Other Public-Key Cryptosystems

Raj Jain
Washington University in Saint Louis
Saint Louis, MO 63130
Jain@cse.wustl.edu

Audio/Video recordings of this lecture are available at:

http://www.cse.wustl.edu/~jain/cse571-11/

1. Diffie-Hellman Key Exchange
2. ElGamal Cryptosystem
3. Elliptic Curve Arithmetic
4. Elliptic Curve Cryptography
5. Pseudorandom Number Generation using Asymmetric Cipher

These slides are based partly on Lawrie Brown's slides supplied with William Stallings's book "Cryptography and Network Security: Principles and Practice," 5th Ed, 2011.

Diffie-Hellman Key Agreement

- Allows two party to agree on a secret key using a public channel
- A selects q=large prime, and α=a primitive root of q
- A selects a random # X_A, B selects another random # S_B

A		B
X_A, α, q		X_B
$Y_A = \alpha^{X_A} \bmod q$	α, q, Y_A →	$Y_B = \alpha^{X_B} \bmod q$
$Y_{AB} = Y_B^{X_A} \bmod q$	← Y_B	$Y_{AB} = Y_A^{X_B} \bmod q$

$$Y_{AB} = g^{X_A X_B} \bmod q$$

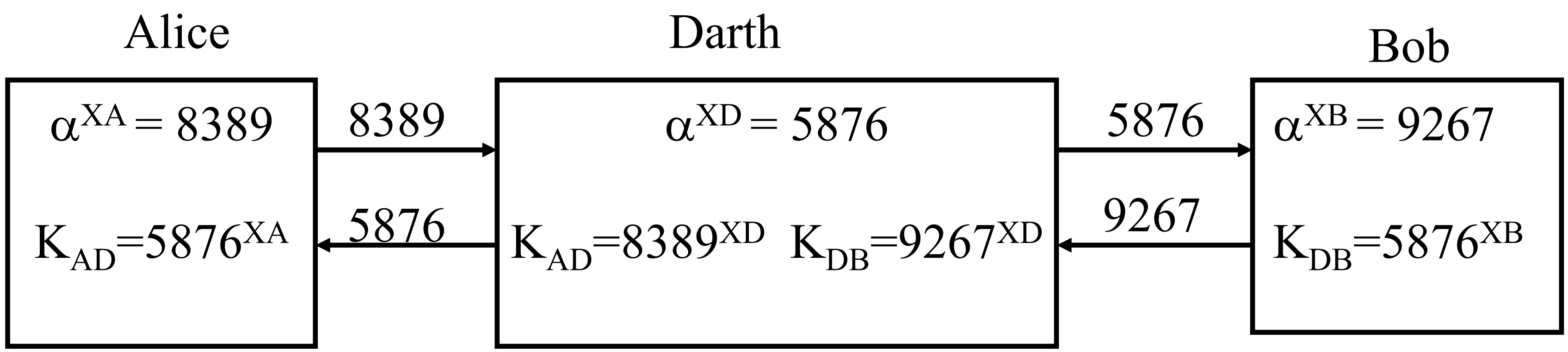
- Eavesdropper can see Y_A, α, q but cannot compute X_A
- Computing X_A requires discrete logarithm - a difficult problem

Diffie-Hellman (Cont)

- Example: α=5, q=19
  - A selects 6 and sends 5^6 mod 19 = 7
  - B selects 7 and sends 5^7 mod 19 = 16
  - A computes K = 16^6 mod 19 = 7
  - B computes K = 7^7 mod 19 = 7
- Preferably (q-1)/2 should also be a prime.
- Such primes are called safe prime.

Man-in-Middle Attack on Diffie-Hellman

- Diffie-Hellman does not provide authentication



- X can then intercept, decrypt, re-encrypt, forward all messages between Alice & Bob
- You can use RSA authentication and other alternatives

ElGamal Cryptography

- Public-key cryptosystem related to D-H
- Uses exponentiation in a finite (Galois)
- Security based difficulty of computing discrete logarithms
- X_A is the private key, $\{\alpha, q, Y_{A\}}$ is the public key

X_A, α, q

$Y_A = \alpha^{X_A} \bmod q$

$K=(C_1)^{XA}$

$M=C_2K^{-1} \bmod q$

α, q, Y_A →

← C_1, C_2

Select random key k

$K=(Y_A)^k \bmod q$

$C_1=\alpha^k \bmod q$

$C_2=KM \bmod q$

$K=\alpha^{kxa}$

- k must be unique each time. Otherwise insecure.

ElGamal Cryptography Example

- Use field GF(19) $q=19$ and $\alpha=10$
- Alice chooses $x_A=5$,
- Bob wants to sent message $M=17$, selects a random key $k=6$

Alice:

$X_A=5$, $\alpha=10$, $q=19$

$Y_A = \alpha^{XA} \bmod q$

$=10^5 \bmod 19 = 3$

$K=(C_1)^{XA}$

$=11^5 \bmod 19 = 7$

$K^{-1}= 7^{-1} = 11$

$M=C_2K^{-1} \bmod q$

$=5 \times 11 \bmod 19=17$

Alice → Bob: $\alpha=10$, $q=19$, $Y_A=3$

Bob → Alice: $C_1=11$, $C_2=5$

Bob:

Select random key $k=6$

$K=(Y_A)^k \bmod q$

$=3^6 \bmod 19 = 7$

$C_1=\alpha^k \bmod q$

$=10^6 \bmod 19 = 11$

$C_2=KM \bmod q$

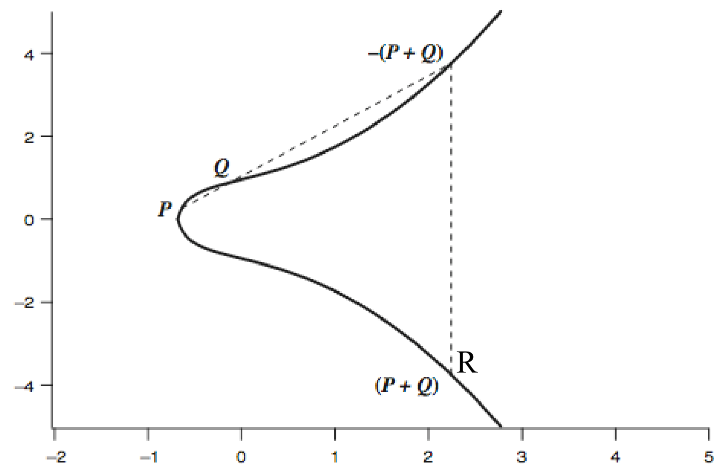
$=7\times17 \bmod 19 =5$

Elliptic Curve Cryptography

- Majority of public-key crypto (RSA, D-H) use either integer or polynomial arithmetic with very large numbers/polynomials
- Imposes a significant load in storing and processing keys and messages
- An alternative is to use elliptic curves
- Offers same security with smaller bit sizes
- Newer, but not as well analyzed

Elliptic Curves over Real Numbers

- An elliptic curve is defined by an equation in two variables x & y,
  - $y^2 = x^3 + ax + b$
  - Where x, y, a, b are all real numbers
  - $4a^3+27b^2\neq 0$
- The set of points E(a, b) forms an abelian group with respect to "addition" operation defined as follows:
  - P+Q is reflection of the intersection R
  - O (Infinity) acts as additive identity
  - To double a point P, find intersection of tangent and curve
  - Closure: P+Q ε E
  - Associativity: P+(Q+R) = (P+Q)+R
  - Identity: P+O=P
  - Inverse: -P ε E
  - Commutative: P+Q = Q+P



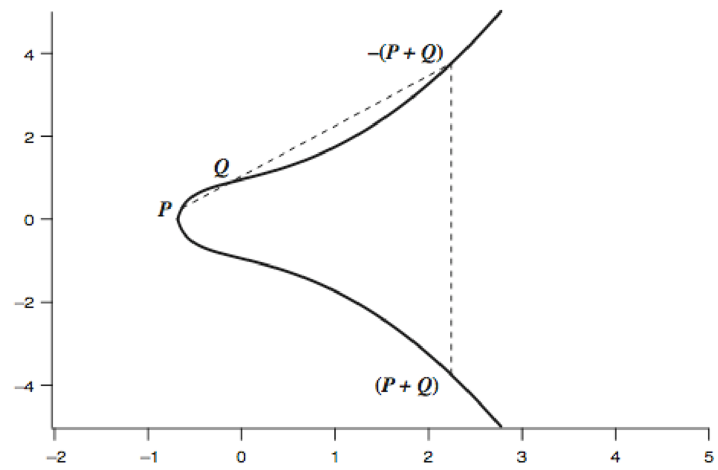
(b) $y^2 = x^3 + x + 1$

Elliptic Curve over Real Numbers (Cont)

- Slope of line PQ is:
  - D = (yQ-yP)/(xQ-xP)
- The sum R=P+Q is:
  - xR=D2-xP-xQ
  - yR=-yp+D(xP-xR)
- P+P=2P=R

$$x_R = \left(\frac{3x_P^2+a}{2y_P}\right)^2 - 2x_P$$

$$y_r = \left(\frac{3x_P^2+a}{2y_P}\right)(x_P - x_R) - y_P$$



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

Finite Elliptic Curves

- Elliptic curve cryptography uses curves whose variables & coefficients are defined over GF
  - **Prime curves**: $E_p(a,b)$ defined over Z_p
    - Use integers modulo a prime
    - Easily implemented in software
  - **Binary curves**: $E_{2^m}(a,b)$ defined over $GF(2^n)$
    - Use polynomials with binary coefficients
    - Easily implemented in hardware
- Cryptography: Addition in elliptic = multiplication in Integer
  - Repeated addition = Exponentiation
  - Easy to compute Q=P+P+…+P=kP, where Q, P ε E
  - Hard to find k given Q, P (Similar to discrete log)

Finite Elliptic Curve Example

- $E_p(a,b)$: $y^2=x^3+ax+b \bmod p$
- $E_{23}(1,1)$: $y^2=x^3+x+1 \bmod 23$
- Consider only +ve x and y
- R=P+Q
  - $x_R=(\lambda^2-x_P-x_Q) \bmod p$
  - $y_R=(\lambda(x_P-x_R)-y_P) \bmod p$
  - Where

$$\lambda = \begin{cases} \left(\frac{y_Q-y_P}{x_Q-x_P}\right) \bmod p & \text{if } P \neq Q \\ \left(\frac{3x_P^2+a}{2y_P}\right) \bmod p & \text{if } P = Q \end{cases}$$

- Example: (3,10)+(9,7)

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

$x_R=(6^2-3-3) \bmod 23 = 7$

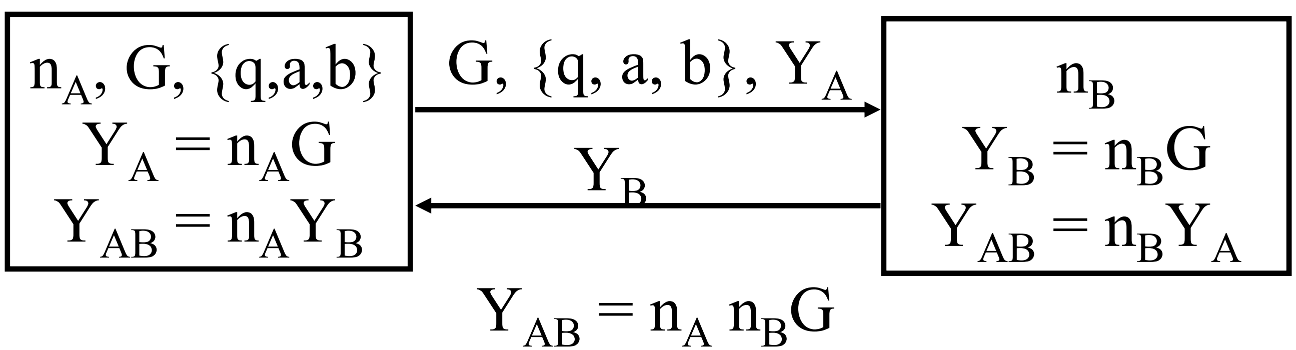
$y_R=(6(3-7)-10) \bmod 23 = 12$

Table 10.1 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)

ECC Diffie-Hellman

- 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: $Y_A=n_AG$, $Y_B=n_BG$
- Compute shared key: $K=n_AY_B$, $K=n_BY_A$
  - Same since $K=n_An_BG$
- Attacker would need to find K, hard



ECC Encryption/Decryption

- Several alternatives, will consider simplest
- Select suitable curve & point G
- Encode any message M as a point on the elliptic curve P_m
- Each user chooses private key $n_A < n$
- Computes public key $P_A = n_A G$
- 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
- 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

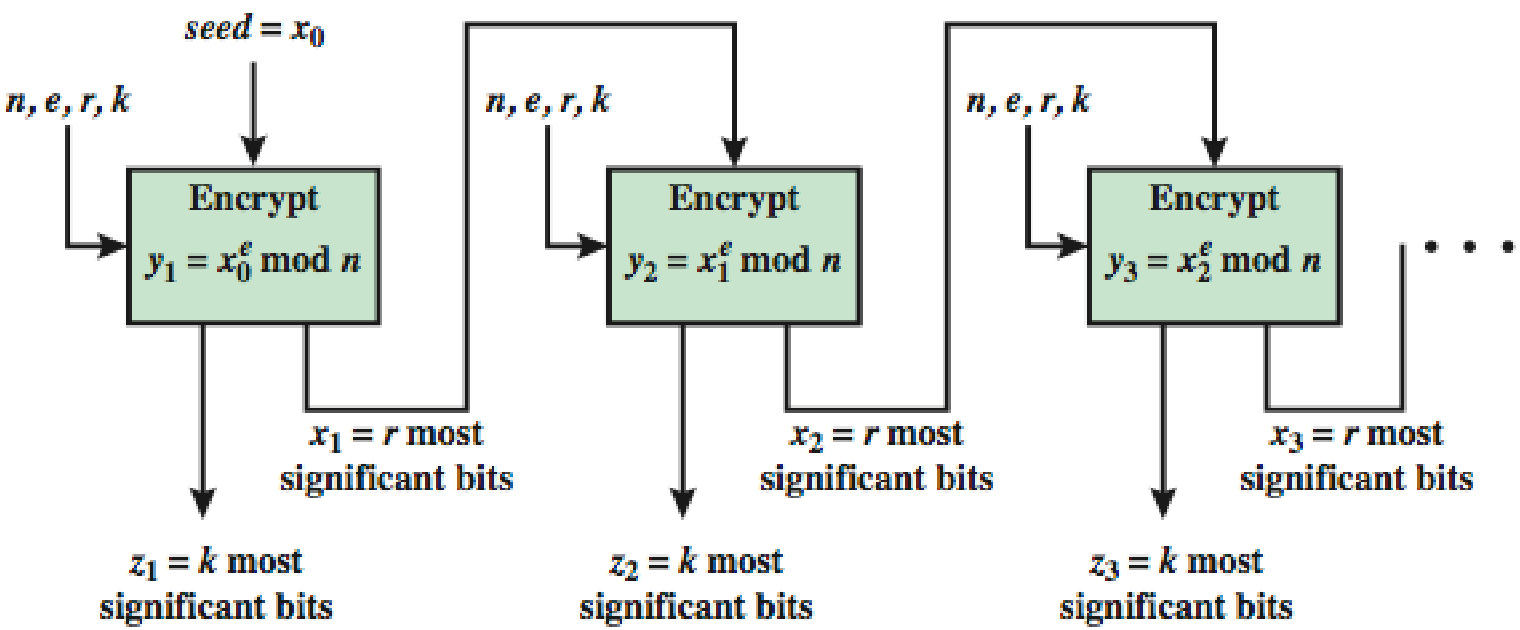
Symmetric scheme (key size in bits)	**ECC-based scheme (size of *n* in bits)**	**RSA/DSA (modulus size in bits)**
56	112	512
80	160	1024
112	224	2048
128	256	3072
192	384	7680
256	512	15360

PRNG based on Asymmetric Ciphers

- Asymmetric encryption algorithms produce apparently random output
- Hence can be used to build a pseudorandom number generator (PRNG)
- Much slower than symmetric algorithms
- Hence only use to generate a short pseudorandom bit sequence (e.g., key)

PRNG based on RSA

- Micali-Schnorr PRNG using RSA
  - in ANSI X9.82 and ISO 18031



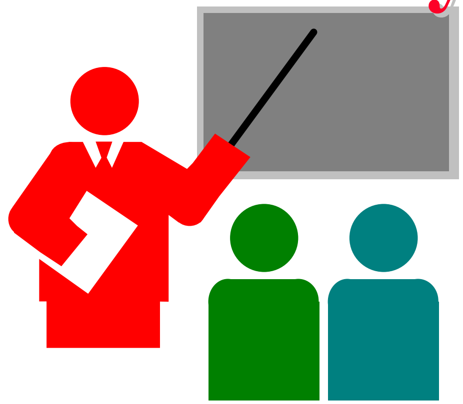
PRNG based on ECC

- Dual elliptic curve PRNG
  - NIST SP 800-9, ANSI X9.82 and ISO 18031
- Some controversy on security /inefficiency
- Notation: x(P) = x coordinate of P. $\text{lsb}_i(x)=i$ least sig bits of x
- Algorithm

```
for i = 1 to k do
set s_i = x(s_{i-1} P )
set r_i  = lsb_240 (x(s_i Q))
end for
return r_1 , . . . , r_k
```

- Only use if just have ECC

Summary



1. Diffie-Hellman key exchange allows creating a secret in public based on exponentiation
2. ElGamal cryptography uses D-H
3. Elliptic Curve cryptography is based on defining addition of points on an elliptic curve in GF(p) or GF(2^n)
4. Public key cryptography (both RSA and ECC) can also be used to generate cryptographically secure pseudorandom numbers.

Homework 10

- Submit answers to problems 10.6 and 10.15