# An Introduction to the Asymmetrical Cryptosystems


Ahmet Koltuksuz, Ph.D., Assoc. Prof.

<ahmet.koltuksuz@yasar.edu.tr>

Yasar University

College of Engineering

Department of Computer Engineering

İzmir, Turkey

# AGENDA

1. Introduction
2. Discrete Logarithm Problem
3. RSA Cryptosystem
4. Elliptic Curve Cryptosystems
5. Lattice Cryptosystems
6. Comparisons of Asymmetrical Cryptosystems

# Part 1: Introduction

- Definitions
- Cryptosystems, Asymmetrix
- Public Key Cryptosystems
- One way functions
- Trapdoor functions

# Introduction

## CRYPTOLOGY

In Greek
kruptos = hidden,
kruptein =to hide
graphein = to write

- CRYPTOGRAPHY
  encryption, decryption
- CRYPTANALYSIS

# A CRYPTOGRAPHIC SYSTEM

is a family *T* of transformations on **P**laintext and **C**iphertext.

family member index

**key (k)**

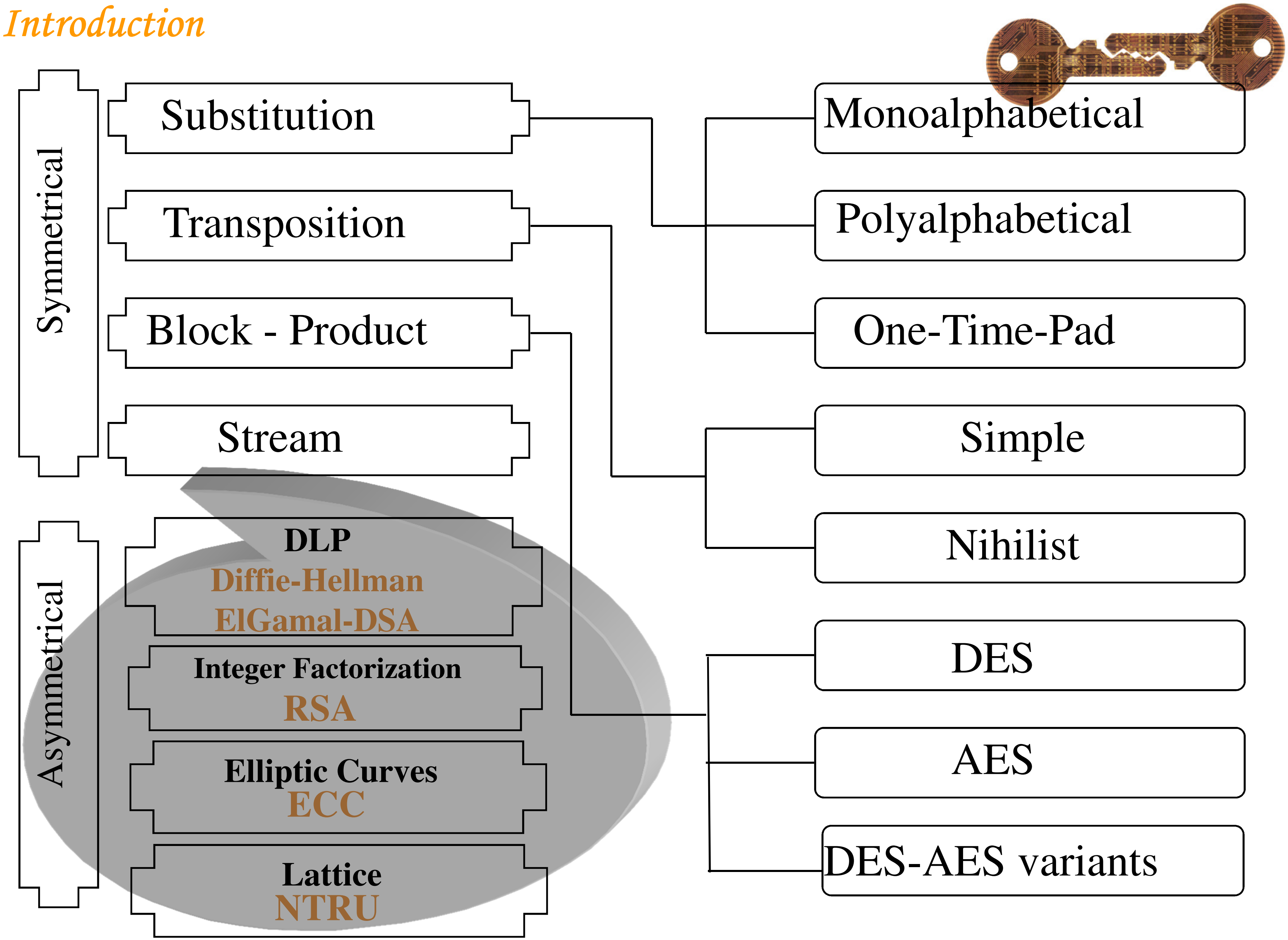
| encryption | decryption |
|---|---|
| E(P) = C | D(C) = P |
| $E_k(P) = C$ | $D_k(C) = P$ |

Now; $E_k = D_k$ ⇒ **Symmetrical**

$E_k \neq D_k$ ⇒ **Asymmetrical**

**"Public Key Cryptosystems"**

# Introduction



# Introduction

**Discrete Logarithm Problem**

**Diffie-Hellman, 1976**

$y = g^x \pmod{p}$

**Diffie-Hellman key exchange**

**ElGamal cryptosystem**

**Digital Signature Algoritm (DSA)**

**Factorization**

**Rivest-Shamir-Adleman, 1978**

$n = pq$

**RSA, PGP**

**Elliptic Curve Discrete Logarithm Problem**

**Miller-Koblitz, 1985-87**

$y^2 = x^3 + ax + b \pmod{p}$

**ECC, ECDH, ECDSA**

**Lattice Cryptosystems**

**Goldreich,Goldwasser-Halevi 1997-1999**

$$L(\mathbf{b}_1, \ldots, \mathbf{b}_n) = \left\{ \sum_{i=1}^{n} x_i \mathbf{b}_i : x_i \in \mathbb{Z} \right\}$$

**GGH, NTRU**

Asymmetrical

**DLP**
Diffie-Hellman
ElGamal-DSA

**Integer Factorization**
RSA

**Elliptic Curves**
ECC

**Lattice**
NTRU

## Public Key Cryptography

- The idea: Differentiate between a (public) encryption key and a (secret) decryption key.
- Main Property: It should not be possible to retrieve the decryption key from the public key.

## Public Key Cryptography

- The public key cryptography is based on the theory of computational complexity.
- The running time of the encryption and decryption algorithms is a function of a security parameter *k*.
- The value of *k* is fixed when the system is initialized.
- An algorithm runs in *polynomial time* if its running time is bounded by a quantity that is polynomially related to *k*.

One-way functions:

- Informally: A *one-way* function is a function that is *easy* to compute but *hard* to invert.
- *easy* means that the function is computable in probabilistic polynomial time.
- *hard* means that the function is easy to invert only for a negligible fraction of the inputs.

## One-way functions: A formal definition

A function **$f: \{0,1\}^* \rightarrow \{0, 1\}^*$** is a one-way-function if

1. There exist an efficient algorithm that on input x computes f(x).

2. For every efficient algorithm $A$ there is a negligible function $\mu_A$

   such that for sufficiently large k

$$\mathbf{Pr[\ f(z)=y:\ x \in R\ \{0,1\}^k\ ;\ f(x)=y\ ;\ A(k,y)=z\ ] \leq \mu_A(k)}$$

## Trapdoor functions: A definition

- Informally a *trapdoor function* is a one-way function with the additional property that it becomes easy to invert if & when some additional information (the trapdoor) is provided.
- it is possible to construct simple (but not completely secure) cryptosystems from trapdoor functions.

## Trapdoor functions: A simplified example

Step #1. Bob publishes a trapdoor function $f$ (but keeps secret the trapdoor $sk$).

Step #2. Alice encrypts a message $m$ as $c = f(m)$

Step #3. Bob decrypts $c$ by computing $f^{(-1)}(c) = m$ (using $sk$)

# Part 2: Discrete Logarithm Problem

- DLP: The Definition
- Diffie-Hellman Key Exchange (DHKE) Algorithm
- El Gamal Cryptosystem

## Discrete Logarithm Problem-DLP:

1. G is a finite group, and $g \in G$
2. Let p be a prime,

and $Z_p^*$ denote the multiplicative group of integers modulo a prime ( $Z_p^* = \{1, 2, ..., p-1\}$ )

3. Let g be a generator of $Z_p^*$

Then the function DL[g](x) = $g^x$ (mod p) is conjectured to be one-way.

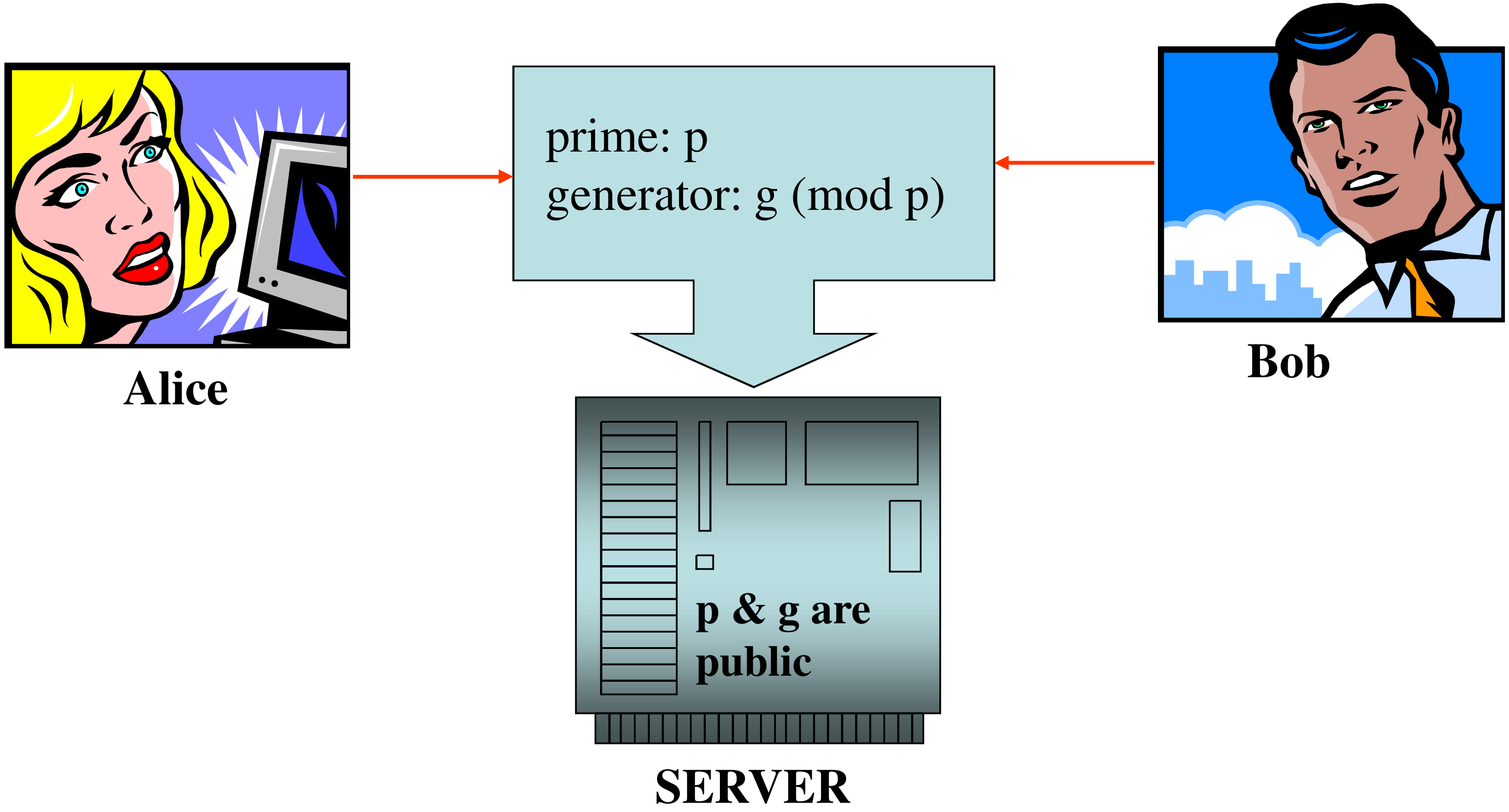
**Which means:**
**Given: y = $g^x$ (mod p), and g. Now;**
**It seems computationally intractable to retrieve x**

## Applications on DLP

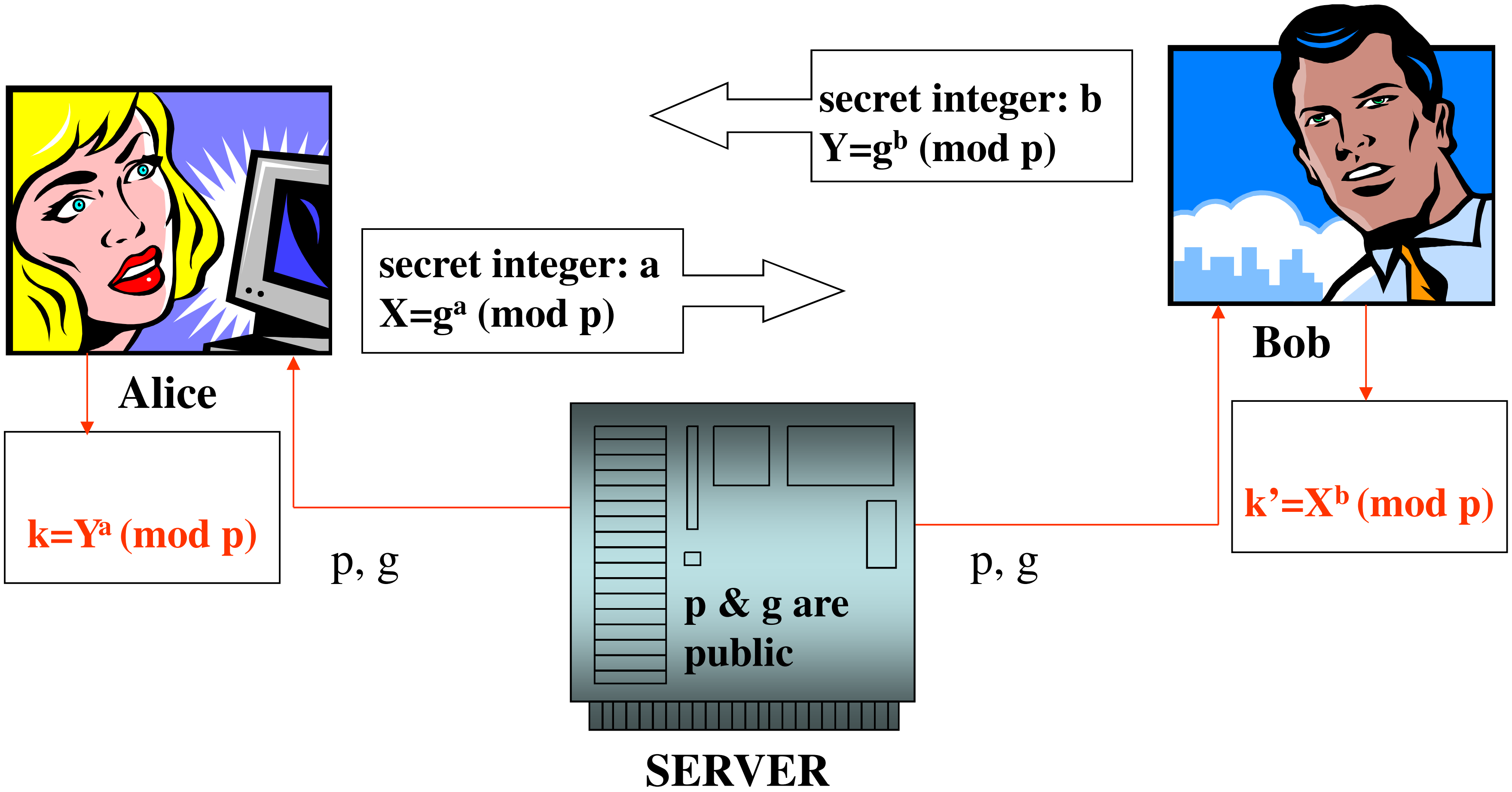
### 1. Diffie-Hellman Key Exchange Algorithm

- Primarily used for key exchange.
- Two or more party can create their own keys independent of each other.
- It is *not* a cryptosystem.
- Based on DLP

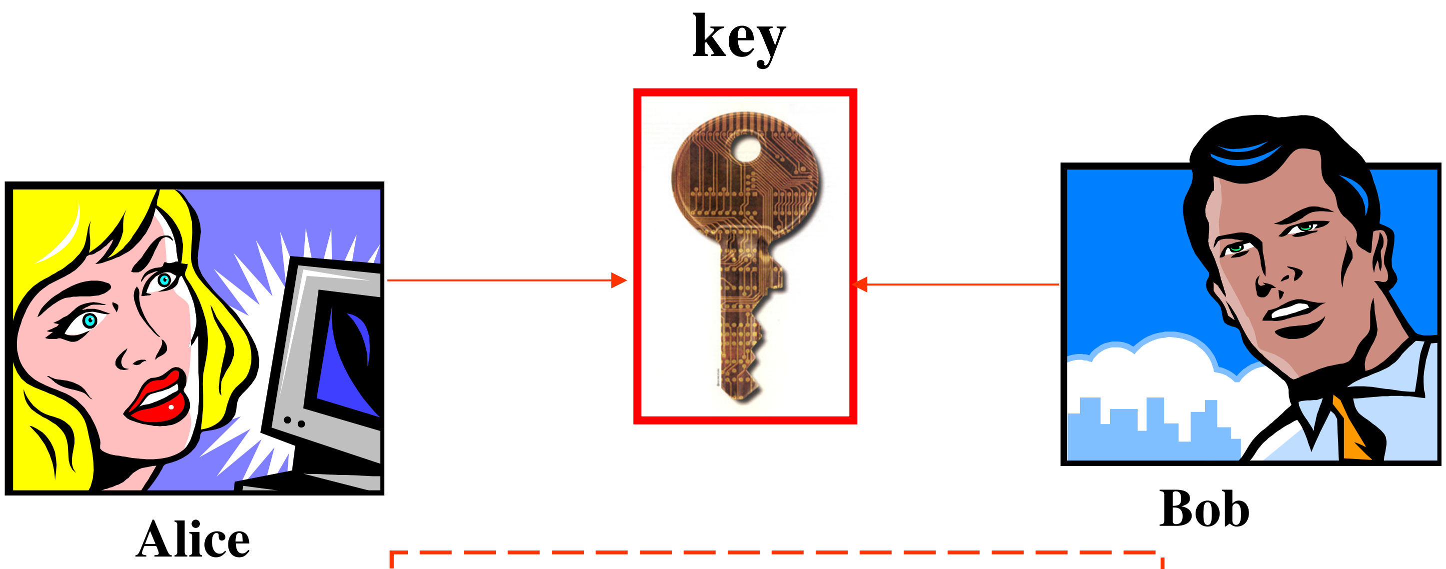
# 1. Diffie-Hellman Key Exchange Algorithm



# 1. Diffie-Hellman Key Exchange Algorithm



## 1. Diffie-Hellman Key Exchange Algorithm




$k = k' \; g^{ab} \pmod{p}$

p,g,X,Y: public

a, b: secret

k = k'; and created independently by each party

# 1. Diffie-Hellman Key Exchange Algorithm

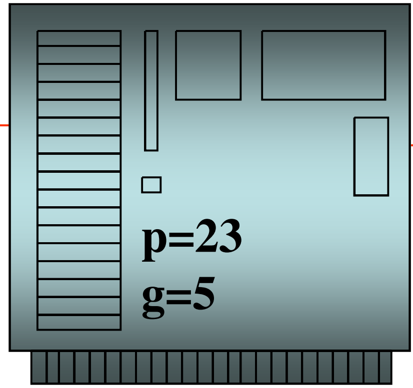
**secret integer: b=15**
**$Y=g^b \pmod p$**
**$Y=5^{15} \pmod{23} = 19$**

**Alice**

**secret integer: a=6**
**$X=g^a \pmod p$**
**$X=5^6 \pmod{23} = 8$**

**Bob**

**$k=Y^a \pmod p$**
**$k=19^6 \pmod{23}$**
**$k=2$**




**SERVER**

**$k'=X^b \pmod p$**
**$k'=8^{15} \pmod{23}$**
**$k'=2$**

## 2. ElGamal Cryptosystem

- Let p be a prime and $(Z_p^*, .)$
- Let generator $g \in Z_p^*$
- $K = \{ (p, g , a, \beta): \beta = g^a \pmod p \}$
- p, g and β are the public key and a is the secret key.
- For $K = (p, g , a, \beta)$, and for a secret random number $k \in Z_{p-1}$ define

$$\mathbf{e_K(x, k) = (y_1, y_2)}$$

where $y_1 = g^k \pmod p$ and, $y_2 = x\beta^k \pmod p$

- For $y_1, y_2 \in Z_p^*$, define

$$\mathbf{d_K(y_1, y_2) = y_2(y_1{}^g)^{-1} \pmod p}$$

## 2. ElGamal Cryptosystem

**Sender chooses:**

$p=2579$
$g=2$
$a=765$ and,
$k=853$ So,
$\beta = 2^{765} \pmod{2579} = 949$

+

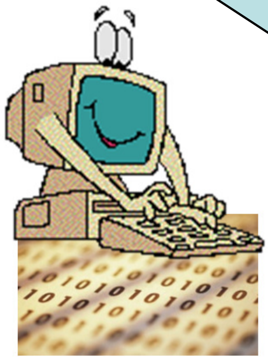
Plaintext is
$x=1299$

$y_1=2^{853} \pmod{2579} =435$
$y_2=1299 \times 949^{853} \pmod{2579}=2396$
ciphertext: $y=(435, 2396)$

**Sender encrypts**

$X = 2396 \cdot (435^{765})^{-1} \pmod{2579} = 1299$

which is the plaintext.



**Receiver decrypts**

# Part 3: RSA Cryptosystem

- Rivest-Shamir-Adleman:RSA
- Formal Definition and Parameters
- The RSA protocol
- An Example for RSA Cryptosystem

RSA algorithm, one of the earliest and simplest public key cryptosystems, gets its strength from the difficulty of factoring large numbers.

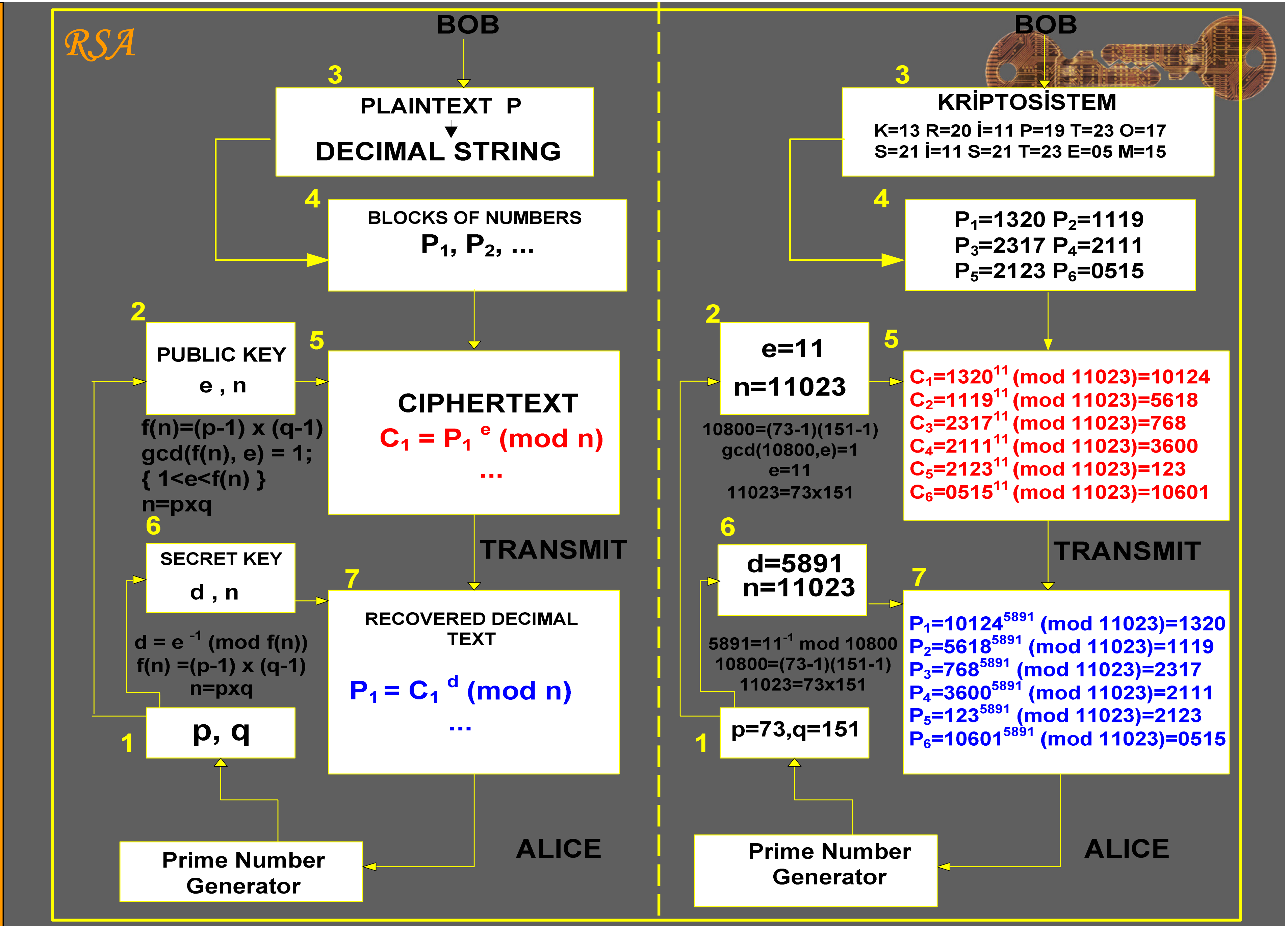
The main arithmetic operation in the RSA cryptosystem is modular exponentiation,

## RSA Domain Parameters

| Parameters | Definition |
|---|---|
| select p & q | p & q are both prime |
| $n = p \times q$ | n is modulus |
| $\Phi(n) = (p-1) \times (q-1)$ | Euler totient function |
| $\gcd(\Phi(n), e) = 1;\ \{ 1<e<\Phi(n) \}$ | Find an integer e |
| $d \equiv e^{-1} \mod (\Phi(n))$ | calculate d |
| $k_p = \{e, n\}$ | Public key |
| $k_s = \{d\}$ | Secret key |
| $e \cdot d \equiv 1 \pmod{\Phi(n)}$ | holds true |

## RSA Protocol

| Encryption | Decryption |
|---|---|
| Plaintext: $M < n$ | Ciphertext: $C$ |
| Ciphertext: $C = M^e \pmod{n}$ | Plaintext: $M = C^d \pmod{n}$ |



RSA
BOB
3
PLAINTEXT P
DECIMAL STRING
4
BLOCKS OF NUMBERS
P1, P2, ...
2
PUBLIC KEY
e , n
5
CIPHERTEXT
C1 = P1 e (mod n)
...
f(n)=(p-1) x (q-1)
gcd(f(n), e) = 1;
{ 1<e<f(n) }
n=pxq
6
SECRET KEY
d , n
TRANSMIT
7
RECOVERED DECIMAL TEXT
P1 = C1 d (mod n)
...
d = e -1 (mod f(n))
f(n) =(p-1) x (q-1)
n=pxq
1
p, q
Prime Number Generator
ALICE
BOB
3
KRİPTOSİSTEM
K=13 R=20 İ=11 P=19 T=23 O=17
S=21 İ=11 S=21 T=23 E=05 M=15
4
P1=1320 P2=1119
P3=2317 P4=2111
P5=2123 P6=0515
2
e=11
n=11023
5
C1=1320^11 (mod 11023)=10124
C2=1119^11 (mod 11023)=5618
C3=2317^11 (mod 11023)=768
C4=2111^11 (mod 11023)=3600
C5=2123^11 (mod 11023)=123
C6=0515^11 (mod 11023)=10601
10800=(73-1)(151-1)
gcd(10800,e)=1
e=11
11023=73x151
6
d=5891
n=11023
TRANSMIT
7
P1=10124^5891 (mod 11023)=1320
P2=5618^5891 (mod 11023)=1119
P3=768^5891 (mod 11023)=2317
P4=3600^5891 (mod 11023)=2111
P5=123^5891 (mod 11023)=2123
P6=10601^5891 (mod 11023)=0515
5891=11^-1 mod 10800
10800=(73-1)(151-1)
11023=73x151
1
p=73,q=151
Prime Number Generator
ALICE

# Part 4: Elliptic Curve Cryptosystems

- Diophantine equations
- Weierstrass equation & sample elliptic curves
- Chord & tangent rule, point operations
- DLP-ECDLP
- ECC domain parameters & a protocol
- An example for the elliptic curve cryptosystem
- Elliptic Curve Diffie-Helmann (ECDH) Key Exchange
- Menezes-Qu-Vanstone (MQV) Key Exchange

*Diophantine* equations:

let a,b,c be integers where a, b ≠ 0 and,

let d=gcd(a,b) then the equation

ax+by=c

has an integer solution x, y iff c is a multiple of d, in which case there are infinitely many solutions. The solution pairs are:

$$x = x_0 + \frac{bn}{d}, y = y_0 - \frac{an}{d}, (n \in Z)$$

*Diophantine* equations:

Examples:

Pythagora's famous theorem $x^2 + y^2 = z^2$

Fermat's 4th degree equation $x^4 + y^4 = z^4$

3rd degree (Elliptic curve) equations $y^2 = x^3+ax^2+bx+c$

## ***Weierstrass* equation:**

An elliptic curve E over a field K is defined by a Weierstrass equation

$$E : y^2+a_1xy+a_3y = x^3+a_2x^2+a_4x+a_6$$

where $a_1, a_2, a_3, a_4, a_6 \in K$ and $\Delta \neq 0$,

where $\Delta$ is the discriminant of E and is defined as

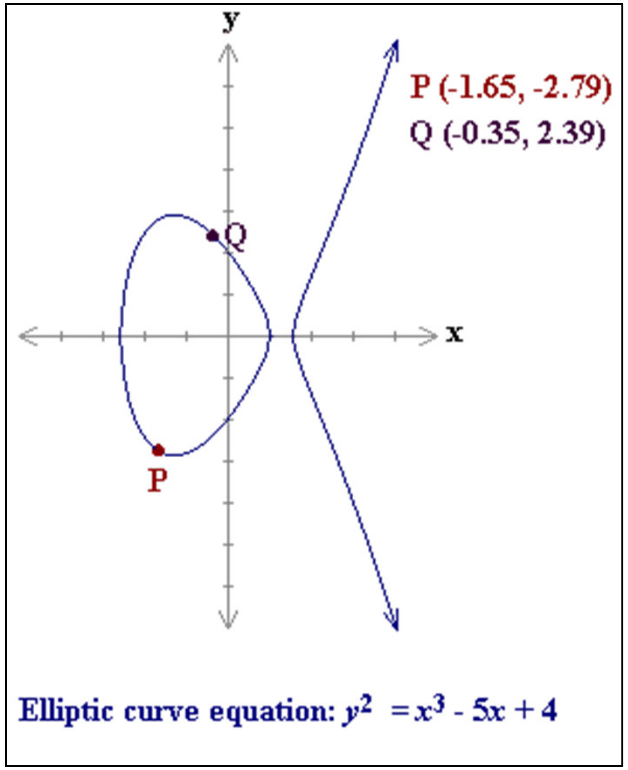
$$\Delta = -d_2^2d_8-8d_4^3-27d_6^2+9d_2d_4d_6$$

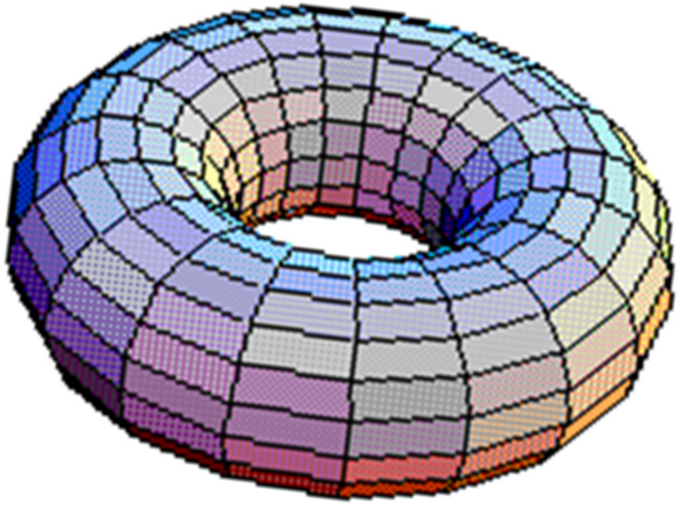
where

$$d_2=a_1^2+4a_2$$
$$d_4=2a_4+a_1a_3$$
$$d_6=a_3^2+4a_6$$
$$d_8=a_1^2a_6+4a_2a_6-a_1a_3a_4+a_2a_3^2-a_4^2$$

$c_4=d_2^2-24d_4$ and $J(E)=c_4^3/\Delta$,
since $\Delta \neq 0$, the elliptic curve E is non-singular

## Example Elliptic Curves:



Suppose given E: $y^2 = x^3+ax^2+bx+c$

and we try to find (x,y) to solve the equation.

Now, the solutions would be in

- rational numbers,
- integers numbers and,
- in a modula p where p is a prime

## Example:

- let E : $y^2 = x^3 + 17$
- for this elliptic curve the solutions would be:
  (-2,3), (-1,4) and (2,5).
- And, other than "trial-and-error" we could apply:

  draw a tangent over P = (-2,3):

  let slope = 1; then $y - 3 = x + 2$.

  place the $y = x + 5$ in E to get $y^2 = x^3 + 17$ and, $(x + 5)^2 = x^3 + 17$

  respectively, $0 = x^3 - x^2 - 10x - 8 = (x + 2)(x^2 - 3x - 4)$ and

  roots are $x=-1$ and $x=4$ for $(x^2 - 3x - 4)$

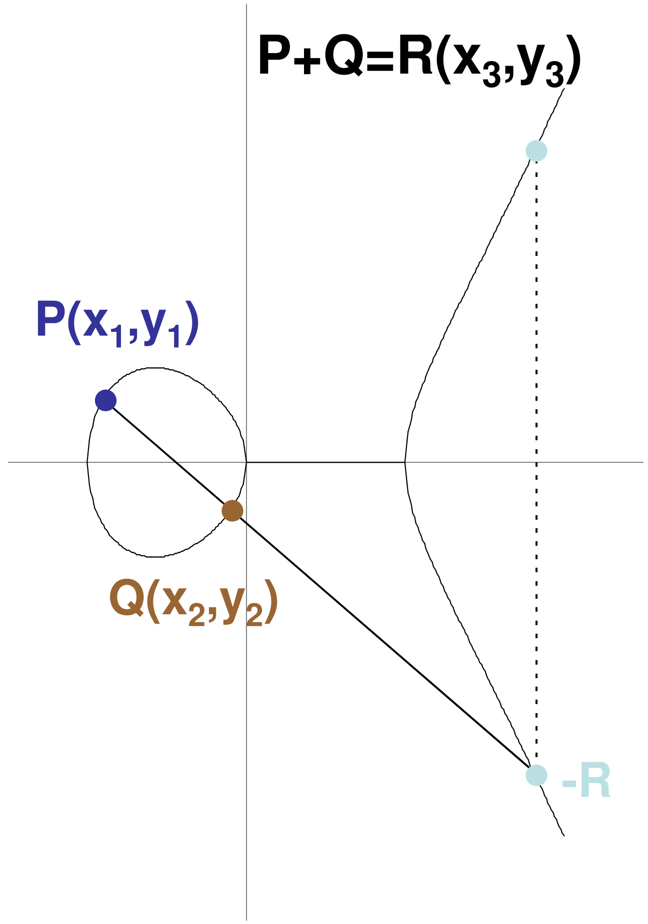
  insert the values of $x$ in $y = x + 5$ to get new y values

So new solution pairs would be: (-1,4) and (4, 9).

## Elliptic Curve Arithmetic:

- There is a rule, called the "chord-and-tangent" rule, for adding two points on an elliptic curve $E(F_p)$ to give a third elliptic curve point.

- With this addition operation, the set of points $E(F_p)$ forms a group with O serving as its identity.



"chord-and-tangent" rule:

- Let $P(x_1,y_1)$, $Q(x_2,y_2)$ and $P, Q \in E(F_p)$

where $P \neq \pm Q$ then

1. $P+Q=R(x_3,y_3)$
2. point doubling $2P=R(x_3,y_3)$

- respectively;

let $\lambda$ is the slope of the line:

1. if $x_1=x_2$ (point doubling) then $\lambda = (3x_1^2+\mathbf{a})/2y_1$, for prime fields the elliptic equation becomes $y^2 = x^3 + \mathbf{a}x^2 +bx$ hence where the a is coming from.
2. otherwise $\lambda = (y_2-y_1)/(x_2-x_1)$ thus the related coordinates are

$$x_3=\lambda^2-x_1-x_2$$

$$y_3=\lambda(x_1-x_3)-y_1$$

## Example “chord-and-tangent” rule:

- Let $p=23$, where $p$ is a prime
- $E: y^2=x^3+x+4$ defined over $F_{23}$
- Let $P(7,3)$ and $Q(8,8)$ then $P+Q=R(x_3, y_3)$ is computed as:

$$\lambda = (y_2-y_1)/(x_2-x_1) = (8-3)/(8-7) = 5$$

$$x_3 = \lambda^2-x_1-x_2 = (5)^2 -8-7 = 25-15 = 10 \equiv 10 \pmod{23}$$

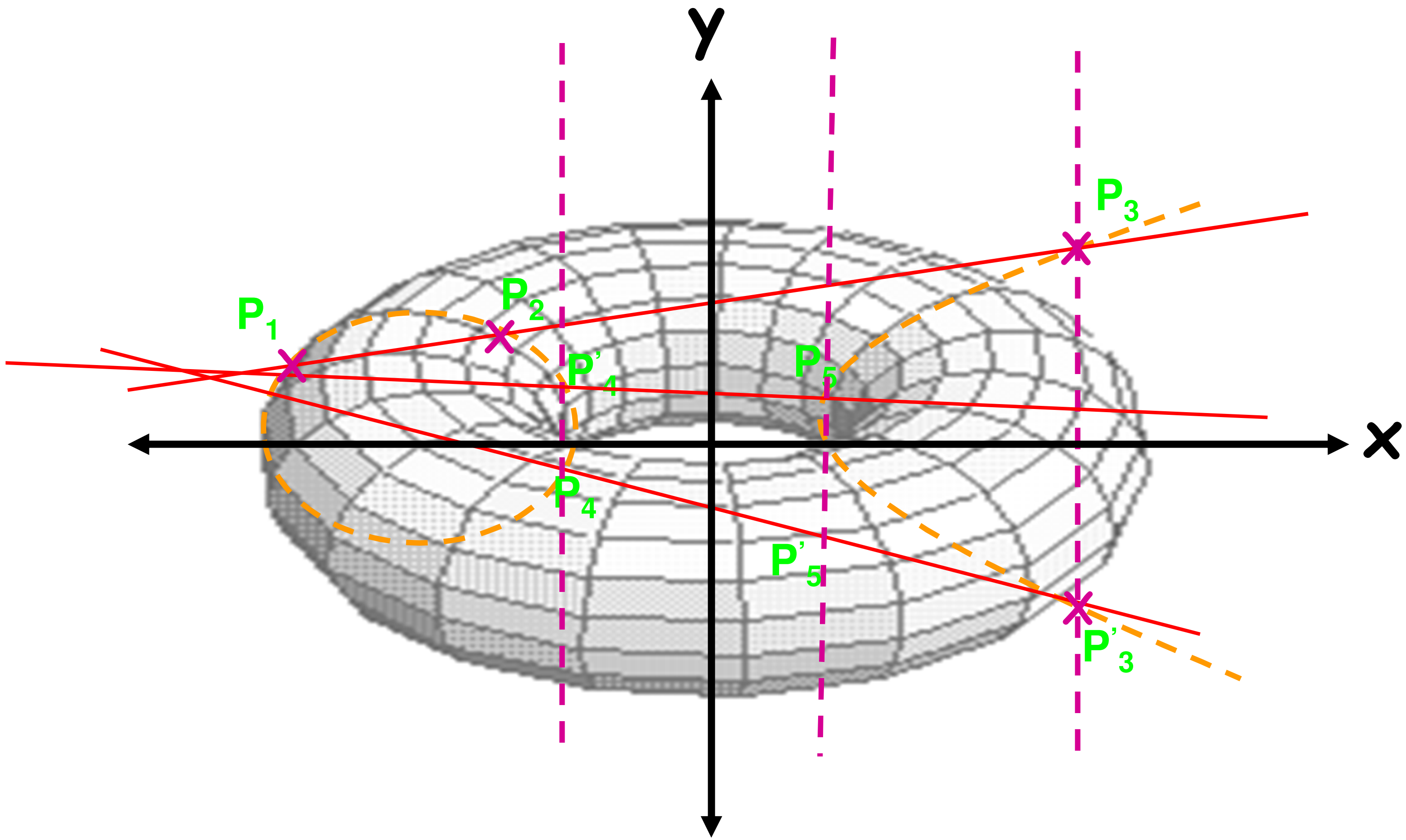
$$y_3= \lambda(x_1-x_3)-y_1 = 5.(7-10)-3 = -18 \equiv 5 \pmod{23}$$

Hence $P + Q = R(10,5)$.

control: $10^3+10+4 \equiv 2 \pmod{23}$ and $5^2 \equiv 2 \pmod{23}$

# Example "chord-and-tangent" rule:

A graphical representation



## Point operations:

- Let p=23, where p is a prime
- E: $y^2=x^3+x+4$ defined over $F_{23}$
- let a=1 and b=4 then the discriminant is
- $\Delta E=4a^3+27b^2 = 4(1)^3 + 27(4)^2 \equiv 22 \pmod{23}$, thus
- E is indeed an elliptic curve.
- So the valid points on $E(F_{23})$:

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| (0,1 ) | (0,21) | (1,11) | (1,12) | (4,7) | (4,16) | (7,3) | (7, 20) | (8,8) | (8,15) |
| (9,11) | (9,12) | (10,5) | (10,18) | (11,9) | (11,14) | (13,11) | (13,12) | (14,15) | (14,18) |
| (15,16) | (15,17) | (17,9) | (17,14) | (18,9) | (18,14) | (22,5) | (22,19) | | |

## Point operations:

- In order to find new points on E drawing a line from the known points is the essential approach. Thus one can find infinite number of points.
- let p is a prime
- let E: $y^2=x^3+x \pmod{p}$ defined over $F_p$
- the number of points for different p primes are:

| p | points on E : $y^2=x^3+x \pmod{p}$ | $N_p$ |
|---|---|---|
| 2 | (0,0), (1,0) | 2 |
| 3 | (0,0), (2,1), (2,2) | 3 |
| 5 | (0,0), (2,0), (3,0) | 3 |
| 7 | (0,0), (1,3), (1,4), (3,3), (3,4), (5,2), (5,5) | 7 |
| 13 | (0,0), (2,6), (2,7), (3,2), (3,11), (4,4), (4,9), (5,0), (6,1), (6,12),(7,5),(7,8),(8,0), (9,6), (9,7), (10,3), (10,10), (11,4), (11,9) | 19 |
| ... | ... | ... |

## DLP-ECDLP

Let $F_p^* = \{1,2,\ldots,p\text{-}1\}$ denote the multiplicative group of integers modulo a prime.

Let $g \in F_p^*$

Then the DLP is:

Given $y \in F_p^*$

Find the integer $a$ such that $y=g^a$ while y and g are known!

A very hard problem!!!

- Let P is a point on E having an order of n and let Q is another point on E.
- Now, Elliptic Curve Discrete Logarithm Problem-ECDLP is to find an integer k such that $Q = kP$ where $0<k<n-1$

And it is a problem since there is no known method of finding it like index calculus for DLP, ECDLP is infinitely more complicated than DLP in finite fields.

## 1. Domain parameters

- let GF be a finite Galois field.
- let E(x) be an elliptic curve defined over GF, and P is a point defined on E.
- GF, E(x), P and $Z_p$ public!

## 2. Key generation

- generate a random number k where $k \in Z_{p-1}$
- calculate $Q = k.P$

Now;

> point Q is the public key.
>
> k is the secret key.

## Encryption

- receiver's public key: Q
- message m gets broken into $(m_1,m_2)$ pairs such that $m_1 \in GF$, $m_2 \in GF$
- select a random integer a such that $a \in Z_{p-1}$
- calculate point $(x_1,y_1) = aP$
- calculate point $(x_2, y_2) = aQ$
- $(m_1$ and $m_2)$ and $(x_2$ and $y_2)$ field elements are combined into $(c_1$ and $c_2)$ field elements
- Send data $m_e = (x_1, y_1, c_1, c_2)$ to receiver

Decryption

- Receive the message from the sender:

$m_e = (x1, y1, c1, c2)$

- calculate the $(x2, y2)=k(x1\ , y1)$ by using k of which

known only by the receiver

- decrypt $m_1$ and $m_2$ by $m_e$

# Elliptic Curve Cryptosystems: An Example

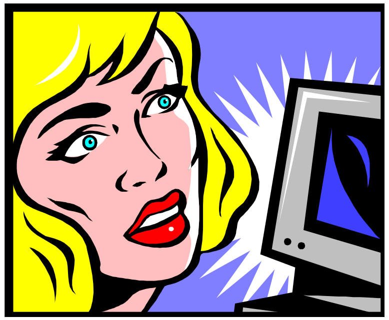
| DOMAIN PARAMETERS | | |
|---|---|---|
| **Steps** | **Parameter** | **Definition** |
| 1 | $F_{11}$ | q=11 and is a prime number which defines the finite Galois field (GF) |
| 2 | E: $y^2=x^3+x+6$ and $\#E(F_{11})$ = n = 13 | a=1, b=6 and the order of the elliptic curve is 13 over $F_{11}$. |
| 3 | Determine r = 13 | r is a prime divisor of $\#E\,(Fq)$, r should be the largest prime factor. |
| 4 | Determine k = 1 | k is a cofactor |
| 5 | $G\in E\,(Fq)$ of order $n$, $G=(2,7)$ | Determine the base point G on E |
| | *P=(2,7),* 2P = (5,2), 3P = (8,3), 4P = (10,2), 5P = (3,6), 6P = (7,9), 7P = (7,2), 8P = (3,5), 9P = (10,9), 10P = (8,8), 11P = (5,9), 12P = (2,4) The selected base point is (2,7) and the other points on the curve are generated from this base until point at infinity (x, 0) is reached through point addition and doubling operations. This operation defines the order of points on the curve and define a new field over $F_{11}$. | |
| | Announce the domain parameters : D(q, E, G, n,a,b)=(11, $y^2=x^3+x+6$, (2,7), 13, 1, 6) | |
| 6 | $W\in E\,(Fq)$ | Public key |
| 7 | $s\in$ [1, 12] and s = 7 | Secret key |
| 8 | *W=sG, W = 7.(2,7) = (7,2)* | Should be on the curve and different from point at infinity |

# *Elliptic Curve Cryptosystems: An Example*

| PROTOCOL | | |
|---|---|---|
| **Steps** | **Encryption** | **Decryption** |
| 1 | The plaintext message $m_p$ is identified with a point on the curve, such as $m_p$=(10,9). | Receive the message from the sender:<br>me = ($x_1$, $y_1$, $c_1$, $c_2$)=((8,3), (10,2)) |
| 2 | select a random integer α,<br>such that<br>1< α <13, α=3 | calculate<br>($x_2$, $y_2$) = s($x_1$, $y_1$) = 7.(8,3)<br>s is the receiver's secret key and is known only by the receiver. |
| 3 | Calculate point<br>(x1,y1) = 3.(2,7)=(8,3)<br>calculate point<br>(x2, y2)= 3.(7,2) =(3,5) | (m1,m2)=($c_1$,$c_2$) – ($x_2$, $y_2$) (mod n)<br>(m1,m2)=(10,2) – 7.(8,3) (mod 13)<br>(m1,m2)=(10,2) + 6.(8,3) (mod 13)<br>(m1,m2)=(10,2) +(3,6) = (10,9)<br>decrypt m1 and m2 by me |
| 4 | (c1 , c2) = (m1, m2) + (x2 ,y2)<br>(c1 , c2) =(10,9) + (3,5) = (10,2) | |
| 5 | Send below data to receiver<br>me = (x1, y1, c1, c2)=((8,3), (10,2)) | |

## Elliptic Curve Diffie-Hellman (ECDH) Key Exchange Algorithm

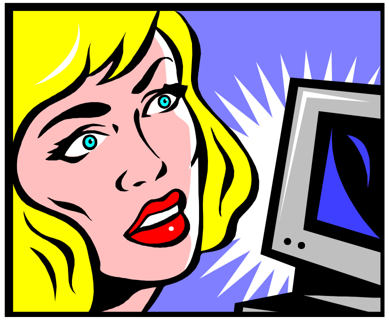
- Alice and Bob agrees upon to use a randomly chosen point P on curve E as a key plus on a methodology to convert that point to an integer.
- Now, E is an elliptic curve over Fq and P is a starting point on E.



**Alice**

**Bob**

# Elliptic Curve Diffie-Hellman (ECDH) Key Exchange Algorithm



**Alice**

**calculates $(k_A.P)$ and sends it over to Bob**

**calculates $(k_B.P)$ and sends it over to Alice**

**Bob**

**Selects a secret random integer $k_A$**

**Selects a secret random integer $k_B$**

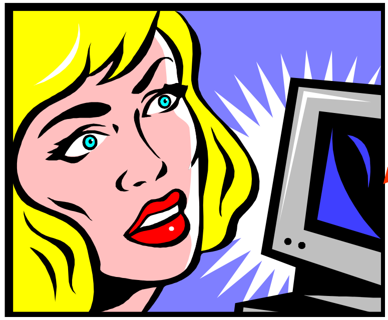
**receives $(k_B.P)$ from Bob**

**receives $(k_A.P)$ from Alice**

**Shared key is $P=(k_A.k_B.P)$**

# Elliptic Curve Diffie-Hellman (ECDH) Key Exchange Algorithm

let p is a prime, p=13
let E: $y^2=x^3+x \pmod p$
Let P=(2,7)



**Alice**

**Bob**

$k_A=2$

$(k_A.P)=2(2,7)$
$=(3,2)$

$(k_B.P)=7(2,7)$
$=(6,1)$

$k_B=7$

**Shared key is**
**$P=(k_A.k_B.P)$**
**$P=(14(2,7))=(10,10)$**

# ECMQV

Elliptic Curve Menezes-Qu-Vanstone (ECMQV) Key Exchange Algorithm

- MQV (Menezes-Qu-Vanstone) is an authenticated protocol for key agreement based on the Diffie-Hellman scheme.
- The protocol can be modified to work in an arbitrary finite group, and, in particular, elliptic curve groups, where it is known as elliptic curve MQV (ECMQV).
- MQV was initially proposed by Menezes, Qu and Vanstone in 1995. It was modified by Law and Solinas in 1998.
- There are one-, two- and three-pass variants.

## Elliptic Curve Menezes-Qu-Vanstone (ECMQV) Key Exchange Algorithm

- MQV is incorporated in the public-key standard IEEE P1363.
- MQV has some (alleged) weaknesses that were (allegedly) fixed by HMQV in 2005.
- ECMQV is also specified by the National Security Agency as part of the "Suite B" set of cryptographic standards for securing US Federal government communications up to the TOP SECRET classification.

## *ECMQV*

Elliptic Curve Menezes-Qu-Vanstone (ECMQV) Key Exchange Algorithm. ANSI X9-42

Steps are as follows:

1. (p, q, g): a set of domain parameters.
2. $y_V$ : V's static public key, an element in GF(p).
3. $x_U$: U's static private key, an integer.
4. $t_V$ : V's ephemeral public key, an element in GF(p).
5. $r_U$: U's ephemeral private key, an integer.
6. $t_U$: U's ephemeral public key, an element in GF(p).
7. w: an integer, w = ||q||/2.

- Input: (p, q, g), $x_U$, $y_V$, $r_U$, $t_U$, $t_V$, w.
- Output: Z.

## Elliptic Curve Menezes-Qu-Vanstone (ECMQV) Key Exchange Algorithm: ANSI X9-42

### ACTIONS

Calculate:

$$\overline{t_U} = t_U \pmod{2^w} + 2^w$$

$$S_U = (r_U + \overline{t_U}\, x_U) \bmod q$$

$$\overline{t_V} = t_V \pmod{2^w} + 2^w$$

$$Z = t_V (y_V \wedge \overline{t_V}) \wedge S_U \bmod p$$

Output:

$$Z$$

# Part 5: Lattice Cryptosystems

- Lattice Theory
- Hard Problems
- NTRU Lattice
- Quotient Polynomial Ring
- The NTRU Cryptosystem
- An Example

# Lattice Theory

$$L(\mathbf{b}_1, \ldots, \mathbf{b}_n) = \left\{ \sum_{i=1}^{n} x_i \mathbf{b}_i : x_i \in Z \right\}$$

$B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ is the *lattice basis* in $R^m$.

m : *dimension* of the lattice
n : *rank* of the lattice

# Lattice Theory

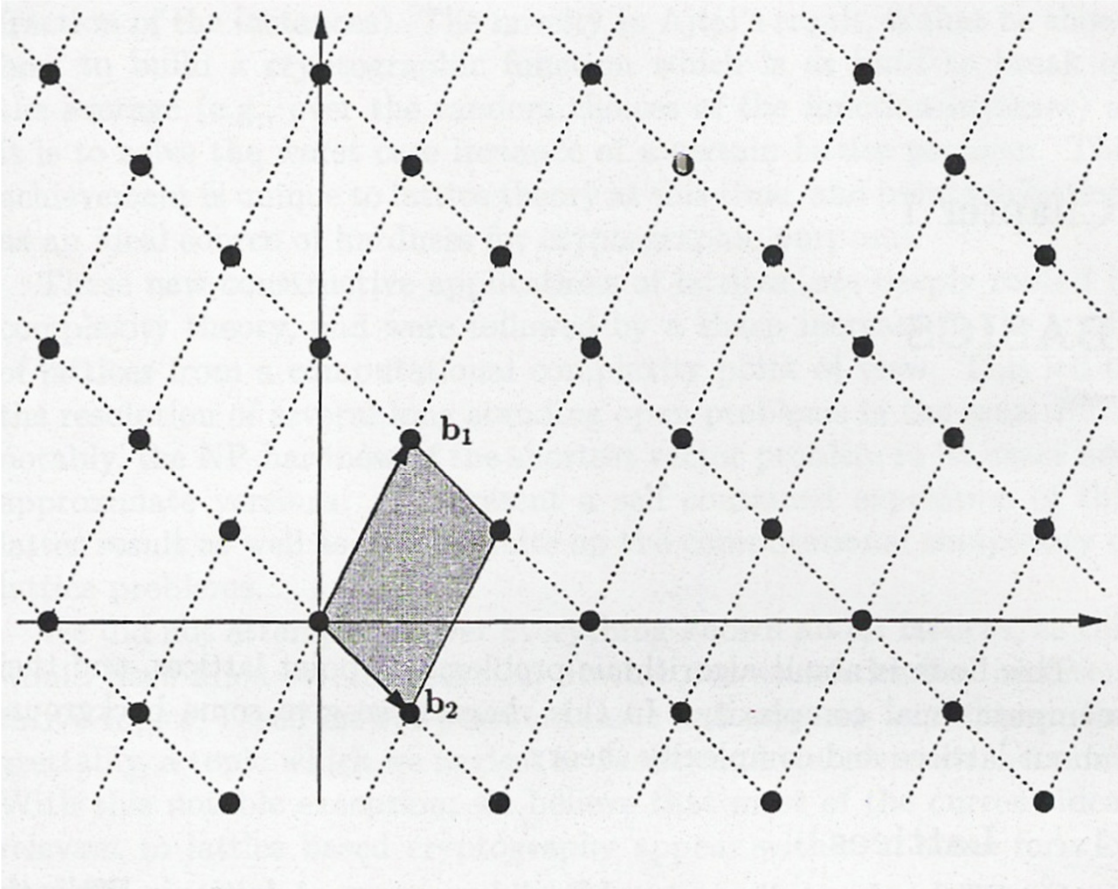



*Figure 1.1.* A lattice in $\mathbb{R}^2$

# *Lattice Hard Problems*

- **The Shortest Vector Problem (SVP)**
  Finding the lattice vector which has the smallest norm / length.

- **The Closest Vector Problem (CVP)**
  Finding the lattice vector which has the smallest distance to a given target vector.

# Lattice Hard Problems

- CVP and SVP also have approximate forms.

- These problems are NP-Complete.

# NTRU Lattice

$$L_{CML} = rowspan\left(\begin{bmatrix} b\mathbf{I} & \mathbf{H} \\ \mathbf{0} & q\mathbf{I} \end{bmatrix}\right)$$

$$\mathbf{H} = \begin{bmatrix} h_0 & h_1 & \cdots & h_{n-1} \\ h_{n-1} & h_0 & \cdots & h_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ h_1 & h_2 & \cdots & h_0 \end{bmatrix}$$

# Quotient Polynomial Ring

$$R = Z\,[x]/\left(x^n - 1\right)$$

$$f = \sum_{i=0}^{n-1} f_i x^i = \left[f_0, f_1, \ldots, f_{n-1}\right]$$

# Quotient Polynomial Ring

- Addition
  - Traditional polynomial addition. (add up the coefficients)
- Multiplication
  - Star Multiplication (Convolution Product)

$$\boxed{h = f * g}$$

$$\boxed{h_k = \sum_{i=0}^{k} f_i g_{k-i} + \sum_{i=k-1}^{n-1} f_i g_{n+k-i} = \sum_{i+j \equiv k \pmod{n}} f_i g_j}$$

# Quotient Polynomial Ring

- Example

$N = 3$, $a = 2 - x + 3x^2$, $b = 1 + 2x - x^2$

$$a + b = 3 + x + 2x^2$$

$$\begin{aligned} a * b &= 2 + 3x - x^2 + 7x^3 - 3x^4 \\ &= 2 + 3x - x^2 + 7 - 3x \\ &= 9 - x^2 \end{aligned}$$

# *NTRU Cryptosystem*

- Domain Parameters

$n$ max degree; n is prime

$p$ small modulus

$q$ large modulus with gcd(p,q) =1

$L_f$ class of polynomial f; $f^{-1}$ exists

$L_g$ class of polynomial g

$L_m$ class of polynomial m

$L_r$ class of polynomial r

# NTRU Cryptosystem

- Key Generation

Choose an $f$ such that

$$\boxed{f_p^{-1} * f \equiv 1 \pmod{p}}$$ and $$\boxed{f_q^{-1} * f \equiv 1 \pmod{q}}$$

**Public Key**

$$\boxed{h \equiv p\, f_q^{-1} * g \pmod{q}}$$

**Private Key**

$$\boxed{f}$$

- Encryption

Choose a random polynomial $r$

Calculate the polynomial $m$ representing plaintext

Encrypt the message $m$ using public key $h$

$$\boxed{c \equiv r * h + m \pmod{q}}$$

# *NTRU Cryptosystem*

- Decryption

Calculate the polynomial *a* by

$$a \equiv f * c \pmod{q}$$

Calculate the plaintext polynomial *m* (representing plaintext) by

$$m \equiv f_p^{-1} * a \pmod{p}$$

# An example

$N = 11 \quad q = 32 \quad p = 3$

$$f = -1 + X + X^2 - X^4 + X^6 + X^9 - X^{10}$$

$$g = -1 + X^2 + X^3 + X^5 - X^8 - X^{10}$$

$$f_p = 1 + 2X + 2X^3 + 2X^4 + X^5 + 2X^7 + X^8 + 2X^9$$

$$f_q = 5 + 9X + 6X^2 + 16X^3 + 4X^4 + 15X^5 + 16X^6 + 22X^7 + 20X^8 + 18X^9 + 30X^{10}$$

$$h = pf_q{*}g = 8 + 25X + 22X^2 + 20X^3 + 12X^4 + 24X^5 + 15X^6 + 19X^7 + 12X^8 + 19X^9 + 16X^{10} \pmod{32}.$$

# An example

$N = 11$ $q = 32$ $p = 3$

$\mathbf{h} = 8 + 25X + 22X^2 + 20X^3 + 12X^4 + 24X^5 + 15X^6 + 19X^7 + 12X^8 + 19X^9 + 16X^{10} \pmod{32}$

$\mathbf{m} = -1 + X^3 - X^4 - X^8 + X^9 + X^{10}$

$\mathbf{r} = -1 + X^2 + X^3 + X^4 - X^5 - X^7$

$\mathbf{e} = \mathbf{r}*\mathbf{h} + \mathbf{m} = 14 + 11X + 26X^2 + 24X^3 + 14X^4 + 16X^5 + 30X^6 + 7X^7 + 25X^8 + 6X^9 + 19X^{10} \pmod{32}$

# An example

$N = 11 \quad q = 32 \quad p = 3$

$\mathbf{f} = -1 + X + X^2 - X^4 + X^6 + X^9 - X^{10}$

$\mathbf{e} = 14 + 11X + 26X^2 + 24X^3 + 14X^4 + 16X^5 + 30X^6 + 7X^7 + 25X^8 + 6X^9 + 19X^{10} \pmod{32}$

$\mathbf{a} = \mathbf{f}*\mathbf{e} = 3 - 7X - 10X^2 - 11X^3 + 10X^4 + 7X^5 + 6X^6 + 7X^7 + 5X^8 - 3X^9 - 7X^{10} \pmod{32}.$

$\mathbf{b} = \mathbf{a} = - X - X^2 + X^3 + X^4 + X^5 + X^7 - X^8 - X^{10} \pmod{3}$

$\mathbf{c} = \mathbf{f_p}*\mathbf{b} = - 1 + X^3 - X^4 - X^8 + X^9 + X^{10} \pmod{3}.$

# Part 6: Comparisons

- ECC vs. RSA
- Lattice vs. ECC

# *ECC vs. RSA*

| | BANDWIDTH | | KEY LENGTH | |
|---|---|---|---|---|
| | Signature length for 2000 bits long messages (bit) | Length of the 100 bits message after encryption (bit) | public key (bit) | secret key (bit) |
| RSA | 1024 | 1024 | 1088 | 2048 |
| DLP-DSA | 320 | 2048 | 1024 | 160 |
| ECC | 320 | 321 | 161 | 160 |

# Lattice vs. ECC

**Table 2.** Public Key Sizes (in bits).

| Security Level (bits) | NTRU | ECC | RSA |
|---|---|---|---|
| 80 | 2008 | 160 | 1024 |
| 112 | 3033 | 224 | 2048 |
| 128 | 3501 | 256 | 3072 |
| 160 | 4383 | 320 | 4096 |
| 192 | 5193 | 384 | 7680 |
| 256 | 7690 | 521 | 15360 |

# *Lattice vs. ECC*

Table 3. Key Generation, Encryption and Decryption Times.

| Crytosystem | Security Level (bits) | Key Generation* (msec) | Encryption* (msec) | Decryption* (msec) |
|---|---|---|---|---|
| NTRU-251 | 80 | 75.65 | 1.68 | 8.22 |
| ECC-192 | between 80, 112 | 57.87 – 152.73 | 37.81 – 116.39 | 19.15 – 57.68 |
| NTRU-347 | 112 | 144.16 | 3.11 | 15.70 |
| ECC-224 | 112 | 234.11 – 367.98 | 52.52 – 164.50 | 26.35 – 81.52 |
| NTRU-397 | 128 | 188.92 | 3.97 | 20.26 |
| ECC-256 | 128 | 478.22 – 656.63 | 68.72 – 223.29 | 35.00 – 111.16 |
| NTRU-491 | 160 | 288.31 | 5.97 | 30.96 |
| NTRU-587 | 192 | 412.10 | 8.42 | 44.42 |
| ECC-384 | 192 | 947.43 – 1429.11 | 182.35 – 586.20 | 90.61 – 290.94 |
| NTRU-787 | 256 | 738.75 | 14.49 | 79.48 |
| ECC-521 | 256 | 2055.04 – 3175.87 | 423.25 – 1257.56 | 211.38 – 626.33 |

*ECC timings are given as minimum - maximum of the values observed over all coordinate systems.