Elliptic Curve Cryptography

Elliptic curve Addition Algorithm

Theorem:

Let *E: $y^2 = x^3 + ax + b$ is an elliptic curve and*

Let P and Q be two points on E

(a) **If** $P = 0$ then $P + Q = Q$

(b) Otherwise if $Q = 0$, then $P + Q = P$

(c) Otherwise, write $P = (x1, y1)$ and $q = (x2, y2)$

(d) If $x1 = x2$ and $y1 = - y2$, then $P + Q = 0$

(e) $P = P$, assume $P \neq 0$ and $Q \neq 0$

(f) **Otherwise define** λ

Contd. To next slide

Elliptic curve Addition Algorithm

Contd.

$\Lambda = (y2 - y1) / (x2 - x1)$ *if* $P \neq Q$

$\Lambda = (3x1^2 + a) / (2\ y1)$ *if* $P = Q$

$X3 = \lambda^2 - x1 - x2$

$Y3 = (\lambda\ (x1 - x3) - y1)$

Then $P + Q = (x3, y3)$

Elliptic curve Addition Algorithm

Proof:

Parts (a) and (b) are clear.

(d) Is the case that the line through P and Q is vertical, so $P + Q = 0$.

For (e), if $P \neq Q$ then *λ is the slope of the line through P and Q and if $P = Q$ then λ is the slope of the tangent line at P .*

In either case, L: $y = \lambda x + c$ *with* $c = y1 - \lambda x1$

Elliptic curve Addition Algorithm

Proof (contd.)

Substituting L on E

$(\lambda x + c)^2 = x^3 + ax + b$

$x^3 - \lambda^2 x^2 + (a - 2\lambda c)x + (b - c^2) = 0$

We know that this cubic equation has two root x1 and x2. If we cal thethird root as x3, then it factors as

$x^3 - \lambda^2 x^2 + (a - 2\lambda c)x + (b - c^2) = (x - x1)(x - x2)(x - x3)$

Multiply and look at the coefficient of x^2 on each side.

Elliptic curve Addition Algorithm

Proof (contd.)

The coefficient of x^2 on the right hand side is

$-x1 - x2 - x3$

Which must equal to $-\lambda^2$, the coefficient of x^2 on the left hand side.

This solves $x3 = \lambda^2 - x1 - x2$ and then y-coordinate of third intersection point of L and E

$$Y3 = \lambda\, x3 + c = \lambda\, x3 + c = \lambda\, x3 + y1 - \lambda\, x1$$

$$= -(\lambda\,(x1 - x3) - y1)$$

So the y-coordinate of (P + Q) is $(\lambda\,(x1 - x3) - y1)$

$\square$

Finite Elliptic Curves

- Elliptic curve cryptography uses curves whose variables & coefficients are finite
- have two families commonly used:
  - prime curves $E_p(a,b)$ defined over Z_p
    - use integers modulo a prime
    - best in software
  - binary curves $E_{2m}(a,b)$ defined over $GF(2^n)$
    - use polynomials with binary coefficients
    - best in hardware

Elliptic Curves over Finite Fields

- **Define Elliptic curve over Fp as**

$y^2 = x^3 + ax + b$ **with a, b ε Fp satisfying** $4a^3 + 27b^2 \neq 0$

$E(Fp) = \{(x,y), x, y \in Fp$ **satisfy** $y^2 = x^3 + ax + b\} \cup \{o\}, p \geq 3$

Example: Consider the elliptic curve

$E: y^2 = x^3 + ax + b$ **over the field F13**

Elliptic Curve on a finite field

- Consider $y^2 = x^3 + 3x + 8$ (**mod 13**)

  $x = 0 \Rightarrow y^2 = 8 \Rightarrow 8$ is not a square mod 13

  $x = 1 \Rightarrow y^2 = 12 = 1 \Rightarrow y = 1{,}5 \pmod{13}$

  $x = 1 \Rightarrow y^2 = 12 = 1 \Rightarrow y = 1{,}8 \pmod{13}$

  $5^2 = 12 \bmod 13,\ \ 8^2 = 12 \bmod 13$

  $x = 2 \Rightarrow y^2 = 22 = 9 \Rightarrow y = 2,\ 3 \pmod{13}$

  $x = 2 \Rightarrow y^2 = 22 = 9 \Rightarrow y = 2,\ 10 \pmod{13}$

  $3^2 = 9 \bmod 13,\ \ 10^2 = 9 \bmod 13$

- Then points on the elliptic curve are

  $E(F13) = \{0, (1,5),(1,8), (2,3), (2,10), (9,6), (9, 7),(12, 2),(12, 11)\}$

Elliptic curve Addition over Finite Field

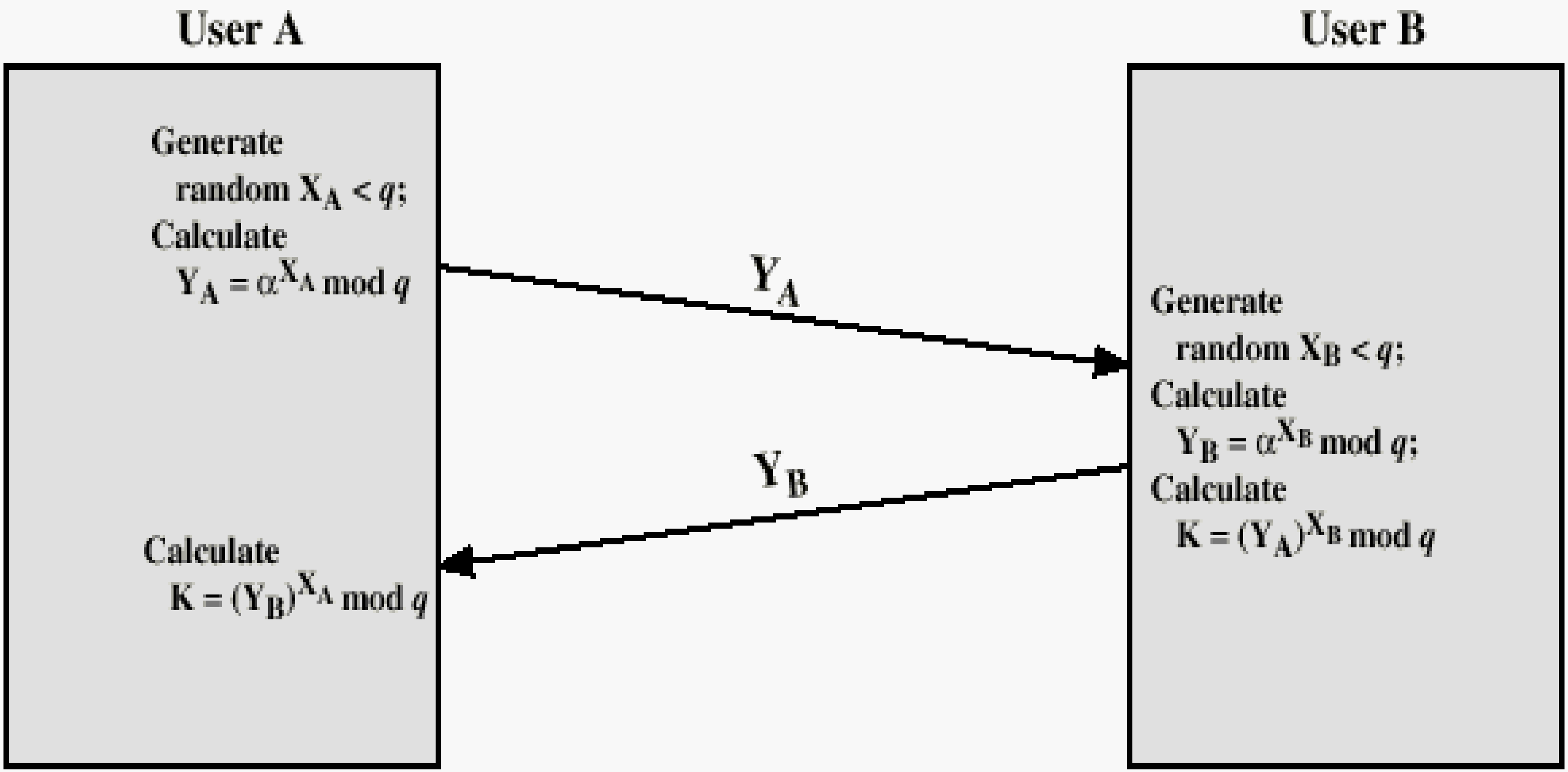
Theorem:

Let E be an elliptic curve over Fp and let P and Q be points on E(Fp)

(a) The elliptic cure addition algorithm applied P and Q yields a point in E(Fp). We denote this point by P + Q

(b) This addition law on E(Fp) satisfies all of the properties additions defined geometrically on elliptic curve i.e. E(Fp) forms a finite group.

Diffie-Hellman (DH) Key Exchange



Elliptic Curve Cryptography

- ECC addition is analog of modulo multiply
- ECC repeated addition is analog of modulo exponentiation
- need "hard" problem equiv to discrete log
  - Q=kP, where Q,P belong to a prime curve
  - is "easy" to compute Q given k,P
  - but "hard" to find k given Q,P
  - known as the elliptic curve logarithm problem
- Certicom example: $E_{23}(9,17)$

Elliptic Curve on a finite field

Example:

Consider the group E23(9, 17), E: $y^2 = x^3 + 9x + 17 \pmod{23}$

- What is discrete logarithm k of Q = (4,5) to the base P = (16,5) ?
- Brute Force : P = (16, 5); 2P = (20,20); 3P = (14, 14);
  4P = (19, 20); 5P = (13, 10); 6P = (7, 3); 7P = (8,7);
  8P = (12, 17); 9P = (4, 5);
- k= 9 , the discrete logarithm of Q(4,5) to the base P(16,5)

ECC Diffie-Hellman

- can do key exchange analogous to D-H
- users select a suitable curve $E_q(a,b)$
- 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: $P_A=n_AG$, $P_B=n_BG$
- compute shared key: $K=n_AP_B$, $K=n_BP_A$
  - same since $K=n_An_BG$
- attacker would need to find k, hard

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 $P_A = n_A G$
- to encrypt P_m : $C_m = \{kG, P_m + kP_b\}$, k random
- decrypt C_m compute:

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

ECC Security

- relies on elliptic curve logarithm problem
- fastest method is “Pollard rho method”
- compared to factoring, can use much smaller key sizes than with RSA, etc.
- for equivalent key lengths computations are roughly equivalent
- hence for similar security ECC offers significant computational advantages

Applications of ECC

- Many devices are small and have limited storage and computational power
- Where can we apply ECC?
  - **Wireless communication devices**
  - Smart cards
  - Web servers that need to handle many encryption sessions
  - **Any application where security is needed but lacks the power, storage and computational power that is necessary for our current cryptosystems**

Advantages of ECC

- Shorter key lengths
  - Encryption, Decryption and Signature Verification speed up
  - Storage and bandwidth savings

Advantage of ECC

- "**Hard problem**" analogous to discrete log
  - **Q=kP, where Q,P belong to a prime curve**

    given k,P → "easy" to compute Q

    given Q,P → "hard" to find k
  - **known as the elliptic curve logarithm problem**
    - **k must be large enough**
- ECC security relies on elliptic curve logarithm problem
  - compared to factoring, can use much smaller key sizes than with RSA etc
  - **for similar security ECC offers significant computational advantages**

Key Sizes for Equivalent Security

Symmetric scheme (key size in bits)	ECC-based scheme (size of *n* in bits)	RSA/DSA (modulus size in bits)
56	112	512
80	160	1024
112	224	2048
128	256	3072
192	384	7680
256	512	15360