Elliptic Curve Cryptography (ECC)

- For the same length of keys, faster than RSA
- For the same degree of security, shorter keys are required than RSA
- Standardized in IEEE P1363
- Confidence level not yet as high as that in RSA
- Much more difficult to explain than RSA

Elliptic Curve Cryptography (cont'd)

- Named so because they are described by cubic equations (used for calculating the circumference of an ellipse)
- Of the form $y^2 + axy + by = x^3 + cx^2 + dx + e$

  where all the coefficients are real numbers satisfying some simple conditions
- Single element denoted *O* and called the *point at infinity* or the *zero point*

Elliptic Curve Cryptography (cont'd)

- Define the rules of addition over an elliptic curve
  - O serves as the additive identity. Thus $O = -O$; for any point P on the elliptic curve, $P + O = P$.
  - $P_1 = (x,y)$, $P_2 = (x,-y)$. Then, $P_1 + P_2 + O = O$, and therefore $P_1 = -P_2$.
  - To add two points Q and R with different x coordinates, draw a straight line between them and find the third point of intersection P_1. If the line is tangent to the curve at either Q or R, then $P_1 = Q$ or R. Finally, $Q + R + P_1 = O$ and $Q + R = -P_1$.

Elliptic Curve Cryptography (cont'd)

- Define the rules of addition over an elliptic curve (cont'd)
  - To double a point Q, draw the tangent line and find the other point of intersection S. Then $Q + Q = 2Q = -S$.

Elliptic Curve Cryptography (cont'd)

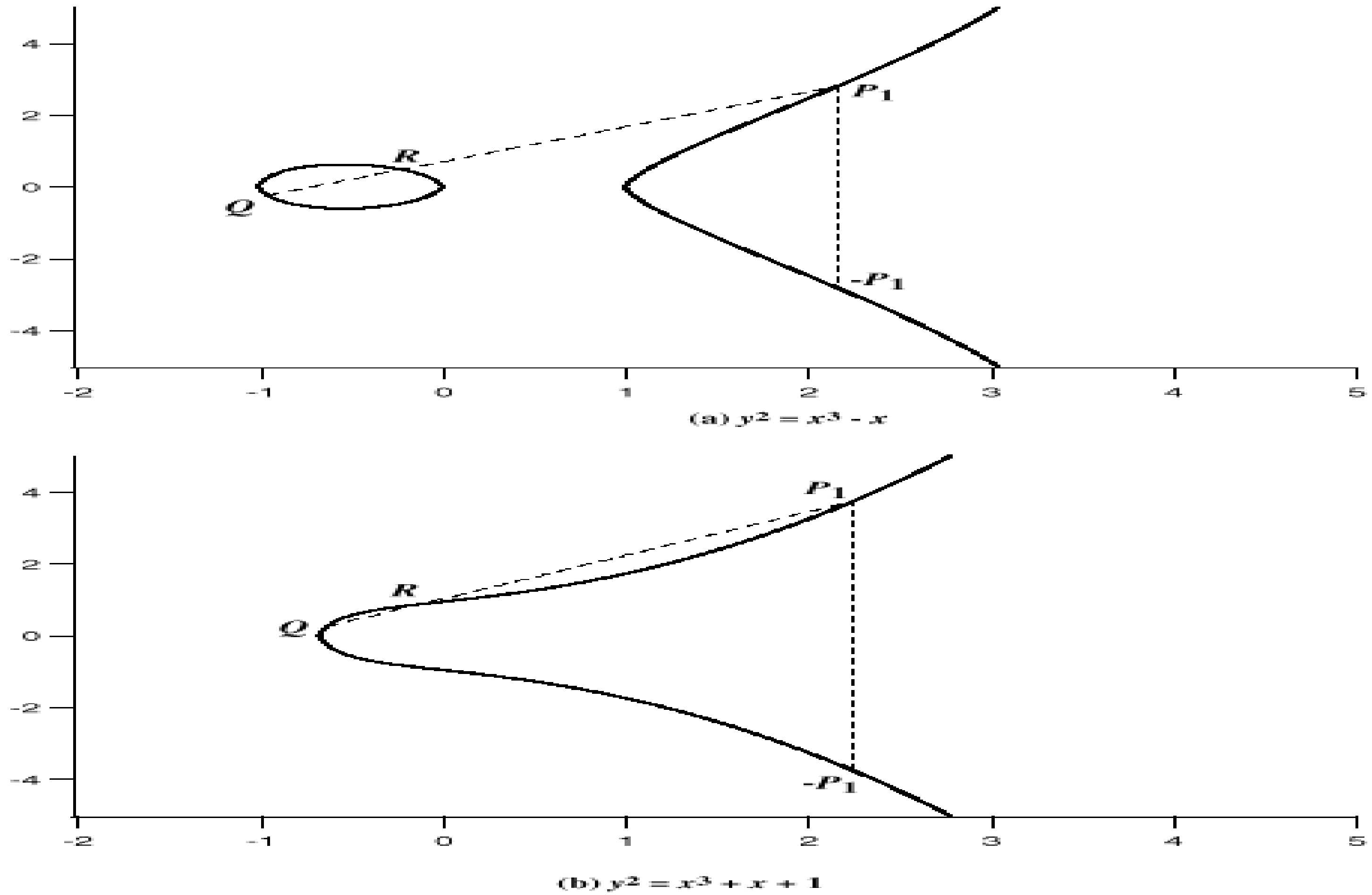

Figure 6.18 Example of Elliptic Curves

Elliptic Curve Cryptography (cont'd)

- Elliptic curves over finite field
  - Define ECC over a finite field
  - The elliptic group mod *p*, where *p* is a prime number
  - Choose 2 nonnegative integers *a* and *b*, less than *p* that satisfy

    $[4a^3 + 27b^2] \pmod{p} \neq 0$

  - $E_p(a,b)$ denotes the elliptic group mod *p* whose element (x,y) are pairs of non-negative integers less than *p* satisfying

    $y^2 \equiv x^3 + ax + b \pmod{p}$, with O

Elliptic Curve Cryptography (cont'd)

- Elliptic curves over finite field (cont'd)
  - Example: Let p = 23, a = b = 1. This satisfies the condition for an elliptic curve group mod 23.

Elliptic Curve Cryptography (cont'd)

- Generation of nonnegative integer points from (0,0) to (p,p) in E_p

1. For each x such that $0 \le x < p$, calculate $x^3 + ax + b$ (mod p).
2. For each result from the previous step, determine if it has a square root mod p. If not, there are no points in $E_p(a, b)$ with this value of x. If so, there will be two values of y that satisfy the square root operation (unless the value is the single y value of 0). These (x, y) values are points in $E_p(a, b)$.

Elliptic Curve Cryptography (cont'd)

- Rules of addition over $E_p(a,b)$

1. $P + O = P$.
2. If $P = (x, y)$, then $P + (x, -y) = O$. The point $(x, -y)$ is the negative of P, denoted as $-P$. Observe that $(x, -y)$ is a point on the elliptic curve, as seen graphically (Figure 6.18b) and in $E_p(a, b)$. For example, in $E_{23}(1, 1)$, for $P = (13,7)$, we have $-P = (13, -7)$. But -7 mod 23 = 16. Therefore, $-P = (13, 16)$, which is also in $E_{23}(1, 1)$.

Table 6.4 Points on the Elliptic Curve $E_{23}(1, 1)$

(0,1)	(6,4)	(12,19)
(0,22)	(6,19)	(13,7)
(1,7)	(7,11)	(13,16)
(1,16)	(7,12)	(17,3)
(3,10)	(9,7)	(17,20)
(3,13)	(9,16)	(18,3)
(4,0)	(11,3)	(18,20)
(5,4)	(11,20)	(19,5)
(5,19)	(12,4)	(19,18)

Elliptic Curve Cryptography (cont’d)

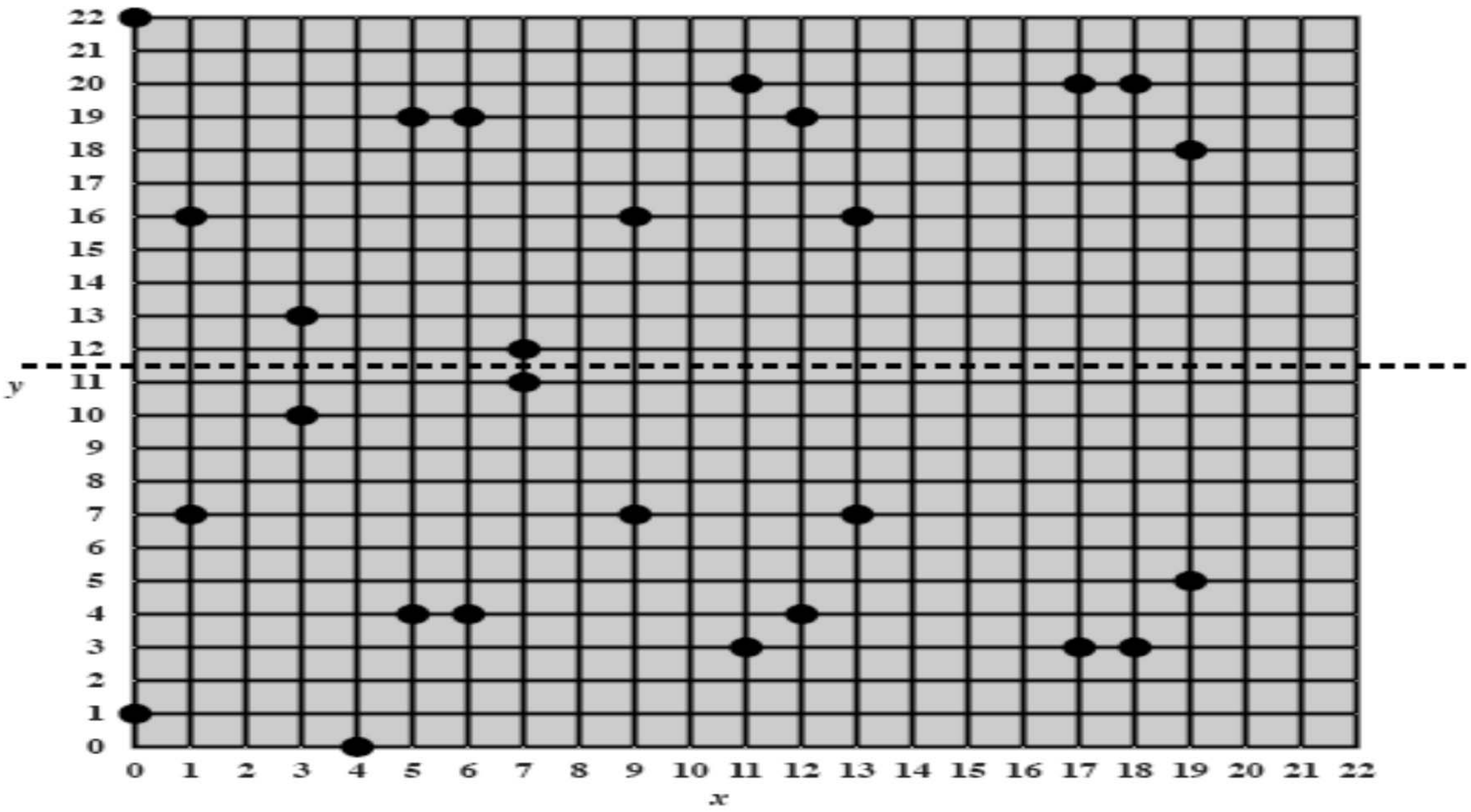

Figure 10.10 The Elliptic Curve $E_{23}(1,1)$

Elliptic Curve Cryptography (cont'd)

- Rules of addition over $E_p(a,b)$ (cont'd)

3. If $P=(x_1, y_1)$ and $Q = (x_2, y_2)$ with $P \neq -Q$, then $P + Q = (x_3, y_3)$ is determined by the following rules:

$$x_3 \equiv \lambda^2 - x_1 - x_2 \pmod p$$

$$y_3 \equiv \lambda(x_1 - x_3) - y_1 \pmod p \text{, where}$$

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & if\ P \neq Q \\ \dfrac{3x_1^{\ 2} + a}{2y_1} & if\ P = Q \end{cases}$$

We look at two examples, taken from [JUR197]. Let $P = (3, 10)$ and $Q = (9, 7)$. Then

$$\lambda = \frac{7-10}{9-3} = \frac{-3}{6} = \frac{-1}{2} \equiv 11 \bmod 23$$

$$x_3 = 11^2 - 3 - 9 = 109 \equiv 17 \bmod 23$$

$$y_3 = 11(3-(-6)) - 10 = 89 \equiv 20 \bmod 23$$

So $P + Q = (17, 20)$. To find $2P$,

$$\lambda = \frac{3(3^2)+1}{2 \times 10} = \frac{5}{20} = \frac{1}{4} \equiv 6 \bmod 23$$

$$x_3 = 6^2 - 3 - 3 = 30 \equiv 7 \bmod 23$$

$$y_3 = 6(3-7) - 10 = -34 \equiv 12 \bmod 23$$

and $2P = (7,12)$. Again, multiplication is defined as repeated addition; for example, $4P = P + P + P + P$.

Elliptic Curve Cryptography (cont'd)

- Analogy of Diffie-Hellman key exchange
  - Pick a prime number *p* in the range of 2^{180}.
  - Choose *a* and *b*.
  - Define the elliptic group of points $E_p(a,b)$.
  - Pick a generator (base) point $G = (x,y)$ in $E_p(a,b)$ such that the smallest value of *n* for which $nG = O$ be a very large number (referred of the order of *G*).
  - $E_p(a,b)$ and *G* are known to the participants.

Elliptic Curve Cryptography (cont'd)

- Analogy of Diffie-Hellman key exchange (cont'd)

  1. A selects an integer n_A less than n. This is A's private key. A then generates a public key $P_A = n_A \times G$; the public key is a point in $E_p(a, b)$.
  2. B similarly selects a private key n_B and computes a public key P_B.
  3. A generates the secret key $K = n_A \times P_B$. B generates the secret key $K = n_B \times P_A$.

Elliptic Curve Cryptography (cont'd)

- Analogy of Diffie-Hellman key exchange (cont'd)
  - Example: p = 211; for $E_p(0,-4)$, choose G = (2,2). Note that 241G = O. n_A =121, and P_A = 121(2,2) = (115,48). n_B = 203 and P_B = 203(2,2) = (130,203). The shared secret key is then 121(130,203) = 203(115,48) = (161,169).
  - For choosing a single number as the secret key, we could simply use the *x* coordinates or some simple function of the *x* coordinate.

Elliptic Curve Cryptography (cont'd)

- Elliptic curve encryption/decryption
  - Encode the plain text m to be sent as an x-y point P_m.
  - There are relatively straightforward techniques to perform such mappings.
  - Require a point G and an elliptic group $E_p(a,b)$ as parameters.
  - Each user A selects a private key n_A and generates a public key $P_A = n_A \times G$

Elliptic Curve Cryptography (cont'd)

- Elliptic curve encryption/decryption (cont'd)
  - To encrypt and send a message P_m from A to B
    - A chooses a random positive integer k.
    - A then produces the ciphertext C_m consisting of the *pair* of points:

      $$C_m = \{kG, P_m + k\,P_{\mathrm{B}}\}.$$
  - A has used B's public key P_{B}.
  - Two instead of one piece of information are sent.

Elliptic Curve Cryptography (cont'd)

- Elliptic curve encryption/decryption (cont'd)
  - To decrypt C_m

    $$P_m + k\,P_{\mathrm{B}} - n_{\mathrm{B}}(kG) = P_m + k\,(n_{\mathrm{B}}G) - n_{\mathrm{B}}(kG) = P_m.$$

  - A has masked P_m by adding $k\,P_{\mathrm{B}}$ to it.
  - An attacker needs to compute k given G and kG, which is assumed hard.

Elliptic Curve Cryptography (cont'd)

- Elliptic curve encryption/decryption (cont'd)
  - Example: Take p = 751, $E_p(-1,188)$ and G = (0,376). Assume that P_m = (562,201) is to be sent and that the sender chooses a random number k = 386. Assume that the receiver's public key is P_B = (201,5). We have 386(0,376) = (676,558), and (562,201) + 386(201,5) = (385,328). Consequently, {(676,558), (385,328)} is sent as the ciphertext.

Elliptic Curve Cryptography (cont'd)

- Computational effort for cryptanalysis of elliptic curve cryptography compared to RSA

Key Size	MIPS-Years
150	3.8*10^10
205	7.1*10^18
234	1.6*10^28

(a) Elliptic Curve Logarithms Using the Pollard rho Method

Key Size	MIPS-Years
512	3*10^4
768	2*10^8
1024	3*10^11
1280	1*10^14
1536	3*10^16
2048	3*10^20

(b) Integer Factorization Using the General Number Field Sieve

Elliptic Curve Cryptography (cont’d)

	1024-bits RSA	163-bits ECC
Security Level	= 163-bits ECC	= 1024-bits RSA
Certificate Size (key and signature)	Over 256-bytes	Over 62-bytes
Key Generation (ms)	285,630	397
Signature Generation (ms)	20,208	528
Signature Verification (ms)	900	1,142

Source: Motorola, 2001 (on a Palm Pilot)