Course: Cryptography and Network Security
Code: CS-34310
Branch: M.C.A - 4th Semester

Lecture – 10 : Elgamal and ECC
ASYMMETRIC-KEY CRYPTOGRAPHY

Faculty & Coordinator : Dr. J Sathish Kumar (JSK)

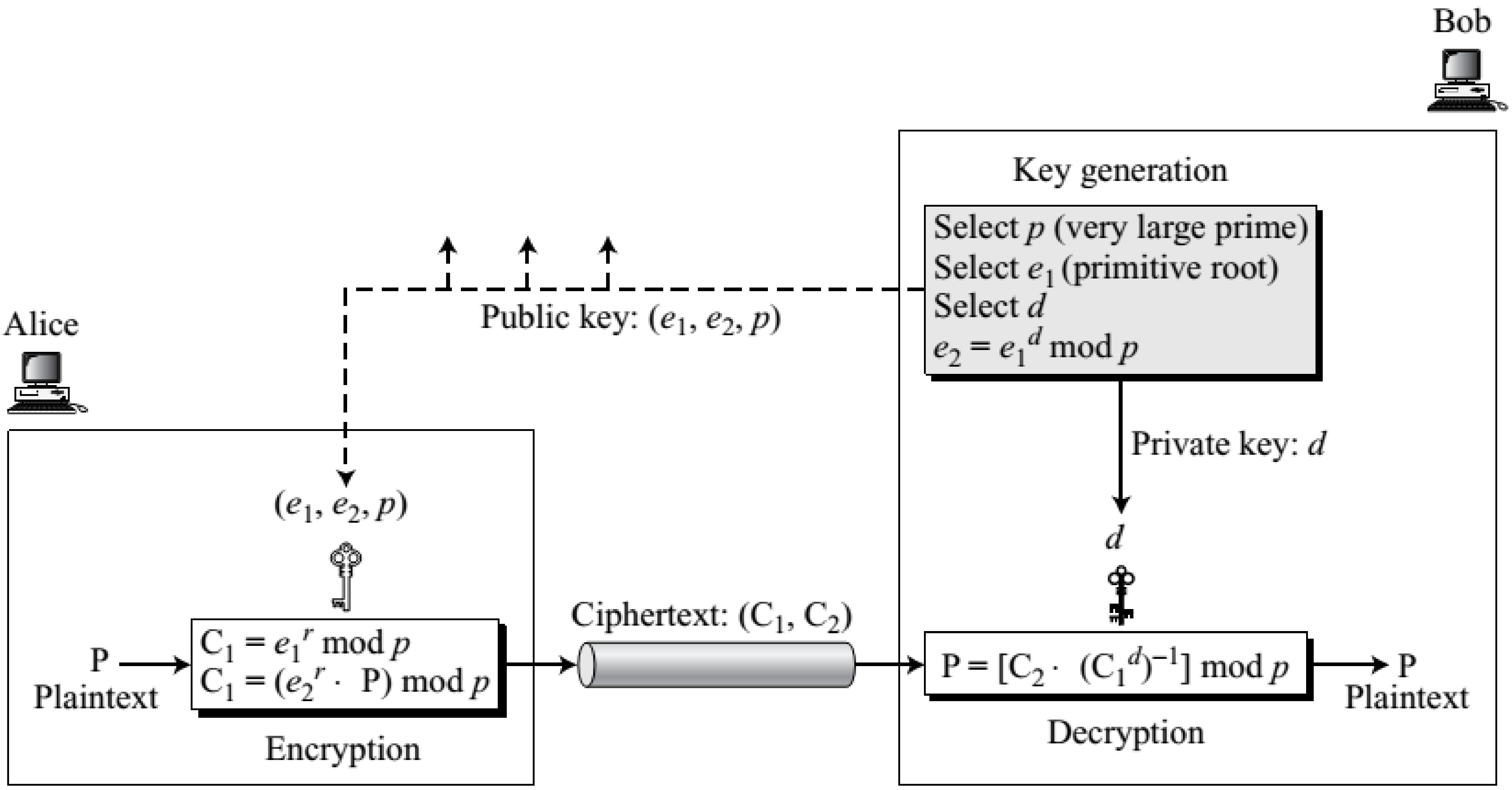
Department of Computer Science and Engineering

Motilal Nehru National Institute of Technology Allahabad,
Prayagraj-211004

ELGAMAL CRYPTOSYSTEM

- If p is a very large prime,
- e_1 is a primitive root in the group $G = <Zp^*, \times>$ and
- r is an integer, then $e_2 = e_1^r \bmod p$ is easy to compute using the fast exponential algorithm (square-and-multiply method),
- but given e_2, e_1, and p, it is infeasible to calculate $r = \log_{e1} e2 \bmod p$ (discrete logarithm problem).

ELGAMAL CRYPTOSYSTEM



ELGAMAL CRYPTOSYSTEM: Key Generation

ElGamal_Key_Generation

{

Select a large prime p

Select d to be a member of the group $\mathbf{G} = < \mathbf{Z}_p{}^*, \times >$ such that $1 \leq d \leq p - 2$

Select e_1 to be a primitive root in the group $\mathbf{G} = < \mathbf{Z}_p{}^*, \times >$

$e_2 \leftarrow e_1{}^d \bmod p$

Public_key $\leftarrow (e_1, e_2, p)$ // To be announced publicly

Private_key $\leftarrow d$ // To be kept secret

return Public_key and Private_key

}

ELGAMAL CRYPTOSYSTEM Encryption and Decryption

ElGamal_Encryption $(e_1, e_2, p, \mathrm{P})$ // P is the plaintext

{

Select a random integer r in the group $\mathbf{G} = \langle \mathbf{Z}_p{}^*, \times \rangle$

$\mathrm{C}_1 \leftarrow e_1{}^r \bmod p$

$\mathrm{C}_2 \leftarrow (\mathrm{P} \times e_2{}^r) \bmod p$ // C_1 and C_2 are the ciphertexts

return C_1 and C_2

}

ElGamal_Decryption $(d, p, \mathrm{C}_1, \mathrm{C}_2)$ // C_1 and C_2 are the ciphertexts

{

$\mathrm{P} \leftarrow [\mathrm{C}_2\,(\mathrm{C}_1{}^d)^{-1}] \bmod p$ // P is the plaintext

return P

}

ELGAMAL CRYPTOSYSTEM

- The ElGamal decryption expression $C_2 \times (C_1{}^{d})^{-1}$ can be verified to be P
- Proof

$$[C_2 \times (C_1{}^{d})^{-1}] \bmod p = [(e_2{}^{r} \times P) \times (e_1{}^{rd})^{-1}] \bmod p = (e_1{}^{dr}) \times P \times (e_1{}^{rd})^{-1} = P$$

ELGAMAL CRYPTOSYSTEM: Example

- Bob chooses 11 as p. He then chooses $e_1 = 2$. Note that 2 is a primitive root in Z_{11}^*. Bob then chooses d = 3 and calculates $e_2 = e_1^d = 8$. So the public keys are (2, 8, 11) and the private key is 3. Alice chooses r = 4 and calculates C_1 and C_2 for the plaintext 7.

Plaintext: 7
$C_1 = e_1^r \bmod 11 = 16 \bmod 11 = 5 \bmod 11$
$C_2 = (P \times e_2^r) \bmod 11 = (7 \times 4096) \bmod 11 = 6 \bmod 11$
Ciphertext: (5, 6)

Bob receives the ciphertexts (5 and 6) and calculates the plaintext.

$[C_2 \times (C_1^d)^{-1}] \bmod 11 = 6 \times (5^3)^{-1} \bmod 11 = 6 \times 3 \bmod 11 = 7 \bmod 11$
Plaintext: 7

Security of ElGamal

- Low-Modulus Attacks
  - If the value of p is not large enough, Eve can use some efficient algorithms to solve the discrete logarithm problem to find d or r.
  - If p is small, Eve can easily find $d = \log_{e_1} e_2 \bmod p$ and store it to decrypt any message sent to Bob.
  - This can be done once and used as long as Bob uses the same keys.
  - It is recommended that p be at least 1024 bits (300 decimal digits).

Security of ElGamal

- Known-Plaintext Attack
  - If Alice uses the same random exponent r, to encrypt two plaintexts P and P′, Eve discovers P′ if she knows P.
  - Assume that $C_2 = P \times (e_2^r) \bmod p$ and $C'_2 = P' \times (e_2^r) \bmod p$.
  - Eve finds P′ using the following steps

1. $(e_2^r) = C_2 \times P^{-1} \bmod p$
2. $P' = C'_2 \times (e_2^r)^{-1} \bmod p$

It is recommended that Alice use a fresh value of r to thwart the known-plaintext attacks

ELLIPTIC CURVE CRYPTOSYSTEMS

- Although RSA and ElGamal are secure asymmetric-key cryptosystems, their security comes with a price, their large keys.
- Researchers have looked for alternatives that give the same level of security with smaller key sizes.
- One of these promising alternatives is the elliptic curve cryptosystem (ECC).
- The system is based on the theory of elliptic curves.

ELLIPTIC CURVE CRYPTOSYSTEMS

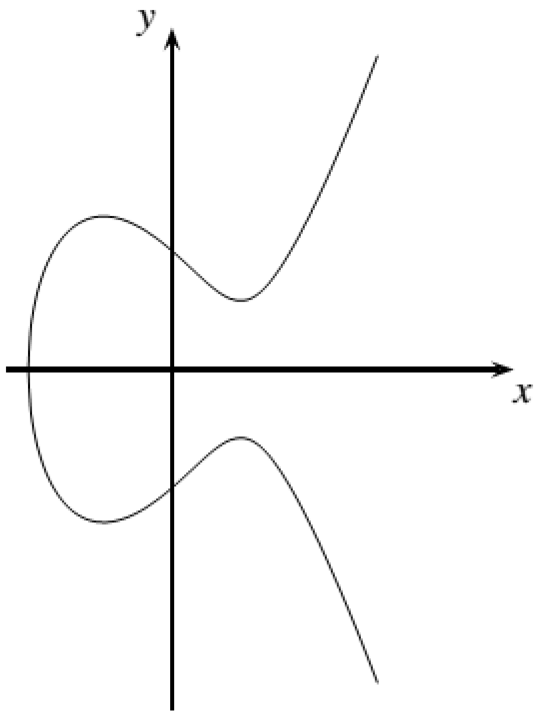
The elliptic curve *over* $\mathbb{Z}_p$, $p > 3$, *is the set of all pairs* $(x, y) \in \mathbb{Z}_p$ *which fulfill*

$$y^2 \equiv x^3 + a \cdot x + b \bmod p$$

together with an imaginary point of infinity $\mathcal{O}$, *where*

$$a, b \in \mathbb{Z}_p$$

and the condition $4 \cdot a^3 + 27 \cdot b^2 \neq 0 \bmod p$.



$4.a^3 + 27.b^2 = 0$ => Singular Ellipic Curve => No three distinct roots

$y^2 = x^3 - 3x + 3$ over $\mathbb{R}$

Finding Points on the Curve

ellipticCurve_points (p, a, b) // p is the modulus

{

$x \leftarrow 0$

while $(x < p)$

{

$w \leftarrow (x^3 + ax + b) \bmod p$ // w is y^2

if (w is a perfect square in $\mathbf{Z}_p$) output $(x, \sqrt{w})$ $(x, -\sqrt{w})$

$x \leftarrow x + 1$

{

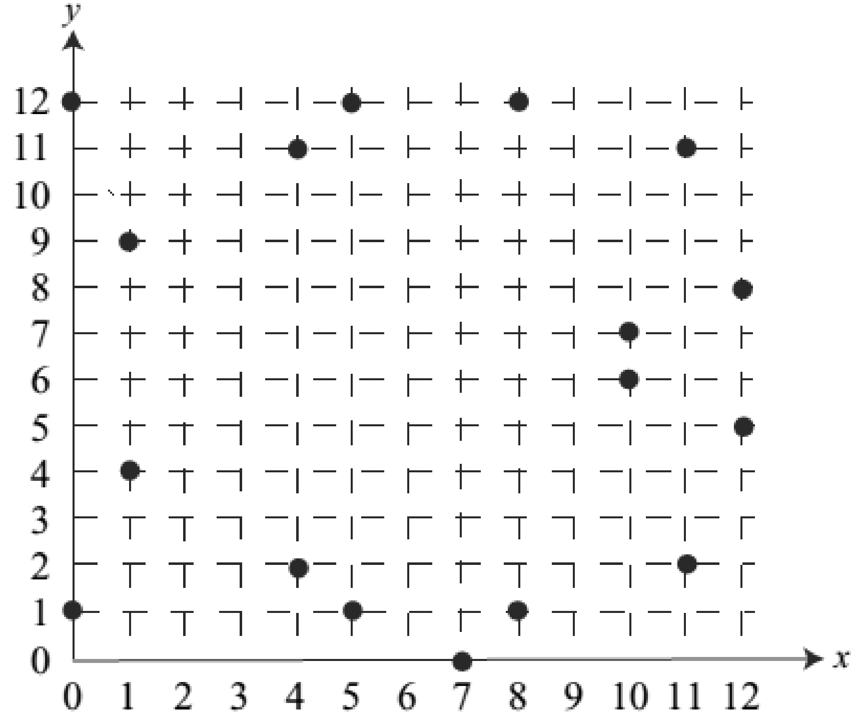
}

Finding Points on the Curve

- Define an elliptic curve $E_{13}(1, 1)$. The equation is $y^2 = x^3 + x + 1$ and the calculation is done modulo 13. Points on the curve can be found as shown in Figure

(0, 1)	(0, 12)
(1, 4)	(1, 9)
(4, 2)	(4, 11)
(5, 1)	(5, 12)
(7, 0)	(7, 0)
(8, 1)	(8, 12)
(10, 6)	(10, 7)
(11, 2)	(11, 11)
(12, 5)	(12, 8)

Points



Graph

Finding Points on the Curve

- Some values of y^2 do not have a square root in modulo 13 arithmetic. These are not points on this elliptic curve. For example, the points with $x = 2$, $x = 3$, $x = 6$, and $x = 9$ are not on the curve.
- Each point defined for the curve has an inverse. The inverses are listed as pairs. Note that (7, 0) is the inverse of itself.
- Note that for a pair of inverse points, the y values are additive inverses of each other in Z_p. For example, 4 and 9 are additive inverses in Z_{13}. So we can say that if 4 is y, then 9 is −y.
- The inverses are on the same vertical lines.

Group Operations on Elliptic Curves

- The group operation with the addition symbol2 "+".
- "Addition" means that given two points and their coordinates, say *P* = (*x*1*, y*1) and *Q* = (*x*2*, y*2), we have to compute the coordinates of a third point *R* such that:

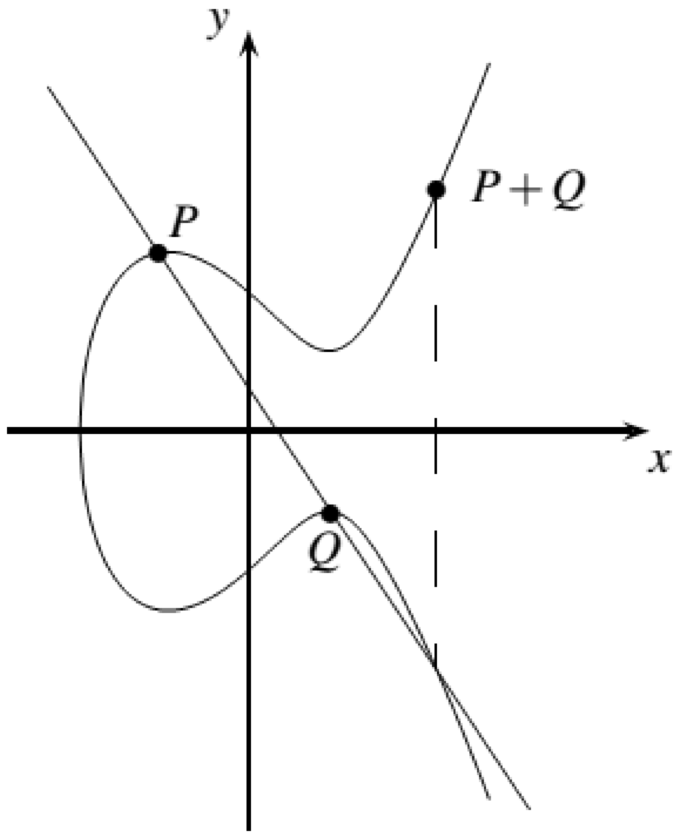
$$P + Q = R$$
$$(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$$

- **Point Addition P+ Q**
- **Point Doubling P+ P**

$$P_1 \oplus P_2 = \begin{cases} \mathcal{O}_E, & \text{if } x_1 = x_2 \ \&\ y_1 = -y_2 \\ (x_3, y_3), & \text{otherwise.} \end{cases}$$

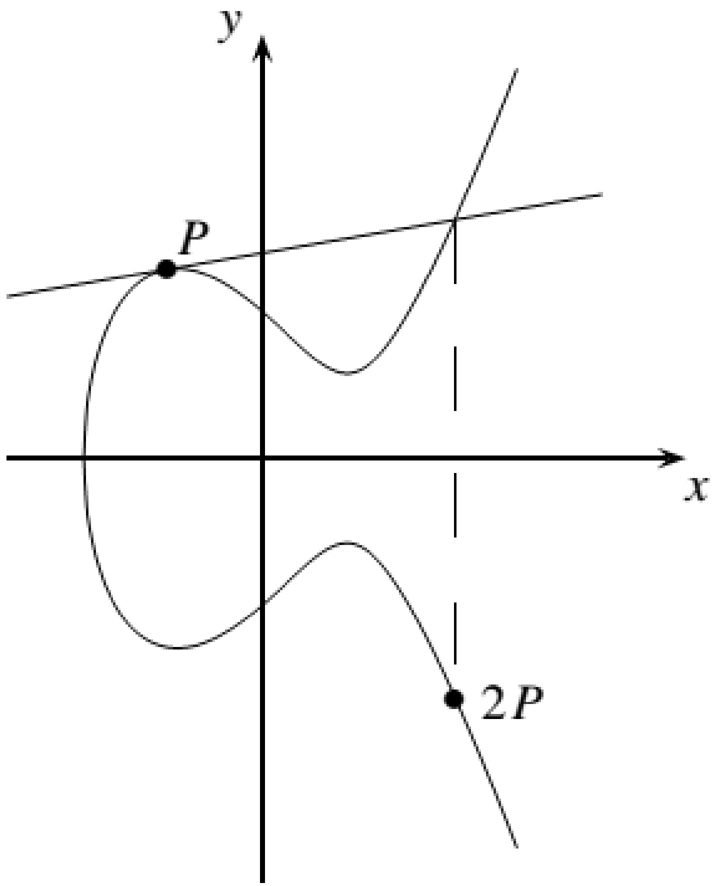
Group Operations on Elliptic Curves

- **Point Addition P+ Q**
- This is the case where we compute $R = P + Q$ and $P \neq Q$.
- The construction works as follows:
  - Draw a line through *P* and *Q* and obtain a third point of intersection between the elliptic curve and the line.
  - Mirror this third intersection point along the *x*-axis.
  - This mirrored point is, by definition, the point *R*.
  - Figure shows the point addition on an elliptic curve over the real numbers.



Group Operations on Elliptic Curves

- **Point Doubling P+ P**
- This is the case where we compute $P + Q$ but $P = Q$.
- Hence, we can write $R = P + P = 2P$.
- We need a slightly different construction here.
- We mirror the point of the second intersection along the x-axis.
- This mirrored point is the result R of the doubling.
- Figure shows the doubling of a point on an elliptic curve over the real numbers.



Group Operations on Elliptic Curves

Elliptic Curve Point Addition and Point Doubling

$$x_3 = s^2 - x_1 - x_2 \bmod p$$
$$y_3 = s(x_1 - x_3) - y_1 \bmod p$$

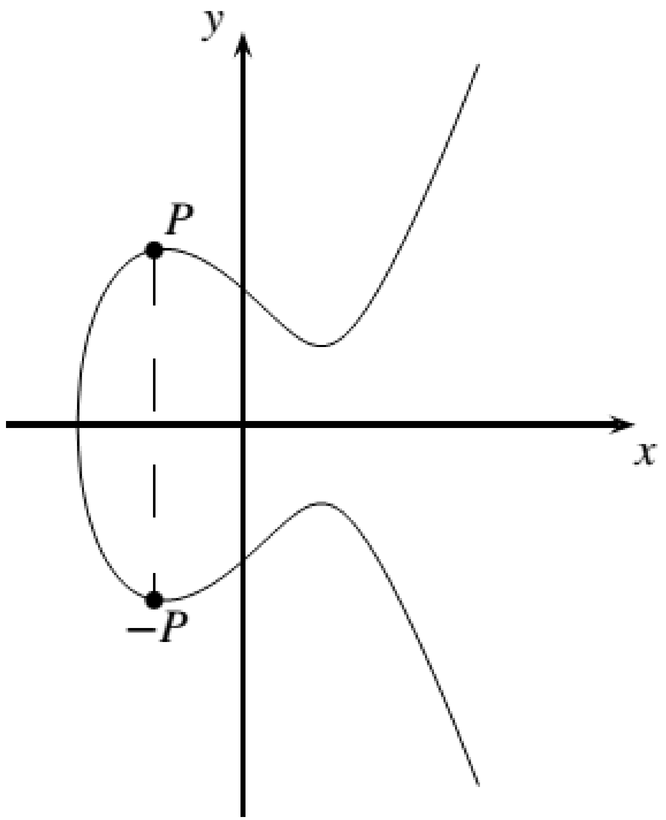
where

$$s = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} \bmod p \text{ ; if } P \neq Q \text{ (point addition)} \\ \frac{3x_1^2 + a}{2y_1} \bmod p \text{ ; if } P = Q \text{ (point doubling)} \end{cases}$$

s is the slope of the line

Group Operations on Elliptic Curves

- Given P, how do we find $-P$?
- If we apply the tangent-and-chord method from above, it turns out that the inverse of the point $P = (x_p, y_p)$ is the point $-P = (x_p, -y_p)$, i.e., the point that is reflected along the x-axis.
- We simply take the negative of its y coordinate.
- In the case of elliptic curves over a prime field $GF(p)$, this is easily achieved since $-y_p \equiv p - y_p \bmod p$, hence, $-P = (x_p, p - y_p)$.



Group Operations on Elliptic Curves

- Example: We consider a curve over the small field Z_{17}:

$$E: y^2 \equiv x^3 + 2x + 2 \bmod 17.$$

$$y^2 \equiv x^3 + 2 \cdot x + 2 \bmod 17$$
$$3^2 \equiv 6^3 + 2 \cdot 6 + 2 \bmod 17$$
$$9 = 230 \equiv 9 \bmod 17$$

We want to double the point $P = (5,1)$.

$$2P = P + P = (5,1) + (5,1) = (x_3, y_3)$$
$$s = \frac{3x_1^2 + a}{2y_1} = (2 \cdot 1)^{-1}(3 \cdot 5^2 + 2) = 2^{-1} \cdot 9 \equiv 9 \cdot 9 \equiv 13 \bmod 17$$
$$x_3 = s^2 - x_1 - x_2 = 13^2 - 5 - 5 = 159 \equiv 6 \bmod 17$$
$$y_3 = s(x_1 - x_3) - y_1 = 13(5 - 6) - 1 = -14 \equiv 3 \bmod 17$$
$$2P = (5,1) + (5,1) = (6,3)$$

Multiplying a Point by a Constant

- In arithmetic, multiplying a number by a constant k means adding the number to itself k times.
- The situation here is the same. Multiplying a point P on an elliptic curve by a constant k means adding the point P to itself k times.
- For example, in E_{13} (1, 1),
    - if the point (1, 4) is multiplied by 4, the result is the point (5, 1).
    - If the point (8, 1) is multiplied by 3, the result is the point (10, 7).

Discrete Logarithm Problem with Elliptic Curves

Hasse's theorem

Given an elliptic curve E modulo p, the number of points on the curve is denoted by $\#E$ and is bounded by:

$$p+1-2\sqrt{p} \leq \#E \leq p+1+2\sqrt{p}.$$

Elliptic Curved Discrete Logarithm Problem (ECDLP)

Given is an elliptic curve E. We consider a primitive element P and another element T. The DL problem is finding the integer d, where $1 \leq d \leq \#E$, such that:

$$\underbrace{P+P+\cdots+P}_{d\ \ times} = dP = T.$$

Elliptic Curve Cryptography Simulating ElGamal

- Generating Public and Private Keys

  1. Bob chooses E(a, b) with an elliptic curve over GF(p) or GF(2^n). (#Tutorial)

  2. Bob chooses a point on the curve, $e_1(x_1, y_1)$.

  3. Bob chooses an integer d.

  4. Bob calculates $e_2(x_2, y_2) = d \times e_1(x_1, y_1)$. Note that multiplication here means multiple addition of points as defined before.

  5. Bob announces E(a, b), $e_1(x_1, y_1)$, and $e_2(x_2, y_2)$ as his public key; he keeps d as his private key.

Elliptic Curve Cryptography Simulating ElGamal

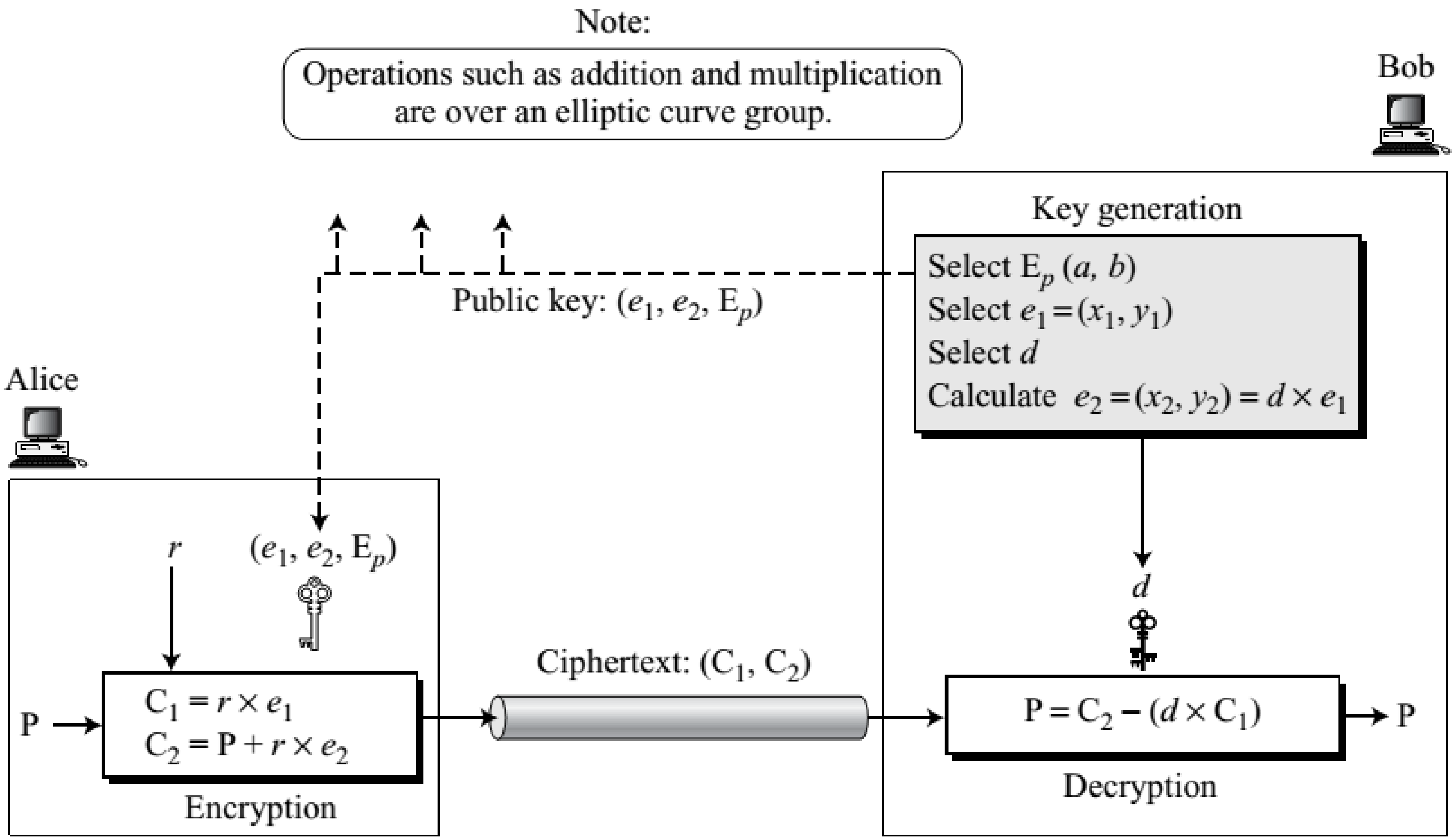
- Encryption
  - Alice selects P, a point on the curve, as her plaintext, P.
  - She then calculates a pair of points on the text as ciphertexts:

$$C_1 = r \times e_1 \qquad C_2 = P + r \times e_2$$

- Decryption
  - Bob, after receiving C_1 and C_2, calculates P, the plaintext using the following formula.
  - $P = C_2 - (d \times C_1)$ The minus sign here means adding with the inverse.

$$\mathrm{P} + r \times e_2 - (d \times r \times e_1) = \mathrm{P} + (r \times d \times e_1) - (r \times d \times e_1) = \mathrm{P} + \boldsymbol{O} = \mathrm{P}$$

Elliptic Curve Cryptography Simulating ElGamal



Elliptic Curve Cryptography Simulating ElGamal

- Example

Here is a very trivial example of encipherment using an elliptic curve over **GF**(p).

1. Bob selects $E_{67}(2, 3)$ as the elliptic curve over **GF**(p).
2. Bob selects $e_1 = (2, 22)$ and $d = 4$.
3. Bob calculates $e_2 = (13, 45)$, where $e_2 = d \times e_1$.
4. Bob publicly announces the tuple (E, e_1, e_2).
5. Alice wants to send the plaintext $P = (24, 26)$ to Bob. She selects $r = 2$.
6. Alice finds the point $C_1 = (35, 1)$, where $C_1 = r \times e_1$.
7. Alice finds the point $C_2 = (21, 44)$, where $C_2 = P + r \times e_2$.
8. Bob receives C_1 and C_2. He uses $2 \times C_1$ (35, 1) to get (23, 25).
9. Bob inverts the point (23, 25) to get the point (23, 42).
10. Bob adds (23, 42) with $C_2 = (21, 44)$ to get the original plaintext $P = (24, 26)$.

Security of ECC

- To decrypt the message, Eve needs to find the value of r or d.

a. If Eve knows r, she can use $P = C_2 - (r \times e_2)$ to find the point P related to the plaintext. But to find r, Eve needs to solve the equation $C_1 = r \times e_1$. This means, given two points on the curve, C_1 and e_1, Eve must find the multiplier that creates C_1 starting from e_1. This is referred to as the **elliptic curve logarithm problem,** and the only method available to solve it is the Polard rho algorithm, which is infeasible if r is large, and p in $GF(p)$ or n in $GF(2^n)$ is large.

b. If Eve knows d, she can use $P = C_2 - (d \times C_1)$ to find the point P related to the plaintext. Because $e_2 = d \times e_1$, this is the same type of problem. Eve knows the value of e_1 and e_2; she needs to find the multiplier d.

Modulus Size

- For the same level of security (computational effort), the modulus, n, can be smaller in ECC than in RSA.
- For example, ECC over the $GF(2^n)$ with n of 160 bits can provide the same level of security as RSA with n of 1024 bits.