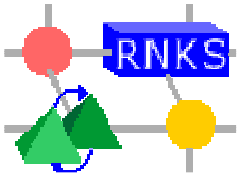


Brandenburg Technical University of Cottbus

Computer Networking Group

A Tutorial on Elliptic Curve Cryptography (ECC)

Fuwen Liu

lfw@informatik.tu-cottbus.de

Contents

I. Introduction

II. Elliptic Curves over Real Number

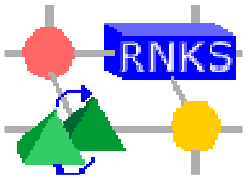
III. Elliptic Curves over Prime Field and Binary Field

IV. Security Strength of ECC System

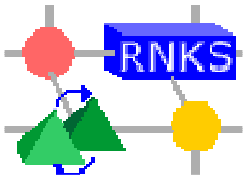
V. ECC Protocols

VI. Patents and Standards

VII. Final Remarks



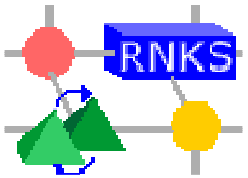
I.
Introduction



Basic concept

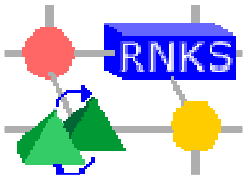
- **Cryptography is a mathematical based technology to ensure the information security over a public channel. There are two objectives:**
  - Privacy: No information is accessible to unauthorized parties
  - Authentication: Information is not alerted in transition and the communication parties are legitimate.

- **Cryptography systems can be distinguished in two categories[1]:**
  - Unconditionally secure system: It resist any cryptanalytic attack no matter how much computation is used.
    - One-time pad system is a typical example
    - Require the length of the key stream equivalent to that of plaintext
    - Rarely deployed in practice
  - Conditionally secure system: It is computationally infeasible to be broken, but would succumb to an attack with unlimited computation.
    - Basically modern cryptographic systems are constructed on the basis of the conditionally secure principle.



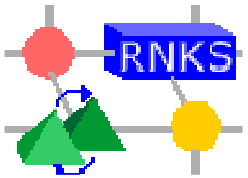
Motivation

- **Public key cryptographic algorithms (asymmetric key algorithms) play an important role in providing security services:**
  - Key management
  - User authentication
  - Signature
  - Certificate
- **Public key cryptography systems are constructed by relying on the hardness of mathematical problems**
  - RSA: based on the integer factorization problem
  - DH: based on the discrete logarithm problem
- **The main problem of conventional public key cryptography systems is that the key size has to be sufficient large in order to meet the high-level security requirement.**
  - This results in lower speed and consumption of more bandwidth
  - Solution: Elliptic Curve Cryptography system

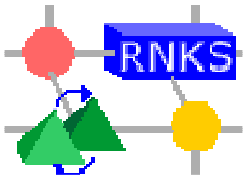


History of ECC

- **In 1985, Neal Koblitz [2] and Victor Miller [3] independently proposed using elliptic curves to design public key cryptographic systems.**
- **In the late 1990`s, ECC was standardized by a number of organizations and it started receiving commercial acceptance.**
- **Nowadays, it is mainly used in the resource constrained environments, such as ad-hoc wireless networks and mobile networks.**
- **There is a tend that conventional public key cryptographic systems are gradually replaced with ECC systems.**
  - As computational power evolves, the key size of the conventional systems is required to be increased dramatically.

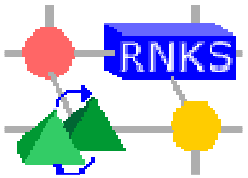


II.
Elliptic Curves over Real Numbers



Overview

- **Elliptic curves have been studied by mathematicians for over a hundred years. They have been deployed in diverse areas**
  - Number theory: proving Fermat`s Last Theorem in 1995 [4]
    - The equation $x^n + y^n = z^n$ has no nonzero integer solutions for x, y, z when the integer n is grater than 2.
  - Modern physics: String theory
    - The notion of a point-like particle is replaced by a curve-like string.
  - Elliptic Curve Cryptography
    - An efficient public key cryptographic system.



Definition

- **An elliptic curve *E* over *R* (real numbers) 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,$a_5 \in K$ and$\Delta \ddagger 0$. Δ is the discriminant of *E* and is defined as follows:

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

$$d_2 = a_1^2 + 4a_2$$

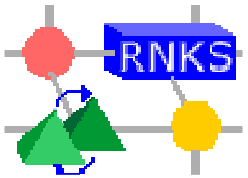
$$d_4 = 2a_4 + a_1 a_3$$

$$d_6 = a_3^2 + 4a_6$$

$$d_8 = a_1^2 a_6 + 4a_2 a_6 - a_1 a_3 a_4 + a_2 a_3^2 - a_4^2$$

- **Points: If both the coordinates of the point *P*∈*E* or *P*=∞ (the point at infinity, or zero element *O*). The set of points on *E* is:**

$$E(L) = \{(x, y) \in R \times R : y^2 + a_1xy + a_3y - x^3 - a_2x^2 - a_4x - a_6 = 0\} \cup \{o\}$$



Simplified Weierstrass Equations

- The Weierstrass equations can be simplified by performing the following change of variables:

$$(x, y) \rightarrow (x - \frac{a_2}{3}, y - \frac{a_1 x + a_3}{2})$$

and set $a_1 = 0, a_3 = 0$

$$a = 1/9a_2^2 + a_4, b = 2/27a_2^3 - 1/3a_2 a_4 a_6$$

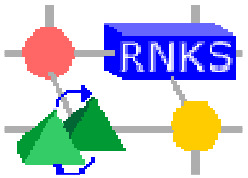
we get one of the simplified Weierstrass equations: $y^2 = x^3 + ax + b$

- By performing the following change of variables:

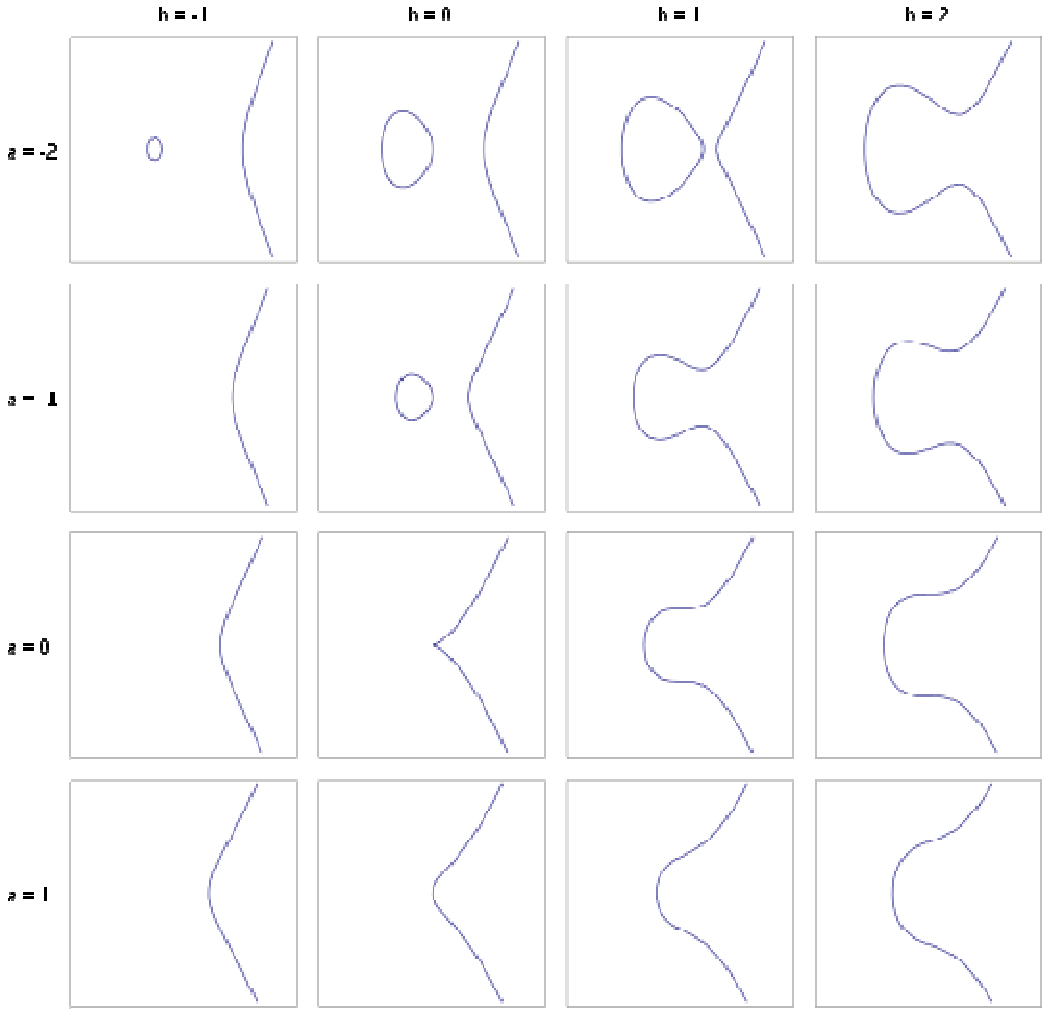
$$(x, y) \rightarrow (a_1^2 x + \frac{a_3}{a_1}, a_1^3 y + \frac{a_1^2 a_4 + a_3^2}{a_1^3})$$

We get another important simplified Weiertstrass equations:

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

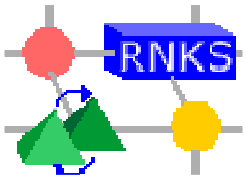


Example Curves of $y^2 = x^3 + ax + b$



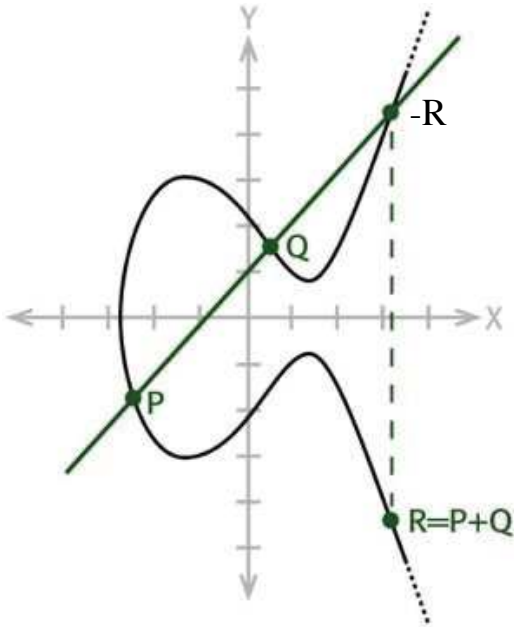
Addition law

- **Addition law of elliptic curve *E* has the following properties:**
  - Identity: $P+\mathcal{O}=\mathcal{O}+P=P$ $\quad \forall P \in E$
  - Inverse: $P+(-P)=\mathcal{O}$ $\quad \forall P \in E$
  - Associative: $P+(R+Q)=(P+R)+Q$ $\quad \forall P,Q,R \in E$
  - Commutative: $P+Q=Q+R$ $\quad \forall P, Q \in E$
- **The addition law makes the points of *E* into an abelian group.**



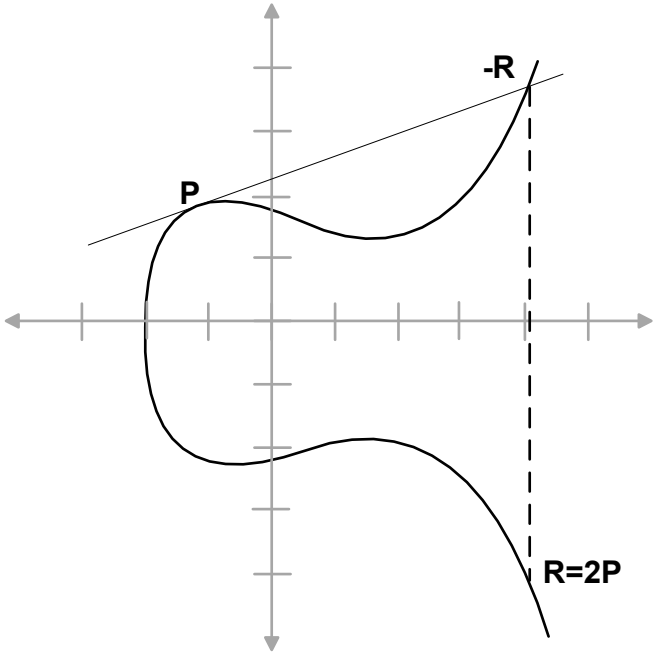
Point addition

- **Geometry approach:**
  - *To add two distinct points P and Q on an elliptic curve, draw a straight line between them. The line will intersect the elliptic cure at exactly one more point –R. The reflection of the point –R with respect to x-axis gives the point R, which is the results of addition of points R and Q*



Point doubling

- **Geometry approach:**
  - *To the point P on elliptic curve, draw the tangent line to the elliptic curve at P. The line intersects the elliptic cure at the point –R. The reflection of the point –R with respect to x-axis gives the point R, which is the results of doubling of point P.*



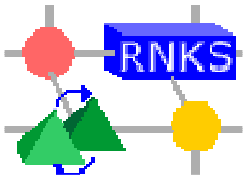
Algebraic Formulae of Point Addition

- For the curve E: $y^2 = x^3 + ax + b$. Let $P=(x_P,y_P)$ and $Q=(x_Q,y_Q) \in E$ with $P \neq Q$, then $R=P+Q=(x_R,y_R)$ is determined by the following formulae:

$$x_R = \lambda^2 - x_P - x_Q$$
$$y_R = \lambda(x_p - x_R) - y_P$$
$$where \quad : \lambda = \frac{y_Q - y_P}{x_Q - x_p}$$

- In the same way, for the curve E: $y^2 + xy = x^3 + ax^2 + b$ $R=P+Q=(x_R,y_R)$ can be determined by the following formulae:

$$x_R = \lambda^2 + \lambda + x_P + x_Q + a$$
$$y_R = \lambda(x_p + x_R) + x_R + y_P$$
$$where \quad : \lambda = \frac{y_Q + y_P}{x_Q + x_p}$$



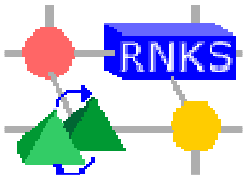
Algebraic Formulae of Point Doubling

- For the curve *E*: $y^2 = x^3 + ax + b$. Let $P=(x_P,y_P) \in E$ with $P \neq -P$, then $R=2P=(x_R,y_R)$ is determined by the following formulae:

$$x_R = \lambda^2 - 2x_P$$
$$y_R = \lambda(x_p - x_R) - y_P$$
$$where \ : \lambda = \frac{3x_P^2 + a}{2y_p}$$

- In the same way, for the curve *E*: $y^2 + xy = x^3 + ax^2 + b$ $R=2P=(x_R,y_R)$ can be determined by the following formulae:

$$x_R = \lambda^2 + \lambda + a$$
$$y_R = x_P^2 + \lambda x_R + x_R$$
$$where \ : \lambda = x_P + y_P / x_P$$



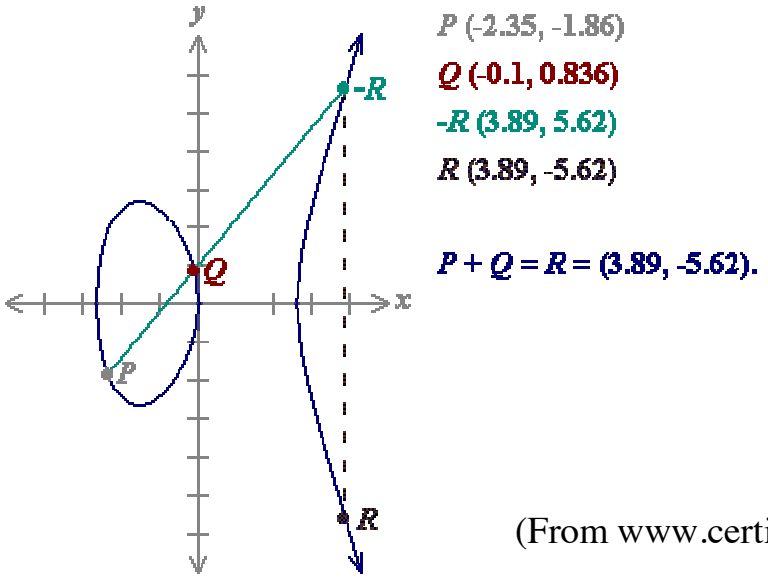
Example of Point Addition

- **Point addition in the curve** $y^2 = x^3 - 7x$

$$x_R = \lambda^2 - x_P - x_Q = 1.1982^{\,2} + 2.35 + 0.1 = 3.89$$

$$y_R = \lambda(x_p - x_R) - y_P = 1.1982\,(-2.35 - 3.89) + 1.86 = -5.62$$

$$where\ : \lambda = \frac{y_Q - y_P}{x_Q - x_p} = \frac{0.836 + 1.86}{-0.1 + 2.35} = 1.1982$$



P (-2.35, -1.86)
Q (-0.1, 0.836)
-R (3.89, 5.62)
R (3.89, -5.62)

P + *Q* = *R* = (3.89, -5.62).

(From www.certicom.com)

$y^2 = x^3 - 7x$

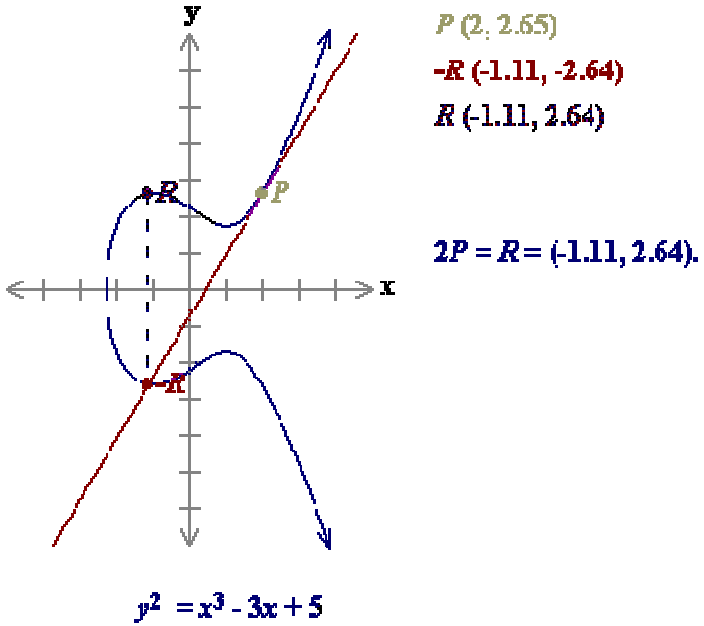
Example of Point Doubling

- **Point doubling in the curve** $y^2 = x^3 - 3x + 5$

$$x_R = \lambda^2 - 2x_P = 1.698^2 - 2 * 2 = -1.11$$

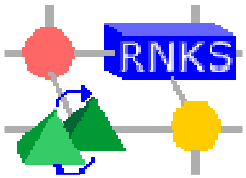
$$y_R = \lambda(x_p - x_R) - y_P = 1.698\,(2 + 1.11) - 2.65 = 2.64$$

$$where \quad : \lambda = \frac{3x_P^2 + a}{2y_p} = \frac{3 * 2^2 + (-3)}{2 * 2.65} = 1.698$$



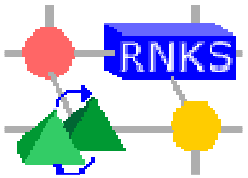
(From www.certicom.com)

III.
Elliptic Curves over Prime Field and Binary Field



Motivation

- **Elliptic curves over real numbers**
  - Calculations prove to be slow
  - Inaccurate due to rounding error
  - Infinite field
- **Cryptographic schemes need fast and accurate arithmetic**
  - In the cryptographic schemes, elliptic curves over two finite fields are mostly used.
    - Prime field $\mathbb{F}_p$, where p is a prime.
    - Binary field $\mathbb{F}_{2^m}$, where m is a positive integer.



EC over $\mathbb{F}_p$

- **The equation of the elliptic curve over $\mathbb{F}_p$ is defined as:**

$$y^2 \bmod p = (x^3 + ax + b) \bmod p$$
$$where : (4a^3 + 27b^2) \bmod p \neq 0$$
$$x, y, a, b \in [0, p-1]$$

- **The points on *E* are denoted as:**

$E(\mathbb{F}_p)=\{(x,y):x,y\in\mathbb{F}_p$ satisfy $y^2=x^3+ax+b\}\cup\{0\}$

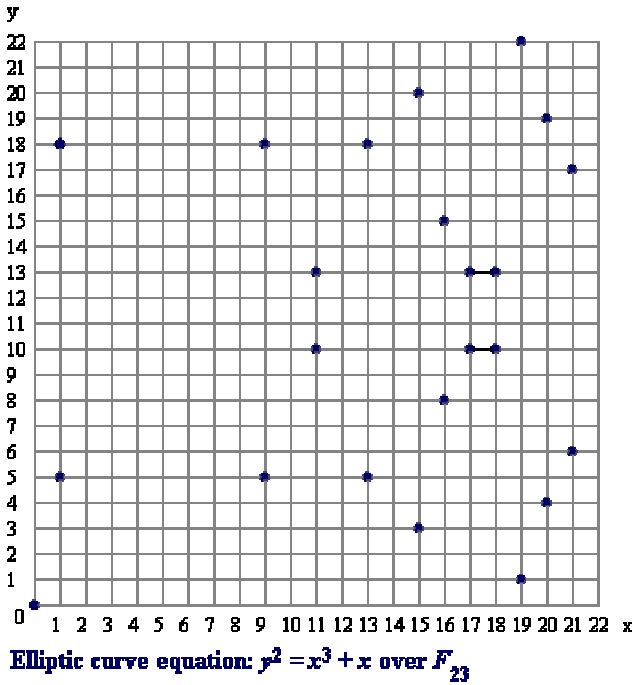
- **Example: Elliptic curve $y^2 = x^3 + x$ over the Prime field $\mathbb{F}_{23}$. The points in the curve are the Following:**

(0,0) (1,5) (1,18) (9,5) (9,18) (11,10) (11,13) (13,5)

(13,18) (15,3) (15,20) (16,8) (16,15) (17,10) (17,13) (18,10)

(18,13) (19,1) (19,22) (20,4) (20,19) (21,6) (21,17)

(From www.certicom.com)



Elliptic curve equation: $y^2 = x^3 + x$ over F_{23}

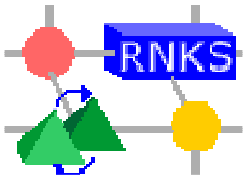
Point Addition and Doubling for EC over $\mathbb{F}_p$

- **Point addition:**

$$x_R = (\lambda^2 - x_P - x_Q) \bmod p$$
$$y_R = (\lambda(x_p - x_R) - y_P) \bmod p$$
$$where \ : \lambda = \frac{y_Q - y_P}{x_Q - x_p} \bmod p$$

- **Point doubling:**

$$x_R = (\lambda^2 - 2x_P) \bmod p$$
$$y_R = (\lambda(x_p - x_R) - y_P) \bmod p$$
$$where \ : \lambda = \frac{3x_P^2 + a}{2y_p} \bmod p$$



Example for point addition and doubling

- Let *P*=(1,5) and *Q*=(9,18) in the curve $y^2 = x^3 + x$ over the Prime field $\mathbb{F}_{23}$. Then the point R(x_R,y_R) can be calculated as

$$\lambda = \frac{18-5}{9-1} \bmod 23 = \frac{13}{8} \bmod 23 = 13 \bmod 23 \times \frac{1}{8} \bmod 23 = 13 \times 3 \bmod 23 = 16$$

$$x_R = (16^2 - 1 - 9) \bmod 23 = 246 \bmod 23 = 16$$

$$y_R = (16(1-16) - 5) \bmod 23 = -245 \bmod 23 = -15 \bmod 23 = 8$$

So the *R*=*P*+*Q*=(16,8)

The doubling point of *P* can be computed as:

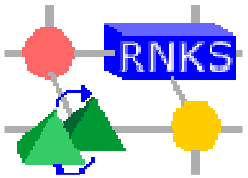
$$\lambda = \frac{3 \times 1^2 + 1}{2 \times 5} \bmod 23 = \frac{2}{5} \bmod 23 = 2 \bmod 23 \times \frac{1}{5} \bmod 23 = 2 \times 14 \bmod 23 = 5$$

$$x_R = (5^2 - 1 - 1) \bmod 23 = 23 \bmod 23 = 0$$

$$y_R = (5(1-0) - 5) \bmod 23 = 0 \bmod 23 = 0$$

So the *R*=2*P*=(0,0)

- Point addition and doubling need to perform modular arithmetic (addition, subtraction, multiplication, inversion)



EC over $\mathbb{F}_{2^m}$

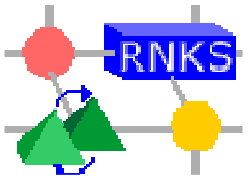
- **A elliptic curve *E* over the finite field $\mathbb{F}_{2^m}$ is given through the following equation.**

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

Where $x, y, a, b \in \mathbb{F}_{2^m}$

- **The points on *E* are denoted as:**

$E(\mathbb{F}_{2^m})=\{(x,y):x,y\in\mathbb{F}_{2^m}$ satisfy $y^2+xy=x^3+ax^2+b\}\cup\{\mathcal{O}\}$



Example Elliptic Curve over $\mathbb{F}_{2^m}$

- **Assume the finite field $\mathbb{F}_{2^4}$ has irreducible polynomial $f(x)=x^4+x+1$. The element g = (0010) is a generator for the field . The powers of g are:**

$g^0 = (0001)$ $g^1 = (0010)$ $g^2 = (0100)$ $g^3 = (1000)$ $g^4 = (0011)$ $g^5 = (0110)$

$g^6 = (1100)$ $g^7 = (1011)$ $g^8 = (0101)$ $g^9 = (1010)$ $g^{10} = (0111)$ $g^{11} = (1110)$

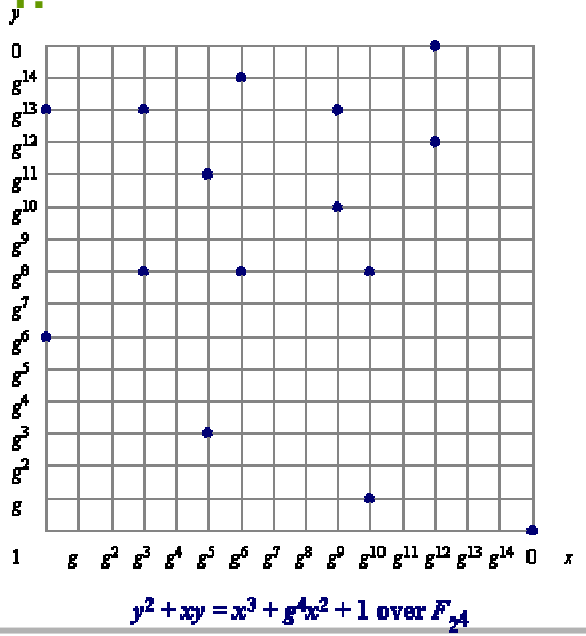
$g^{12} = (1111)$ $g^{13} = (1101)$ $g^{14} = (1001)$ $g^{15} = (0001)$

- **Consider the elliptic curve $y^2 + xy = x^3 + g^4x^2 + 1$. The points on *E* are:**

$(1, g^{13})$ (g^3, g^{13}) (g^5, g^{11}) (g^6, g^{14}) (g^9, g^{13}) (g^{10}, g^8) (g^{12}, g^{12})

$(1, g^6)$ (g^3, g^8) (g^5, g^3) (g^6, g^8) (g^9, g^{10}) (g^{10}, g) $(g^{12}, 0)$ $(0, 1)$

(From www.certicom.com)



$y^2 + xy = x^3 + g^4x^2 + 1$ over F_{24}

Point Addition and Doubling over $\mathbb{F}_{2^m}$

- Let $P=(x_P, y_P)$, $Q=(x_Q, y_Q)$ on the curve $y^2 + xy = x^3 + ax^2 + b$
  Then $R=P+Q$ can be computed:

$$x_R = \lambda^2 + \lambda + x_P + x_Q + a$$

$$y_R = \lambda(x_p + x_R) + x_R + y_P$$

$$where \ : \lambda = \frac{y_Q + y_P}{x_Q + x_p}$$

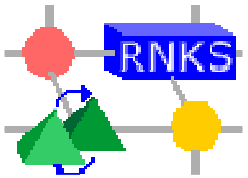
- Let $P=(x_P, y_P)$ on the curve curve $y^2 + xy = x^3 + ax^2 + b$
  Then $R=2P$ can be computed:

$$x_R = \lambda^2 + \lambda + a$$

$$y_R = x_P^2 + \lambda x_R + x_R$$

$$where \ : \lambda = x_P + y_P / x_P$$

- Note that all calculations are performed using the rules of arithmetic in $\mathbb{F}_{2^m}$



An Example of Point Addition and Doubling over $\mathbb{F}_{2^m}$

- Let $P=(g^5, g^3)$, $Q=(g^9, g^{13})$ on the curve $y^2 + xy = x^3 + g^4 x^2 + 1$
  Then $R(x_R, y_R)=P+Q$ can be computed:

$$\lambda = \frac{y_Q + y_P}{x_Q + x_p} = \frac{g^3 + g^{13}}{g^5 + g^9} = \frac{g^8}{g^6} = g^2$$

$$x_R = \lambda^2 + \lambda + x_P + x_Q + a = (g^2)^2 + g^2 + g^5 + g^9 + g^4 = g^3$$

$$y_R = \lambda(x_p + x_R) + x_R + y_P = g^2(g^5 + g^3) + g^3 + g^3 = g^{13}$$

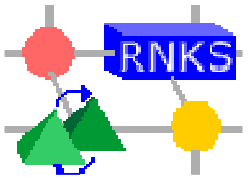
- Let $P=(x_P, y_P)$ on the curve curve $y^2 + xy = x^3 + ax + b$
  Then $R=2P$ can be computed:

$$\lambda = x_P + y_P / x_P = g^5 + g^3 / g^5 = g^5 + g^{13} = g^7$$

$$x_R = \lambda^2 + \lambda + a = (g^7)^2 + g^7 + g^4 = 1$$

$$y_R = x_P^2 + \lambda x_R + x_R = (g^5)^2 + g^7 + 1 = g^{13}$$

- Point addition and doubling need to perform the polynomial arithmetic (addition, subtraction, multiplication, and division)



Point Representation

- **The normal (*x, y*) pairs are denoted as affine coordinates. It has disadvantages in performing point addition and doubling.**
  - Expensive inverse operations are involved.

- **The normal (x, y) pairs can be represented by the triplet (*X, Y, Z*), which is called the projective coordinates. The relationship between (*x, y*) and (*X, Y,Z*) is:**

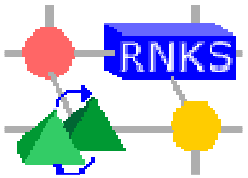
$$(X, Y, Z) = (\lambda^c x, \lambda^d y, \lambda)$$

$$(x, y) = (X / Z^c, Y / Z^d)$$

$$where \quad : \lambda \neq 0$$

- **There are a number of types of coordinates when *c, d* are set different values, such as standard[5], Jacobian[5], Lopez-Dahab[6].**

- **The use of projective coordinates can avoid the expensive inverse operations. But it requires more multiplications in the field operation. If the ratio of Inverse/Multiplication is big, the resulting computation cost of point addition is less than that using affine coordinates.**

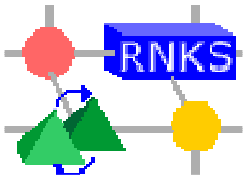


Usage of Elliptic Curves

- **An elliptic curve over $\mathbb{F}_p$ is defined as prime curve. An elliptic curve over $\mathbb{F}_{2^m}$ is defined as binary curve.**

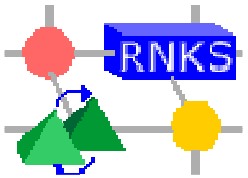
- **As pointed out in [7], prime curves are best for software applications.**
  - They do not need the extended bit-fiddling operations required by binary curves.

- **As shown in [7], binary curves are best for hardware applications.**
  - They can takes less logic gates to create a cryptosystem compared to prime curves.



Elliptic Curve Cryptography (ECC)

- Elliptic curves are used to construct the public key cryptography system
- The private key d is randomly selected from [1,n-1], where n is integer. Then the public key Q is computed by dP, where P,Q are points on the elliptic curve.
- Like the conventional cryptosystems, once the key pair (d, Q) is generated, a variety of cryptosystems such as signature, encryption/decryption, key management system can be set up.
- Computing dP is denoted as scalar multiplication. It is not only used for the computation of the public key but also for the signature, encryption, and key agreement in the ECC system.



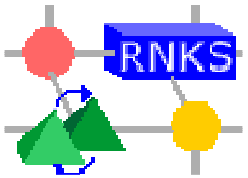
Scalar Multiplication

- **Intuitive approach:**

$$dP = \underbrace{P+P+\ldots+P}_{d \text{ times}}$$

It requires *d*-1 times point addition over the elliptic curve.

- **Observation: To compute 17 *P*, we could start with 2*P*, double that, and that two more times, finally add *P*, i.e. 17*P*=2(2(2(2*P*)))+*P*. This needs only 4 point doublings and one point addition instead of 16 point additions in the intuitive approach. This is called Double-and-Add algorithm.**



Double-and-Add algorithm

- **Let $d=(d_{t-1}, d_{t-2},\ldots d_0)$ be the binary representation of d, then**

$$d = \sum_{i=0}^{t-1} d_i 2^i$$

$$dP = (\sum_{i=0}^{t-1} d_i 2^i)P = (d_{t-1}2^{t-1}P) + \ldots + (d_1 2P) + d_0 P$$

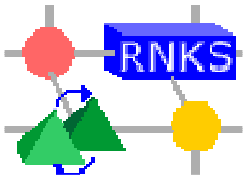
$$= 2(2(\ldots 2(2d_{t-1}P) + d_{t-2}P) + \ldots) + d_1 P) + d_0 P$$

- **Double-and-Add algorithm:**

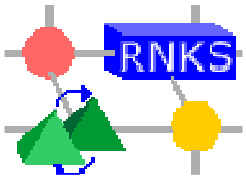
Input: $d=(d_{t-1}, d_{t-2},\ldots d_0)$, $P \in E$.

1. $Q \leftarrow \mathcal{O}$
2. For i from 0 to t-1 do
   - 2.1 If $d_i=1$ then $Q \leftarrow Q+P$
   - 2.2 $P \leftarrow 2P$

Output: $dP=Q$

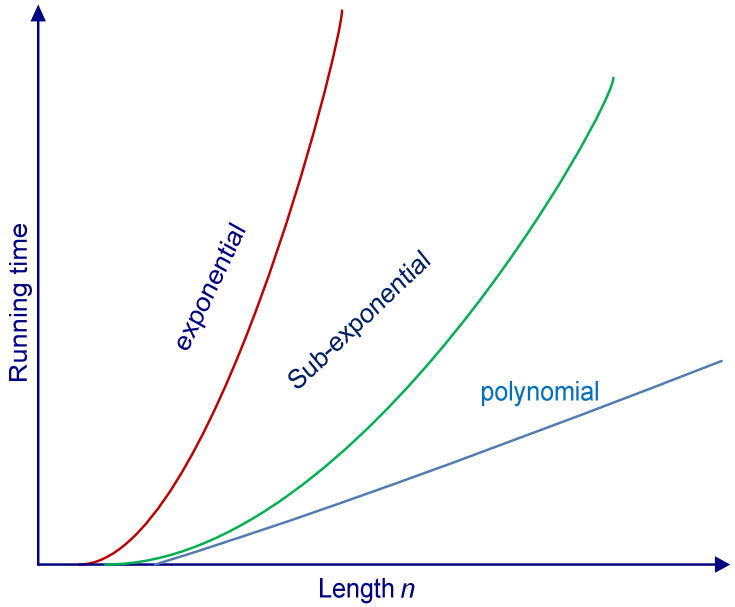


IV.
Security Strength of ECC System



Complexity of an Algorithm

- **Definition: Let A be an algorithm whose input has bit-length *n***
  - A is a *polynomial-time algorithm* if its running time is $O(n^c)$ for some constant $c>0$, such as n^{10}
  - A is a *subexponential-time algorithm* if its running time is $O(e^{o(n)})$, such as $e^{n^{1/3}}$.
  - A is an *exponential-time algorithm* if its running time is $O(c^n)$ or $O(n^{f(n)})$ for $c>1$, such as 1.1^n and n^{n^2}.

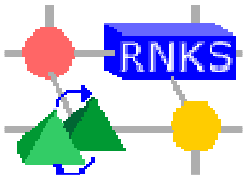


Security of Public Key Cryptosystems

- **A Public key cryptosystem is constructed on the basis of hardness of some mathematic problems.**
  - RSA depends on the intractability of factoring problem
  - DH protocol relies on the hardness of discrete logarithm
  - ECC is secure due to the elliptic curve discrete logarithm problem (ECDLP).

- **A public key cryptosystem consist of a private key that is kept secret, and a public key which is accessible to the public.**

- **The straightforward way to break the public key cryptosystem is to draw the private key from the public key. But the required computation cost is equivalent to solving these difficult mathematic problems.**



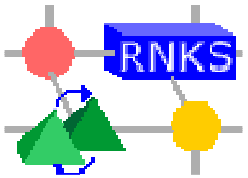
RSA

- **RSA key pair generation.**
  - Randomly select two large primes p and q, and $p \neq q$
  - Compute $n=pq$ and $\phi=(p-1)(q-1)$
  - Select an arbitrary integer e with $1<e<\phi$ and gcd $(e,\phi)=1$.
  - Compute the integer d satisfying $1<d<\phi$ and $ed \equiv 1 \pmod{\phi}$

  The public key is (n,e), the private key is d.

- **Observation: If we can derive the primes p and q from n, $\phi=(p-1)(q-1)$ can be computed. The enables the determination of the private key $d \equiv e^{-1} \pmod{\phi}$.**

- **Multiplying two prime integers together is easy, but factoring the product of two prime numbers is much more difficult.**



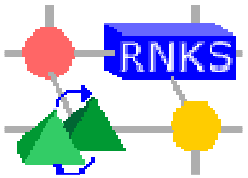
Factoring problem

- **Definition**
  - Given a positive integer *n*, find its two prime factorization *p* and *q*.

- **The best published solution to the factoring problem is the general number field sieve (GNFS) algorithm, which, for a number *n*, its running time is:**

$$L_n[1/3, 1.923] = O(e^{1.923(\log n)^{1/3}(\log\log n)^{2/3}})$$

- **GNFS is a subexponential time algorithm.**



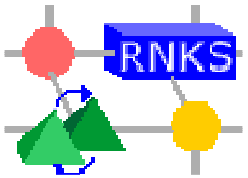
Deffie-Hellman

- **DH key pair generation**
  - *G* is finite group with generator *g*, *p* is a prime and *q* is a prime divisor of *p*-1.
  - Randomly select *x* from [1, *q*-1]
  - Compute $y=g^x \pmod p$

  The public key is *y*, and private key is *x*.

- **Observation: $x=log_g\ y \pmod p$, x is called the discrete logarithm of *y* to the base *g*.**

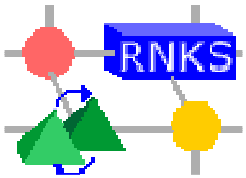
- **Given *g*,*x*, and *p*, it is trivial to calculate *y*. However, given *y*, *g*, and *p* it is difficult to calculate *x*.**



Discrete Logarithm Problem

- **Definition**
  - Given a prime *p*, generator *g*, and an element *y* in group G, find the integer *x*, such that $y=g^x \pmod{p}$.

- **The fastest algorithm known for solving discrete logarithm problem is still GNFS which has a subexponential running time.**



ECC

- **Key pair generation**
  - Randomly select $d \in [1, n-1]$.
  - Compute $Q=dP$, P, Q is a point on the curve

  Public key is Q, private key is d

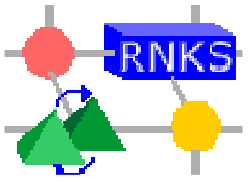
- **The naive algorithm to draw the *d* from *Q* is the computation of a sequence of points *P, 2P, 3P, 4P*, until *Q=dP*.**

- **Hasse Theorem: the number of points of $E(\mathbb{F}_q)$ is denoted by $\# E(\mathbb{F}_q)$, which is determined by**

$$\# E(F_q) = q + 1 - t$$

$$where |t| \le 2\sqrt{q}$$

- **Usually *q* is a large prime number whose length is greater than 160 bit. So $\# E(\mathbb{F}_q)$ is also a big number. Thus it is computationally infeasible to solve *d* from *Q* by using the naive algorithm.**



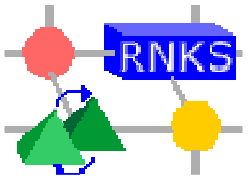
Elliptic Curve Discrete Logarithm Problem (ECDLP)

- **Definition**
  - Given an elliptic curve E defined over a finite field $\mathbb{F}_q$, a point $P \in E(\mathbb{F}_q)$ of order n, and a point $Q \in E$, find the integer $d \in [0, n-1]$ such that $Q=dP$.

- **The fastest algorithm to solve ECDLP is Pollard's rho algorithm, its running time is**

$$\sqrt{\frac{\pi q}{2}}$$

- **Pollard's rho algorithm is an exponential-time algorithm**

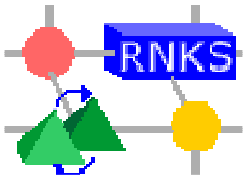


Key Size Comparsion

NIST recommended key sizes

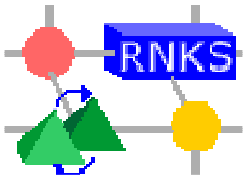
Symmetric algorithm (bit)	RSA and DH (bit)	ECC (bit)
56	512	112
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	521

- **The reason is that there exist subexponential-time algorithms for factoring and discrete logarithm problem, whilst only exponential-time algorithms for ECDLP.**

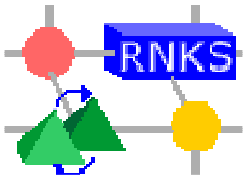


Selecting an Appropriate Elliptic Curve

- **Conditions to be satisfied:**
  - $\#E(\mathbb{F}_q)$ should be divisible by a sufficiently large prime, in order to resist against the Pollard ρ-attack.
  - $\#E(\mathbb{F}_q)$ should not to be equal to q, to avoid the Semaev-Smart-Satoh-Araki attack.
  - To resist the MOV reduction attack, n should not divide q^k-1 for all $1 \leq k \leq 30$.



V.
Elliptic Curve Protocols



Elliptic Curve Digital Signature Algorithm (ECDSA)

Alice

Private key d_A, Public key $Q_A=d_AP$.

Signature generation

1.Select a random k from $[1, n-1]$

2.Compute $kP=(x_1,y_1)$ and $r=x_1 \bmod n$. if $r=0$ goto step 1

3.Compute $e=H(m)$, where H is a hash functon, m is the message.

4.Compute $s=k^{-1}(e+d_Ar) \bmod n$. If $s=0$ go to step 1.

(r, s) is Alice's signature of message m

$m,(r,s)$

Bob

Signature verification

1.Verify that r, s are in the interval $[1, n-1]$

2.Compute $e=H(m)$, where H is a hash functon, m is the message.

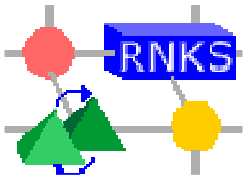
3.Compute $w=s^{-1} \bmod n$

4.Compute $u_1=ew \bmod n$ and $u_2=rw \bmod n$.

5.Compute $X=u_1P+u_2Q_A=(x_1,y_1)$

6.Compute $v=x_1 \bmod n$

7.Accept the signature if and only if $v=r$



Proof the correctness of ECDSA

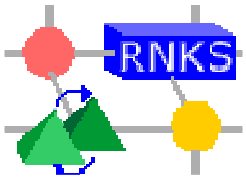
Proof

- If a signature (r,s) on a message m was authentic, then $s=k^{-1}(e+d_A r) \bmod n$. It can be rewritten as:

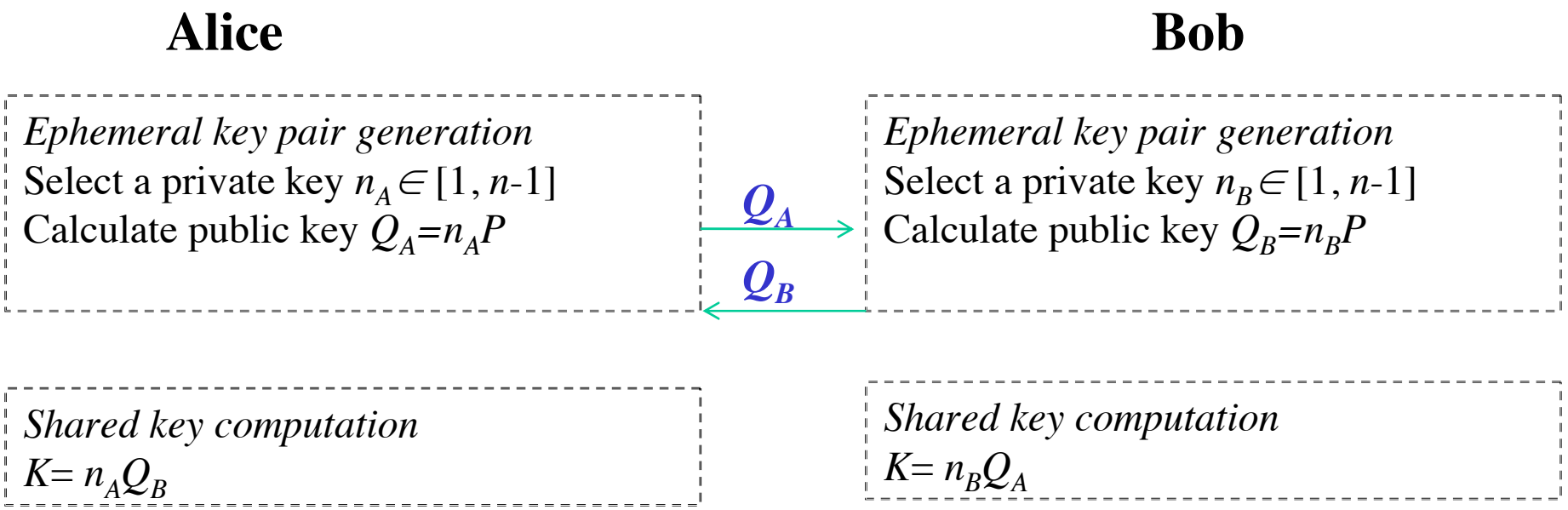
$$k \equiv s^{-1}(e+d_A r) \equiv s^{-1}e+s^{-1}rd_A \equiv we+wrd_A \equiv u_1+u_2 d_A \pmod n$$

Thus $X=u_1P+u_2Q_A=(u_1+u_2d_A)P=kP$.

So v=r is required.



Elliptic Curve Deffie-Hellmen (ECDH)



- **Consistency: $K = n_A Q_B = n_A n_B P = n_B Q_A$**
- **ECDH is vulnerable to the man-in-the-middle attack**

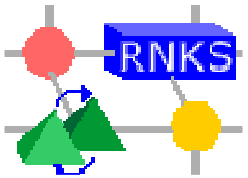
An Example of ECDH

- Alice and Bob make a key agreement over the following prime, curve, and point.
  p=3851, E:$y^2=X^3+324x+1287$, $P=(920,303)\in E(\mathbb{F}_{3851})$

- Alice chooses the private key n_A=1194,
  computes $Q_A=1194P=(2067,2178)\in E(\mathbb{F}_{3851})$, and sends it to Bob.

- Bob chooses the private key n_B=1759
  computes $Q_A=1759P=(3684,3125)\in E(\mathbb{F}_{3851})$, and sends it to Alice.

- Alice computes $n_AQ_B=1194(3684,3125)=(3347,1242)\in E(\mathbb{F}_{3851})$
  Bob computes $n_BQ_A=1759(2067,2178)=(3347,1242)\in E(\mathbb{F}_{3851})$



Authenticated Key Agreement Protocol ECMQV

Alice (Q_A, d_A)

1. ID_A, R_A →
2. ← $ID_B, R_B, t_B=MAC_{kl}(2, ID_B, ID_A, R_B, R_A)$
3. $t_A=MAC_{kl}(3, ID_A, ID_B, R_A, R_B)$ →

Bob (Q_B, d_B)

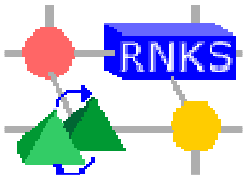
1. Select a random k_A from $[1, n-1]$, compute ephemeral public key $R_A=k_AP$, sends ID_A, R_A to Bob

3. Receiving message **2**, Alice does the following
3.1 Compute $s_A=(k_A+\overline{R}_A d_A) \bmod n$ and $Z=hs_A(R_B+\overline{R}_B Q_B)$
3.2 $(k_1, k_2)\leftarrow KDF(x_Z)$
3.3 Compute $t=MAC_{kl}(2, ID_B, ID_A, R_B, R_A)$ and verify that $t=t_B$
3.4 Compute $t_A=MAC_{kl}(3, ID_A, ID_B, R_A, R_B)$
3.5 Send t_A to Bob

2. Receiving message **1**, Bob does the following
2.1 Generate the ephemeral public key $R_B=k_BP$
2.2 Compute $s_B=(k_B+\overline{R}_B d_B) \bmod n$ and $Z=hs_B(R_A+\overline{R}_A Q_A)$, where $\overline{R}_B, \overline{R}_A$ is the integer representation of the x-coordinate of R_B, R_A, h *is* one of EC domain parameters.
2.3 $(k_1, k_2)\leftarrow KDF(x_Z)$, where x_Z is the x-coordinate of Z, *KDF* is a key derivation function
2.4 Compute $t_B=MAC_{kl}(2, ID_B, ID_A, R_B, R_A)$
2.5 Send ID_B, R_B, t_B to Alice.

4. Receiving message 3, Bob computes $t= MAC_{kl}(3, ID_A, ID_B, R_A, R_B)$ and verify that $t=t_A$

***Z* is the shared secret**

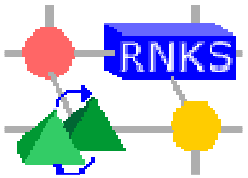


Explanation of ECMQV

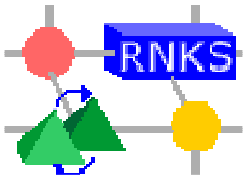
- **Resist against the man-in-the-middle attack**
  - The quantity $s_A=(k_A+\overline{R}_A d_A) \bmod n$ serves as an implicit signature for Alice. It is a signature in the sense that only person who knows Alice's private key d_A can produce s_A. Bob indirectly verifies its validity by using $s_A P=R_A+\overline{R}_A Q_A$
  - In the same way, the quantity $s_B=(k_B+\overline{R}_B d_B) \bmod n$ serves as an implicit signature for Bob.

- The shared secret between Alice and Bob is Z
  - Bob computes $Z=hs_B(R_A+\overline{R}_A Q_A)=hs_B s_A P$
  - Alice computes $Z=hs_A(R_B+\overline{R}_B Q_B)=hs_A s_B P$

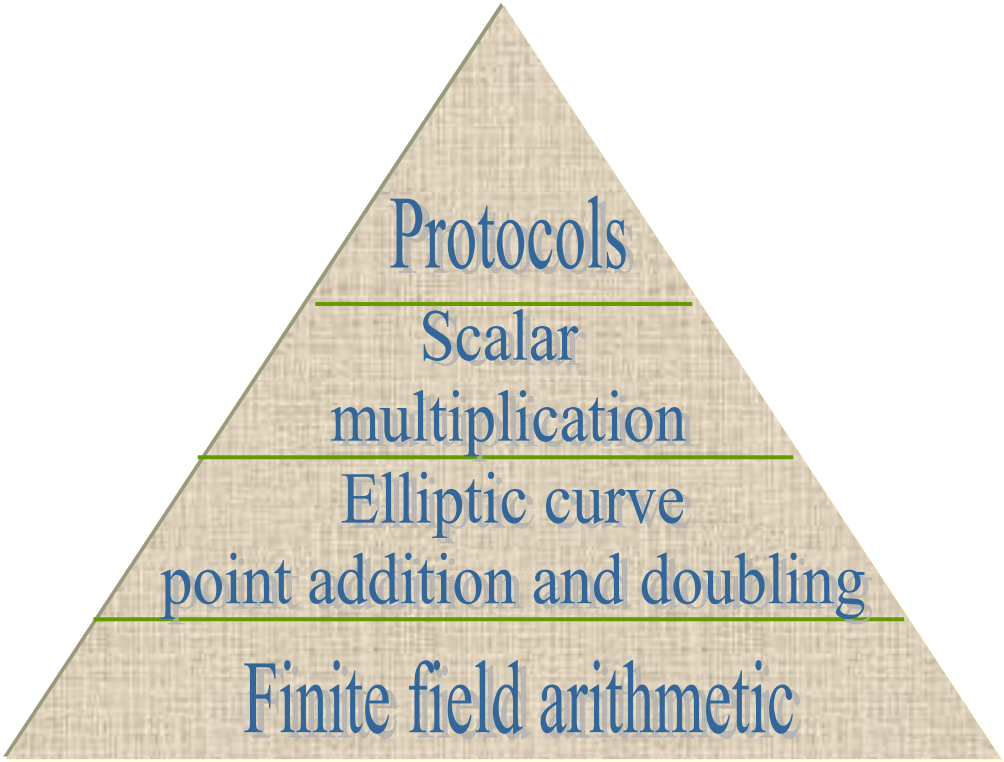
- The function of MAC is to ensure that the messages exchanged between Alice and Bob are authentic.



VI.
Patents and Standards



ECC System Structure



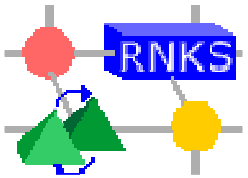
Patents

- The general idea of ECC was not patented [8], but there are a number of patents regarding the efficient implementation from the underlying layer (finite field arithmetic) to the highest layer (protocols)

- The patent issue for elliptic curve cryptosystems is the opposite of that for RSA and Diffie-Hellman, where the cryptosystems themselves have patents, but efficient implementation techniques often do not [8].

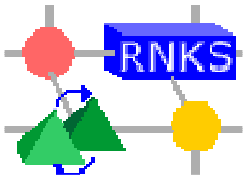
- Certicom holds more than 130 patents related to ECC. It has sold 26 patents to NSA and NISA in the value of 26 million US$, which covers the prime field curves with primes of 256 bits, 384 bits and 521 bits.

- Certicom was taken over by the RIM(Research in Motion) with the offer of 130 million C$ in 2009.

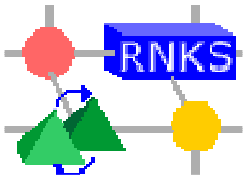


Standards

Standard	Titel
ANSI X9.63	Public Key Cryptography for the Financial Services Industry, Key Agreement and Key Transport Using Elliptic Curve Cryptography
IEEE P1363	Standard Specifications For Public-Key Cryptography
ISO 15946	*Information technology — Security techniques — Cryptographic techniques based on elliptic curves*
NIST SP 800-56	Recommendation on key establishment schemes

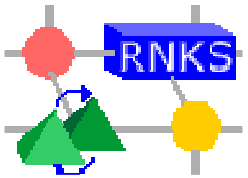


VII.
Final Remarks

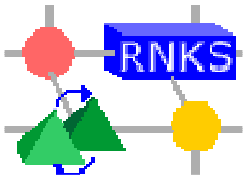


Final Remarks

- **The mathematic background of ECC is more complex than other cryptographic systems**
  - Geometry, abstract algebra, number theory
- **ECC provides greater security and more efficient performance than the first generation public key techniques (RSA and Diffie-Hellman)**
  - Mobile systems
  - Systems required high security level (such as 256 bit AES)
- **The next step is to apply the ECDH principle to the group key management protocol.**
- **Unless the explicit statement of sources, the materials used in this tutorial are from Hankerson's book[9] and www.certicom.com.**



Thanks



Reference

(1) W. Diffie and M. Hellman: New Directions in Cryptography. IEEE Transactions on Information Theory, 22:644-654,1976.

(2) N. Koblitz. Elliptic curve cryptosystems. Mathematics of Computation, 48:203–209,1987.

(3) V. Miller. Use of elliptic curves in cryptography. Advances in Cryptology—CRYPTO ’85 (LNCS 218) [483], 417–426, 1986.

(4) G. Faltings (July 1995): The Proof of Fermat's Last Theorem by R. Taylor and A. Wiles. Notices of the AMS 42 (7): 743–746. ISSN 0002-9920. July 1995.

(5) I. Blake, G. Seroussi, and N. Smart: Elliptic Curves in Cryptography. Cambridge, U.K.: Cambridge University Press, 1999, vol 265.

(6) J. Lopez and R. Dahab: Improved algorithms for elliptic curve arithmetic in GF(2^n). Selected Areas in Cryptography—SAC ’98 (LNCS 1556) [457], 201–212, 1999.

(7) A. Fernandes: Elliptic Curve cryptography. Dr. Dobb’s Journal, December 1999.

(8) RSA Laboratory: FAQ. http://www.rsa.com/rsalabs/node.asp?id=2325

(9) D. Hankerson, A. Menezes, and S. Vanstone: Guide to Elliptic Curve Cryptography. Springer, 2004.