SNS COLLEGE OF ENGINEERING

Kurumbapalayam(Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NAAC-UGC with 'A' Grade

Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

DEPARTMENT OF INFORMATION TECHNOLOGY

Course Code and Name : 19IT602– CRYPTOGRAPHY AND CYBER SECURITY

III YEAR / VI SEMESTER

Unit 3: ASYMMETRIC KEY CRYPTOGRAPHY

Topic : Elliptic curve cryptography

Elliptic Curve Arithmetic

- Most of the products and standards that use public-key cryptography for encryption and digital signatures use RSA
  - The key length for secure RSA use has increased over recent years and this has put a heavier processing load on applications using RSA
- Elliptic curve cryptography (ECC) is showing up in standardization efforts including the IEEE P1363 Standard for Public-Key Cryptography
- Principal attraction of ECC is that it appears to offer equal security for a far smaller key size
- Confidence level in ECC is not yet as high as that in RSA

Abelian Group

- A set of elements with a binary operation, denoted by •, that associates to each ordered pair *(a, b)* of elements in *G* an element *(a • b)* in *G,* such that the following axioms are obeyed:

(A1) Closure: If *a* and *b* belong to *G*, then *a* • *b* is also in *G*

(A2) Associative: $a \bullet (b \bullet c) = (a \bullet b) \bullet c$ for all *a, b, c* in *G*

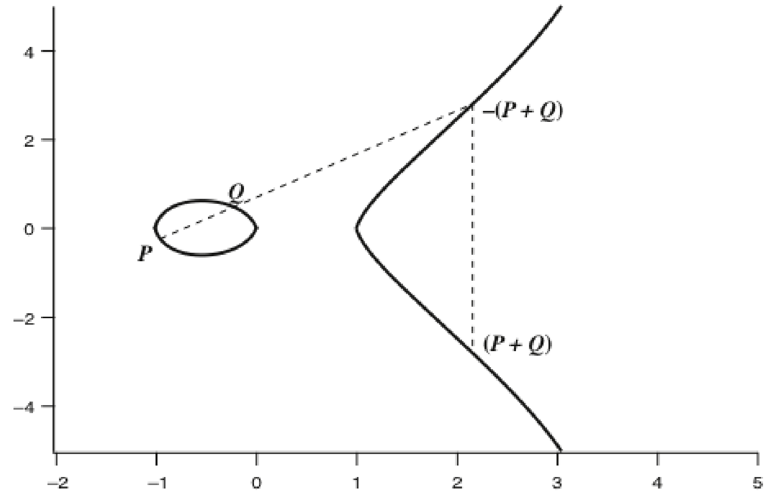
(A3) Identity element:

There is an element *e* in *G* such that $a \bullet e = e \bullet a = a$ for all *a* in *G*

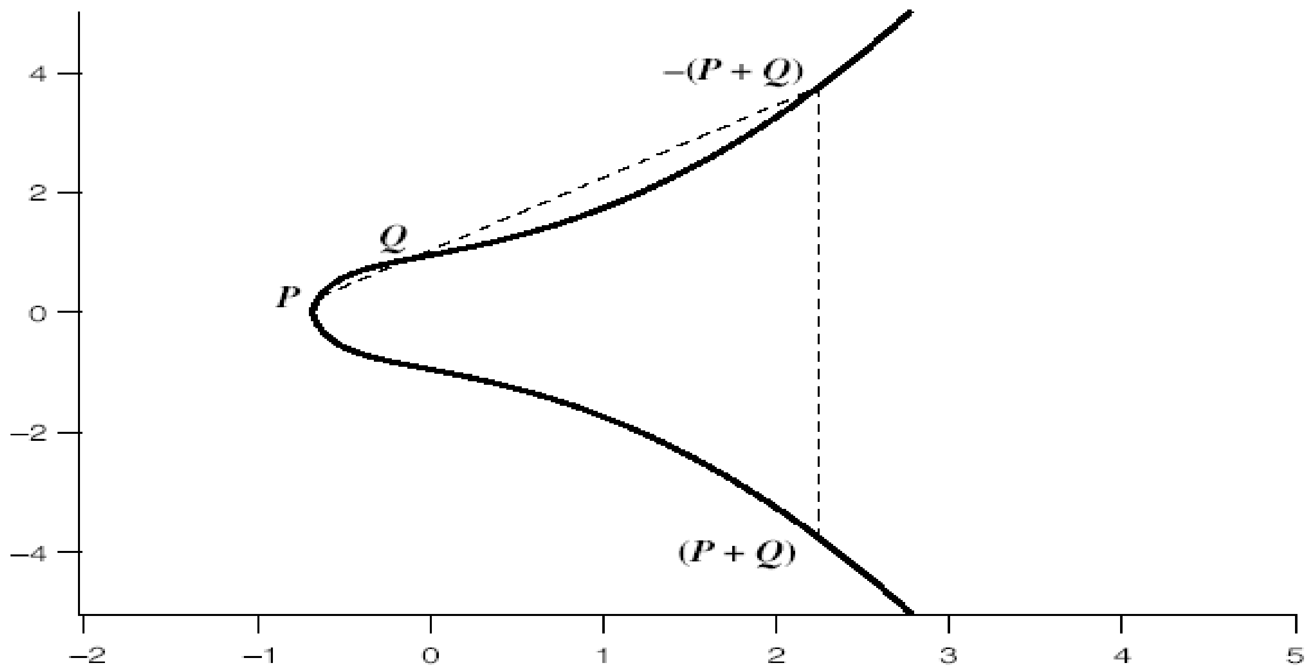
(A4) Inverse element:

For each *a* in *G* there is an element *a′* in *G* such that $a \bullet a' = a' \bullet a = e$

(A5) Commutative: $a \bullet b = b \bullet a$ for all *a, b* in *G*



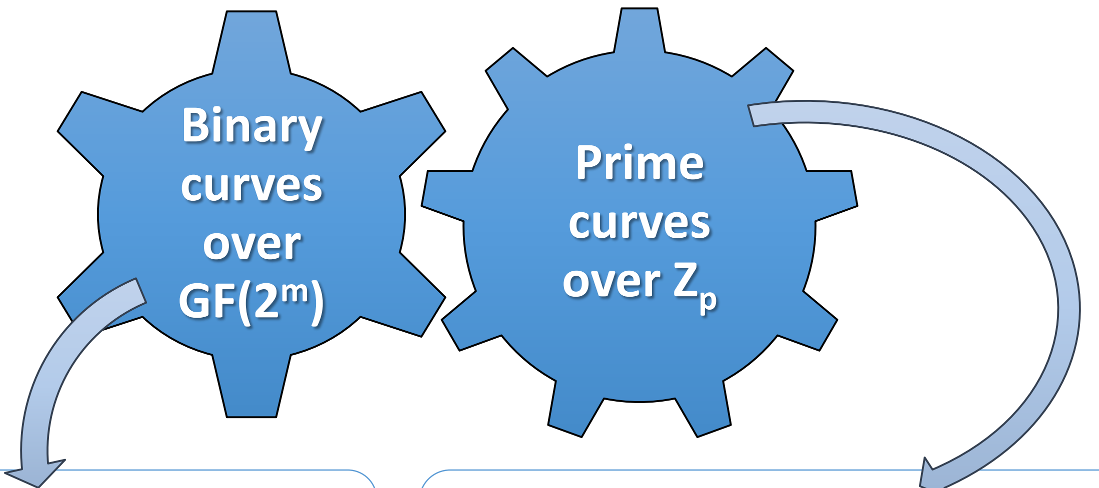
(a) $y^2 = x^3 - x$



(b) $y^2 = x^3 + x + 1$

Elliptic Curves Over Z_p

- Elliptic curve cryptography uses curves whose variables and coefficients are finite
- Two families of elliptic curves are used in cryptographic applications:



- **Variables and coefficients all take on values in $GF(2^m)$ and in calculations are performed over $GF(2^m)$**
- **Best for hardware applications**

- **Use a cubic equation in which the variables and coefficients all take on values in the set of integers from 0 through p-1 and in which calculations are performed modulo p**
- **Best for software applications**

Points (other than *O*) 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)

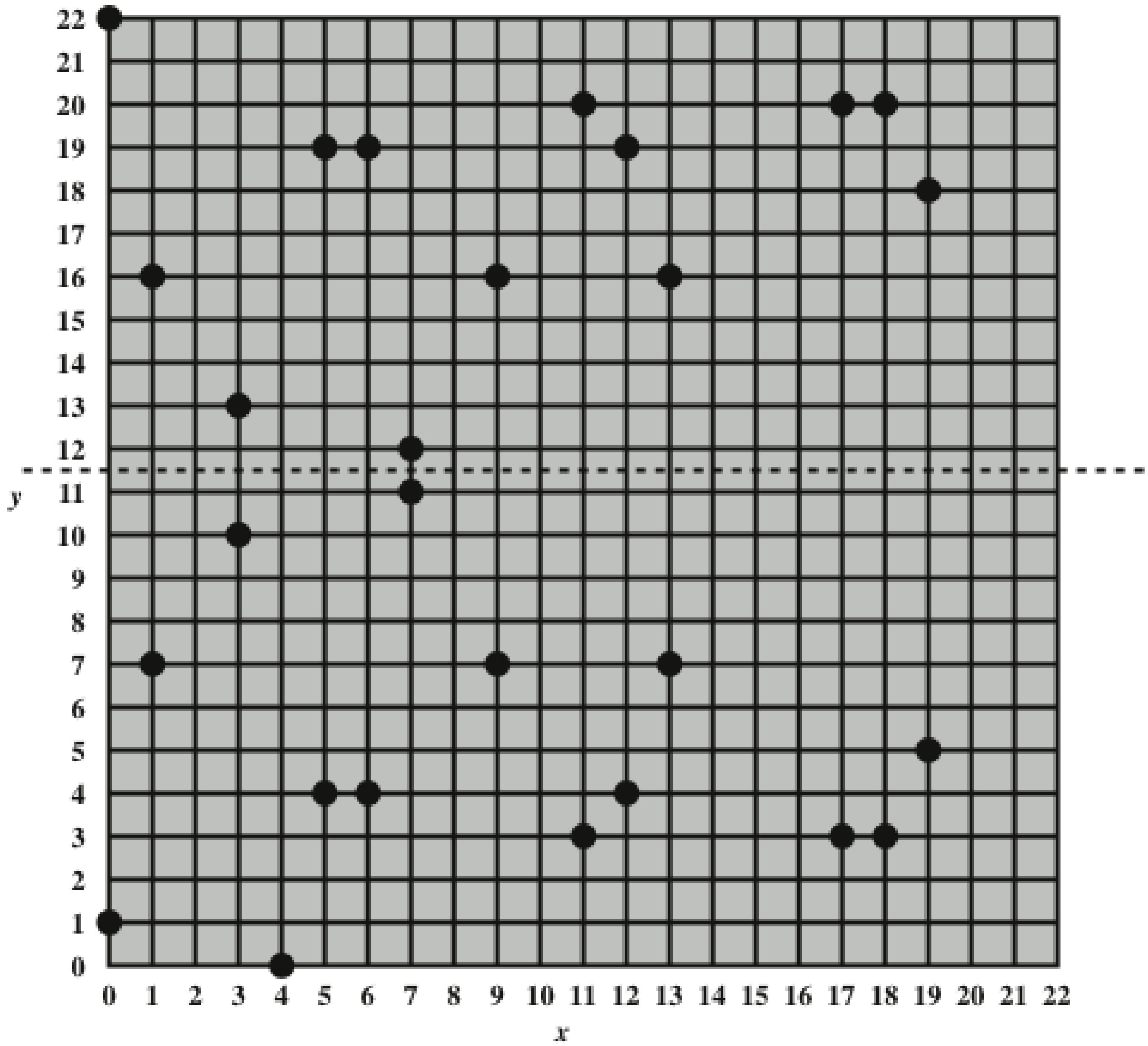


Figure 10.5 The Elliptic Curve $E_{23}(1,1)$

Elliptic Curves Over GF(2^m)

- Use a cubic equation in which the variables and coefficients all take on values in GF(2^m) for some number *m*
- Calculations are performed using the rules of arithmetic in GF(2^m)
- The form of cubic equation appropriate for cryptographic applications for elliptic curves is somewhat different for GF(2^m) than for Z_p
  - It is understood that the variables *x* and *y* and the coefficients *a* and *b* are elements of GF(2^m) and that calculations are performed in GF(2^m)

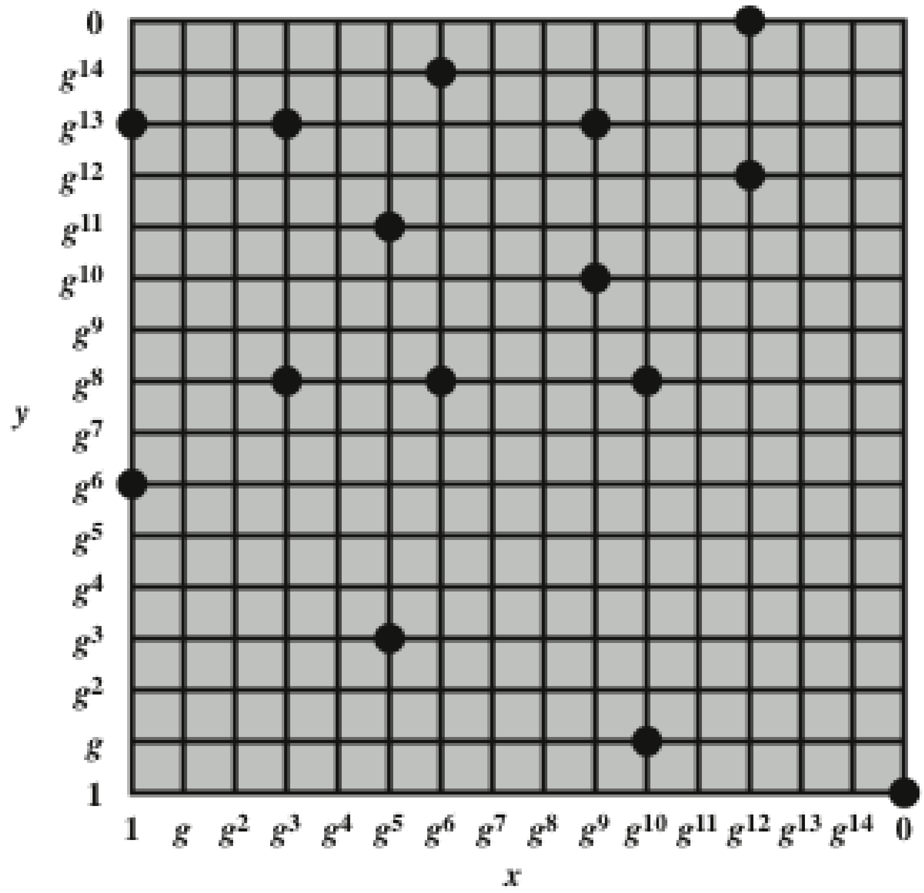


Figure 10.6 The Elliptic Curve $E_{2^4}(g^4, 1)$

Elliptic Curve Cryptography (ECC)

- Addition operation in ECC is the counterpart of modular multiplication in RSA
- Multiple addition is the counterpart of modular exponentiation

To form a cryptographic system using elliptic curves, we need to find a “hard problem” corresponding to factoring the product of two primes or taking the discrete logarithm

- *Q=kP,* where *Q, P* belong to a prime curve
- Is “easy” to compute *Q* given *k* and *P*
- But “hard” to find *k* given *Q*, and *P*
- Known as the elliptic curve logarithm problem

Global Public Elements	
$E_q(a, b)$	elliptic curve with parameters a, b, and q, where q is a prime or an integer of the form 2^m
G	point on elliptic curve whose order is large value n

User A Key Generation	
Select private n_A	$n_A < n$
Calculate public P_A	$P_A = n_A \times G$

User B Key Generation

Select private n_B	$n_B < n$
Calculate public P_B	$P_B = n_B \times G$

Calculation of Secret Key by User A

$$K = n_A \times P_B$$

Calculation of Secret Key by User B

$$K = n_B \times P_A$$

ECC Diffie-Hellman

- can do key exchange analogous to D-H
- users select a suitable curve $E_p(a,b)$
- select base point $G=(x_1,y_1)$ with large order n s.t. $nG=O$
- A & B select private keys $n_A<n$, $n_B<n$
- compute public keys: $P_A=n_A \times G$, $P_B=n_B \times G$
- compute shared key: $K=n_A \times P_B$, $K=n_B \times P_A$
  - same since $K=n_A \times n_B \times G$

ECC Encryption/Decryption

- several alternatives, will consider simplest
- must first encode any message M as a point on the elliptic curve P_m
- select suitable curve & point G as in D-H
- each user chooses private key $n_A < n$
- and computes public key $P_A = n_A \times G$
- to encrypt P_m : $C_m = \{kG, P_m + k\, P_b\}$, k random
- decrypt C_m compute:

$$P_m + kP_b - n_B(kG) = P_m + k(n_BG) - n_B(kG) = P_m$$

Security of Elliptic Curve Cryptography

- Depends on the difficulty of the elliptic curve logarithm problem
- Fastest known technique is “Pollard rho method”
- Compared to factoring, can use much smaller key sizes than with RSA
- For equivalent key lengths computations are roughly equivalent
- Hence, for similar security ECC offers significant computational advantages

Comparable Key Sizes in Terms of Computational Effort for Cryptanalysis (NIST SP-800-57)

Symmetric key algorithms	Diffie-Hellman, Digital Signature Algorithm	RSA (size of n in bits)	ECC (modulus size in bits)
80	$L = 1024$ $N = 160$	1024	160–223
112	$L = 2048$ $N = 224$	2048	224–255
128	$L = 3072$ $N = 256$	3072	256–383
192	$L = 7680$ $N = 384$	7680	384–511
256	$L = 15,360$ $N = 512$	15,360	512+

Note: L = size of public key, N = size of private key

THANK YOU