CPE5021
Advanced Network Security

--- Advanced Cryptography: Elliptic Curve Cryptography ---

Lecture 3

Outline

- Principle of public key systems
  - Discrete Logarithm Problem (DLP)
- Introduction to Elliptic Curve Cryptography
  - EC with real numbers
  - EC with finite groups
- ECC in practice

Acknowledgment

- Acknowledgement:
  - Some of the figures in the lecture are borrowed from *certicom*.
  - Some diagrams are borrowed from other universities.

Some references

- http://www.certicom.com/index.php?action=res,ecc_faq (good introduction papers)
- http://cnscenter.future.co.kr/crypto/algorithm/ecc.html (more materials)
- http://www.cs.mdx.ac.uk/staffpages/m_cheng/link/ecc_simple.pdf (good introduction for students with strong maths background)

(You can find many more from the Web)

Elliptic curve cryptosystem (ECC)

Symmetric key size (in bits)	Example algorithm	DLP key size for equivalent security (*p* in bits)	RSA key size for equivalent security (*n* in bits)	ECC key size for equivalent security (*n* in bits)	Key size ratio of RSA to ECC (approx)
56	-	512	512	112	5:1
80	SKIPJACK22	1024	1024	160	6:1
112	Triple DES	2048	2048	224	9:1
128	AES-128	3072	3072	256	12:1
192	AES-192	7680	7680	384	20:1
256	AES-256	15360	15360	512	30:1

Extract from my student's Thesis – Markku N.M. Pekkarinen
Key Size Equivalence Against Best Known Attacks
(Based on López and Dahab, 2000 and Fibíková, 2002)

RSA and ECC challenges

Year	Number of decimal digits	Number of bits	MIPS Years	Calendar Time to Solution	Method (year method developed)
1994	129	429	5000	8 months, using 1600 computers	Quadratic Sieve (1984)
1995	119	395	250		
1996	130	432	750		General Number Field Sieve (1989)
1999	140	466	2000		
1999	155	512	8000	3.7 months	General Number Field Sieve (1989)

Progress in Integer Factorisation (Certicom 1997)

Discrete Logarithm Problem (DLP)

- For a group G,

Given group elements, α, β
find an integer **x** such that $\beta = \alpha^x$

x is called the ***discrete log*** of β to the base α.
- It is easy to compute β
- It is hard to find **x**, knowing α and β

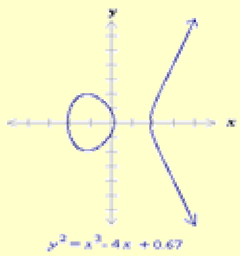
DLP - Example

- If $a^b = c$, then $\log_a c = b$
- Example:
  - $2^3 = 8 \Leftrightarrow \log_2 8 = 3$
  - $10^3 = 100 \Leftrightarrow \log_{10} 1000 = 3$
- Computing a^b and $\log_a c$ are both easy for real numbers.
- However, when working with field such as (Zp,mod), it is easy to calculate $c = a^b \bmod p$, but given *c*, *a* and *p* it is very difficult to find *b*.
- Given an integer *n* it is hard to find two integers *p*, *q* such that *n* = *p*•*q* (factorisation problem as in RSA)

Real Elliptic Curves

- An elliptic curve is defined by an equation in two variables x & y, with coefficients:
  - $y^2 + axy + by = x^3 + cx^2 + dx + e$ *(general form)*
- Consider a cubic elliptic curve of form
  - $y^2 = x^3 + ax + b$; where x,y,*a*,*b* are all real numbers. Eg.
    - $y^2 = x^3 + x + 1$.
    - $y^2 = x^3 + 2x + 6$.

Example of EC



Elliptic curve over real number

- **Let's consider the equation:**

  ***$y^2 = x^3 + ax + b$*, where x, y, a and b are real numbers, where $4a^3 + 27b^2 \neq 0$ – condition for distinct single roots (smooth curve).**
- **All (x,y) points satisfying above equation along with a infinite point *O* and addition operation (+), form a group. *O* and (+) are defined in the next slide.** *O* is the identity of the group.

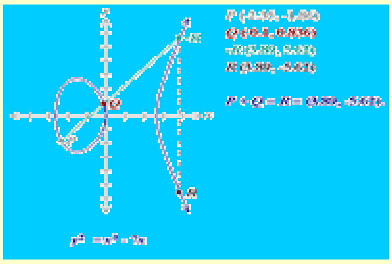
EC over a group (G,+) – E(G,+)

An EC over a group (G,+) is defined with the following:

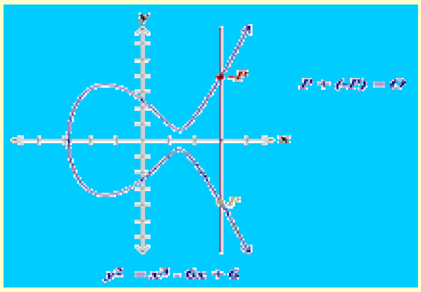
1. *Addition*: If P and Q are distinct, and P ≠ -Q, define P+Q as follows:
   - Draw a line through P and Q, then the line will intersect with the curve, the intersected point is denoted as –R, and define P+Q=R.
2. *For every P, define P + (-P) =* ***O***
3. *If P=(x,0), then P+P =* ***O***, (a vertical line)

   Otherwise, draw a **tangent line through P, the intersected point is defined as –R, then P+P =2P =R.**

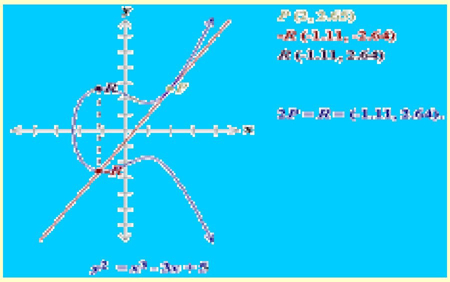
Definition of P+Q = R



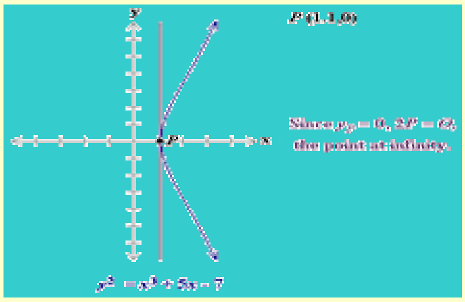
Definition of P+(-P)



Definition of P+P (where y!=0)



Definition of P+P (where y=0)



Elliptic Curve : An Algebraic Approach

1. **Adding distinct points P and Q (1)**
   When P = (x_P, y_P) and Q = (x_Q, y_Q) and P≠ Q, P ≠ -Q,
   P + Q = R(x_R, y_R) with $x_R = s^2 - x_P - x_Q$ and $y_R = s(x_P - x_R) - y_P$
   where $s = (y_P - y_Q) / (x_P - x_Q)$
2. **Doubling the point P (2)**
   When y_P is not ***O***,
   2P = R(x_R, y_R) with $x_R = s^2 - 2x_P$ and $y_R = s(x_P - x_R) - y_P$
   where $s = (3x_P^2 + a) / (2y_P)$
3. **P + (-P) = *O* (3)**
4. **If P = (x_P, y_P) and y_P =0,** then **P + P = 2P = *O* (4)**

Finite Elliptic Curves on discrete Fields

- Cryptography works with finite field and Elliptic curve cryptography uses curves whose variables and coefficients are finite
- There are two commonly used ECC families:
  - prime curves $E_p(a,b)$ defined over Z_p
    - use modulo with a prime number p
    - efficient in software
  - binary curves $E_{2^m}(a,b)$ defined over $GF(2^n)$
    - use polynomials with binary coefficients
    - efficient in hardware

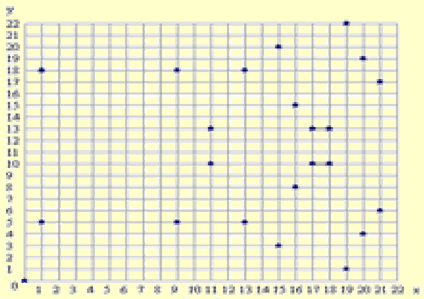
Elliptic Curve Groups over Z_p

- (Zp, mod) = {0,1,…,p-1} is a group
  - Where p is a prime number
- Define the elliptic curve
  - $y^2 = x^3 + ax + b$ mod p
  - Where a and b are in Zp, and x, y are also in Zp.
  - $(4a^3 + 27b^2 \pmod p) \neq 0$.

EC over (Zp, mod)- examples

- p=11, Zp=Z_{11}. $y^2 = x^3 + x + 6 \pmod{11}$
  - E(Z_{11}, mod) = {(2,4),(2,7), (3,5),(3,6), (5,2),(5,9), (7,2),(7,9), (8,3),(8,8), (10,2),(10,9)}
- p=23, Zp=Z_{23}. $y^2 = x^3 + x \pmod{23}$
  - E(Z_{23}, mod) = {(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)}
- p=23, Zp=Z_{23}. $y^2 = x^3 + x + 1 \pmod{23}$
  - E(Z_{23},mod) = { (0,1), (0,22), (1,7), (1,16), (3,10), (3,13), (4,0), (5,4), (5,19), (6,4), (6,19), (7,11), (7,12), (9,7), (9,16), (11,3), (11,20), (12,4), (12,19), (13,7), (13,16), (17,3), (17,20), (18,3), (18,20), (19,5), (19,18)}

$y^2 = x^3 + x$ mod 23



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

Operations on E(Z_{11},mod)

- Consider the E(Z_{11},mod):

Let P and Q on E(Z_{11},mod)

1. **P** = (10,2) **and Q=** (5,2) then **P + Q** = (10,2) + (5,2) = (7,9).
2. **P =** (2,7); **P + P** = (5,2).
3. **P =** (2,7); -P = (2,-7); P + -P =?

Elliptic Curve Cryptography (ECDLP)

- Assume that we are working with E(Z_p, mod)
  - Let Q and P be on E(Z_p, mod); and 1 < k < p-1
- Define a hard problem which is equivalent to the DLP: Q=kP
  - It is "easy" to compute Q given k and P
  - but it is "hard" to find k given Q,P
  - known as the *elliptic curve logarithm problem (ECDLP)*

ECC in Practice – simple method

- Suppose **A** wants to send a message ***m*** to **B** using EC over group (G,+) = {0, ..n-1} with generator g
  - Key generation: B selects a random integer B_s from the interval [1, n-1] as private key and publish $B_p = B_s g$ as B's public key
  - Encryption: A selects a random integer A_s as A's secrete key and send to B: ($A_s g$, $A_s B_p$ + m) to B as ciphertext ($A_s g = A_p$ is A's public key).
  - B decrypts the message by computing

$m + A_s B_p - B_s(A_s g) = m + A_s B_s g - B_s A_s g = m$

$E(Z_{11}, mod)$ with generator

- $y^2 = x3 + x + 6 \mod 11$.
  - ➔ **E(Z$_{11}$,mod) = {**(2,4),(2,7), (3,5),(3,6), (5,2),(5,9), (7,2),(7,9), (8,3),(8,8), (10,2),(10,9)**}**

$E(Z_{11}, mod)$ with generator

- **Let's select g =(2,7) as a generator.**
- **Compute 2g, 3g, … as using the following:**
  - ➔ $2P = R(x_R, y_R)$ with $x_R = s^2 - 2x_P$ and $y_R = s(x_P - x_R) - y_P$ where $s = (3x_P^2 + a) / (2y_P)$

{g=(2,7), 2g=(5,2), 3g=(8,3), 4g=(10,2)
5g=(3,6), 6g=(7,9), 7g=(7,2), 8g=(3,5)
9g=(10,9), 10g=(8,8),11g=(5,9),12g=(2,4)**}**

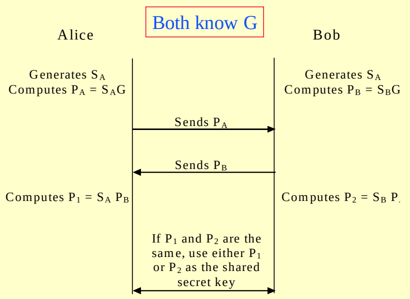
ECC example on $E(Z_{11}, mod)$ $y^2 = x3 + x + 6 \mod 11$

- **Suppose message is m =(3,6) (a point on E).**
- **B** selects a random B_s **=3,** then send B_p **to A;**
- $B_p = B_s g = 3g = g+g+g = (8,3)$; where g =(2,7)
- **A** selects a random number and generates **A's** public key, let's say 2. $A_s = 2$; $A_p = A_s g = 2g = g+g = (5,2)$.
  - ➔ **A encrypts the message:** $(A_s g, A_s B_p + m)$ = ((5,2), 2(8,3) + (3,6)) = ((5,2), (7,9) + (3,6)) = ((5,2), (5,9)).
- **B** decrypts the cipher by first computing $A_s g B_s$
  - ➔ $A_s g B_s$ = 3(5,2) = (5,2) + (5,2) + (5,2) = (10,2) + (5,2) = (7,9);
  - ➔ $A_s B_p + m - A_s g B_s$ = (5,9) – (7,9) = (5,9) + (7,-9) = (5,9) + (7,2) = (3,6)

ECC system (general approach)

- **General steps to construct an EC cryptosystem**
  1. **Selects an underlying field F**
  2. **Selects a representation for the elements of F**
  3. **Implementing arithmetic operations in F**
  4. **Selecting an appropriate EC over F to form E(F)**
  5. **Implementing EC operations in group E(F)**
  6. **Choose a protocol**
  7. **Implement ECC based on the chosen protocol.**

Diffie-Hellman Key Exchange Protocol



Diffie Hellman over ECC

- **Alice chooses a random *a* and compute** a**P** ∈ **E**
- **Bob chooses a random *b* and compute** b**P** ∈ **E**
- **Alice and Bob exchange the computed values**
- **Alice, from** b**P and a can compute S =** ab**P**
- **Bob, from** a**P and b can compute S =** ab**P**

Simple implementation of ECC

- Simple steps to construct an EC cryptosystem
  1. Select an underlying field F and generate a random curve (e.g: $y^2 = x^3 + ax + b$) – store values of a and b
     (should declare data structures to store curve and point parameters prior this)
  1. Find the base point *g* (generator) as public point (Every one knows this point)
  2. Compute share secret key using Diffie Hellman over ECC
  3. Compute public keys:
     1. Alice chooses a random as her secret key A^s and computes her public key $A^p = A^s g$
     2. Bob chooses a random B^s and computes his public key $B^p = B^s g$

     (Both Alice and Bob can now compute shared key $B^s A^s g$)
  - Embed message *m* onto a point, M(x,y), of the curve using Koblitz's method (see next slide)
  - Encrypt and decrypt
    1. Alice encrypts the message M(x,y): (A^p, $A^s B^p + M$) and sends it to Bob.
    2. Bob decrypts the message by computing $A^p B^s$ and then
       $M + A^s B^p - A^p B^s = M + A^s B^s g - A^s g B^s = M$

Embedding plaintext messages as points on an Elliptic Curve

- In order to build an ECC, there must be an accurate and efficient way for embedding a ciphertext message on an EC.
  - There is no known deterministic algorithm for embedding message units as points on an elliptic curve.
  - However, there is a probabilistic method that can be used for embedding message units as points on an elliptic curve.
  - See Koblitz's proposal of representing a message unit as a point on an EC.

Embed a message *m*

- Suppose p is prime with p mod 4 = 3
- Pick k so that $1/2^k$ is small
- Let *m* be the message and allows *m* < (p-k)/k
- For j=0, …, k-1
- Set $x_j = m*k + j$; $w_j = x^3 + a\,x_j + b$; $z_j = w_j^{((p+1)/4)}$
- If ($z^2 = w_j$) then (x_j , z_j) is the point to encode *m*
- If no j works then FAIL with Prob. <= $1/2^k$
- If *m* is embedded as M(x,y) then *m* = [x/k]

Remarks

- The efficiency of any ECC depends on how efficient the EC is represented and computations on points.
- There are may classes of curves that can be efficiently implemented
- There are still may opportunities to improve the current ECCs.
- ECC implementation is more efficient with finite fields of $E(Z_p)$; where p is a prime number or $p = 2^n$.
- There are many versions that make it hard to agree with a proposed standard one
- ECC can be implemented using other protocols. (refer to papers).