Asymmetric key cryptography

[Slide courtesy: Cryptography and network security by Behrouz Fourozan]

1

Introduction

- Symmetric and asymmetric-key cryptography will exist in parallel and continue to serve the community.
- They are complements of each other
  - The advantages of one can compensate for the disadvantages of the other.
- Symmetric-key cryptography is based on sharing secrecy
- Asymmetric-key cryptography is based on personal secrecy.

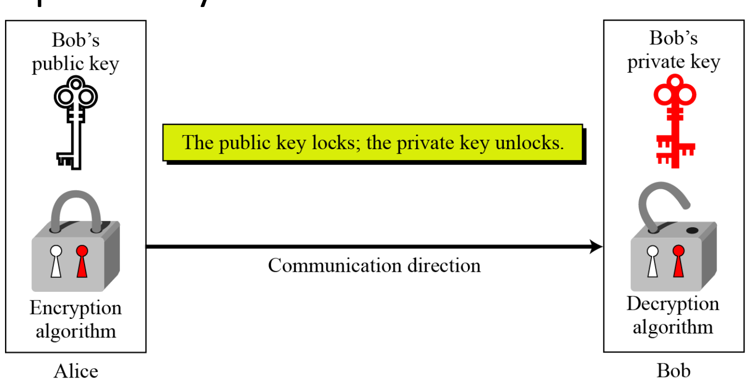
2

Need for Both

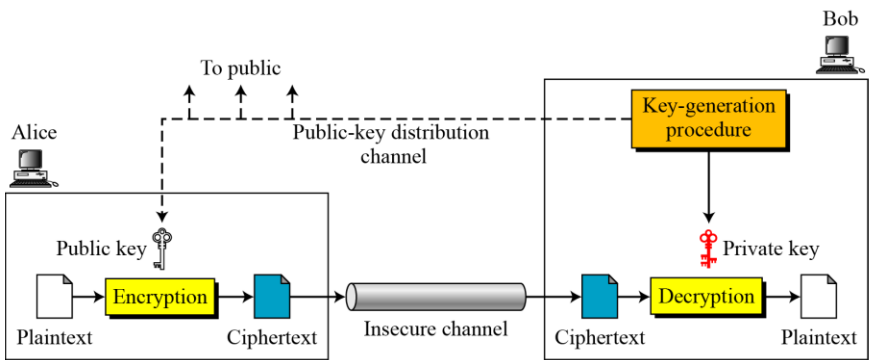
- There is a very important fact that is sometimes misunderstood
- The advent of asymmetric-key cryptography does not eliminate the need for symmetric-key cryptography.

Keys

- Asymmetric key cryptography uses two separate keys



General Idea



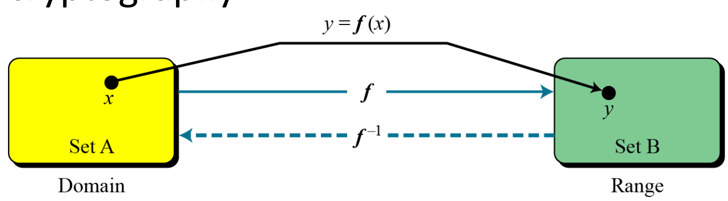
General Idea…

- Plaintext/Ciphertext
  - Unlike in symmetric-key cryptography, plaintext and ciphertext are treated as integers in asymmetric-key cryptography.

$$\boxed{C = f(K_{public}, P) \qquad P = g(K_{private}, C)}$$

Trapdoor One-Way Function

- The main idea behind asymmetric-key cryptography



Trapdoor One-Way Function…

- One-Way Function (OWF)

1. f is easy to compute.
2. f^{-1} is difficult to compute.

- Trapdoor One-Way Function (TOWF)

3. Given y and a trapdoor, x can be computed easily.

Trapdoor One-Way Function…

- Example
  - When n is large, $n = p \times q$ is a one-way function. Given p and q , it is always easy to calculate n ; given n, it is very difficult to compute p and q. This is the factorization problem.
- Example
  - When n is large, the function $y = x^k \bmod n$ is a trapdoor one-way function. Given x, k, and n, it is easy to calculate y. Given y, k, and n, it is very difficult to calculate x. This is the discrete logarithm problem. However, if we know the trapdoor, k′ such that $k \times k' = 1 \bmod \Phi(n)$, we can use $x = y^{k'} \bmod n$ to find x.

9

Merkle-Hellman Knapsack Cryptosystem

- Definition
  - $a = [a_1, a_2, \ldots, a_k]$ and $x = [x_1, x_2, \ldots, x_k]$.

$$\boxed{s = knapsackSum\ (a, x) \ = \ x_1a_1 + x_2a_2 + \cdots + \ x_ka_k}$$

  - Given a and x, it is easy to calculate s. However, given s and a it is difficult to find x.
- Superincreasing Tuple
- $$\boxed{a_i \ge a_1 + a_2 + \ldots + a_{i-1}}$$

10

Merkle-Hellman Knapsack Cryptosystem…

Algorithm 10.1 *knapsacksum and inv_knapsackSum for a superincreasing k-tuple*

knapsackSum (x [1 … k], a [1 … k])	**inv_knapsackSum** (s, a [1 … k])

{
 $s \leftarrow 0$
 for (i = 1 to k)
 {
 $s \leftarrow s + a_i \times x_i$
 }
 return s
 }
	{
	 for ($i = k$ down to 1)
	 {
	 if $s \geq a_i$
	 {
	 $x_i \leftarrow 1$
	 $s \leftarrow s - a_i$
	 }
	 else $x_i \leftarrow 0$
	 }
	 return x [1 … k]
	 }

Merkle-Hellman Knapsack Cryptosystem…

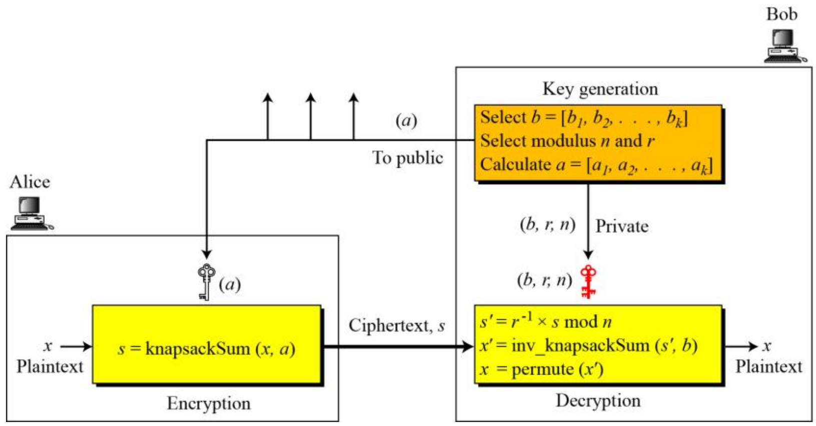
- Example
  - As a very trivial example, assume that a = [17, 25, 46, 94, 201,400] and s = 272 are given. Table 10.1 shows how the tuple x is found using inv_knapsackSum routine in Algorithm 10.1. In this case x = [0, 1, 1, 0, 1, 0], which means that 25, 46, and 201 are in the knapsack.

Table 10.1 *Values of i, a_i, s, and x_i in Example 10.3*

i	a_i	s	$s \geq a_i$	x_i	$s \leftarrow s - a_i \times x_i$
6	400	272	false	$x_6 = 0$	272
5	201	272	true	$x_5 = 1$	71
4	94	71	false	$x_4 = 0$	71
3	46	71	true	$x_3 = 1$	25
2	25	25	true	$x_2 = 1$	0
1	17	0	false	$x_1 = 0$	0

Merkle-Hellman Knapsack Cryptosystem…

- Secret Communication with Knapsacks.



Merkle-Hellman Knapsack Cryptosystem…

1. Key generation:
   a. Bob creates the superincreasing tuple $b = [7, 11, 19, 39, 79, 157, 313]$.
   b. Bob chooses the modulus $n = 900$ and $r = 37$, and [4 2 5 3 1 7 6] as permutation table.
   c. Bob now calculates the tuple $t = [259, 407, 703, 543, 223, 409, 781]$.
   d. Bob calculates the tuple a = permute $(t) = [543, 407, 223, 703, 259, 781, 409]$.
   e. Bob publicly announces a; he keeps n, r, and b secret.

2. Suppose Alice wants to send a single character "g" to Bob.
   a. She uses the 7-bit ASCII representation of "g", $(1100111)_2$, and creates the tuple $x = [1, 1, 0, 0, 1, 1, 1]$. This is the plaintext.
   b. Alice calculates $s = knapsackSum\ (a, x) = 2165$. This is the ciphertext sent to Bob.

3. Bob can decrypt the ciphertext, $s = 2165$.
   a. Bob calculates $s' = s \times r^{-1} \bmod n = 2165 \times 37^{-1} \bmod 900 = 527$.
   b. Bob calculates $x' = Inv\_knapsackSum\ (s', b) = [1, 1, 0, 1, 0, 1, 1]$.
   c. Bob calculates $x = permute\ (x') = [1, 1, 0, 0, 1, 1, 1]$. He interprets the string $(1100111)_2$ as the character "g".

Merkle-Hellman Knapsack Cryptosystem…

- Exercise

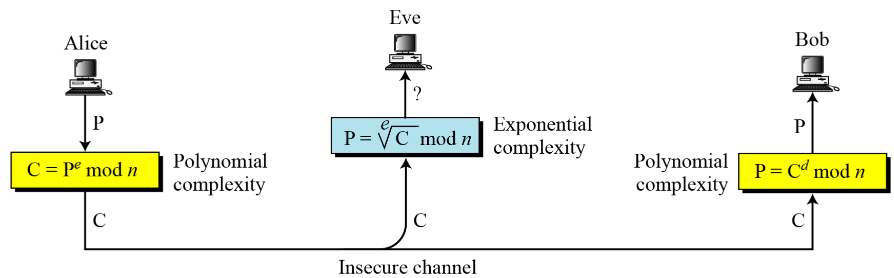
Given the superincreasing tuple b=[7,11,23,43,87,173,357], r=41 and modulus n=1001, encrypt and decrypt the letter 'a' using the Merkle-Hellman knapsack cryptosystem.

Use [7 6 5 1 2 3 4] as the permutation table.

RSA CRYPTOSYSTEM

- The most common public-key algorithm is the RSA cryptosystem, named for its inventors (Rivest, Shamir, and Adleman).

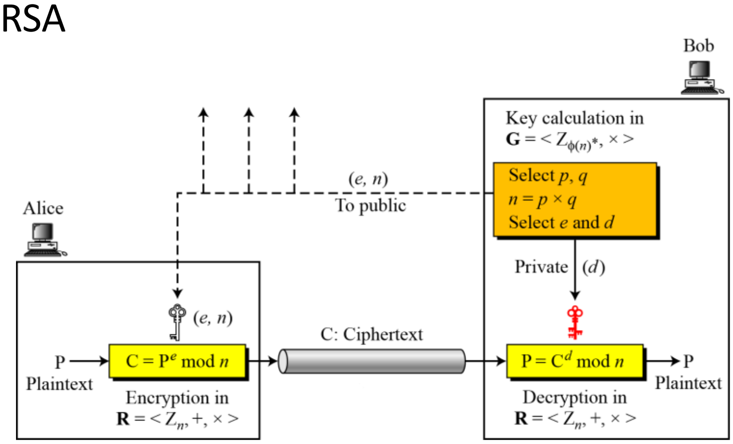
RSA CRYPTOSYSTEM…



RSA uses modular exponentiation for encryption/decryption; To attack it, Eve needs to calculate $\sqrt[e]{C} \bmod n$.

RSA CRYPTOSYSTEM…

- Encryption, decryption, and key generation in RSA



RSA CRYPTOSYSTEM…

- Two Algebraic Structures

Encryption/Decryption Ring: $R = \langle Z_n, +, \times \rangle$

Key-Generation Group: $G = \langle Z_{\phi(n)}{}^*, \times \rangle$

RSA uses two algebraic structures:
a public ring $R = \langle Z_n, +, \times \rangle$ and a private group $G = \langle Z_{\phi(n)}{}^*, \times \rangle$.

In RSA, the tuple (e, n) is the public key; the integer d is the private key.

RSA CRYPTOSYSTEM…

Algorithm 10.2 *RSA Key Generation*

```
RSA_Key_Generation
{
    Select two large primes p and q such that p ≠ q.
    n ← p × q
    φ(n) ← (p − 1) × (q − 1)
    Select e such that 1 < e < φ(n) and e is coprime to φ(n)
    d ← e^−1 mod φ(n)                    // d is inverse of e modulo φ(n)
    Public_key ← (e, n)                  // To be announced publicly
    Private_key ← d                      // To be kept secret
    return Public_key and Private_key
}
```

RSA CRYPTOSYSTEM…

Encryption

Algorithm 10.3 *RSA encryption*

```
RSA_Encryption (P, e, n)                    // P is the plaintext in Zn and P < n
{
   C ← Fast_Exponentiation (P, e, n)        // Calculation of (P^e mod n)
   return C
}
```

In RSA, p and q must be at least 512 bits; n must be at least 1024 bits.

RSA CRYPTOSYSTEM…

Decryption

Algorithm 10.4 *RSA decryption*

```
RSA_Decryption (C, d, n)                    //C is the ciphertext in Zn
{
   P ← Fast_Exponentiation (C, d, n)        // Calculation of (C^d mod n)
   return P
}
```

RSA CRYPTOSYSTEM…

Can you give a proof of RSA?

RSA CRYPTOSYSTEM…

- Proof of RSA

If $n = p \times q$, $a < n$, and k is an integer, then $a^{k \times \phi(n)+1} \equiv a \pmod{n}$.

$P_1 = C^d \bmod n = (P^e \bmod n)^d \bmod n = P^{ed} \bmod n$
$ed = k\phi(n) + 1$ // d and e are inverses modulo $\phi(n)$
$P_1 = P^{ed} \bmod n \rightarrow P_1 = P^{k\phi(n)+1} \bmod n$
$P_1 = P^{k\phi(n)+1} \bmod n = P \bmod n$ // Euler's theorem (second version)

Some Trivial Examples

- Example
  - Bob chooses 7 and 11 as *p* and *q* and calculates *n* = 77. The value of Φ(n) = (7 − 1)(11 − 1) or 60. Now he chooses two exponents, *e* and *d*, from $Z_{60}*$. If he chooses *e* to be 13, then d is 37. Note that *e* × *d* mod 60 = 1 (they are inverses of each other). Now imagine that Alice wants to send the plaintext 5 to Bob. She uses the public exponent 13 to encrypt 5.

Plaintext: 5	$C = 5^{13} = 26 \bmod 77$	Ciphertext: 26

  - Bob receives the ciphertext 26 and uses the private key 37 to decipher the ciphertext:

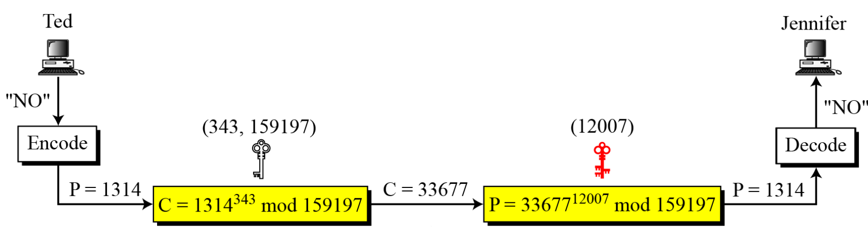
Ciphertext: 26	$P = 26^{37} = 5 \bmod 77$	Plaintext: 5

Some Trivial Examples…

Jennifer creates a pair of keys for herself. She chooses *p* = 397 and *q* = 401. She calculates *n* = 159197. She then calculates Φ(n) = 158400. She then chooses e = 343 and d = 12007. Show how Ted can send a message to Jennifer if he knows *e* and *n*.

Suppose Ted wants to send the message “NO” to Jennifer. He changes each character to a number (from 00 to 25), with each character coded as two digits. He then concatenates the two coded characters and gets a four-digit number. The plaintext is 1314.

Some Trivial Examples…



A realistic example

- A more realistic example
- We choose a 512-bit p and q, calculate n and Φ(n), then choose e and test for relative primeness with Φ(n). We then calculate d. Finally, we show the results of encryption and decryption. The integer p is a 159-digit number.

p =	96130345313583504574191581280615427909309845594996215822583150879647940455056470638491257160180347503120986660649242019180878066742 1096063354219926661209

q =	12060191957231446918276794204450896001555925054637033936061798321731482148483764659215389453209175225273226830107120695604602513887145524969000359660045617

A realistic example...

n =	1159350417396761496889250986461588752377145737545414477548552613761478854083263508172768788159683251684688493006254857641112501624145523391829271625076567727274600970827141277304349605005563472745666280600999240371029914244722922157727985317270338393813346926841373276220009666766718318310883734208234443709 53

- $\Phi(n) = (p-1)(q-1)$ has 309 digits.

$\phi(n)$ =	115935041739676149688925098646158875237714573754541447754855261376147885408326350817276878815968325168468849300625485764111250162414552339182927162507656751054233608492916752034482627988117554787657013923444405716989581728196098226361075467211864612171359107358640614008885170265377277264467341066243857664128

A realistic example...

- Bob chooses e = 35535 and tests it to make sure it is relatively prime with Φ(n). He then finds the inverse of e modulo Φ(n) and calls it d.

e =	35535
d =	580083028600377639360936612896779175946690620896509621804228661113805938528223587317062869100300217108590443384021707298690876006115306202524959884448047568240966247081485817130463240644077704833134010850947385295645071936774061197326557424237217617674620776371642076003370853332885321447088595513667029483 1

A realistic example…

- Example
  - Alice wants to send the message “THIS IS A TEST”, which can be changed to a numeric value using the 00−26 encoding scheme (26 is the space character).

P =	1907081826081826002619041819
C =	47530912364622682720636555061054518094237179607049171652323924305445296061319932856661784341835911415119741125200568297979457173603610127821884789274156609048002350719071527718591497518846588863210114835410336165789846796838676373376577746562507928052114814184404814184430812773059004692874248559166462108656

31

A realistic example…

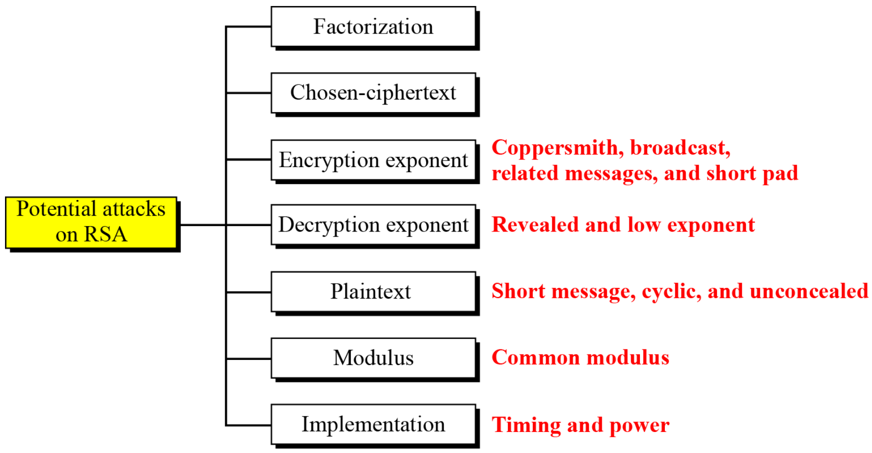
- Bob can recover the plaintext from the ciphertext using $P = C^d$, which is

P =	1907081826081826002619041819

- The recovered plaintext is “THIS IS A TEST” after decoding.

32

Attacks on RSA



33

Attacks on RSA: Chosen Ciphertext

- Eve chooses a random integer X in Zn*.
- Eve calculates $Y=C \times X^e$ modn
- Eve sends Y to Bob for decryption and get $Z=Y^d$modn
- Can you show how Eve can find P ???

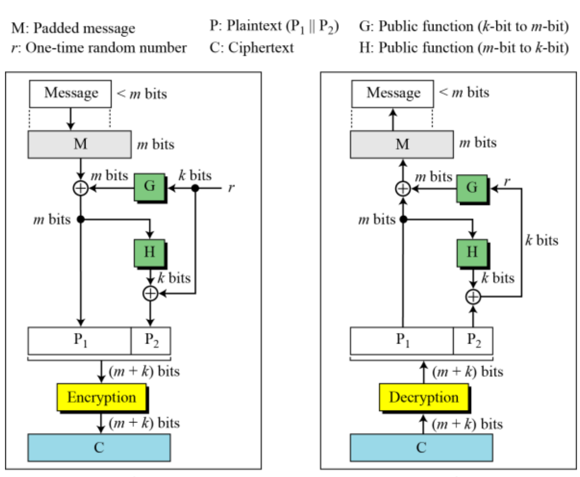
34

Attacks on RSA: Short Message Attack

- Eve knows that Alice always sends 4-digit number to Bob
  - How Eve can mount attack ?
- What could be the possible solution?

35

OAEP:Optimal Asymmetric Encryption Padding



36

Assignment questions

1. BOB chooses p=101, q=113 and therefore n=11413.

   $\phi(n)=11200=2^6 \times 5^2 \times 7$

   Can the following be candidates of e?

   a. 25

   b. 32

   Justify your answer.

37

Assignment questions…

2. Given (e,n), one would not be able to find d. Proove.
3. “If the value of d is leaked, then changing it is not suffice. One needs to change the modulus n.” Comment on the statement.

38

Assignment questions…

4. In an unpadded RSA cryptosystem, a plaintext m is encrypted as $E(m)=m^e \bmod n$, where (e,n) is the public key. Given such a ciphertext, can an adversary construct an encryption of mt for any integer t.

Now, think about the case when RSA is used with OAEP.

Assignment questions (Solution)

1. Solution: No. As both numbers are not coprime to ϕ(n)
2. Do by your own
3. Do by your own
4. Solution: $E(m)=m^e \bmod n$,

   Adversary taps the E(m). Adversary holding the public key e and an integer t of his choise, can create encryption of mt by performing,

   $E(mt)=(m^e \bmod n)\ (t^e \bmod n)$

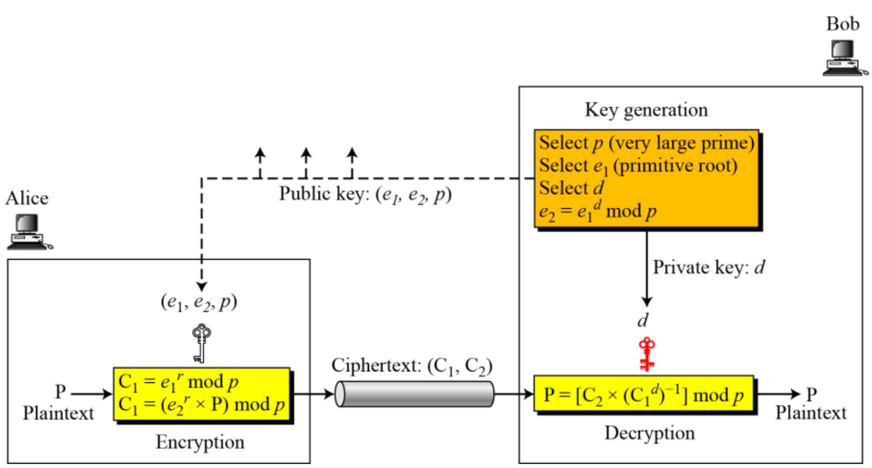
   $= (mt)^e \bmod n$

This attack is not possible if RSA is used with OAEP.

El-Gamal Cryptosystem

- Besides RSA, another public-key cryptosystem is ElGamal.
- ElGamal is based on the discrete logarithm problem

El-Gamal Cryptosystem…



El-Gamal Cryptosystem…

- Key Generation

Algorithm 10.9 *ElGamal 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
}

43

El-Gamal Cryptosystem…

- Encryption

Algorithm 10.10 *ElGamal encryption*

ElGamal_Encryption $(e_1, e_2, p, \mathrm{P})$ // P is the plaintext
{
  Select a random integer r in the group $\mathbf{G} = < \mathbf{Z}_p^*, \times >$
  $\mathrm{C}_1 \leftarrow e_1^{\,r} \bmod p$
  $\mathrm{C}_2 \leftarrow (\mathrm{P} \times e_2^{\,r}) \bmod p$ // C_1 and C_2 are the ciphertexts
  return C_1 and C_2
}

44

El-Gamal Cryptosystem…

- Decryption

Algorithm 10.11 *ElGamal decryption*

```
ElGamal_Decryption (d, p, C1, C2)                 // C1 and C2 are the ciphertexts
{
   P ← [C2 (C1^d) ^-1] mod p                      // P is the plaintext
   return P
}
```

The bit-operation complexity of encryption or decryption in ElGamal cryptosystem is polynomial.

El-Gamal Cryptosystem…

- Example
  - Here is a trivial example. Bob chooses p = 11 and e_1 = 2 and d = 3.
  - $e_2 = e_1^d$ = 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)

El-Gamal Cryptosystem…

- Example…
  - 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

El-Gamal Cryptosystem…

For the ElGamal cryptosystem, *p* must be at least 300 digits and *r* must be new for each encipherment.

- What about known-plaintext attack on El-Gamal
  - If different r is chosen each time
  - If same r is chosen