Computer Security and Cryptography

CS381

来学嘉
计算机科学与工程系 电院3-423室
34205440 1356 4100825 laix@sjtu.edu.cn

2016-04

Organization

- Week 1 to week 16 (2016-02-24 to 2016-06-08)
- 东上院502
- Monday 3-4节; week 9-16
- Wednesday 3-4节; week 1-16
- lecture 10 + exercise 40 + random tests 40 + other 10
- Ask questions **in** class – counted as points
- Turn ON your mobile phone (after lecture)
- Slides and papers:
  - http://202.120.38.185/CS381
    - **computer-security**
  - http://202.120.38.185/references
- TA: '薛伟佳' xue_wei_jia@163.com，'黄格仕' <huang.ge.shi@foxmail.com>
- Send homework to: laix@sjtu.edu.cn and to TAs

Rule: do not disturb others!

2

Contents

3

References

- W. Stallings, *Cryptography and network security - principles and practice*，Prentice Hall.
- W. Stallings, 密码学与网络安全：原理与实践（第4版），刘玉珍等译，电子工业出版社，2006
- Lidong Chen, Guang Gong, *Communication and System Security*, CRC Press, 2012.
- A.J. Menezes, P.C. van Oorschot and S.A. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1997, ISBN: 0-8493-8523-7, http://www.cacr.math.uwaterloo.ca/hac/index.html
- B. Schneier, *Applied cryptography*. John Wiley & Sons, 1995, 2nd edition.
- 裴定一,徐祥, 信息安全数学基础, ISBN 978-7-115-15662-4, 人民邮电出版社,2007.

4

contents

- Public-key cryptosystems:
  - RSA - factorization
  - DH , ElGamal -discrete logarithm
  - ECC
- Math
  - Fermat's and Euler's Theorems & ø(n)
  - Group, Fields
  - Primality Testing
  - Chinese Remainder Theorem
  - Discrete Logarithms

Group

- a set G, and • : $G \times G \rightarrow G$ be a binary operation, satisfying
  - closed：for $a,b \in G$， $a \bullet b \in G$；
  - associative：for $a,b,c \in G$，
    $(a \bullet b) \bullet c = a \bullet (b \bullet c)$；
  - (identity) There is an element $e \in G$, such that for any $a \in G$, $e \bullet a = a \bullet e = a$
  - (Inverse) For any $a \in G$，there exists an element $b \in G$, such that, $a \bullet b = b \bullet a = e$.

  Then $(G, \bullet)$ is called to be a group.

Eample.

- (Z, +)，(Q, +)，(R, +); $(Z_m, +)$
- ? $(Z^* = Z \setminus \{0\}, \bullet)$, $(Z_P^*, \bullet)$

6

Cyclic group

- Order of an element：for $a \in G$, compute $\{a, a^2, \ldots, a^m=1\}$，the least positive integer m such that $a^m=1$ is called to be the order of *a*.
- $\{1, a, a^2, \ldots, a^{m-1}\}$ is a cyclic group with order *m. a* is called the generator of the cyclic group.
- Lemma: if the order of a is m and if $a^n=1$，then $m|n$.
- Lemma: if the order of a is m, then the order of a^k is $m/\gcd(k,m)$.
- Theorem: if the order of group *G* is *n*, then for any subgroup of G, the order of subgroup divides *n*.
- **Cyclic subgroups of (Z_7*, •)**
  - $1^0=1$ **{1}**
  - $2^0=1, 2^1=2, 2^2=4, 2^3=1$ **{1,2,4}**
  - $3^0=1, 3^1=3, 3^2=2, 3^3=6, 3^4=4, 3^5=5, 3^6=1$ **{1,3,2,6,4,5}**
  - $4^0=1, 4^1=4, 4^2=2, 4^3=1$ **{1,2,4}**
  - $5^0=1, 5^1=5, 5^2=4, 5^3=6, 5^4=2, 5^5=3, 5^6=1$ **{1,5,4,6,2,3}**
  - $6^0=1, 6^1=6, 6^2=1$ **{1,6}**

7

Field

- Let *F* be a set, and • and + are binary operations defined over *F*, satisfying
  - (*F*,+) is an Abel additive group with identity 0；
  - (*F*\{0}, •) is a multiplicative group, with identity 1；
  - Distribution law holds for multiplication and addition.

(*F*,+, •) is called to be a field.

Example: let *p* be a prime, then (Z_p,+, •) is a field, called Galois Field, denoted as $GF(P)=F_p$.

Galois,
1811～1832

8

Discrete logarithm

- For any $0<x<p$ in GF(p).
  - Given x and g, compute $y \equiv g^x \pmod{p}$ is called modular exponentiation,
  - Given g and y, to find x such that $y \equiv g^x \pmod{p}$ is called discrete logarithm, written as $x = \log_g y \pmod{p}$
- exponentiation is relatively easy, with computation complexity $O(\log_2(p))$。
- finding discrete logarithms is generally a **hard** problem

9

Diffie-Hellman Key Agreement

W.Diffie and M.E.Hellman, "New Directions in Cryptography", IEEE Transaction on Information Theory, V.IT-22.No.6, Nov 1976, PP.644-654


Parameters: p, g

Alice

Choose a
Compute $g^a \bmod p$

$g^a \bmod p$ →

← $g^b \bmod p$

Bob

Choose b
Compute $g^b \bmod p$

Compute $g^{ab} \bmod p$ (Alice)

Compute $g^{ab} \bmod p$ (Bob)

g^{ab} is the secrete key shared by Alice and Bob

10

ElGamal encryption algorithm

- Set up: GF(p), and g the primitive element.
- Users' key generation：
  - user U randomly chooses $x_U \in$GF(p)*as his private key.
  - Compute $y_U \equiv g^{x_U} (\bmod\ p)$ as his public key.
- Encryption: suppose that Alice wants to send Bob a message $m \in$GF(p). She uses Bob's public key y_B,
  - Alice randomly chooses an integer r, and compute $R = g^r$
  - Alice computes $S = m \bullet y_b^r (\bmod\ p)$;
- Alice sends (R,S) to Bob
- Decryption: Bob uses his own private key to decrypt m from (R,S) : $m = S/R^{x_b} = (m \bullet y_b^r)/(g^r)^{x_b}$

11

ElGamal encryption algorithm

Alice sends Bob a message $m \in$ GF(p). Using Bob's public key

Parameters: p, g

Alice $SK_A=(x_A)$
$PK_A=(y_A) = (g^{xA} \bmod p)$

Bob $SK_B=(x_B)$
$PK_B=(y_B) = (g^{xB} \bmod p)$

Get PK_B,
Compute $R=g^r \bmod p$
Compute $S=m\, y_B^r \bmod p$

$\xrightarrow{(R,S)}$

$m = S/R^{x_b} = (m \bullet y_b^r)/(g^r)^{x_b}$

Compute $m=S/R^{xB} \bmod p$

12

ElGamal Signature Algorithm

- Parameters are chosen as in encryption algorithm.
  - Alice's private key is x_a, and public key is $y_a=g^{x_a}$
  - Bob's private key is x_b, and public key is $y_b=g^{x_b}$
- Signing
  - Alice randomly chooses an integer r, that gcd(r, p–1)=1, and gets $R=g^r$
  - Alice uses her own private key x_a to compute

$$S=r^{-1}(m-x_aR) \pmod{p-1}$$

- Alice sends (m, R, S) to Bob
- Verification
  - Bob verifies $g^m=y_a^R R^S \pmod{p}$

2016/4/12

13

ElGamal Signature Algorithm

Parameters: p, g

Alice $SK_A=x_A$
$PK_A=y_A=(g^{xA} \bmod p)$

Bob $SK_B=x_B$
$PK_B=y_B=(g^{xB} \bmod p)$

Choose r, ***such that*** **gcd(r, p-1)=1**
Compute $R=g^r \bmod p$
Compute $S=r^{-1}(m - x_A R) \bmod p-1$

$\xrightarrow{(m, R, S)}$

Verify $g^m=y_A^R R^S \bmod p$

14

Complexity of Dlog

- Similar to factoring large number n, for discrete logarithm, the complexity of currently known algorithms is about

  exp($b^{1/3}$ $\log^{2/3}$(b))   b=log(p)   (number field sieve)

- The size of p should be at least 1024-bit

- Use strong prime: p-1 has large factors.

15

Attack when p-1 consists of small primes

- Suppose p-1=2^n， g is a generator of Z_p*
- Given $C=g^x$ mod p, to compute x=?
  - Let $x=2^{n-1}x_{n-1}+\ldots+2x_1+x_0$
  - If $C^{2^{n-1}}$=1, then x_0 =0; if $C^{2^{n-1}}$=-1, then x_0 =1.
  - Compute $C_1=C/g^{x_0}$
  - If $C_1^{2^{n-2}}$=1, then x_1 =0, if $C_1^{2^{n-2}}$=-1, then x_1 =1.
  - Compute $C_2=C_1/g^{2x_1}$
  - …
  - If C_{n-1}=1, then x_{n-1} =0, if C_{n-1}=0 then x_{n-1} =1

[sqrt(1)=1 or -1]

2016/4/12

16

Attack when p-1 consists of small primes

- When $p-1=p_1^{e_1}p_2^{e_2}\ldots p_r^{e_r}$
- Given $C=g^x \bmod p$, compute x=?
  - Compute $x=x_1 \bmod p_1^{e_1}$
  - Compute $x=x_2 \bmod p_2^{e_2}$
  - …
  - Compute $x=x_r \bmod p_r^{e_r}$
  - From Chinese remainder theorem

    $x=x \pmod{p-1}$

2016/4/12

17

Chinese Remainder Theorem

- Find a number x that leaves
  - a remainder of 2 when divided by 3,
  - a remainder of 3 when divided by 5,
  - a remainder of 4 when divided by 7.
- If

$$x \equiv 2 \pmod 3$$
$$x \equiv 3 \pmod 5$$
$$x \equiv 4 \pmod 7$$

- x=?

18

☆ Chinese Remainder Theorem

- Let $(m_1, m_2,\ldots,m_k)$ be pairwise relatively prime positive integers. Then the system of congruence
  - $x \equiv b_1 \pmod{m_1}$
  - $x \equiv b_2 \pmod{m_2}$
  - ……
  - $x \equiv b_k \pmod{m_k}$

  has a unique solution (modulo $m_1 m_2 \ldots m_k$)
- Solution

  $M=m_1 m_2 \ldots m_k$, $M_i=M/m_i$, $M_i' = M_i^{-1} \pmod{m_i}$

  $x=b_1 M_1 M_1'+b_2 M_2 M_2'+\ldots+b_k M_k M_k'$

19

Primality Testing

- often we need to find large prime numbers
- traditionally **sieve** using **trial division**
  - i.e. divide by all numbers (primes) in turn less than the square root of the number
  - only works for small numbers
- alternatively can use statistical primality tests based on properties of primes
  - for which all primes numbers satisfy property
  - but some composite numbers, called pseudo-primes, also satisfy the property
- can use a slower deterministic primality test

Miller Rabin Algorithm

- based on Fermat's Theorem: $a^{p-1} = 1 \pmod p$
- TEST (n):
  1. Find integers k, q, $k > 0$, q odd, so that $(n-1)=2^k q$
  2. Select a random integer a, $1<a<n-1$
  3. **if** $a^q \bmod n = 1$ **then** return ("maybe prime");
  4. **for** $j = 0$ **to** $k - 1$ **do**
     5. **if** ($a^{2^j q} \bmod n = n-1$)
        then return("maybe prime")
  6. return ("composite")
- Prob(n maybe prime but not prime) < ¼
  - repeat test with different random a
  - Prob(n is prime after t tests) = $1-4^{-t}$ (0.99999 for t=10)

Primitive Roots

- from Euler's theorem have $a^{ø(n)} \bmod n=1$
- consider $a^m=1 \pmod n$, $GCD(a,n)=1$
  - must exist for $m = ø(n)$ but may be smaller
  - once powers reach m, cycle will repeat
- if smallest is $m = ø(n)$ then a is called a **primitive root**
- if p is prime, then successive powers of a "generate" the group $\bmod p$
- these are useful but relatively hard to find

Exercise 8 – PKC

1. If x=2 (mod 3) x=3 (mod 5) x=4 (mod 7), what is x?
2. Compute $\phi(24)$=#{ ? }, and $\phi(n)$ for $n=p_1^{e1}\,p_2^{e2}\,p_3^{e3}$
3. Prove: in ElGamal Signature Algorithm, the Verification test $g^m = y_a^R R^S$ (mod p) is valid.
4. ElGamal scheme uses a random integer *r* for each message,
   - A) what will happen if *r* is used twice in encryption?
   - *B)* what will happen if *r* is used twice in signature?
5. Is it possible to achieve confidentiality with DH key exchange? Is it possible to achieve authenticity with DH key exchange?

- Deadline: 1 day before next lecture

Elliptic Curves

- an elliptic curve is defined by an equation in two variables x & y, with coefficients
- consider a cubic elliptic curve of form
  - $y^2 = x^3 + ax + b$
  - where x,y,a,b are all real numbers
  - define zero point O(x,∞)
- have addition operation for elliptic curve
  - geometrically sum of Q+R is reflection of intersection R

Elliptic Curve

General form:

$$y^2 = x^3 + ax + b$$

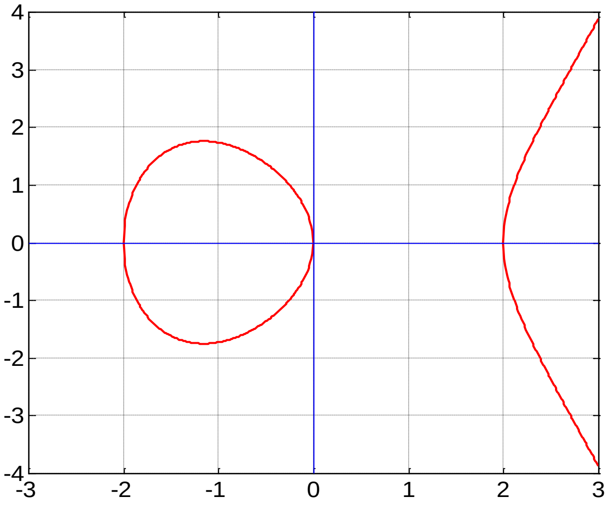
Condition for distinct single roots:

$$4a^3 + 27b^2 \neq 0$$

Example:

$y^2 = x^3 - 4x$

$= x(x-2)(x+2)$

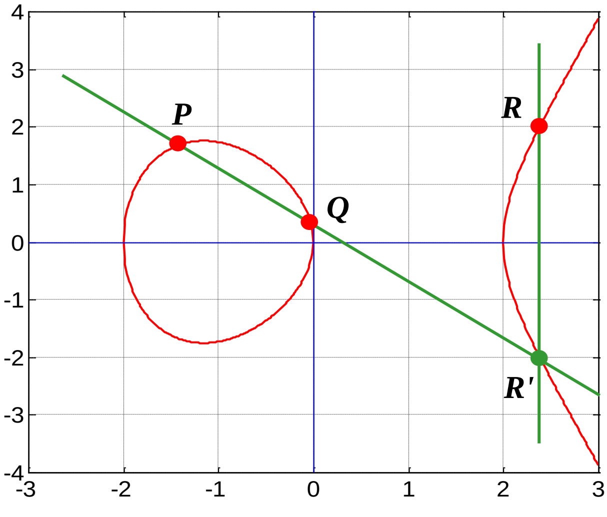


Points P(x,y) on an Elliptic Curve form a Group

Group set:
All points P(x,y) lying on an elliptic curve

Group operation:
Point addition

$$R = P + Q$$

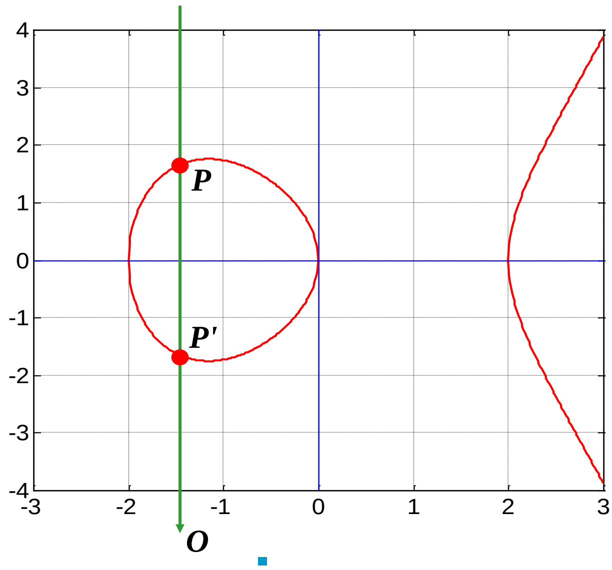


Neutral and Inverse Elements O

Inverse element:
P'(x,-y) = P(x,y)
is mirrored on x-axis

Point addition with inverse element:
P + P' = O
results in a neutral element O(x,∞) at infinity

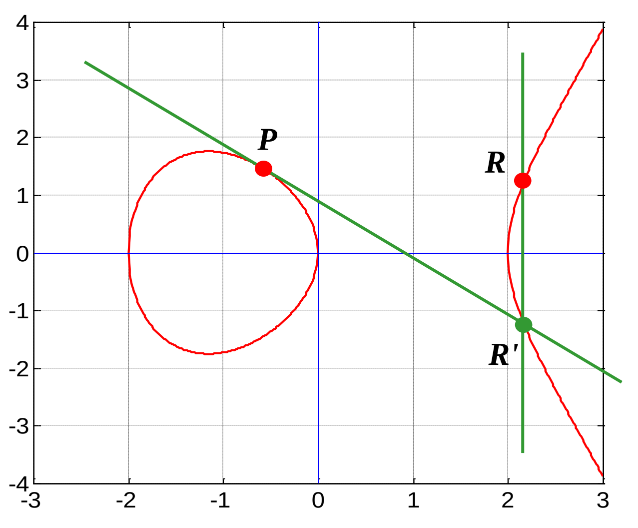
Neutral element:
P + O = P



Point Doubling – Adding a point to itself

Point Doubling:
Form the tangent in Point P(x,y)

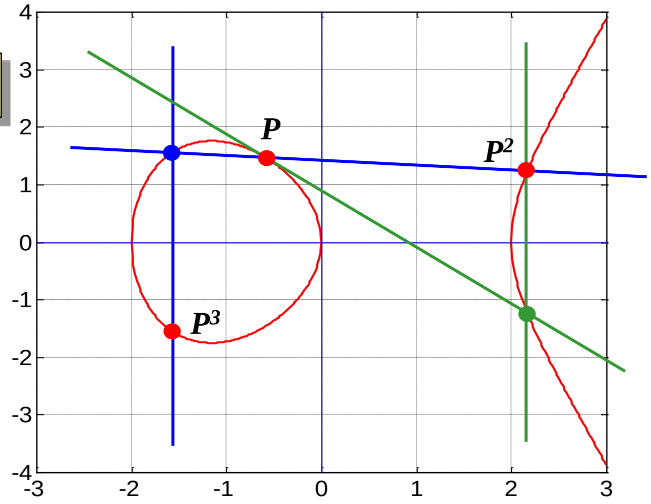
$R = P + P = 2P$



Point Iteration – Adding a point k-1 times to itself

Point Iteration:

$kP = P + P + \ldots + P$



Mathematical Description of Geometrical Transform

Line g: $y = s \cdot x + y_0$

with $s = \frac{y_Q - y_P}{x_Q - x_P}$

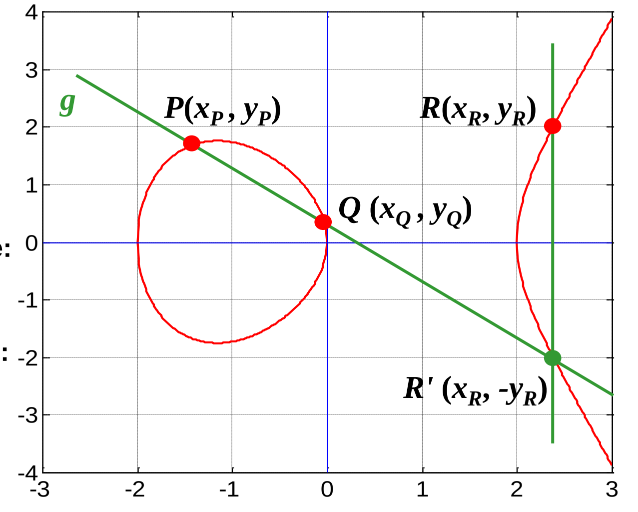
$y_0 = y_P - s \cdot x_P$

Intersection with curve:

$(s \cdot x + y_0)^2 = x^3 + ax + b$

Coordinates of point R:

$$x_R = s^2 - x_P - x_Q$$
$$y_R = -(s \cdot x_R + y_0)$$



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 Curve Discrete Logarithm Problem (ECDLP)

k	kP	s	y_0
1	(2,4)	3	9
2	(5,9)	9	8
3	(8,8)	8	10
4	(10,9)	2	0
5	(3,5)	1	2
6	(7,2)	4	7
7	(7,9)	1	2
8	(3,6)	2	0
9	(10,2)	8	10
10	(8,3)	9	8
11	(5,2)	3	9
12	(2,7)	∞	-
13	O	∞	-

Given an elliptic curve

$y^2 = x^3 + ax + b \mod p$

and a basis point P, we can compute

$Q = kP$

through k-1 iterative point additions. Fast algorithms for this task exist (double and add).

Question: Is it possible to compute k when the point Q is known?

Answer: This is a hard problem known as the Elliptic Curve Discrete Logarithm.

ECC Diffie-Hellman

- **Diffie-Hellman: Basis g and prime p**

$A = g^a \bmod p$ →

$B = g^b \bmod p$ ←

Secret: $s = A^b = B^a = g^{ab} \bmod p$

- **Elliptic Curve Cryptosystem: ECC, basis point P and prime p**

$Q_A = aP$ →

$Q_B = bP$ ←

Secret: $S = bQ_A = aQ_B = abP$

ECC Encryption/Decryption

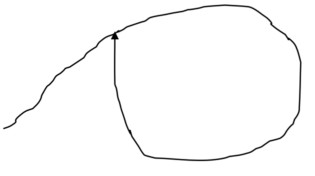
- Setup
  - encode message M as a point on the EC over P_m
  - select suitable curve & point G as in D-H
- Key generation
  - each user chooses private key $n_A < n$
  - and computes public key $P_A = n_A G$
- En/Decryption
  - encrypt P_m : $C_m = \{kG, P_m + kP_b\}$, k random
  - decrypt C_m : $P_m + kP_b - n_B(kG) = P_m + k(n_B G) - n_B(kG) = P_m$

ECC Security

- based on elliptic curve logarithm problem
- fastest method is “Pollard’s rho method”,
- compared with 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 computational advantages

Pollard’s rho method

- F: G -> G
- $F(x), F^2(x), F^3(x), \ldots$
- Exist i,j, $F^i(x)= F^j(x)$



- to find x s.t. $y \equiv g^x (\bmod\ p)$
  - $\{g^i\}$ and $\{y^j\}$ have common element , with $p^{1/2}$ computations
  - $g^i= y^j= g^{xj}$ i=xj mod p-1
- Same for EC Dlog

36

Comparable Key Sizes for Equivalent Security (NIST)

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

New: RSA should be longer, ECC can be shorter

The Digital Signature Algorithm (DSA)

Global Public-Key Components	
p	A prime number of L bits where L is a multiple of 64 and $512 \le L \le 1024$
q	A 160-bit prime factor of p-1
g	$= h^{(p-1)/q} \bmod p$, where h is any integer with $1<h<p-1$, such that $(h^{(p-1)/q} \bmod p)>1$
User's Private Key	
x	A random or pseudorandom integer with $0<x<q$
User's Public Key	
y	$= g^x \bmod p$
User's Per-Message Secret Number	
k	A random or pseudorandom integer with $0<k<q$
Signing	
	$r = (g^k \bmod p) \bmod q$ $s = [k^{-1}(H(M) = xr)] \bmod q$ Signature = (r, s)
Verifying	
	$w = (s')^{-1} \bmod q$ $u_1 = [H(M')w] \bmod q$ $u_2 = (r')w \bmod q$ $v = [(g^{u1}y^{u2}) \bmod p] \bmod q$ Test: $v = r'$

38

Summary

- Public-key cryptosystems:
  - RSA - factorization
  - DH , ElGamal -discrete logarithm
  - ECC
- Math
  - Fermat's and Euler's Theorems & ø(n)
  - Group, Fields
  - Primality Testing
  - Chinese Remainder Theorem
  - Discrete Logarithms