[3] *The linear congruence $ax \equiv b (mod\, n)$ has a solution if and only if $d \mid b$, where $d = gcd(a, n)$. If $d \mid b$, then it has d mutually incongruent solutions modulo n.*

Proof: The given congruence is equivalent to the linear diophantine equation $ax - ny = b$. We know that the latter equation can be solved if and only if $d \mid b$; moreover, if it is solvable and x_0, y_0 is one specific solution, then any other solution has the form

$$x = x_0 + \frac{n}{d}t, \qquad y = y_0 + \frac{a}{d}t$$

for some choice of t. Among the various integers satisfying the first of these formulas, consider those that occur when t takes on the successive values $t = 0, 1, 2, \cdots, d-1$:

$$x_0,\ x_0 + \frac{n}{d},\ x_0 + \frac{2n}{d}, \cdots,\ x_0 + \frac{(d-1)n}{d}$$

We claim that these integers are incongruent modulo n and all other such integers x are congruent to some one of them. If it happened that

$$x_0 + \frac{n}{d}t_1 \equiv x_0 + \frac{n}{d}t_2 (\text{mod } n)$$

where $0 \leq t_1 < t_2 \leq d-1$, then we would have

$$\frac{n}{d}t_1 \equiv \frac{n}{d}t_2 (\text{mod } n)$$

Now $gcd(n/d, n) = n/d$. So

$$t_1 \equiv t_2 (\text{mod } d)$$

which is to say that $d \mid t_2 - t_1$. But this is impossible in the view of inequality $0 < t_2 - t_1 < d$.

It remains to argue that any other solution $x_0 + (n/d)t$ is congruent modulo n to one of the d integers listed above. The division algorithm permits us to write t as $t = qd + r$, where $0 \leq r \leq d-1$. Hence

$$\begin{aligned} x_0 + \frac{n}{d}t &= x_0 + \frac{n}{d}(qd + r) \\ &= x_0 + nq + \frac{n}{d}r \\ &\equiv x_0 + \frac{n}{d}r (\text{mod } n) \end{aligned}$$

with $x_0 + \frac{n}{d}r$ being one of our d selected solutions.

Corollary 1.2.9 [3]. *If $gcd(a, n) = 1$, then the linear congruence $ax \equiv b(mod\ n)$ has a unique solution modulo n.*

Theorem 1.2.10.[3] *The system of linear congruences*

$$ax + by \equiv r(mod\, n)$$

$$cx + dy \equiv s(mod\, n)$$

has a unique solution modulo n, whenever $gcd(ad - bc, n) = 1$.

1.3. Fermat's Theorem and Related results

Theorem 1.3.1 (Fermat)[3]. *Let p be a prime and suppose that $p \nmid a$. Then $a^{p-1} \equiv 1(mod\, p)$.*

Proof: We begin by considering the first $p - 1$ positive multiples of a; i.e the integers

$$a, 2a, 3a, \cdots, (p-1)a.$$

None of these numbers is congruent modulo p to any other, nor is any congruent to zero. Indeed, if it happened that

$$ra \equiv sa(\text{mod } p) \qquad 1 \leq r < s \leq (p-1)$$

then a could be cancelled out to give $r \equiv s(\text{mod } p)$, which is impossible. Therefore the previous set of integers must be congruent modulo p to $1, 2, \cdots, (p-1)$, taken in same order. Multiplying all these congruences together, we get that

$$a.2a.3a \cdots (p-1)a \equiv 1.2.3 \cdots (p-1)(\text{mod } p)$$

whence

$$a^{(p-1)}(p-1)! \equiv (p-1)!(\text{mod } p).$$

Once $(p-1)!$ is cancelled out from both the sides of the preceding congruence, since $p \nmid (p-1)!$, we get that

$$a^{(p-1)} \equiv 1(\text{mod } p).$$

Corollary 1.3.2 [3]. *If p is a prime, then $a^p \equiv a(mod\, p)$, for any integer a.*

Theorem 1.3.3 (Wilson)[3]. *If p is a prime, then $(p-1)! \equiv -1(mod\, p)$.*

Theorem 1.3.4.[3] *The quadratic congruence $x^2 + 1 \equiv 0(mod\, p)$, where p is an odd prime, has a solution if and only if $p \equiv 1(mod\, 4)$.*

Proof: Let a be any solution of $x^2 + 1 \equiv 0(\text{mod } p)$, so that $a^2 \equiv -1(\text{mod } p)$. Because $p \nmid a$, the outcome of applying Fermat's theorem is

$$1 \equiv a^{(p-1)} \equiv (a^2)^{(p-1)/2} \equiv (-1)^{(p-1)/2}(\text{mod } p).$$

The possibility that $p = 4k + 3$, for some k does not arise. If it did, we would have

$$(-1)^{(p-1)/2} = (-1)^{(2k+1)} = -1$$

hence, $1 \equiv -1(\text{mod } p)$. The net result of this is that $p \mid 2$, which is false. Therefore p must be of the form $4k + 1$.

Now for the opposite direction, in the product

$$(p-1)! = 1.2.\cdots\frac{(p-1)}{2}.\frac{(p+1)}{2}\ldots(p-2)(p-1)$$

we have the congruences

$$p - 1 \equiv -1(\text{mod } p)$$

$$p - 2 \equiv -2(\text{mod } p)$$

$$\vdots$$

$$\frac{(p+1)}{2} \equiv -\frac{(p-1)}{2}(\text{mod } p)$$

Rearrangement of the factors produce

$$\begin{aligned}(p-1)! &\equiv 1.(-1).2(-2)\ldots\frac{(p-1)}{2}.\left(-\frac{(p-1)}{2}\right)(\text{mod } p)\\ &\equiv (-1)^{(p-1)/2}\left(1.2.\ldots\frac{(p-1)}{2}\right)^2(\text{mod } p)\end{aligned}$$

because there are $(p-1)/2$ minus signs involved. By Wilson's theorem $(p-1)! \equiv -1(\text{mod } p)$,

$$-1 \equiv (-1)^{(p-1)/2}\left[\left(\frac{p-1}{2}\right)!\right]^2(\text{mod } p)$$

If we assume that p is of the form $4k+1$, then $(-1)^{(p-1)/2} = 1$, leaving us with the congruence

$$-1 \equiv \left[\left(\frac{p-1}{2}\right)!\right]^2 (\text{mod } p).$$

The conclusion is that the integer $[(p-1)/2]!$ satisfies the quadratic congruence $x^2 + 1 \equiv 0(\text{mod } p)$.

1.4. Number Theoretic Functions

Definition 1.4.1.[3] Given a positive integer n; let $\tau(n)$ denotes the number of positive divisors of n and $\sigma(n)$ denote the sum of these divisors.

Theorem 1.4.2.[3] *If $n = p_1^{k_1}.p_2^{k_2}\cdots p_r^{k_r}$ is the prime factorization of $n > 1$, then the positive divisors of n are precisely those integers d of the form*

$$d = p_1^{a_1}.p_2^{a_2}\ldots p_r^{a_r}$$

where $0 \leq a_i \leq k_i \quad (i = 1, 2 \ldots, r)$.

Proof: Note that the divisor $d = 1$ is obtained when $a_1 = a_2 = \cdots = a_r = 0$ and n itself occurs when $a_1 = k_1$, $a_2 = k_2$, $\ldots$,$a_r = k_r$. Suppose that d divides n is non-trivially, say, $n = dd'$ where $d > 1$; $d' > 1$. Express both d and d' as product of (not necessarily distinct) primes.

$$d = q_1.q_2.\ldots.q_s \qquad d' = t_1.t_2.\ldots.t_u$$

with q_i, t_i prime. Then

$$p_1^{k_1}.p_2^{k_2}\ldots p_r^{k_r} = q_1.q_2\ldots q_s.t_1\ldots t_u$$

are two prime factorizations of the positive integer n. By the uniqueness of the prime factorization each prime q_i must be one of the p_j. Collecting the equal primes into a single integral power, we get

$$d = q_1.q_2\ldots q_s = p_1^{a_1}.p_2^{a_2}\ldots p_r^{a_r}$$

where the possibility that $a_i = 0$ is allowed.

Conversely, every number $d = p_1^{a_1}.p_2^{a_2}\dots p_r^{a_r}$ $(0 \le a_i \le k_i)$ turns out to be a divisors of n. For we can write

$$\begin{aligned} n &= p_1^{k_1}.p_2^{k_2}\dots p_r^{k_r} \\ &= (p_1^{a_1}.p_2^{a_2}\dots p_r^{a_r})(p_1^{k_1-a_1}.p_2^{k_2-a_2}\dots p_r^{k_r-a_r}) \\ &= dd' \end{aligned}$$

with $d' = p_1^{k_1-a_1}.p_2^{k_2-a_2}\dots p_r^{k_r-a_r}$ and $k_i - a_i \ge 0$ for each i. Then $d' > 0$ and $d \mid n$.

Theorem 1.4.3.[3] *If $n = p_1^{k_1}, p_2^{k_2}\dots p_r^{k_r}$ is the prime factorization of $n > 1$, then*

(a) $\tau(n) = (k_1+1)(k_2+1)\dots(k_r+1)$, *and*

(b) $\sigma(n) = \dfrac{p_1^{k_1+1}-1}{p_1-1}.\dfrac{p_2^{k_2+1}-1}{p_2-1}.\dots.\dfrac{p_r^{k_r+1}-1}{p_r-1}.$

Definition 1.4.4.[3] A number-theoretic function f is said to be *multiplicative* if

$$f(mn) = f(m).f(n)$$

whenever $gcd(m,n) = 1$.

Theorem 1.4.5.[3] *The functions τ and σ are both multiplicative functions.*

Proof: Let m and n be relatively prime integers. Because the result is trivially true if either m, or n is equal to 1, we may assume that $m > 1$ and $n > 1$. If

$$m = p_1^{k_1}p_2^{k_2}\dots p_r^{k_r} \text{ and } n = q_1^{j_1}q_2^{j_2}\dots q_s^{j_s}$$

are the prime factorizations of m and n, then because $gcd(m,n) = 1$, no p_i can occur among the q_j. It follows that the prime factorization of the product mn is given by

$$mn = p_1^{k_1}p_2^{k_2}\dots p_r^{k_r}q_1^{j_1}q_2^{j_2}\dots q_s^{j_s}.$$

Applying the previous theorem, we obtain

$$\begin{aligned} \tau(mn) &= [(k_1+1)\dots(k_r+1)][(j_1+1)\dots(j_s+1)] \\ &= \tau(m)\tau(n). \end{aligned}$$

In the similar way,

$$\begin{aligned} \sigma(mn) &= \left[\frac{p_1^{k_1+1}-1}{p_1-1}\dots\frac{p_r^{k_r+1}-1}{p_r-1}\right]\left[\frac{q_1^{j_1+1}-1}{q_1-1}\dots\frac{q_s^{j_s+1}-1}{q_s-1}\right] \\ &= \sigma(m)\sigma(n). \end{aligned}$$

Thus, τ and σ are multiplicative functions.

1.5. Euler's Generalization of Fermat's Theorem

Definition 1.5.1.[3] For $n \geq 1$, let $\phi(n)$ denote the number of positive integers not exceeding n that are relatively prime to n. The function ϕ is called the *Euler's phi-function or, Euler's totient function.*

Theorem 1.5.2.[3] *If p is a prime and $k > 0$, then*

$$\phi(p^k) = p^k - p^{(k-1)} = p^k\left(1 - \frac{1}{p}\right).$$

Proof: Clearly $gcd(n, p^k) = 1$ if and only if $p \nmid n$. There are $p^{(k-1)}$ integers between 1 and p^k divisible by p, namely;

$$p, 2p, 3p \ldots (p^{(k-1)}).p$$

Thus the set $\{1, 2, \ldots, p^k\}$ contains exactly $p^k - p^{k-1}$ integers that are relatively prime to p^k and so by the definition of Euler's phi-function, $\phi(p^k) = p^k - p^{k-1}$.

Lemma 1.5.3.[3] *Given integers a, b, c; $gcd(a, bc) = 1$ iff $gcd(a, b) = 1$ and $gcd(a, c) = 1$.*

Theorem 1.5.4.[3] *The function ϕ is a multiplicative function.*

Theorem 1.5.5.[3] *If the integer $n > 1$ has the prime factorization $n = p_1^{k_1} p_2^{k_2} \ldots p_r^{k_r}$, then*

$$\begin{aligned}\phi(n) &= (p_1^{k_1} - p_1^{k_1-1})(p_2^{k_2} - p_2^{k_2-1}) \ldots (p_r^{k_r} - p_r^{k_r-1}) \\ &= n\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right) \cdots \left(1 - \frac{1}{p_r}\right)\end{aligned}$$

Proof: We will prove the above result by induction on r. Clearly, the result is true for $r = 1$. Suppose it holds for $r = i$. As

$$gcd(p_1^{k_1} p_2^{k_2} \ldots p_i^{k_i}, p_{i+1}^{k_{i+1}}) = 1$$

the definition of multiplicative function gives

$$\begin{aligned}\phi((p_1^{k_1} p_2^{k_2} \ldots p_i^{k_i}) p_{i+1}^{k_{i+1}}) &= \phi(p_1^{k_1} p_2^{k_2} \ldots p_i^{k_i}).\phi(p_{i+1}^{k_{i+1}}) \\ &= \phi(p_1^{k_1} p_2^{k_2} \ldots p_i^{k_i})(p_{i+1}^{k_{i+1}} - p_{i+1}^{k_{i+1}-1})\end{aligned}$$

By the induction hypothesis, the first factor on the R.H.S. becomes

$$\phi(p_1^{k_1}p_2^{k_2}\dots p_i^{k_i}) = (p_1^{k_1} - p_1^{k_1-1})(p_2^{k_2} - p_2^{k_2-1})\cdots(p_i^{k_i} - p_i^{k_i-1})$$

and this completes the induction steps.

Theorem 1.5.6.[3] *For $n > 2$, $\phi(n)$ is an even integer.*

Proof: First assume that n is a power of 2. Let's say that $n = 2^k$, with $k \geq 2$. Then

$$\phi(n) = \phi(2^k) = 2^k\left(1 - \frac{1}{2}\right) = 2^{k-1}$$

an even integer. If n is not a power of 2, then it is divisible by an odd prime p, we therefore may write n as $n = p^k m$, where $k \geq 1$ and $gcd(p^k, m) = 1$. Applying the multiplicativity of phi-function we get $\phi(n) = \phi(p^k).\phi(m) = p^{k-1}(p-1).\phi(m)$ which again is even because $2|(p-1)$.

Lemma 1.5.7.[3] *Let $n > 1$ and $gcd(a, n) = 1$. If a_1, a_2, ..., $a_{\phi(n)}$ are the positive integers less than n and relatively prime to n, then*

$$aa_1, aa_2, \dots, aa_{\phi(n)}$$

are congruent modulo n to a_1, a_2, ..., $a_{\phi(n)}$ in some order.

Proof: Note that no two of the integers $aa_1, aa_2, \dots, aa_{\phi(n)}$ are congruent modulo n. For if $aa_i \equiv aa_j (\text{mod } n)$ with $1 \leq i < j \leq \phi(n)$, then the cancellation law yields $a_i \equiv a_j (\text{mod } n)$, and thus $a_i = a_j$, which is a contradiction. Further as $gcd(a_i, n) = 1$, for all i and $gcd(a, n) = 1$, each aa_i is relatively prime to n. Fixing on a particular aa_i, there exists a unique integer b, where $0 \leq b < n$, for which $aa_i \equiv b (\text{mod } n)$. Since $gcd(b, n) = gcd(aa_i, n) = 1$, b must be one of the integers a_1, a_2, ..., $a_{\phi(n)}$. This proves that the numbers $aa_1, aa_2, \dots, aa_{\phi(n)}$ and the numbers a_1, a_2, ..., $a_{\phi(n)}$ are identical (modulo n) in a certain order.

Theorem 1.5.8(Euler)[3]. *If $n \geq 1$ and $gcd(a, n) = 1$, then $a^{\phi(n)} \equiv 1 (mod\, n)$.*

Proof: Let $n > 1$. Let a_1, a_2, ..., $a_{\phi(n)}$ be the positive integers less than n that are relatively prime to n. Since $gcd(a, n) = 1$, it follows from the lemma that $aa_1, aa_2, \dots, aa_{\phi(n)}$ are congruent, not necessarily in the order of appearance, to a_1, a_2, ..., $a_{\phi(n)}$. Then

$$aa_1 \equiv a_1' (\text{mod } n)$$

$$aa_2 \equiv a_2' (\text{mod } n)$$

$$\vdots$$

$$aa_{\phi(n)} \equiv a'_{\phi(n)} (\text{mod } n)$$

where a'_1, a'_2, ..., $a'_{\phi(n)}$ are integers a_1, a_2, ..., $a_{\phi(n)}$ in some order. On taking the product of these $\phi(n)$ congruences, we get

$$\begin{aligned}(aa_1)(aa_2)\dots(aa_{\phi(n)}) &\equiv a'_1 a'_2 \cdots a'_{\phi(n)} (\text{mod } n) \\ &\equiv a_1 a_2 \cdots a_{\phi(n)} (\text{mod } n)\end{aligned}$$

and this implies that

$$a_{\phi(n)}(a_1 a_2 \dots a_{\phi(n)}) \equiv a_1 a_2 \cdots a_{\phi(n)} (\text{mod } n).$$

Since $gcd(a_i, n) = 1$, for each i, we have $gcd(a_1 a_2 \dots a_{\phi(n)}, n) = 1$. Therefore dividing both sides of the above congruence by $a_1 a_2 \dots a_{\phi(n)}$ we get, $a^{\phi(n)} \equiv 1 (\text{mod } n)$.

Corollary 1.5.9 (Fermat)[3]. *If p is a prime such that p does not divide a, then $a^{p-1} \equiv 1 (mod\, p)$.*

Theorem 1.5.10 (Gauss)[3]. *For each positive integer $n \geq 1$,*

$$n = \sum_{d|n} \phi(d)$$

the sum being extended over all positive divisors of n.

Proof: The integers between 1 and n can be separated into classes as follows: If d is a positive divisor of n, we put the integer m in the class S_d provided that $gcd(m, n) = d$. i.e.

$$S_d = \{m : gcd(m, n) = d;\ 1 \leq m \leq n\}$$

Now $gcd(m, n) = d$ iff $gcd(m/d, n/d) = 1$. Thus the number of integers in the class S_d is equal to the number of positive integers not exceeding n/d that are relatively prime to n/d, or equal to $\phi(n/d)$. Since each of the n integers in the set $\{1, 2, \dots, n\}$ lies in exactly one class S_d, we get

$$n = \sum_{d|n} \phi(n/d)$$

But as d runs through all positive divisors of n so does n/d; hence

$$\sum_{d|n} \phi(n/d) = \sum_{d|n} \phi(d).$$

Theorem 1.5.11.[3] *For $n > 1$, the sum of the positive integers less than n and relatively prime to n is $\frac{1}{2}n\phi(n)$.*

Proof. Let $a_1, a_2, \cdots, a_{\phi(n)}$ be the positive integers less than n and relatively prime to n. Now since $gcd(a, n) = 1$ iff $gcd(n - a, n) = 1$, the numbers $n - a_1$, $n - a_2$, $\cdots$, $n - a_{\phi(n)}$ are equal in some order to a_1, a_2, $\ldots$, $a_{\phi(n)}$. Thus

$$\begin{aligned} a_1 + a_2 + \cdots + a_{\phi(n)} &= (n - a_1) + (n - a_2) + \cdots + (n - a_{\phi(n)}) \\ &= \phi(n)n - (a_1 + a_2 + \cdots + a_{\phi(n)}) \end{aligned}$$

Hence, $2(a_1 + a_2 + \cdots + a_{\phi(n)}) = \phi(n)n$.

Chapter 2

Cryptography

2.1. Introduction

Cryptography (from the Greek kryptos means hidden and graphein means to write) provides practical means of protecting information transmitted through public communication networks, such as those using telephone lines, microwaves or satellites etc. In cryptography, codes are called *ciphers*, the information to be concealed is called *plaintext* and, after transformation to a secret form a message is called *ciphertext*.

Both the plaintext and the ciphertext are written in terms of elements from a finite set $\mathcal{A}$, called an *alphabet of definition*. The alphabet of definition may consist of numbers, letters from an alphabet such as the English, Greek, or Russian alphabets, or symbols such as !, @, $*$, or any other symbols that we choose to use when sending messages. The alphabet of definition for the plaintext and ciphertext may differ, but the usual convention is to use the same for both. A *message space*, $\mathcal{M}$, is defined to be a finite set consisting of strings of symbols from the alphabet of definition. Elements of $\mathcal{M}$, which may be anything from binary strings to English text, are called *plaintext message units*. A finite set $\mathcal{C}$, consisting of strings of symbols from an alphabet of definition for the ciphertext, is called the *ciphertext space*, and elements from $\mathcal{C}$ are called *ciphertext message units*. Let $\mathcal{K}$ be a set of parameters, called the *keyspace*, and elements of $\mathcal{K}$ are called keys.

Definition 2.1.1.[4] An *enciphering transformation* (or, *enciphering function*) is a bijective function

$$E_e : \mathcal{M} \to \mathcal{C},$$

where the key $e \in \mathcal{K}$ uniquely determines E_e acting upon plaintext message units $m \in \mathcal{M}$ to get ciphertext message units

$$E_e(m) = c \in \mathcal{C}.$$

A *deciphering transformation* (or, *deciphering function*) is a bijective function

$$D_d : \mathcal{C} \to \mathcal{M},$$

which is uniquely determined by a given key $d \in \mathcal{K}$, acting upon ciphertext message units $c \in \mathcal{C}$ to get plaintext message units

$$D_d(c) = m.$$

The application of E_e to m, is called *enciphering, encoding, or encrypting* $m \in \mathcal{M}$, whereas the application of D_d to c is called *deciphering, decoding, or decrypting* $c \in \mathcal{C}$.

For example, let N=letter alphabets with numerical equivalents $0, 1, 2 \ldots .(N-1)$, b=fixed integer, and f=a shift transformation, that is, the enchiphering function defined by the rule

$$C = f(P) \equiv P + b \pmod{N},$$

where P represents the plaintext and C is the ciphertext.

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

Let the plaintext message be

THE GOD IS GREAT

Using the congruence theory and the enchiphering function defined by

$$C \equiv P + 3 \pmod{26},$$

where P is the digital equivalent of a plaintext letter and C the digital equivalent of the ciphertext, the letters of the message in the above equation are converted to their equivalents:

19 07 04 06 14 03 08 18 06 17 04 00 19

Thus, the ciphertext for the above plaintext message becomes

22 10 07 09 17 06 11 21 09 20 07 03 22

i.e

$$WKH \;\; JRG \;\; LV \;\; JUHDW.$$

To recover the plaintext, the procedure is simply reversed by means of the congruences

$$P \equiv C - 3 \pmod{26} \equiv C + 23 \pmod{26}.$$

Definition 2.1.2.[4] A *cryptosystem* is composed of a set $\{E_e : e \in \mathcal{K}\}$ consisting of enciphering transformations and the corresponding set $\{E_e^{-1} : e \in \mathcal{K}\} = \{D_d : d \in \mathcal{K}\}$ of deciphering transformations. In other words, for each $e \in \mathcal{K}$, there exists a unique $d \in \mathcal{K}$ such that $D_d = E_e^{-1}$, so that $D_d(E_e(m)) = m$ for all $m \in \mathcal{M}$. The keys (e, d) are called a key pair where possibly $e = d$.

2.2. Secret Key Cryptography

With secret key cryptography, a single key is used for both encryption and decryption. The sender uses the key (or some set of rules) to encrypt the plaintext and sends the ciphertext to the receiver. The receiver applies the same key (or ruleset) to decrypt the message and recover the plaintext. Because a single key is used for both functions, secret key cryptography is also called *symmetric encryption*. With this form of cryptography, it is obvious that the key must be known to both the sender and the receiver; that, in fact, is the secret. The biggest difficulty with this approach, of course, is the distribution of the key. Secret key cryptography schemes are generally categorized as being either *stream ciphers* or *block ciphers*.

Definition 2.2.1.[4] A *Block Cipher* is a cryptosystem that separates the plaintext message into strings, called *blocks*, of fixed length $k \in \mathbb{N}$, called the *blocklength*, and enciphers one block at a time.

Classically, block ciphers are divided into two types, *substitution* and *transposition ciphers*. A substitution cipher replaces plaintext symbols with other symbols to produce ciphertext. As an example, the plaintext might be palace, and the ciphertext might be $QZYZXW$ when a, c, e, l, p are replaced by Z, X, W, Y, Q, respectively. With a transposition cipher we permute the places where the plaintext letters sit. That is, we do not change the letters but rather move them around, transpose them, without introducing any new letters.

Definition 2.2.2.[4] A *simple transposition cipher*, also known as a *simple permutation cipher*, is a symmetric-key block cryptosystem having blocklength $r \in \mathbb{N}$, with keyspace $\mathcal{K}$ being the set of permutations on $\{1, 2, \cdots, r\}$. The enciphering transformation is defined, for each $m = (m_1, m_2 \cdots, m_r) \in \mathcal{M}$, and given $e \in \mathcal{K}$, by $E_e(m) = (m_{e(1)}, m_{e(2)}, \cdots, m_{e(r)})$, and for each $c = (c_1, c_2, \cdots, c_r) \in \mathcal{C}$, $D_d(c) = D_{e^{-1}}(c) = (c_{d(1)}, c_{d(2)}, \cdots, c_{d(r)})$.

The cryptosystems in the above definition have keyspace of cardinality $|\mathcal{K}| = r!$. Permutation encryption involves grouping plaintext into blocks of r symbols and applying to each block the permutation e on the numbers $1, 2, \cdots, r$.

Definition 2.2.3.[4] Let $\mathcal{A}$ be an alphabet of definition consisting of n symbols, and let $\mathcal{M}$ be the set of all blocks of length r over $\mathcal{A}$. The keyspace $\mathcal{K}$ will consist of all ordered r-tuples $e = (\sigma_1, \sigma_2, \cdots, \sigma_r)$ of permutations σ_j on $\mathcal{A}$. For each $e \in \mathcal{K}$, and $m = (m_1, m_2, \cdots, m_r) \in \mathcal{M}$ let $E(m) = (\sigma_1(m_1), \sigma_2(m_2), \cdots, \sigma_r(m_r)) = (c_1, c_2, \cdots, c_r) = c \in \mathcal{C}$, and for $d = (d_1, d_2, \cdots, d_r) = (\sigma_1^{-1}, \sigma_2^{-1}, \cdots, \sigma_r^{-1}) = \sigma^{-1}$, $D_d(c) = (d_1(c_1), d_2(c_2), \cdots, d_r(c_r)) = (\sigma_1^{-1}(c_1), \sigma_2^{-1}(c_2), \cdots, \sigma_r^{-1}(c_r)) = m$. This type of cryptosystem is called a *substitution cipher*. If all keys are the same, namely, $\sigma_1, \sigma_2, \cdots, \sigma_r$, then this cryptosystem is called a *simple substitution cipher* or *monoalphabebetic substitution cipher*. If the keys differ, then it is called a *polyalphabetic substitution cipher*.

Definition 2.2.4.[4] *Affine Cipher* is also a type of block cipher. Let $a, b, n \in \mathbb{N}$ and for $m \in \mathbb{Z}$ define

$$E_e(m) = am + b(mod\ n),$$

where the key e is the ordered pair (a, b). Notice that for $a = 1$ we have $E_e(m) = m + b(mod\ n)$, the *Shift Cipher*, where the key is b. Such a transformation is called an *Affine function*. In order to guarantee that the deciphering transformation exists, we need to know that the inverse of the affine function exists. This means that $f^{-1}(c) \equiv a^{-1}(c - b)(\text{mod } n)$ must exist and this can happen only if $gcd(a, n) = 1$. We know that there are $\phi(n)$ natural numbers less than n and relatively prime to it. Hence, since b can be any of the choices of natural numbers less than n, there are exactly $n\phi(n)$ possible Affine Ciphers, the product of the

possible choices for a with the number for b, since this is the total number of possible keys. Thus we have

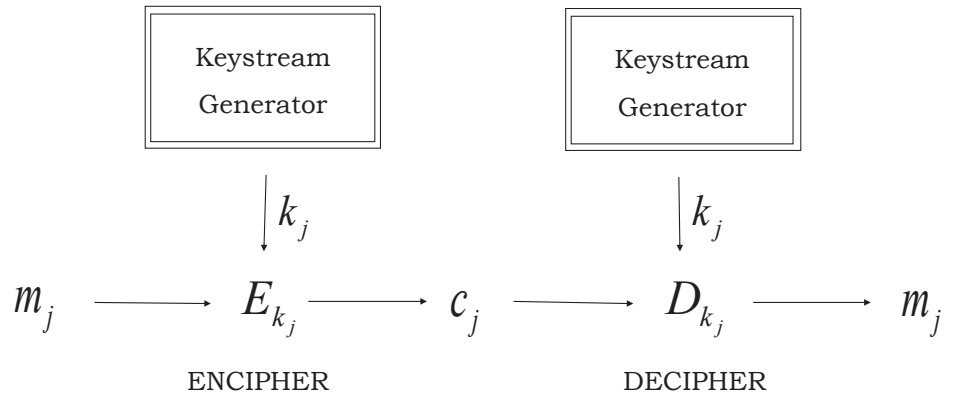
Let $\mathcal{M} = \mathcal{C} = \mathbb{Z}/n\mathbb{Z}$, $n \in \mathbb{N}$, $\mathcal{K} = \{(a,b) : a,b \in \mathbb{Z}/n\mathbb{Z}$ *and* $gcd(a,n) = 1\}$, *and for* $e,d \in \mathcal{K}$, *and* $m,c \in \mathbb{Z}/n\mathbb{Z}$, *let* $E_e(m) \equiv am + b(mod\ n)$, *then* $D_d(c) \equiv a^{-1}(c-b)(mod\ n)$.

Thus, $e = (a,b)$ since e is multiplication by a followed by addition of b *modulo* n, and $d = (a^{-1}, -b)$ is subtraction of b followed by multiplication with a^{-1}. In the case of the Shift Cipher, the inverse is additive and in the case of the Affine Cipher, the inverse is multiplicative. Of course, these coincide precisely when $a = 1$. In either case, knowing e or d allows us to easily determine the other, so they are symmetric-key cryptosystems. They are also Block Ciphers with the trivial blocklengths of $k = 1$.

Definition 2.2.5.[4] *Stream ciphers* operate on a single bit (byte or computer word) at a time and implement some form of feedback mechanism so that the key is constantly changing. A block cipher is so-called because the scheme encrypts one block of data at a time using the same key on each block. In general, the same plaintext block will always encrypt to the same ciphertext when using the same key in a block cipher whereas the same plaintext will encrypt to different ciphertext in a stream cipher.

The following is the simplest flow chart for a stream cipher.

Figure 1: A Stream Cipher



Definition 2.2.6.[4] If $\mathcal{K}$ is the keyspace for a set of enciphering transformations, then a sequence $k_1k_2\cdots \in \mathcal{K}$ is called a *keystream*. A keystream is either randomly chosen or generated by an algorithm, called a *keystream generator*, which generates the keystream from an initial small input keystream called a *seed*. Keystream generators that eventually repeat their output are called *periodic*.

The *Vernam cipher* is a *stream cipher* with alphabet of definition $A = \{0, 1\}$ that enciphers in the following fashion. Given a bitstring

$$m_1m_2 \ldots m_n \in \mathcal{M}$$

and a key stream

$$k_1k_2 \ldots k_n \in \mathcal{K}$$

the enciphering transformation is given by

$$E_{k_j}(m_j) = m_j + k_j = c_j \in \mathcal{C},$$

and the deciphering transformation is given by

$$D_{k_j}(c_j) = c_j + k_j = m_j,$$

where + is addition modulo 2. The keystream is randomly chosen and never used again. For this reason, the Vernam cipher is also called the one-time pad.

Example 2.2.7.[4]
Let $n = |\mathcal{A}|$ where $\mathcal{A}$ is the alphabet of definition. We call

$$k_1k_2 \ldots k_r \qquad \text{for } \ 1 \leq r \leq n$$

a priming key. Then given a plaintext message unit

$$m = (m_1, m_2, \ldots, m_s) \quad where \quad s > r,$$

we generate a keystream as follows.

$$k = k_1k_2 \ldots k_rm_1m_2 \ldots m_{s-r}.$$

Then we encipher via:

$$E_{k_j}(m_j) = m_j + k_j \ (mod\ n) = c_j, \quad \text{for} \quad j = 1, 2, \ldots r, \text{ and}$$

$$E_{k_j}(m_j) = m_j + m_{j-r} \ (mod\ n) = c_j, \quad \text{for} \quad j > r,$$

and we decipher via

$$D_{k_j}(c_j) = c_j - k_j \ (mod\ n) = m_j, \quad \text{for} \quad j = 1, 2, \ldots r, \text{ and}$$

$$D_{k_j}(c_j) = c_j - m_{j-r} \ (mod\ n) = m_j \quad \text{for} \quad j > r.$$

This cryptosystem is non-synchronous since the plaintext serves as the key, from the $(r+1)^{st}$ position onwards, with the simplest case being $r = 1$.

Definition 2.2.8.[4] A stream cipher is said to be *synchronous* if the keystream is generated without use of either the plaintext or of the ciphertext, called keystream generation independent of the plaintext and ciphertext. A stream cipher is called *self synchronizing* (or *asynchronous*) if the keystream is generated as a function of the key and a fixed number of previous ciphertext units. If the stream cipher utilizes plaintext in the keystream generation, then it is called *non synchronous*.

The following two flow charts illustrate a general *synchronous* and a general *asynchronous* cipher respectively.

Figure 2: A Synchronous Stream Cipher

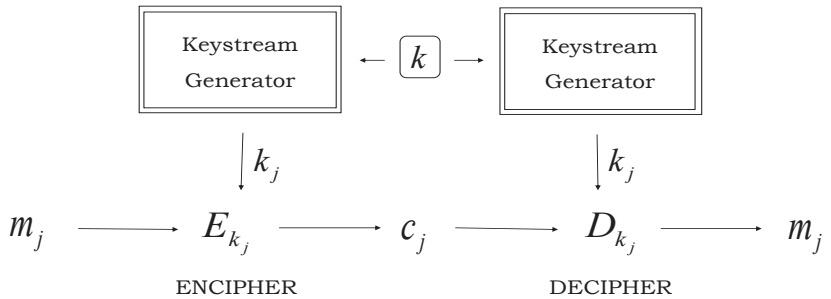
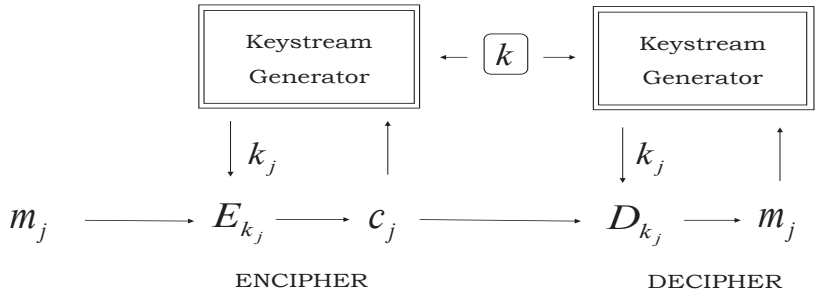


Figure 3: A Asynchronous Stream Cipher



Data Encryption Standard (DES):[4] The best-known symmetric-key block cipher, (now replaced by the Advanced Encryption Standard), is the Data Encryption Standard (DES). We will begin with an overview of the mechanisms behind S-DES, which is a simplified version of DES. In practice, any text to be sent is first converted to a string of numbers, for example by assigning the numerical ASCII codes that correspond to ordinary keyboard characters. These are then written in binary (base 2) notation, so that the text becomes a string of 0's and 1's.

Algorithm for S-DES encryption:[4]

(i) First the message to be encrypted is divided into blocks of 8-bits. Let us denote each such block by m. The key k used in the encryption process has bitlength 10.

(ii) From the 10-bit key k generate two subkeys k_1 and k_2 of 8-bits each.

(iii) For k_1 apply a permutation P_{10} (of 10 symbols) to k, divide it into two equal parts 5-bits each and apply left-shift by 1 on these two sets of 5 bits. To the resulting 10-bits apply the permutation P_8 (of 8 symbols). This will generate the first subkey k_1 of 8-bits.

(iv) For k_2 start from the two sets of 5-bits obtained after applying left-shift by 1 and on both apply left shift by 2, then P_8 to obtain the second subkey k_2 of 8-bits.

(v) Apply an initial permutation IP to m.

(vi) Then divide the resulting 8-bits into two equal parts. Let $L(t)$ represents the first 4-bits and $R(t)$ represents the remaining 4-bits.

(vii) Apply the first round function $f_{k_1}(t) = (L(t) \oplus F(R(t), k_1), R(t))$. Here $\oplus$ denotes sum modulo 2. The function F first uses the expansion permutation EP to convert 4-bit input $R(t)$ into 8-bit output. Then add this 8-bit output to the subkey k_1 modulo 2. Denote the first four bits of this result by $L(y)$ and the remaining four bits by $R(y)$. Now apply the S-boxes S_0 and S_1 to $L(y)$ and $R(y)$, respectively. This process will give us 4-bits. At last apply the permutation P_4 to get the final output of the function F.

(viii) Now apply the switch function SW which swaps the set of first four bits and the set of remaining four bits of the output of f_{k_1}.

(ix) Divide the resulting 8-bits into two equal parts. Let $L(t)$ represents the first 4-bits and $R(t)$ represents the remaining 4-bits. Apply the second round function $f_{k_2}(t) = (L(t) \oplus F(R(t), k_2), R(t))$ as described above by using the second subkey k_2.

(x) At last apply the inverse of the initial permutation IP^{-1} to get the ciphertext c.

Algorithm for S-DES decryption:

(i) Apply the initial permutation IP to c.

(ii) Then divide the resulting 8-bits into two equal parts. Let $L(t)$ represents the first 4-bits and $R(t)$ represents the remaining 4-bits.

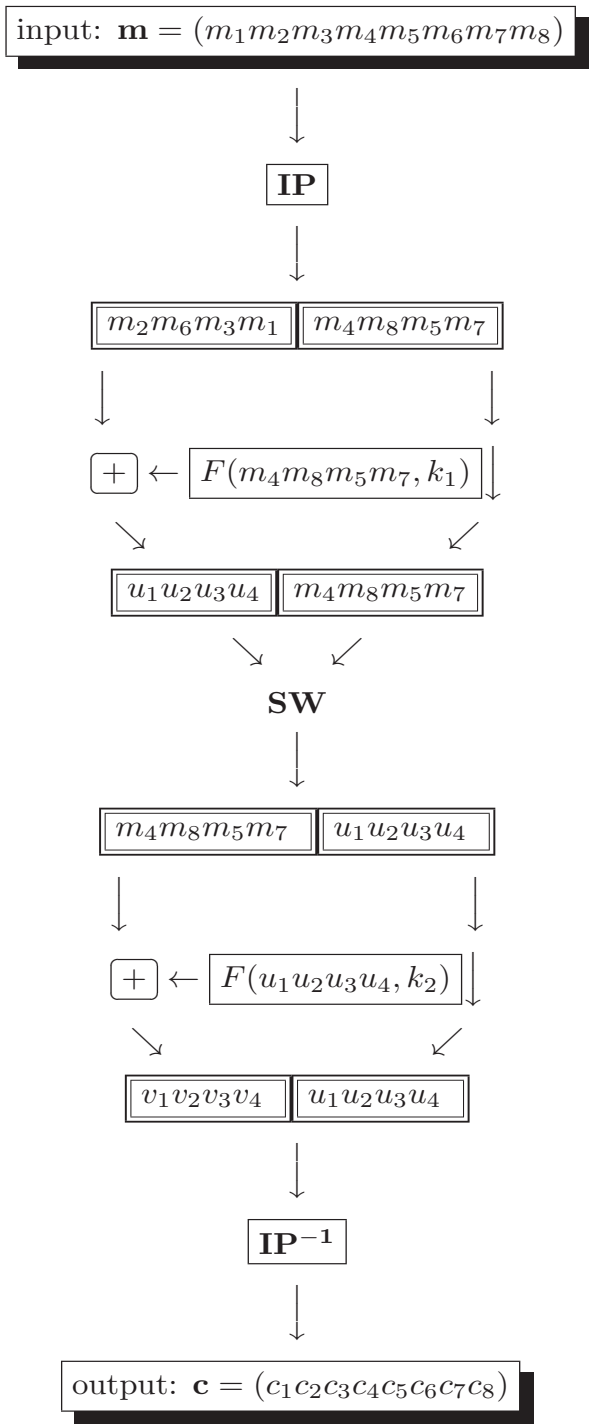
(iii) Apply the second round function $f_{k_2}(t) = (L(t) \oplus F(R(t), k_2), R(t))$.

(iv) Now apply the switch function SW which swaps the set of first four bits and the set of remaining four bits of the output of f_{k_2}.

(v) Divide the resulting 8-bits into two equal parts. Let $L(t)$ represents the first 4-bits and $R(t)$ represents the remaining 4-bits. Apply the first round function $f_{k_1}(t) = (L(t) \oplus F(R(t), k_1), R(t))$.

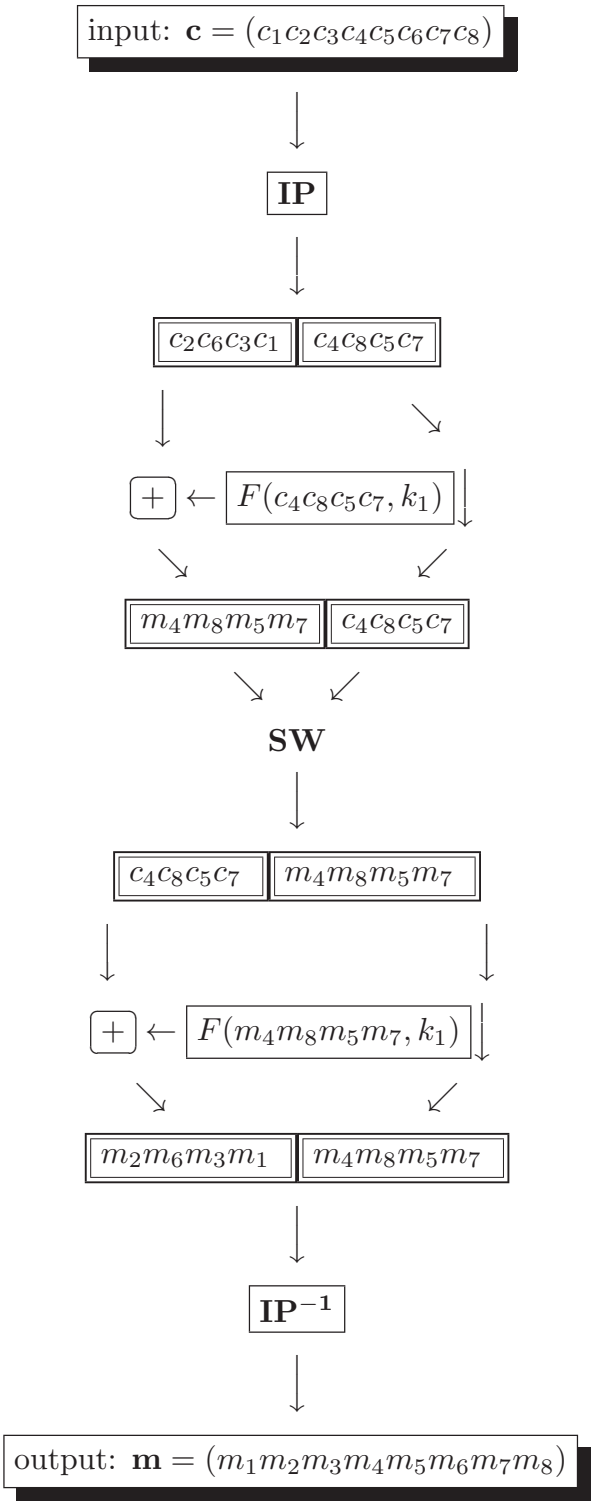
(vi) At last apply the inverse of the initial permutation IP^{-1} to get the original plaintext m.

The S-DES Encryption Flowchart



The action between **IP** and **SW** is round 1, namely, the execution of f_{k_1}, and the action between **SW** and $\mathbf{IP}^{-1}$ is round 2, the action of f_{k_2}.

The S-DES Decryption Flowchart



Example 2.2.9.[4] Suppose we are given plaintext bitstring $m = (10100101)$ and key bitstring $k = (0010010111)$. Suppose that IP(Initial Permutation), EP(Expansion Permutation), P_{10}, P_8, P_4, IP^{-1} and the two S-boxes, S_0 and S_1 are given as follows:

IP

j	1	2	3	4	5	6	7	8
$IP(j)$	2	6	3	1	4	8	5	7

P_{10}

j	1	2	3	4	5	6	7	8	9	10
$P_{10}(j)$	3	5	2	7	4	10	1	9	8	6

P_8

j	1	2	3	4	5	6	7	8
$P_8(j)$	6	3	7	4	8	5	10	9

P_4

j	1	2	3	4
$P_4(j)$	2	4	3	1

EP

j	1	2	3	4	5	6	7	8
$EP(j)$	4	1	2	3	2	3	4	1

IP^{-1}

j	1	2	3	4	5	6	7	8
$IP^{-1}(j)$	4	1	3	5	7	2	8	6

$\mathbf{S_0}$		x_2	0	0	1	1
		x_3	0	1	0	1
x_1	x_4					
0	0		01	00	11	10
0	1		11	10	01	00
1	0		00	10	01	11
1	1		00	01	11	10

and

$\mathbf{S_1}$		x_2	0	0	1	1
		x_3	0	1	0	1
x_1	x_4					
0	0		00	01	10	11
0	1		10	00	01	11
1	0		11	00	01	10
1	1		10	01	00	11

These S-Boxes or substitution boxes for S-DES are four-by-four matrices with entries from $\mathbb{Z}/4\mathbb{Z}$ (put into binary) with rows and columns labelled from 0 to 3 (put into binary) that take a 4-bit input and output a 2-bit string as follows. If $(x_1x_2x_3x_4) = (1101)$ is our input bitstring of length 4, then by using the first S-Box, we get $S_0(1101) = (11)$, since $(x_1x_4) = (11)$ represents the fourth row, and $(x_2x_3) = (10)$ represents the third column, the entry at the intersection of which is (11). Similarly, if we want to use the S-Box S_1, then $S_1(1101) = (00)$.

First we generate our subkeys as follows:

(1) $P_{10}(k) = 1000010111$.

(2) $LS1(10000) = (00001)$ and $LS1(10111) = (01111)$.

(3) $P_8(0000101111) = (00101111) = k_1$.

(4) $LS2(00001) = (00100)$ and $LS2(01111) = (11101)$ (applying $LS2$ to the output of step 2)

(5) $P_8(0010011101) = (11101010) = k_2$ (applying P_8 to the output of step 4).

Now we encrypt as follows. First we calculate $IP(m) = (01110100)$. Then we will calculate the round function for the first round $f_{k_1}(01110100) = (L(01110100) \oplus F(R(01110100), k_1), R(01110100))$.

(1) $EP(0100) = (00101000)$.

(2) $EP(0100) \oplus k_1 = (00101000) \oplus (00101111) = (00000111)$.

(3) $S_0(0000) = (01)$ *and* $S_1(0111) = (11)$.

(4) $P_4(0111) = (1110) = F(R(01110100), k_1)$.

(5) $L(01110100) \oplus F(R(01110100), k_1) = (0111) \oplus (1110) = (1001)$.

(6) $f_{k_1}(01110100) = (10010100)$.

Now we apply the switch function, $SW(10010100) = (01001001)$. Similarly,

$$f_{k_2}(01001001) = (L(01001001) \oplus F(R(01001001), k_2), R(01001001)) = (01101001)$$

At last, we apply the inverse of the initial permutation, $IP^{-1}(01101001) = (00110110)$, which is the ciphertext.

To decrypt, we reverse the process. First feed c into IP to get

$$IP(c) = (01101001),$$

then apply f_{k_2} to get

$$f_{k_2}(0110 \oplus F(1001, k_2), 1001) = (01001001).$$

Then $SW(01001001) = (10010100)$. Next,

$$f_{k_1}(1001 \oplus F(0100, k_1), 0100) = (01110100),$$

then the final application yields the original plaintext, $IP^{-1}(01110100) = (10100101) = m$.

Schaefer relabelled S-DES as baby DES since it is a much simpler block cipher than DES. In terms of composition of functions, all of the above discussion of S-DES can be combined as follows.

$$(IP^{-1} \circ f_{k_2} \circ SW \circ f_{k_1} \circ IP)(m) = IP^{-1}(f_{k_2}(SW(f_{k_1}(IP(m))))) = c.$$

Full DES takes 64-bit plaintext blocks, a 56-bit key, from which sixteen 48-bit subkeys are generated, and correspondingly there are sixteen round functions f_{k_j} for $j = 1, 2, \cdots, 16$. Hence, we may specify (full) DES now as a single composition of functions.

$$(IP^{-1} \circ f_{k_{16}} \circ SW \circ f_{k_{15}} \circ SW \circ \cdots \circ f_{k_1} \circ IP)(m) = c.$$

Moreover, in DES, we have eight S-Boxes S_j for $j = 1, 2, \cdots, 8$, each having four rows and sixteen columns, where $S_j(m_1m_2m_3m_4m_5m_6)$ picks out the entry in row (m_1m_6) and column $(m_2m_3m_4m_5)$, which represents sixteen possible entries, in binary, for each such row. Also, P_4 in S-DES, is replaced by P_{32} in DES, which is half the bitlength of the input in either case.

2.3. Public Key Cryptography

In Public key cryptography encryption and decryption are carried out using two different keys. The two keys in such a key pair are referred to as the public key and the private key. Public-key cryptography is also known as asymmetric-key cryptography.

Party A, if wanting to communicate confidentially with party B, can encrypt a message using B's publicly available key. Such a communication would only be decipherable by B as only B would have access to the corresponding private key. Party A, if wanting to send an authenticated message to party B, would encrypt the message with A's own private key. Since this message would only be decipherable with A's public key, that would establish the authenticity of the message meaning that A was indeed the source of the message. The public-key encryption can be used to provide both confidentiality and authentication at the same time. Note that confidentiality means that we want to protect a message from eavesdroppers and authentication means that the recipient needs a guarantee as to the identity of the sender.

A's public and private keys are designated as PU_A and PR_A. B's public and private keys are designated as PU_B and PR_B. Suppose that A wants to send a message M to B with both authentication and confidentiality. The processing steps undertaken by A to convert M into its encrypted form C that can be placed on the wire are:

$$C = E(PU_B, E(PR_A, M))$$

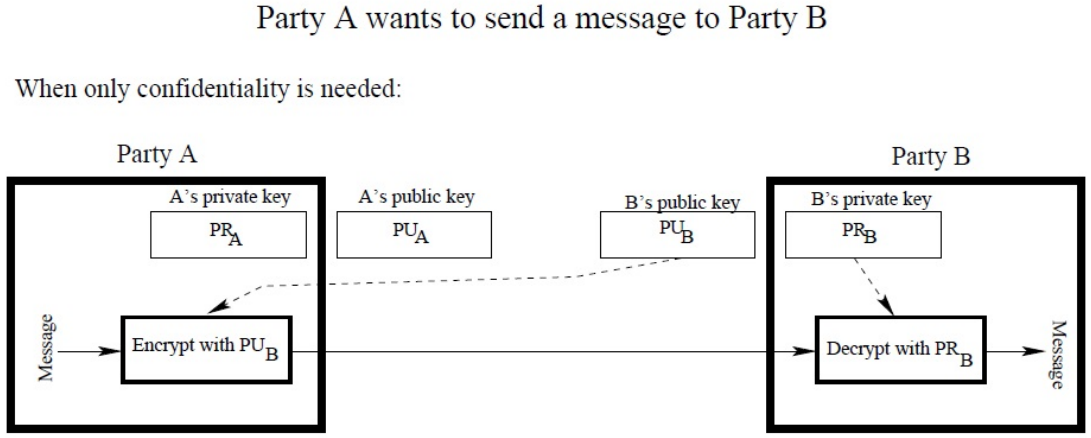
where $E()$ stands for encryption. The processing steps undertaken by B to recover M from C are

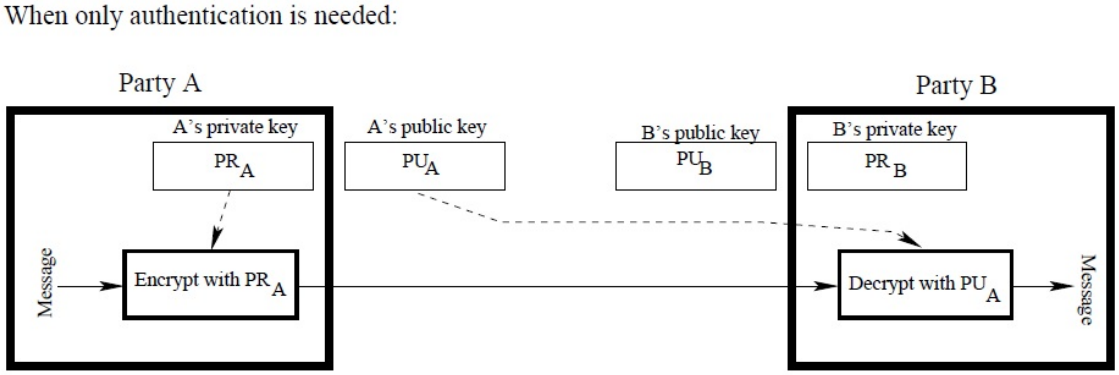
$$M = D(PU_A, D(PR_B, C))$$

where $D()$ stands for decryption.

The sender A encrypting his/her message with its own private key PR_A provides authentication. The sender A further encrypting his/her message with the receivers public key PU_B

provides confidentiality.

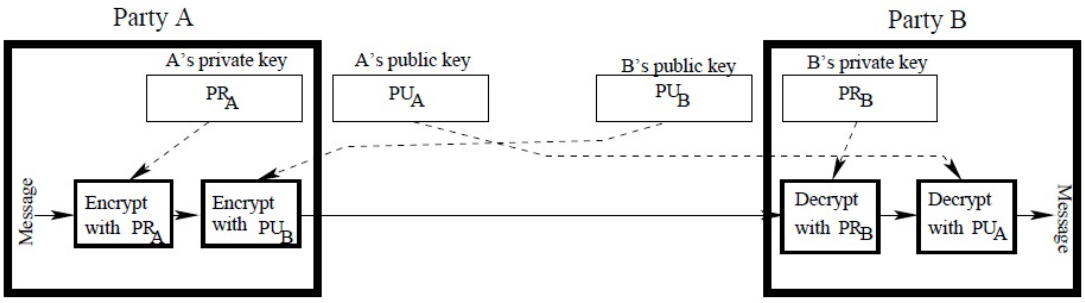




The Rivest-Shamir-Adleman (RSA) Algorithm:[1]

The RSA is one of the first practicable public key cryptosystems and widely used for secure data transmission. In such a cryptosystem, the encryption key is public and differs from the decryption key which is kept secrete. In RSA, the asymmetric key is based on the practical difficulty of factoring the product of two large prime numbers. Considering arithmetic modulo n, let's say that e is an integer that is co-prime to the totient $\varphi(n)$ of n. Further, let d be the multiplicative inverse of e *modulo* $\varphi(n)$.

When both confidentiality and authentication are needed:



An individual A who wishes to receive messages confidentially will use the pair of integers (e, n) as his/her public key. At the same time, this individual can use the pair of integers (d, n) as the private key. The definitions of n, e, and d are as given above. Another party B wishing to send a message M to A confidentially will encrypt M using A's public key (e, n) to create cipher text C. Subsequently, only A will be able to decrypt C using his/her private key (d, n). If the plain text message M is too long, B may choose to use RSA as a block cipher for encrypting the message meant for A. When RSA is used as a block cipher, the block size is likely to be half the number of bits required to represent the modulus n. If the modulus required, say, 1024 bits for its representation, message encryption would be based on 512-bit blocks.

RSA ALGORITHM:[1]

The RSA algorithm involves 3 steps:

(1) Key Generation

(2) Encryption

(3) Decryption

(1) Key Generation:

Step-1: Choose two distinct large prime numbers p and q. For security purpose, the integers p and q should be chosen at random and should be of similar bit length.

Step-2: Compute $n = pq$. n is used as the modulus of both public and private key.

Step-3: Compute $\varphi(n) = \varphi(p)\varphi(q) = (p-1)(q-1) = n-(p+q-1)$, where φ is Euler totient

function.

Step-4: Choose an integer e such that $1 < e < \varphi(n)$ and $gcd(e, \varphi(n))) = 1$ i.e e and $\varphi(n)$ are coprime.

Step-5: Determine d, the multiplicative inverse of $e(\text{mod}\, \varphi(n))$.

(2) Encryption: Let A transmits her public key (e, n) to B and keeps the private key secret. B then wishes to send message M to A. He first turns M into m such that $0 \leq m < n$. Then he computes the cipher text C corresponding to $C \equiv m^e (mod\ n)$.

(3) Decryption: A can recover m from C by using her private key exponent via computing

$$m \equiv C^d (mod\ n)$$

Example 2.3.1. [1]

(1) Choose two distinct prime numbers such as $p = 61$ and $q = 53$.

(2) Then $n = p \times q$ gives $n = 61 \times 53 = 3233$.

(3) The totient of the product given by $\varphi(n) = (p-1) \times (q-1)$ is

$$\varphi(3233) = (61-1) \times (53-1) = 3120.$$

(4) Choose any number $1 < e < 3120$ that is co-prime to 3120. Let $e = 17$.

(5) Then d, the modular multiplicative inverse of $e(\text{mod}\varphi(n))$ yields

$$d \equiv e^{-1} (\text{mod}\, \varphi(n))$$

$$d \equiv 17^{-1} (\text{mod}\, \varphi(3120))$$

$$d = 2753$$

The public key is $(n = 3233, e = 17)$. For plain text message m, the encryption function is

$$C(m) = m^{17} (\text{mod}\, 3233).$$

The private key is $(n = 3233, d = 2753)$. For an encrypted cipher text C, the decryption function is

$$m(C) = m^{17} (\text{mod}\, 3233)$$

For instant; in order to encrypt $m = 65$, $C = 65^{17} (\text{mod}\, 3233) = 2790$. To decrypt $C = 2790$, we calculate $m = 2790^{2753} (\text{mod}\, 3233) = 65$.

Digital Signature Algorithm (DSA) [4]: A digital signature scheme typically consists of three algorithms:

(1) A key generation algorithm that selects a private key uniformly at random from a set of possible private keys. The algorithm outputs the private key and a corresponding public key.

(2) A signing algorithm that, given a message and a private key, produces a signature.

(3) A signature verifying algorithm that, given a message, public key and a signature, either accepts or rejects the message's claim to authenticity.

Definition 2.3.2. A *hash function* is a computationally efficient function that maps bitstrings of arbitrary length to bitstrings of fixed length, called *hash values*. A *one-way hash function* $f : \mathcal{M} \to \mathcal{C}$ is a hash function that satisfies the property that $f(m)$ is easy to compute for all $m \in \mathcal{M}$, but for randomly chosen c in the image of f, finding an $m \in \mathcal{M}$ such that $c = f(m)$ is computationally infeasible, namely we can easily compute f, but it is computationally infeasible to compute f^{-1}. One-way hash functions are called *cryptographic hash functions* since these functions prevent unauthorized retrieval of the original bitstring.

(I) Key Generation:

(1) The sender selects a prime q with 160 bits. Then she selects a prime p with bitlength a multiple of 64 between 512 and 1024, satisfying the property that q divides $p - 1$.

(2) She chooses an $\alpha \in \mathbb{Z}_p^*$ of order q modulo p. This can be done, for instance, by selecting a primitive root a modulo p and setting $\alpha \equiv a^{(p-1)/q}(mod\ p)$. (If $m \in \mathbb{Z}$, $n \in \mathbb{N}$ and $ord_n(m) = \phi(n)$, then m is called a primitive root modulo n. Let $m \in \mathbb{Z}$, $n \in \mathbb{N}$ and $gcd(m, n) = 1$. Then the order of m modulo n is the smallest $e \in \mathbb{N}$ such that $m^e \equiv 1(mod\ n)$, denoted by $e = ord_n(m)$.)

(3) A cryptographic hash function $h : \mathbb{Z}_p^* \to \mathcal{B}_{160}$ (bitstrings of length 160) is selected. She chooses a private key $e \in \mathbb{N}$ such that $e < q$ and computes $\beta \equiv \alpha^e(mod\ p)$.

(4) She publishes (p, q, α, β) and keeps private her key e.

(II) Signing: The sender performs the following in order to sign a message $m \in \mathbb{Z}_p^*$. In what follows, we will assume that any powers of α or β have been reduced modulo p before being used in any congruence modulo q:

(1) Select a random $r \in \mathbb{N}$ such that $r \leq q-1$.

(2) Compute $\gamma \equiv \alpha^r (mod\ q)$.

(3) Compute $\sigma \equiv r^{-1}(h(m) + e\gamma)(mod\ q)$.

(4) She sends m and $sig_k(m, r) = (\gamma, \sigma)$ to the receiver.

(III) Verifying: The receiver executes the following steps:

(1) Obtain sender's public data (p, q, α, β).

(2) Compute $\delta_1 \equiv \sigma^{-1}h(m)(mod\ q)$ and $\delta_2 \equiv \sigma^{-1}\gamma(mod\ q)$.

(3) Compute $\delta \equiv \alpha^{\delta_1}\beta^{\delta_2}(mod\ q)$.

(4) $ver_k(m, (\gamma, \sigma)) = 1$ if and only if $\delta \equiv \gamma(mod\ q)$, in which case he accepts, and rejects otherwise.

2.4. Elliptic Curve Cryptography [2]

An elliptic curve in its *standard form* is described by

$$y^2 = x^3 + ax + b$$

for some fixed real values for the parameters a and b. This equation is also referred to as *Weierstrass Equation of characteristic* 0. For an elliptic curve to be smooth the following condition on the discriminant of the polynomial $f(x) = x^3 + ax + b$ must be satisfied:

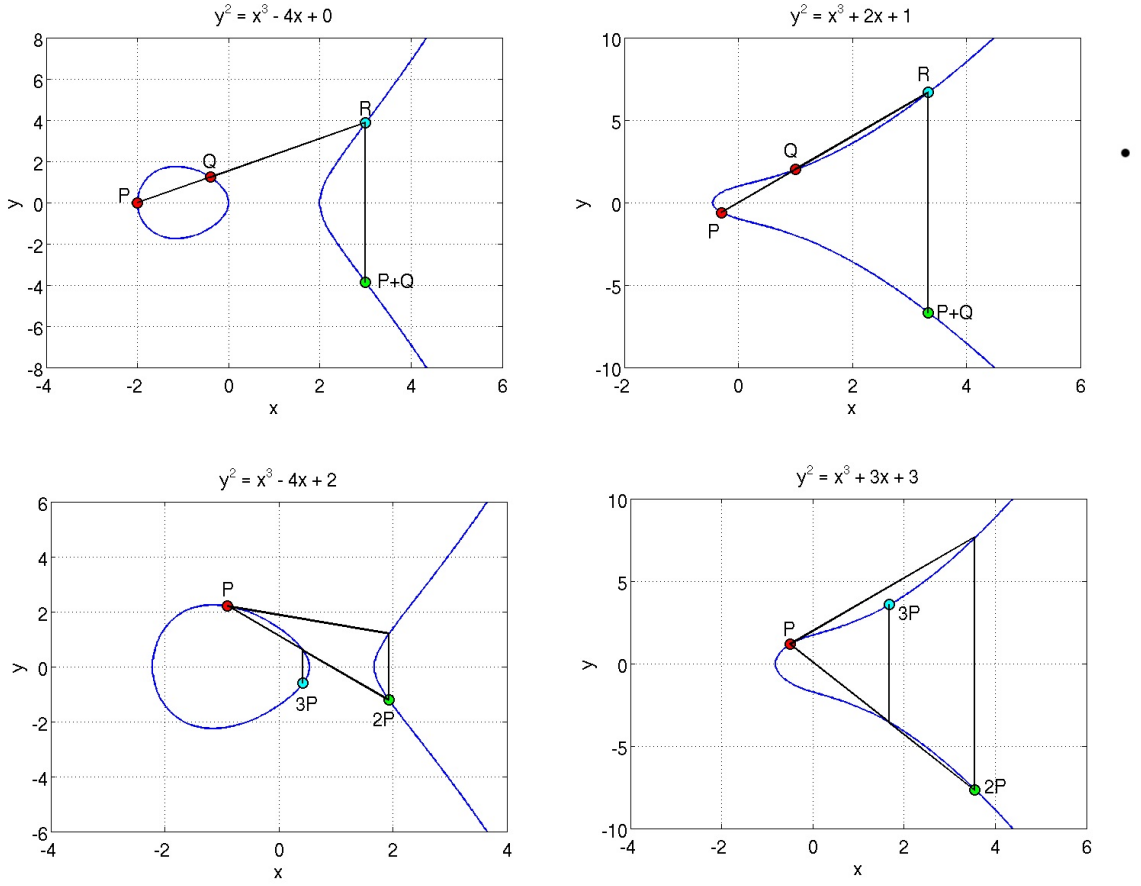
$$4a^3 + 27b^2 \neq 0.$$

If the discriminant is zero, then the curve have a cusp or some other form of non-smoothness. Non-smooth curves are called singular. It is not safe to use singular curves for cryptography. For example, $y^2 = x^3 - 4x$ and $y^2 = x^3 + 2x + 1$ are smooth elliptic curves. Note that since we can write $y = \pm\sqrt{x^3 + ax + b}$ elliptic curves in their standard form will be symmetric about the x-axis. The points on an elliptic curve can be shown to constitute a group. The group

operator for the points on an elliptic curve is, by convention, called addition. To add a point P on an elliptic curve to another point Q on the same curve, we use the following rule:

We first join P with Q with a straight line. The third point of the intersection of this straight line with the curve, if such an intersection exists, is denoted by R. The mirror image of this point with respect to the x-coordinate is the point $P+Q$. If the third point of intersection does not exist, we say it is at infinity.

We denote the point at infinity by the special symbol O and, this serves as the additive identity element for the group operator. We stipulate that $P + O = P$ for any point on the curve.



We define the additive inverse of a point P as its mirror reflection with respect to the x-coordinate. So if Q on the curve is the mirror reflection of P on the curve, then $Q = -P$. For any such two points, it would obviously be the case that the third point of intersection

with the curve of a line passing through the first two points will be at infinity. That is, the point of intersection of a point and its additive inverse will be the distinguished point O. We will further stipulate that that $O + O = O$, implying that $-O = O$. Therefore, the mirror reflection of the point at infinity is the same point at infinity.

The additive inverse of a point where the tangent is parallel to the y-axis, is the point itself. That is, if the tangent at P is parallel to the y-axis, then $P + P = O$. In general, addition of P to itself means to take two distinct points P and Q and let Q approach P. The line joining P and Q will obviously become a tangent at P in the limit. Therefore, the operation $P+P$ means that we must draw a tangent at P, find the intersection of the tangent with the curve, and then take the mirror reflection of the intersection.

For an elliptic curve $y^2 = x^3+ax+b$ we define the set of all points on the curve along with the distinguished point O by $E(a, b)$. $E(a, b)$ is an abelian group with the addition operator as we defined earlier.

The elliptic curves considered so far were for the field of real numbers which is of characteristic zero. The addition operator similar to above can also be defined for fields of characteristic 2 or, 3. But now the elliptic curve $y^2 = x^3 + ax + b$ becomes singular. While singular elliptic curves do admit group laws defined above, such groups, although defined over the points on the elliptic curve, become isomorphic to either the multiplicative or the additive group over the underlying field itself, depending on the type of singularity. That fact makes singular elliptic curves unsuitable for cryptography because they are easy to crack. Therefore, in general, when using the elliptic curve equation $y^2 = x^3 + ax + b$, we avoid underlying fields of characteristic 2 or 3 because of the nature of the constraints they place on the parameters a and b in order for the curve to not become singular.

An Algebraic expression for adding two points on an Elliptic Curve:[2] Given two points P and Q on an elliptic curve $E(a, b)$, we have already seen how to compute the point $P + Q$, we first draw a straight line through P and Q. We next find the third intersection of this line with the elliptic curve. We denote this point of intersection by R. Then $P + Q$ is equal to the mirror reflection of R about the x-axis. In other words, if P, Q, and R are the three intersections of the straight line with the curve, then $P + Q = -R$. This implies that

the three intersections of a straight line with the elliptic curve must satisfy $P + Q + R = O$. We will next examine the algebraic implications of the above relationship between the three points of intersection. The equation of the straight line that runs through the points P and Q must be of the form $y = \alpha x + \beta$, where α is the slope of the line, which is given by $\alpha = \dfrac{y_Q - y_P}{x_Q - x_P}$. For a point (x, y) to lie at the intersection of the straight line and the elliptic curve $E(a, b)$, the following equality must hold

$$(\alpha x + \beta)^2 = x^3 + ax + b$$

since $y = \alpha x + \beta$ on the straight line through the points P and Q and since the equation of the elliptic curve is $y^2 = x^3 + ax + b$. For there to be three points of intersection between the straight line and the elliptic curve, the cubic form in equation $(\alpha x + \beta)^2 = x^3 + ax + b$ must have three roots. We already know two of these roots, since they must be x_P and x_Q, corresponding to the points P and Q. Being a cubic equation, since $(\alpha x + \beta)^2 = x^3 + ax + b$ has at most three roots, the remaining root must be x_R, the x-coordinate of the third point R. Expressing this cubic equation in the form $x^3 - \alpha^2 x^2 + (a - 2\alpha\beta)x + (b - \beta^2) = 0$ we notice that the coefficient of x^2 is $-\alpha^2$. Therefore, we have $x_P + x_Q + x_R = \alpha^2$. We therefore have the following result for the x-coordinate of R:

$$x_R = \alpha^2 - x_P - x_Q.$$

Since the point (x_R, y_R) must be on the straight line $y = \alpha x + \beta$, we can write for y_R:

$$y_R = \alpha x_R + \beta = \alpha x_R + (y_P - \alpha x_P) = \alpha(x_R - x_P) + y_P.$$

To summarize, ordinarily a straight line will intersect an elliptic curve at three points. If the coordinates of the first two points are (x_P, y_P) and (x_Q, y_Q), then the coordinates of the third point are

$$x_R = \alpha^2 - x_P - x_Q, \qquad y_R = \alpha(x_R - x_P) + y_P.$$

We started out with the following relationship between P, Q, and R: $P + Q = -R$, we can therefore write the following expressions for the x and the y coordinates of the addition of two points P and Q:

$$x_{P+Q} = \alpha^2 - x_P - x_Q, \qquad y_{P+Q} = \alpha(x_P - x_R) - y_P,$$

since the y-coordinate of the reflection $-R$ is negative of the y-coordinate of the point R on the intersecting straight line.

Similarly we can obtain the following algebraic expression for calculating $2P$ from P:

$$x_{2P} = \alpha^2 - 2x_P, \qquad y_{2P} = \alpha(x_P - x_R) - y_P, \qquad \left(\alpha = \frac{3x_P^2 + a}{2y_P}\right).$$

Elliptic curves over $\mathbb{Z}_p$ for a prime p:[2] The elliptic curve arithmetic we described so far was over real numbers. These curves cannot be used as such for cryptography because calculations with real numbers are prone to round-off error. Cryptography requires error-free arithmetic. By restricting the values of the parameters a and b, the value of the independent variable x, and the value of the dependent variable y to some prime finite field $\mathbb{Z}_p$, we obtain elliptic curves that are more appropriate for cryptography. Such curves would be described by

$$y^2 \equiv (x^3 + ax + b) (\text{mod } p).$$

The points on such curves would be subject to the modulo p version of the same smoothness constraint on the discriminant as we had for the case of real numbers:

$$(4a^3 + 27b^2) \neq 0 (\text{mod } p).$$

We will use the notation $E_p(a, b)$ to represent all the points (x, y) that obey the conditions laid down above. $E_p(a, b)$ will also include the distinguished point O, the point at infinity. So the points in $E_p(a, b)$ are the set of coordinates (x, y), with $x, y \in \mathbb{Z}_p$, such that the equation $y^2 = x^3 + ax + b$, with $a, b \in \mathbb{Z}_p$ is satisfied modulo p and such that the condition $4a^3 + 27b^2 \neq 0 (\text{mod } p)$ is fulfilled. Obviously, then, the set of points in $E_p(a, b)$ is no longer a curve, but a collection of discrete points in the (x, y) plane (or, even more precisely speaking, in the plane corresponding to the Cartesian product $\mathbb{Z}_p \times \mathbb{Z}_p$). Since the points in $E_p(a, b))$ can no longer be connected to form a smooth curve, we cannot use the geometrical construction to illustrate the action of the group operator. However, the algebraic expressions we derived for these operations continue to hold good provided the calculations are carried out modulo p.

Note that for a prime finite field $\mathbb{Z}_p$, the value of p is its characteristic. Elliptic curves over prime finite fields with $p \leq 3$, while admitting the group law, are not suitable for cryptography. The set $E_p(a, b)$ of points, with the elliptic curve defined over a prime finite field $\mathbb{Z}_p$, constitutes a group, the group operator being as defined earlier.

Hasse's Theorem addresses the question of how many points are on an elliptic curve that is defined over a finite field. This theorem says that if N is the number of points on $E_p(a, b)$ when the curve is defined on a finite field $\mathbb{Z}_p$ with p elements, then N is bounded by

$$|N - (p + 1)| \leq 2\sqrt{p}$$

That is the number of points, N, on an elliptic curve must be in the interval $[p + 1 - \sqrt{p}, p + 1 + \sqrt{p}]$. As mentioned previously, N includes the additive identity element O.

Elliptic Curve Cryptography:[2] That elliptic curves over finite fields could be used for cryptography was suggested independently by Neal Koblitz and Victor Miller in 1985. Just as RSA uses multiplication as its basic arithmetic operation (exponentiation is merely repeated multiplication), ECC uses the addition group operator as its basic arithmetic operation (multiplication is merely repeated addition). Suppose G is a user-chosen base point on the curve $E_p(a, b)$, where p is a prime and the underlying finite field is the prime finite field $\mathbb{Z}_p$. The core notion that ECC is based on is that, with a proper choice for G, whereas it is relatively easy to calculate $C = M \times G$, it can be extremely difficult to recover M from C even when an adversary knows the curve $E_p(a, b)$ and the G used. An adversary could try to recover M from $C = M \times G$ by calculating $2G, 3G, 4G, \cdots, kG$ with k, in the worst case, spanning the size of the set $E_p(a, b)$, and then seeing whether or not the result matched C. But if p is sufficiently large and if the point G on the curve $E_p(a, b)$ is chosen carefully, that would take much too long.

Elliptic curve Diffie-Hellman Secret key exchange:[2] Choose the parameters p, a, and b for an elliptic-curve based group $E_p(a, b)$, and a base point $G \in E_p(a, b)$. Party A selects an integer X_A to serve as his/her private key. A then generates $Y_A = X_A \times G$ to serve as his/her public key. A makes publicly available the public key Y_A. B designates an integer X_B

to serve as his/her private key. As was done by A, B also calculates his/her public key by $Y_B = X_B \times G$. Then by choosing suitable enciphering and deciphering functions E and D, respectively, the encryption and decryption can be carried out by using the following steps.

$$C = E(Y_B, E(X_A, M)),$$

$$M = D(Y_A, D(X_B, C)),$$

where M is the plaintext and C is the ciphertext.

In order to create a shared secret key (that could subsequently be used for, say, a symmetric-key based communication link), both A and B carry out the following operations:
A calculates the shared session key by $K = X_A \times Y_B$.
B calculates the shared session key by $K = X_B \times Y_A$.

$$\begin{aligned} K \text{ as calculated by } A &= X_A \times Y_B \\ &= X_A \times (X_B \times G) \\ &= (X_A \times X_B) \times G \\ &= (X_B \times X_A) \times G \\ &= X_B \times (X_A \times G) \\ &= X_B \times Y_A \\ &= K \text{ as calculated by } B \end{aligned}$$

References

[1] A. Kak, *Public-Key Cryptography and the RSA Algorithm*, Lecture Notes on Computer and Network Security, Purdue University, (2013).

[2] A. Kak, *Elliptic Curve Cryptography and Digital Rights Management*, Lecture Notes on Computer and Network Security, Purdue University, (2014).

[3] D. M. Burton, *Elementary Number Theory*, McGraw-Hill, New York, (2007).

[4] K. H. Rosen, *An Introduction to Cryptography*, Taylor and Francis, (2007).