Institute of Information Technology

Jahangirnagar University

জাহাঙ্গীরনগর বিশ্ববিদ্যালয়

IT-4259: Computer Network Security

for

4th Year 2nd Semester of B.Sc (Honors) in IT (5th Batch)

Lecture: 13

RSA and Other Cryptosystems

Prepared by:

K M Akkas Ali

akkas_khan@yahoo.com, akkas@juniv.edu

Associate Professor

Institute of Information Technology (IIT)

Jahangirnagar University, Dhaka-1342

Objectives of this Lecture:

- To discuss the RSA cryptosystem
- To discuss the Rabin cryptosystem
- To discuss the ElGamal cryptosystem

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

RSA Cryptosystem

- RSA is the most commonly used public-key cryptography algorithm, which uses **prime factorization** as the **trapdoor one-way function**. That is, it is based on the presumed difficulty of factoring large integers.
- It is named so after the surnames of its inventors Ron **R**ivest, Adi **S**hamir, and Leonard **A**dleman of the Massachusetts Institute of Technology (MIT).
- It was first published in 1978.
- This algorithm relies on one way function. A one way function is easy to compute but hard to invert. For example it is easy to take the product of two prime numbers but given the product, it is difficult to split it into the original prime factors.
- **This algorithm lets you choose the size of your public key.**
- The 512-bit keys are considered insecure or weak, but the 768-bit keys are secure from everything but the National Security Administration (NSA).
- The 1024-bit keys are secure from everything virtually.
- RSA is embedded in major products such as Windows, Netscape Navigator etc.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

How the RSA Cryptosystem Works?

- Briefly, the RSA algorithm involves multiplying two large prime numbers and through additional operations deriving a set of two numbers ***e*** and ***d*** where ***e*** is the public key and ***d*** is the private key.
- Once the keys have been developed, the original prime numbers are no longer important and can be discarded. Both the public and the private keys are needed for encryption /decryption but only the owner of a private key ever needs to know it. Using the RSA system, the private key never needs to be sent across the Internet.
- Anyone can use the public key to encrypt a message. But the message can be decrypted only by the owner of the private key.
  - Thus, if Alice wants to send a message to Bob, she can find out Bob's public key (but not his private key) from a central administrator.
  - After getting the public key of Bob, Alice then encrypt the message using Bob's public key and sends the encrypted message to Bob.
  - When Bob receives it, he decrypts it with his private key.
- In addition to encrypting messages (which ensures privacy), Bob can authenticate himself to Alice (so Alice knows that it is really Bob who sent the message) by using Bob's private key to encrypt a digital certificate. When Alice receives it, she can use Bob's public key to decrypt it.

Steps in RSA Algorithm

- The RSA algorithm involves three steps:
  1. Key generation (Generating public and private key)
  2. Encryption (Encrypting the message)
  3. Decryption (Decrypting the message)
- RSA involves a public key and a private key.
  - The public key can be known by everyone and is used for encrypting messages.
  - Messages encrypted with the public key can only be decrypted in a reasonable amount of time using the private key.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

RSA Algorithm: Key Generation

The keys for the RSA algorithm are generated by the following ways:

1. Choose two large and distinct prime numbers ***p*** and ***q***.

- For security purposes, the integers ***p*** and ***q*** should be chosen at random, and should be of similar bit-length.
- In RSA, ***p*** and ***q*** must be at least 512 bits; ***n*** must be at least 1024 bits.
- Prime integers can be efficiently found using a primality test.

2. Compute ***n*** = p * q

- ***n*** is used as the modulus for both the public and private keys. Its length, usually expressed in bits, is the key length.

3. Compute the number of integers less than ***n*** that are coprime with ***n*** (otherwise known as the **totient or Euler's Phi function**):

$$\varphi(n) = \varphi(p*q) = \varphi(p)* \varphi(q)=(p - 1) * (q - 1)$$

4. Choose an integer ***e*** such that **1 < e < φ(n)** and **gcd(e, φ(n)) = 1**; i.e. ***e*** and ***φ(n)*** are coprime.

- e is released as the public key exponent (encryption exponent).
- e having a short bit-length results in more efficient encryption– most commonly $2^{16} + 1 = 65,537$. However, much smaller values of e (such as 3) have been shown to be less secure in some settings.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

RSA Algorithm: Key Generation

5. Determine the multiplicative inverse ***d*** of ***e***; i.e., compute a value for ***d*** such that it satisfies the relation: (***d*** * ***e***) mod φ(n) = 1

- ***d*** is kept as the private key exponent (decryption exponent).
- ***d*** is often computed using the extended Euclidean algorithm.
- ***d*** must be kept secret.
- ***p***, ***q***, and ***φ(n)*** must also be kept secret because they can be used to calculate ***d***.

6. The public key consists of the modulus ***n*** and the public key exponent ***e***; i.e., the public key is **(e, n).**

7. The private key consists of the modulus ***n*** and the private key exponent ***d***; i.e., the private key is **(d, n).**

8. To encrypt message ***m*** using the public key, use the relation:

$$\mathbf{c} = m^{\mathbf{e}} \bmod \mathbf{n}$$

9. To decrypt ***c*** using the private key, use the relation:

$$m = c^{\mathbf{d}} \bmod \mathbf{n}$$

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

RSA Algorithm: Encryption

- Bob transmits his public key (***e***, ***n***) to Alice and keeps the private key (***d***, ***n***) secret.
- Alice then wishes to send message ***M*** to Bob.
- The message is encrypted by the following ways:

1. Alice first turns message ***M*** into an integer ***m***, such that $0 \leq m < n$.

- That is, the message is represented as an integer between 0 and (n-1).
- Large messages can be broken up into a number of blocks. Each block would then be represented by an integer in the same range.

2. After turning the message into integer, Alice then computes the ciphertext ***c*** using the following relation:

$$c = m^e \bmod n$$

3. After computing ciphertext, Alice then transmits ***c*** to Bob.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

RSA Algorithm: Decryption

- Bob can recover ***m*** from ***c*** by using his private key exponent ***d*** using the following relation:

$$m = c^d \bmod n$$

- After having ***m***, Bob can recover the original message ***M*** by reversing the padding scheme.
- The encryption, decryption and key generation in RSA is shown in the figure below.

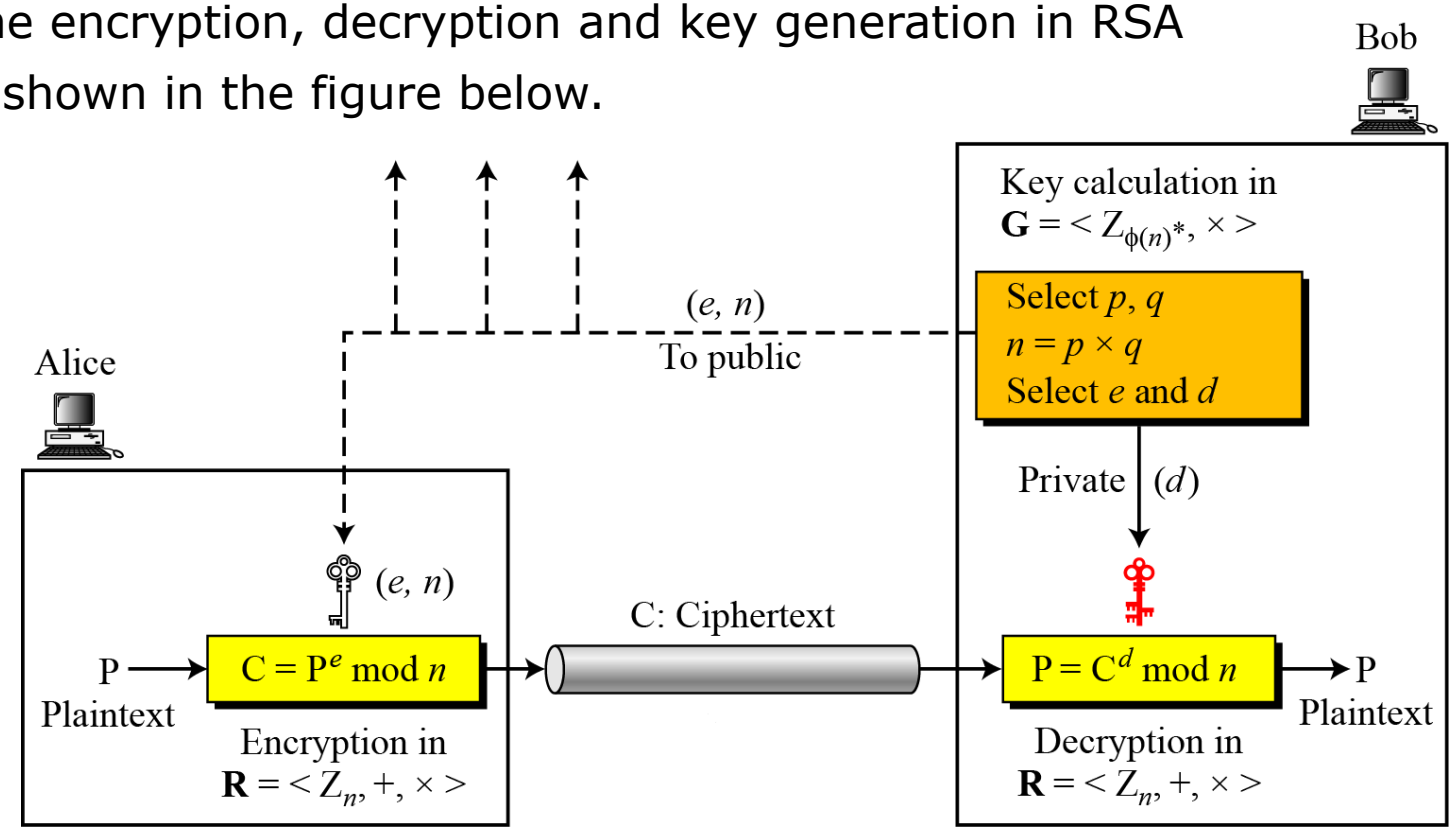


Figure: Encryption, decryption, and key generation in RSA

RSA Cryptosystem: Trivial Examples

Example-1:

1. Choose p = 3 and q = 11
2. Compute n = p * q = 3 * 11 = 33
3. Compute φ(n) = φ(p*q) = φ(p)* φ(q)=(p - 1) * (q - 1) = 2 * 10 = 20
4. Choose e such that 1 < e < φ(n) and e and φ(n) are coprime. We have several choices for e: 7, 11, 13, 17, 19. (We cannot use 5 as e, because 20 is divisible by 5). Let e = 7
5. Compute a value for d such that (d * e) mod φ(n) = 1. One solution is d = 3 [(3 * 7) % 20 = 1] [d is the multiplicative inverse of e]
6. Public key is (e, n) => (7, 33)
7. Private key is (d, n) => (3, 33)
8. The encryption of m = 2 is $c = m^e \bmod n = 2^7 \bmod 33 = 29$
9. The decryption of c = 29 is $m = c^d \bmod n = 29^3 \bmod 33 = 2$

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

RSA Cryptosystem: Trivial Examples

Example-2:

- Bob chooses 7 and 11 as p and q.
- He 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).
- 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

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

RSA Cryptosystem: Trivial Examples

Example-3:

- Now assume that another person, John, wants to send a message to Bob.
- John can use the same public key announced by Bob (probably on his website), 13.
- John's plaintext is 63. John calculates the following:

Plaintext: 63	$C = 63^{13} = 28 \bmod 77$	Ciphertext: 28

- Bob receives the ciphertext 28 and uses his private key 37 to decipher the ciphertext:

Ciphertext: 28	$P = 28^{37} = 63 \bmod 77$	Plaintext: 63

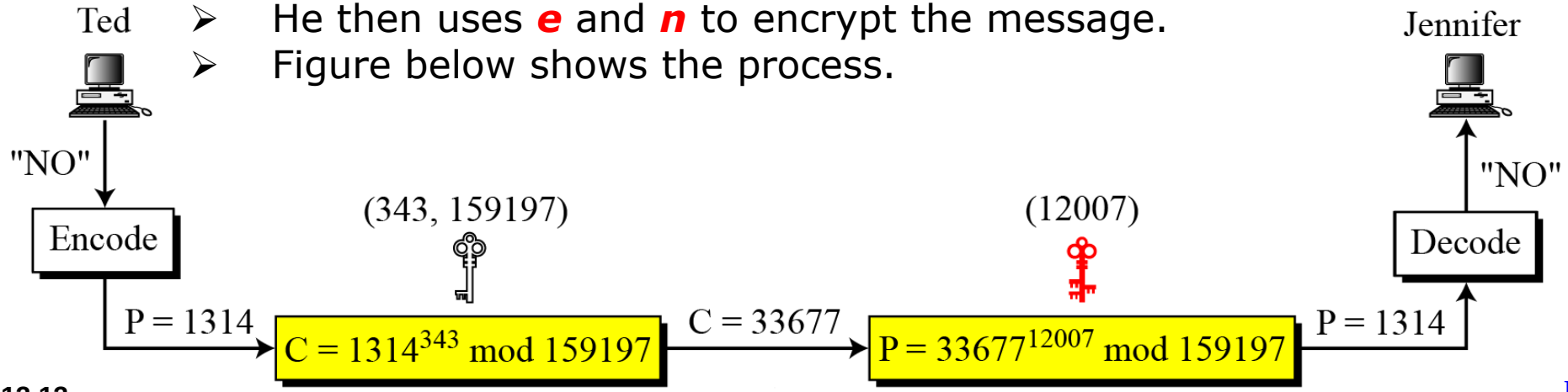
RSA Cryptosystem: Trivial Examples

Example-4:

- 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**.

Solution:

- 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.
- He then uses ***e*** and ***n*** to encrypt the message.
- Figure below shows the process.



Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

RSA Cryptosystem: Realistic Examples

Example-5:

- Here is a more realistic example.
- We choose a 512-bit ***p*** and ***q***, calculate ***n*** and **φ(n).**
- We 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.

$p =$	96130345313583504574191581280615427909309845594996215822583150879647940455056470638491257160180347503120986666064924201918087806674210960633542199266612 09

$q =$	12060191957231446918276794204450896001555925054637033936061798321731482148483764659215389453209175225273226830107120695604602513887145524969000359660045617

...li, Associate Professor, IIT, JU

RSA Cryptosystem: Realistic Examples

Example-5 (Continued…):

- The modulus **n** = p × q. It has 309 digits.

$$n = 115935041739676149688925098646158875237714573754541447754855261376$$
$$147885408326350817276878815968325168468849300625485764111250162414$$
$$552339182927162507656772727460097082714127730434960500556347274566$$
$$628060099924037102991424472292215772798531727033839381334692684137$$
$$32762200096667667183183108837342082344437095 3$$

- **φ(n)** = (p − 1)(q − 1) has 309 digits.

$$\phi(n) = 115935041739676149688925098646158875237714573754541447754855261376$$
$$147885408326350817276878815968325168468849300625485764111250162414$$
$$552339182927162507656751054233608492916752034482627988117554787657$$
$$013923444405716989581728196098226361075467211864612171359107358640$$
$$614008885170265377277264467341066243857664128$$

RSA Cryptosystem: Realistic Examples

Example-5 (Continued…):

- Bob chooses ***e*** = 35535 (the ideal is 65537) 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
$d =$	5800830286003776393609366128967791759466906208965096218042286611138059385282235873170628691003002171085904433840217072986908760061153062025249598844480475682409662470814858171304632406440777048331340108509473852956450719367740611973265574242372176176746207763716420760033708533328853214470885955136670294831

RSA Cryptosystem: Realistic Examples

Example-5 (Continued…):

- 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

- The ciphertext calculated by Alice is $C = P^e$, which is

C =	475309123646226827206365550610545180942371796070491716523239243054 452960613199328566617843418359114151197411252005682979794571736036 101278218847892741566090480023507190715277185914975188465888632101 148354103361657898467968386763733765777465625079280521148141844048 141844308127730590046928742485591664621086 56

RSA Cryptosystem: Realistic Examples

Example-5 (Continued…):

- 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.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

Attacks on RSA Cryptosystem

- No devastating attacks on RSA have been yet discovered.
- Several attacks have been predicted based on the weak plaintext, weak parameter selection, or inappropriate implementation.
- Figure below shows the category of potential attacks on RSA.

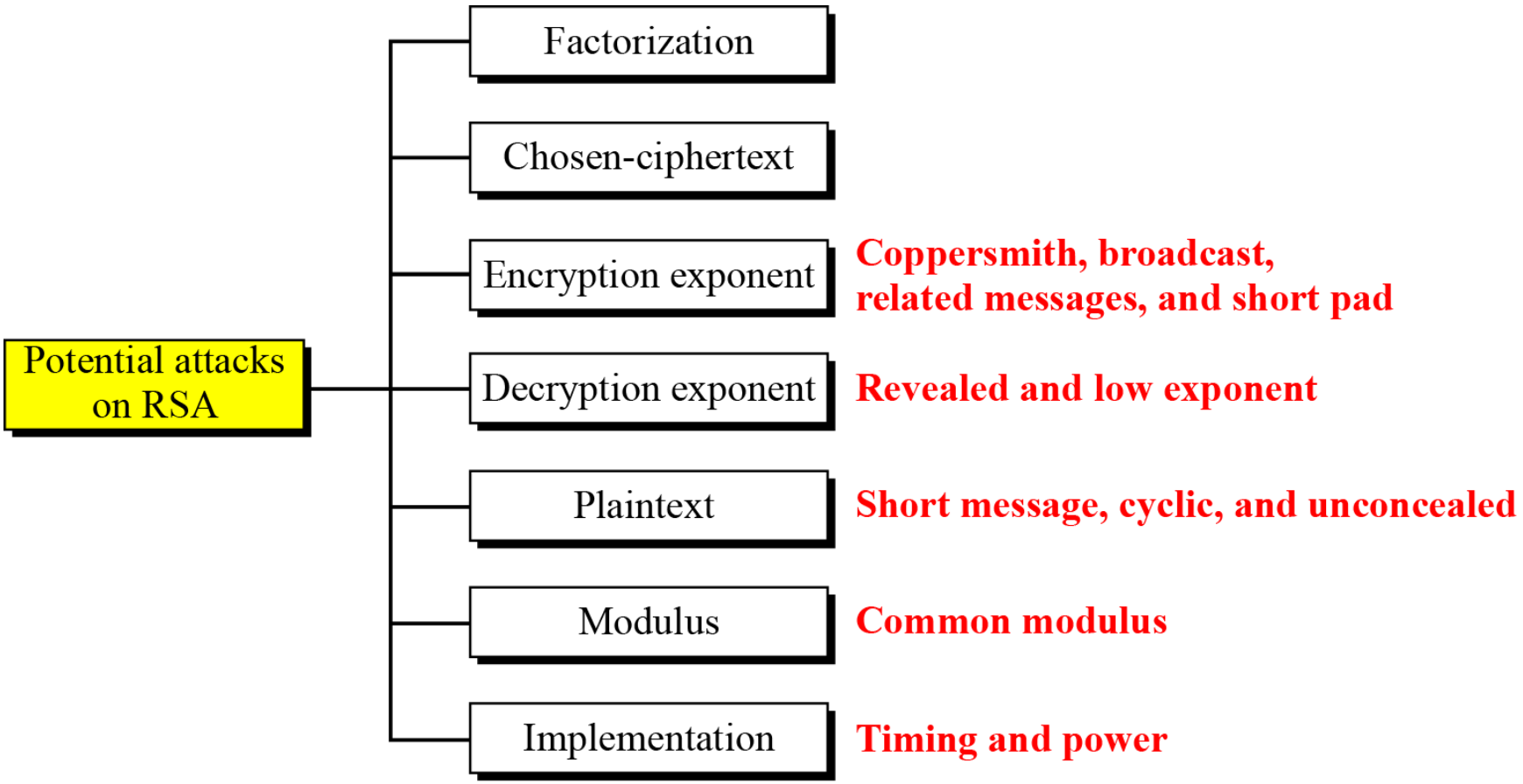


Figure: Taxonomy of potential attacks on RSA

RSA Cryptosystem: Cracking the Code

- The essential requirement of the Public Key Cryptography, like RSA, is that- the public and secret keys are mathematically related, but this relationship must be made very hard to determine by an outsider.
- As we saw in the preceding text, everything starts with ***p*** and ***q***, from which we calculated ***n***.
- The public key consists of two numbers: ***e*** and ***n***, where ***e*** is calculated from ***φ(n)***, and ***φ(n)*** is calculated from ***p*** and ***q***.
- The secret key ***d***, was calculated from ***e*** and ***φ(n)*** and, as we just stated, ***e*** and ***φ(n)*** are calculated from ***p*** and ***q***.
- It follows then, that ***d*** is also calculated from ***p*** and ***q***, which proves that the public and private keys are mathematically related.
- So, if an adversary (like Eve) wanted to find the secret key ***d***, by only knowing ***n***, he must break down ***n*** into the two prime numbers that were used to produce it (remember that ***n*** = ***p*** * ***q***).
- Now, here is the real crux of the bisquit: Decomposing a very large ***n*** into ***p*** and ***q*** is really difficult to do. It is easy with the small numbers that we have used in our demonstration, but, for example, if 100 digit numbers are used for p and q, the resulting n will be approximately 200 digits. Then decomposing ***n*** into ***p*** and ***q*** will be very hard. The fastest known factoring algorithm would take far too long for an attacker to ever break the code.
- Well, if you have some free time on your hands, try this challenge: n=13289. Find p and q. If you can find, you may even earn some money.
- Any cryptographic technique which can resist a concerted attack is regarded as secure. At this point in time, the RSA algorithm is considered secure.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

RSA Cryptosystem

- In RSA cryptosystem, encryption using public keys is normally computationally intensive. So, in practice-
  - The sender encrypts the message with a secret key that is randomly generated.
  - The secret key is encrypted using the public key of the recipient and sent with the encrypted message.
  - The recipient decrypts the secret key using his private key and using that secret key, he decrypts the rest of the message.
- The following lists all the steps in the process:
  1. The client and server go through a handshaking procedure.
  2. The handshake begins when the client connects to a SSL enabled server requesting a secure connection and presents a list of encryption algorithms and hash functions that it supports.
  3. From this list the server chooses the most secure encryption algorithm and hash function that it also supports and lets the client know about its choice.
  4. In the above transaction, the server also sends it identification in the form of a digital certificate. The digital certificate contains the server's name, the trusted Certificate Authority, and the server's public encryption key.
  5. The client may contact the trusted Certificate Authority for verification.
  6. The client generates a random number and encrypts it with the server's public key and sends it to the server. Only the server can decrypt this with its private key.
  7. The random number generated by the client is then used in the encryption and decryption process on both the client and server sides.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

RSA Cryptosystem: Applications

- Although RSA can be used to encrypt and decrypt actual messages, it is very slow if the message is long.
- Therefore, RSA is useful for short messages.
- RSA is used in digital signature and other cryptosystems that often need to encrypt a small message without having access to a symmetric key.
- RSA is also used for authentication.

Rabin Cryptosystem

- The Rabin cryptosystem (published in January 1979 by Michael O. Rabin) is an asymmetric cryptographic technique, whose security, like that of RSA, is related to the difficulty of factorization.
- This cryptosystem is a variation of the RSA cryptosystem:
  - RSA is based on the exponentiation congruence;
  - Rabin is based on the quadratic congruence;
- The Rabin cryptosystem can be thought of as an RSA cryptosystem in which the value of ***e*** and ***d*** are fixed, i.e., ***e***=2 and ***d***=1/2. This means, in Rabin cryptosystem, the formula for encryption is $C \equiv P^2 \bmod n$ and the formula for decryption is $P \equiv C^{1/2} \bmod n$.
- The public key in the Rabin cryptosystem is ***n*** and the private key is (***p,q***). Everyone can encrypt a message using n; only Bob can decrypt the message using p and q.
- Decryption of the message is infeasible for Eve because she does not know the values of p and q.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

Rabin Cryptosystem

- In Rabin cryptosystem, Bob chooses two large and distinct prime numbers p and q such that $p \equiv q \equiv 3 \pmod 4$ and then he calculates n=p x q. He announces n as the public key and keeps (p, q) to him as the private key.
- Anyone (say Alice) can encrypt a message using the public key based on the formula: $C \equiv P^2 \bmod n$ and can send the encrypted message to Bob.
- Using the private key, Bob alone can decrypt the message.
- The process is shown in the figure below.

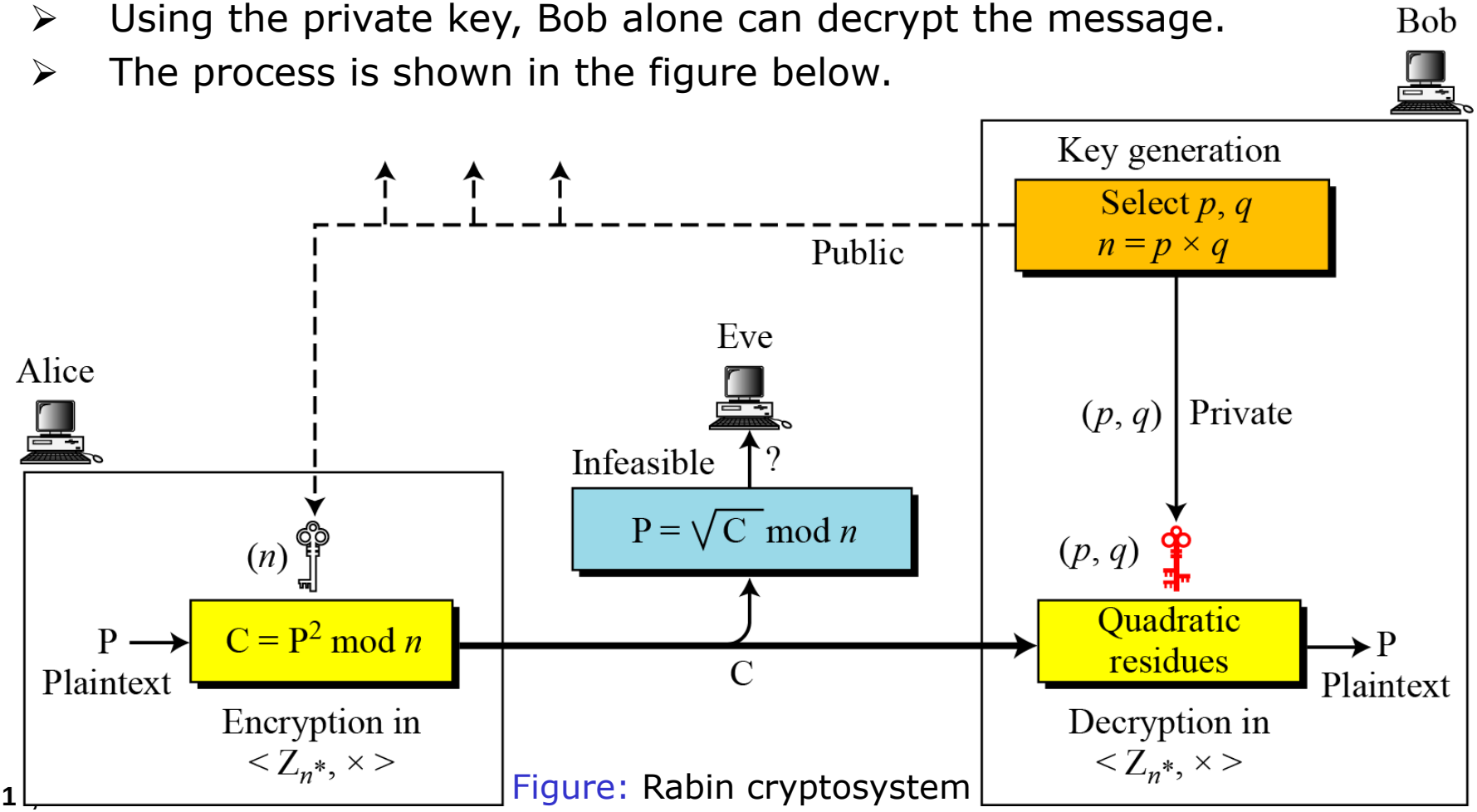


Figure: Rabin cryptosystem

Rabin Algorithm: Key Generation

- As with all asymmetric cryptosystems, the Rabin system uses both a public and a private key.
  - The public key is necessary for encryption and can be published.
  - The private key must be possessed only by the recipient of the message.
- The precise key-generation process for Rabin cryptosystem is as follows:

1. Choose two large distinct primes p and q.
   - One may choose p and q such that $p \equiv q \equiv 3 \pmod 4$ to simplify the computation of square roots modulo p and q. But the scheme works with any primes.
2. Compute n = p * q. Then n is the public key. The primes p and q are the private key.
   - To encrypt a message, only the public key n is needed.
   - To decrypt a ciphertext, the factors p and q of n are necessary.

- As a trivial example, if p = 7 and q = 11, then n=77. The public key 77 would be released, and the message is encoded using this key. In order to decode the message, the private keys 7 and 11 would have to be known.

Rabin Algorithm: Encryption

- Anyone can send a message to Bob using his public key n.
- The encryption process is listed below:

1. Represent the message as an integer in the range {0,……, n-1}.
2. Now the ciphertext c is determined by

$$c = m^2 \bmod n$$

- That is, c is the quadratic remainder of the square of the plaintext, modulo the key-number n.

- **Note:**

- In Rabin cryptosystem, encryption is very simple. The operation needs only one multiplication, which can be done quickly. This is beneficial when resources are limited. For example, smart cards have limited memory and need to use short CPU time.

- As a trivial example, let p = 7 and q = 11, then n=77 is the public key which will be used to encrypt the message.
- In our simple example, P = { 0, 2, …., 76} is our plaintext space. We will take m = 20 as our plaintext. The ciphertext is thus $c = m^2 \bmod n = 20^2 \bmod 77 = 400 \bmod 77 = 15$.
- For exactly four different values of m, the ciphertext 15 is produced, i.e. for m in { 13, 20, 57, 64 }. This is true for most ciphertexts produced by the Rabin algorithm, i.e. it is a four-to-one function.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

Rabin Algorithm: Decryption

- To decrypt the ciphertext, the private keys are necessary.
- Decryption is based on the solution of quadratic congruence.
- Because the received ciphertext is the square of the plaintext, it is guaranteed that C has roots (quadratic residues) in Z_n^*. The Chinese remainder algorithm is used to find four square roots.
- Rabin cryptosystem is not deterministic: decrypting a message can produce four different plaintext outputs, of which only one is the correct plaintext. It is up to the receiver of the message to choose one of these four as final answer.
- The decryption is performed by the following algorithm:

Algorithm: *Decryption in Rabin cryptosystem*

Rabin_Decryption (p, q, C) // C is the ciphertext; p and q are private keys

{

$a_1 \leftarrow +(C^{(p+1)/4}) \bmod p$
$a_2 \leftarrow -(C^{(p+1)/4}) \bmod p$
$b_1 \leftarrow +(C^{(q+1)/4}) \bmod q$
$b_2 \leftarrow -(C^{(q+1)/4}) \bmod q$

// The algorithm for the Chinese remainder algorithm is called four times.

$P_1 \leftarrow$ Chinese_Remainder (a_1, b_1, p, q)
$P_2 \leftarrow$ Chinese_Remainder (a_1, b_2, p, q)
$P_3 \leftarrow$ Chinese_Remainder (a_2, b_1, p, q)
$P_4 \leftarrow$ Chinese_Remainder (a_2, b_2, p, q)
return P_1, P_2, P_3, and P_4

}

Rabin Algorithm: Trivial Example

Here is a very trivial example to show the idea.

1. Bob selects p = 23 and q = 7. Note that both are congruent to 3 mod 4.
2. Bob calculates $n = p \times q = 161$.
3. Bob announces n publicly; he keeps p and q private.
4. Alice wants to send the plaintext P = 24. Note that 161 and 24 are relatively prime; 24 is in Z_{161}^*. She calculates $C = 24^2 = 93 \bmod 161$, and sends the ciphertext 93 to Bob.
5. Bob receives 93 and calculates four values:
   $a_1 = +(93^{(23+1)/4}) \bmod 23 = 1 \bmod 23$
   $a_2 = -(93^{(23+1)/4}) \bmod 23 = 22 \bmod 23$
   $b_1 = +(93^{(7+1)/4}) \bmod 7 = 4 \bmod 7$
   $b_2 = -(93^{(7+1)/4}) \bmod 7 = 3 \bmod 7$
6. Bob takes four possible answers, (a_1, b_1), (a_1, b_2), (a_2, b_1), and (a_2, b_2), and uses the Chinese remainder theorem to find four possible plaintexts: 116, 24, 137, and 45. Note that only the second answer is Alice's plaintext.

Evaluation of Rabin Algorithm

Effectiveness:

- In Rabin cryptosystem, decoding produces three false results in addition to the correct one, so that the correct result must be guessed. This is the major disadvantage of the Rabin cryptosystem and one of the factors which have prevented it from finding widespread practical use.
- If the plaintext is intended to represent a text message, guessing is not difficult; however, if the plaintext is intended to represent a numerical value, this issue becomes a problem that must be resolved by some kind of disambiguation scheme. It is possible to choose plaintexts with special structures, or to add padding, to eliminate this problem.
- A way of removing the ambiguity of inversion was suggested by Blum and Williams:
  - the two primes used are restricted to primes congruent to 3 modulo 4 and the domain of the squaring is restricted to the set of quadratic residues. These restrictions make the squaring function into a trapdoor permutation, eliminating the ambiguity.

Evaluation of Rabin Algorithm

Security:

- The Rabin cryptosystem is computationally secure against a chosen plaintext attack provided that the modulus n can not be factored. That is, if p and q are very large, then Rabin system is as secure as RSA.
- The great advantage of the Rabin cryptosystem is that a random plaintext can be recovered entirely from the ciphertext only if the codebreaker is capable of efficiently factoring the public key n. Note that this is a very weak level of security. Extensions of the Rabin cryptosystem achieve stronger notions of security.
- It has been proven that decoding the Rabin cryptosystem is equivalent to the integer factorization problem, which is rather different than for RSA. Thus the Rabin system is 'more secure' in this sense than is RSA, and will remain so until a general solution for the factorization problem is discovered, or until the RSA problem is discovered to be equivalent to factorization. (This assumes that the plaintext was not created with a specific structure to ease decoding.)

RSA Vs. Rabin Cryptosystems

- Both the RSA and the Rabin cryptosystems are asymmetric cryptographic techniques.
- Rabin cryptosystem is a variation of the RSA cryptosystem:
  - RSA is based on the exponentiation congruence;
  - Rabin is based on the quadratic congruence;
- The Rabin cryptosystem can be thought of as an RSA cryptosystem in which the value of ***e*** and ***d*** are fixed, i.e., ***e***=2 and ***d***=1/2. This means, in Rabin cryptosystem, the formula for encryption is $C \equiv P^2 \bmod n$ and the formula for decryption is $P \equiv C^{1/2} \bmod n$.
- The public key in the Rabin cryptosystem is ***n*** and the private key is (***p,q***). But in RSA, the public key is (***e,n***) and private key is ***d***.
- If Bob is using RSA, he can keep d and n; and discard vp, q, and φ(n) after key generation. But, if Bob is using Rabin cryptosystem, he needs to keep p and q.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

ElGamal Cryptosystems

- Besides RSA and Rabin, another public-key cryptosystem is ElGamal, named after its inventor Taher ElGamal.
- This cryptosystem is based on the discrete logarithm problem.
- Figure below shows the key generation, encryption, and decryption in ElGamal cryptosystem.

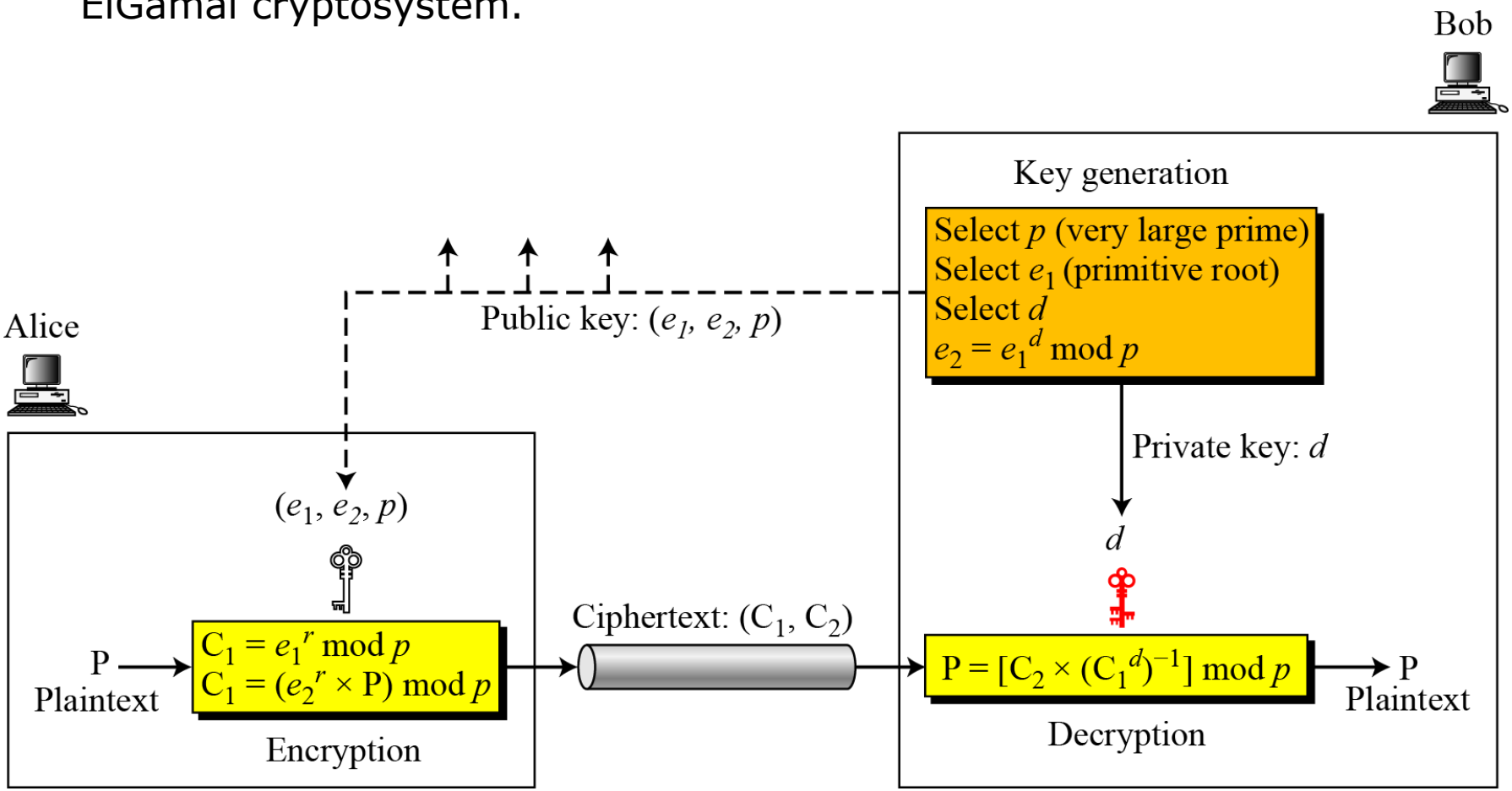


Figure: Key generation, encryption, and decryption in ElGamal

ElGamal Cryptosystems: Key Generation

- Bob uses the steps shown in the algorithm below to create his public and private keys.

Algorithm: *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

}

ElGamal Cryptosystems: Encryption

- Anyone can send a message to Bob using his public key.
- The encryption process is shown in the algorithm below.

Algorithm: *ElGamal encryption*

ElGamal_Encryption (e_1, e_2, p, P) // P is the plaintext

{

Select a random integer r in the group $\mathbf{G} = < \mathbf{Z}_p^*, \times >$

$C_1 \leftarrow e_1^{\,r} \bmod p$

$C_2 \leftarrow (\text{P} \times e_2^{\,r}) \bmod p$ // C_1 and C_2 are the ciphertexts

return C_1 and C_2

}

ElGamal Cryptosystems: Decryption

- Bob can use the following algorithm to decrypt the ciphertext message he received.

Algorithm : *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
}
```

ElGamal Cryptosystems: Trivial Example

Here is a trivial example.

- Bob chooses p = 11 and e_1 = 2. Note that 2 is a primitive root in Z_{11}^*.
- Bob then chooses d = 3 and calculate $e_2 = e_1^d = 8$.
- 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)

Bob receives the ciphertexts (5 and 6) and calculates the plaintext.

Ciphertext: $[C_2 \times (C_1^d)^{-1}] \bmod 11 = 6 \times (5^3)^{-1} \bmod 11 = 6 \times 3 \bmod 11 = 7 \bmod 11$

Plaintext: 7

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

ElGamal Cryptosystems: Trivial Example

Instead of using $P = [C_2 \times (C_1^d)^{-1}] \bmod p$ for decryption, we can avoid the calculation of multiplicative inverse and use

$$P = [C_2 \times C_1^{p-1-d}] \bmod p.$$

In the previous example, we can calculate

$$P = [6 \times 5^{11-1-3}] \bmod 11 = 7 \bmod 11.$$

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

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

ElGamal Cryptosystems: Realistic Example

Bob uses a random integer of 512 bits (the ideal is 1024 bits).
The integer p is a 155-digit number (the ideal is 300 digits).
Bob then chooses e_1, d, and calculates e_2, as shown below:

$p =$	115348992725616762449253137170143317404900945326098349598143469219056898698622645932129754737871895144368891765264730936159299937280611659643473534400085 77
$e_1 =$	2
$d =$	1007
$e_2 =$	97886413043009189508766856938097739043880062887337687610022062233255450707415618921231831770461014167336015088413294085724853770315820660100725587074 55

Bob announces (e_1, e_2, p) as his public key and keeps d as his private key.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

ElGamal Cryptosystems: Trivial Example

Alice has the plaintext P = 3200 to send to Bob.
She chooses r = 545131, calculates C_1 and C_2, and sends them to Bob.

P =	3200
***r* =**	545131
C_1 =	88729706938352847102257047149227566312026006725656212501818835142941722359971268111410536366170517305158153318916540097373635508029573678785690606191528881
C_2 =	70845433304892994457701601238079499956743602183619244696177450692124469615516580077945559308034588961440240859952591957920972162887968135058277956643029 50

Bob calculates the plaintext $P = C_2 \times ((C_1)^d)^{-1} \bmod p = 3200 \bmod p$.

P =	3200

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

ElGamal Cryptosystems: Application

ElGamal can be used whenever RSA can be used.
It is used for key exchange, authentication, and encryption and decryption of small messages.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU