LECTURE 4

RSA cryptosystem

Telecommunication systems department

Lecturer: assistant professor Persikov Anatoliy Valentinovich

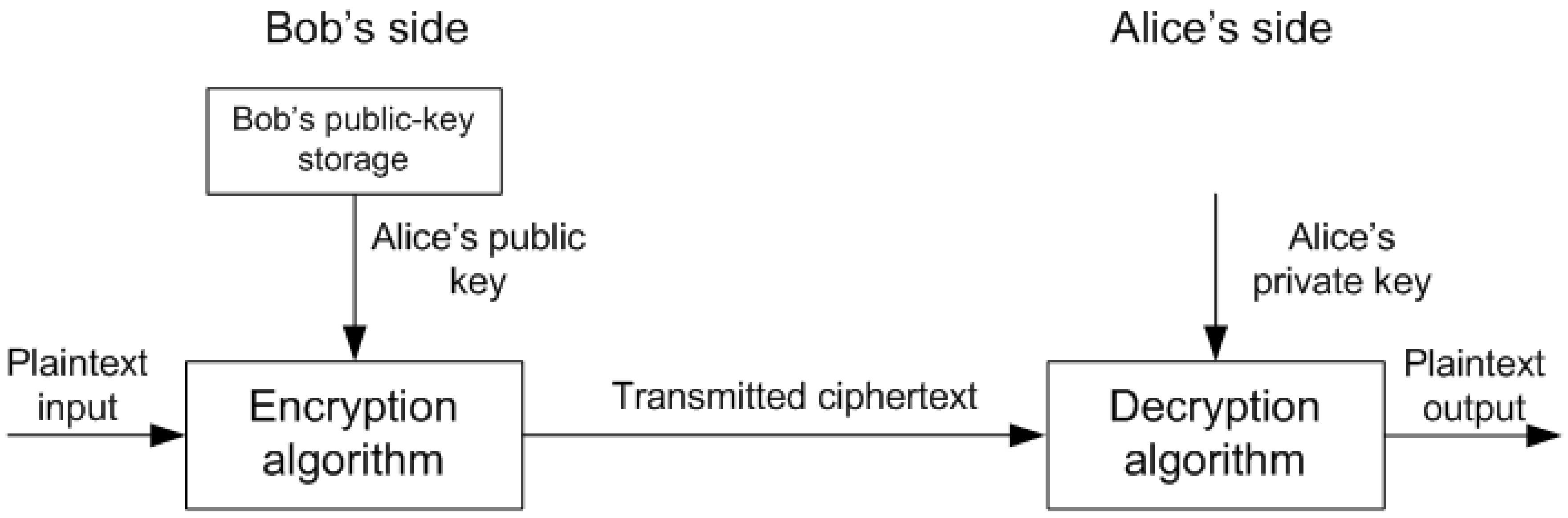
PUBLIC-KEY CRYPTOSYSTEMS

Asymmetric algorithms rely on one key for encryption and a different but related key for decryption. These algorithms have the following important characteristic:

– It is computationally infeasible to determine the decryption key given only knowledge of the cryptographic algorithm and the encryption key.

In addition, some algorithms, such as RSA, also exhibit the following characteristic:

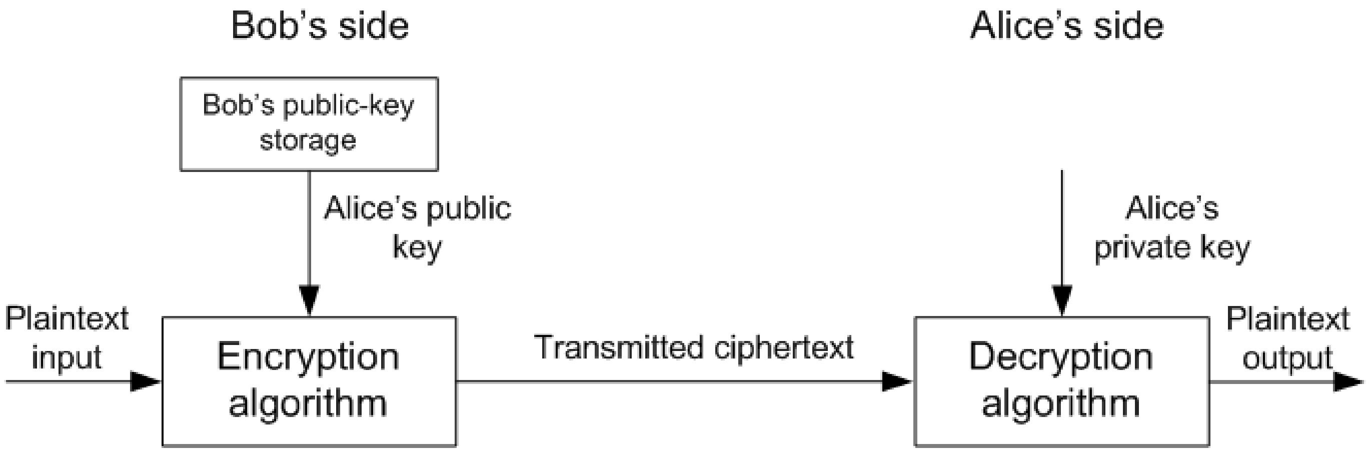
– Either of the two related keys can be used for encryption, with the other used for decryption.



PUBLIC-KEY CRYPTOSYSTEMS

A public-key encryption scheme has six ingredients:

- **Plaintext**: This is the readable message or data that is fed into the algorithm as input.
- **Encryption algorithm**: The encryption algorithm performs various transformations on the plaintext.
- **Public** and **private keys**: This is a pair of keys that have been selected so that if one is used for encryption, the other is used for decryption. The exact transformations performed by the algorithm depend on the public or private key that is provided as input.
- **Ciphertext**: This is the scrambled message produced as output. It depends on the plaintext and the key. For a given message, two different keys will produce two different ciphertexts.
- **Decryption algorithm**: This algorithm accepts the ciphertext and the matching key and produces the original plaintext.



PUBLIC-KEY CRYPTOSYSTEMS

The essential steps are the following:

1) Each user generates a pair of keys to be used for the encryption and decryption of messages.
2) Each user places one of the two keys in a public register or other accessible file. This is the public key. The companion key is kept private. Each user maintains a collection of public keys obtained from others.
3) If Bob wishes to send a confidential message to Alice, Bob encrypts the message using Alice's public key.
4) When Alice receives the message, she decrypts it using her private key. No other recipient can decrypt the message because only Alice knows Alice's private key.

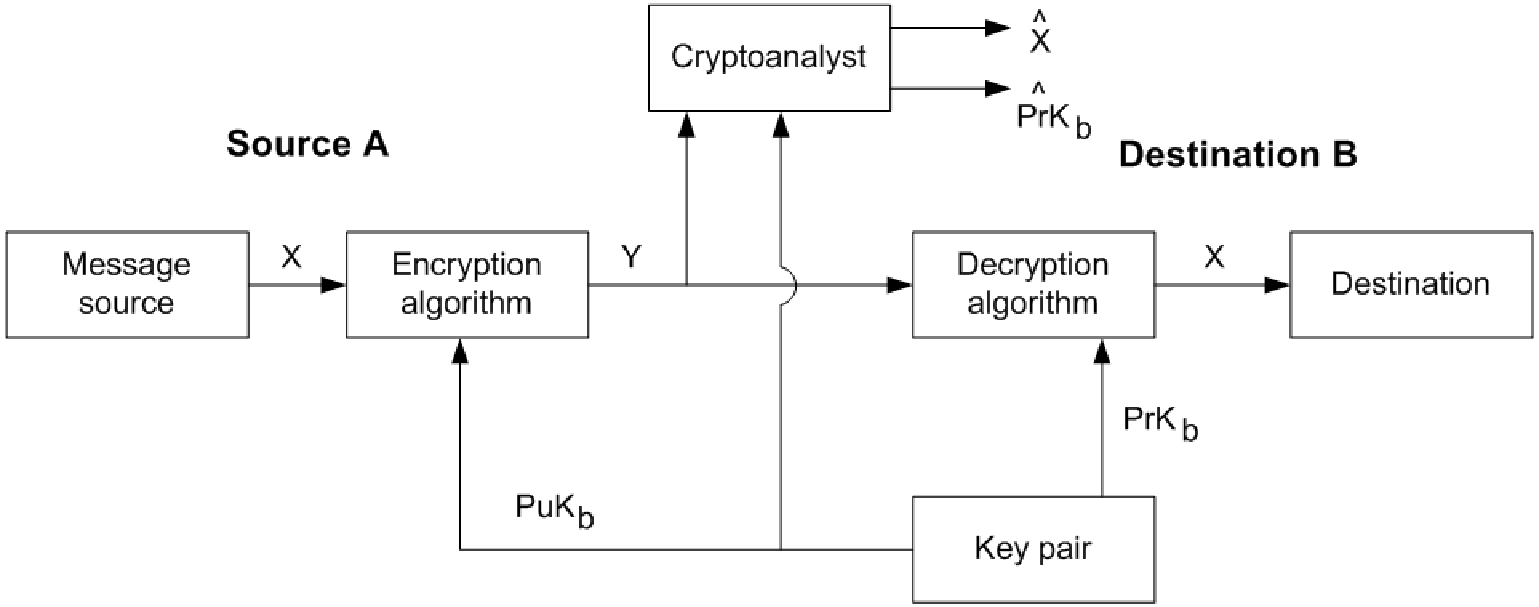
With this approach, all participants have access to public keys, and private keys are generated locally by each participant and therefore need never be distributed. As long as a user's private key remains protected and secret, incoming communication is secure. At any time, a system can change its private key and publish the companion public key to replace its old public key.

CRYPTOSYSTEMS COMPARISON

Conventional Encryption	Public-Key Encryption
Needed to work: 1. The same algorithm with the same key is used for encryption and decryption. 2. The sender and receiver must share the algorithm and the key.	**Needed to work:** 1. One algorithm is used for encryption and decryption with a pair of keys, one for encryption and one for decryption. 2. The sender and receiver must each have one of the matched pair of keys (not the same one).
Needed for security: 1. The key must be kept secret. 2. It must be impossible or at least impractical to decipher a message if no other information is available. 3. Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key.	**Needed for security:** 1. One of the two keys must be kept secret. 2. It must be impossible or at least impractical to decipher a message if no other information is available. 3. Knowledge of the algorithm plus one of the keys plus samples of ciphertext must be insufficient to determine the other key.

PUBLIC-KEY CRYPTOSYSTEM CLOSE LOOK

Let us take a closer look at the essential elements of a public-key encryption scheme. There is some source **A** that produces a message in plaintext, $X \ = [X_1, X_2, \ldots, X_M]$. The M elements of X are letters in some finite alphabet. The message is intended for destination **B**. **B** generates a related pair of keys: a public key, PuK_b, and a private key, PrK_b. PrK_b is known only to B, whereas $PuK_b$is publicly available and therefore accessible by **A**.



PUBLIC-KEY CRYPTOSYSTEM CLOSE LOOK

With the message X and the encryption key PuK_b as input, A forms the ciphertext $Y = [Y_1, Y_2, \ldots, Y_M]$:

$$Y = E(PuK_b, X)$$

The intended receiver, in possession of the matching private key, is able to invert the transformation:

$$X = D(PrK_b, Y)$$

An adversary, observing Y and having access to PuK_b but not having access to PrK_b or X, must attempt to recover X and/or PrK_b. It is assumed that the adversary does have knowledge of the encryption (E) and decryption (D) algorithms. If the adversary is interested only in this particular message, then the focus of effort is to recover X, by generating a plaintext estimate$\hat{X}$. Often, however, the adversary is interested in being able to read future messages as well, in which case an attempt is made to recover PrK_b by generating an estimate $\widehat{PrK_b}$.

APPLICATIONS FOR PUBLIC-KEY CRYPTOSYSTEMS

Before proceeding, we need to clarify one aspect of public-key cryptosystems that is otherwise likely to lead to confusion. Public-key systems are characterized by the use of a cryptographic algorithm with two keys, one held private and one available publicly. The sender uses either the sender's private key or the receiver's public key, or both, to perform some type of cryptographic function. In broad terms, we can classify the use of public-key cryptosystems into three categories:

- **Encryption/decryption:** The sender encrypts a message with the recipient's public key.
- **Digital signature:** The sender "signs" a message with its private key. Signing is achieved by a cryptographic algorithm applied to the message.
- **Key exchange**: Two sides cooperate to exchange a session key. Several different approaches are possible, involving the private key(s) of one or both parties.

Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Elliptic Curve	Yes	Yes	Yes
Diffie-Hellman	No	No	Yes
DSS	No	Yes	No

REQUIREMENTS FOR PUBLIC-KEY CRYPTOGRAPHY

Requirements for public-key cryptography:

1. It is computationally easy for a party **B** to generate a pair (public key PuK_b, private key PrK_b).
2. It is computationally easy for a sender **A**, knowing the public key and the message to be encrypted, M, to generate the corresponding ciphertext: $C = E(PuK_b, M)$
3. It is computationally easy for the receiver **B** to decrypt the resulting ciphertext using the private key to recover the original message: $M = D(PrK_b, C) = D[PrK_b, E(PuK_b, M)]$
4. It is computationally infeasible for an adversary, knowing the public key, PuK_b, to determine the private key, PrK_b.
5. It is computationally infeasible for an adversary, knowing the public key, PuK_b, and a ciphertext, C, to recover the original message, M.

We can add a sixth requirement that, although useful, is not necessary for all public-key applications:

6. The two keys can be applied in either order:

$$M = D[PuK_b, E(PrK_b, M)] = D[PrK_b, E(PuK_b, M)]$$

THE RSA ALGORITHM

The scheme developed by Rivest, Shamir, and Adleman makes use of an expression with exponentials. Plaintext is encrypted in blocks, with each block having a binary value less than some number n. That is, the block size must be less than or equal to $log_2(n)$; in practice, the block size is i bits, where $2^i < n \leq 2^{i+1}$. Encryption and decryption are of the following form, for some plaintext block M and ciphertext block C:

$$C = M^e \bmod n$$

$$M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$$

Both sender and receiver must know the value of n. The sender knows the value of e, and only the receiver knows the value of d. Thus, this is a public-key encryption algorithm with a public key of $PuK = \{e, n\}$ and a private key of $PrK = \{d, n\}$. For this algorithm to be satisfactory for public-key encryption, the following requirements must be met:

1) It is possible to find values of e, d, n such that $M^{ed} \bmod n = M$ for all $M < n$.
2) It is relatively easy to calculate $M^e \bmod n$ and $C^d \bmod n$ for all values of $M < n$.
3) It is infeasible to determine d given e and n.

THE RSA ALGORITHM

For now, we focus on the first requirement and consider the other questions later. We need to find a relationship of the form

$$M^{ed} \bmod n = M$$

The preceding relationship holds if e and d are multiplicative inverses modulo $\varphi(n)$, where $\varphi(n)$ is the **Euler totient function**. For p, q prime, $\varphi(p \cdot q) = (p-1) \cdot (q-1)$ The relationship between e and d can be expressed as

$$e \cdot d \bmod \varphi(n) = 1$$

This is equivalent to saying

$$e \cdot d \equiv 1\, mod\ \varphi(n)$$

$$d \equiv e^{-1} \bmod \varphi(n)$$

That is, e and d are multiplicative inverses $\mathrm{mod}\ \varphi(n)$. Note that, according to the rules of modular arithmetic, this is true only if d (and therefore e) is relatively prime to $\varphi(n)$. Equivalently, $gcd(\varphi(n), d) = 1$.

THE RSA ALGORITHM

We are now ready to state the RSA scheme. The ingredients are the following:

p, q, two prime numbers	(private, chosen)
$n = pq$	(public, calculated)
e, with $\gcd(\varphi(n), e) = 1; 1 < e < \varphi(n)$	(public, chosen)
$d \equiv e^{-1} \bmod \varphi(n)$	(private, calculated)

Key generation

Select p, q	p and q both prime,
$p \neq q$	
Calculate $n = p \times q$	
Calculate $\varphi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\varphi(n), e) = 1;$
$1 < e < \varphi(n)$	
Calculate d	$d \equiv e^{-1} \bmod \varphi(n)$
Public key	$PuK = \{e, n\}$
Private key	$PrK = \{d, n\}$

THE RSA ALGORITHM

Encryption

Plaintext	$M < n$
Ciphertext	$C = M^e \bmod n$

Decryption

Ciphertext	C
Plaintext	$M = C^d \bmod n$

THANKS FOR ATTENTION