Lecture 20: Public-key Encryption & Hybrid Encryption

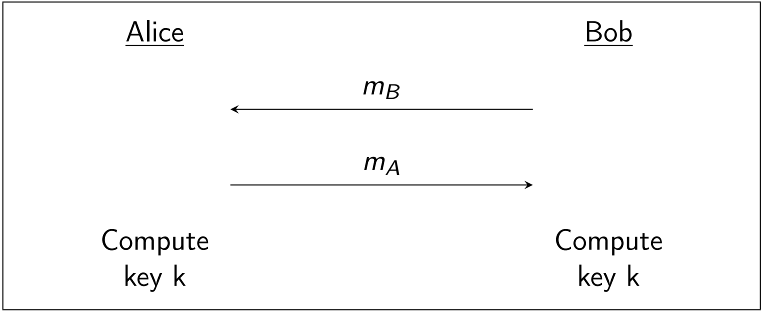
Lecture 20: Public-key Encryption & Hybrid Encryption

Overview

- Suppose there is a 2-round Key-Agreement protocol. This means that there exists a protocol where
  - Bob sends the first message m_B
  - Alice sends the second message m_A
  - Now, parties can compute a secret key key that is hidden from an eavesdropper (who got to see the first message by Bob and the second message by Alice)
  - For example, the Diffie-Hellman key-exchange protocol. Bob sends $m_B = g^b$, Alice sends $m_A = g^a$, and both parties compute the key key $= g^{ab}$, but it remains hidden from any computationally bounded adversary who sees only $A = g^a$ and $B = g^b$.
- Using this 2-round key-agreement protocol we can construct a public-key encryption scheme. For example, using the Diffie-Hellman key-exchange protocol, we shall construct the ElGamal public-key encryption scheme

First Component: 2-round Key-Agreement Protocol I

- Suppose we have a protocol $\Pi_{2-\mathsf{KA}}$, which is a 2-round key-agreement protocol that looks like the following



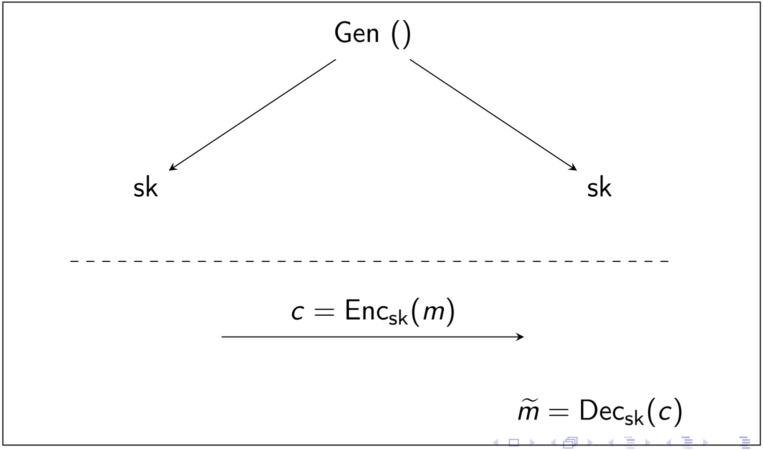
- Note that $\Pi_{2-\mathsf{KA}}$ can be any 2-round key-agreement protocol. One such example is the Diffie-Hellman key-agreement protocol. The next slide presents this protocol in this template.

First Component: 2-round Key-Agreement Protocol II

- For example, we consider $\Pi_{2-\mathsf{KA}}$ to be the Diffie-Hellman key agreement protocol

Alice		Bob
	$\xleftarrow{m_B = g^b}$	
	$\xrightarrow{m_A = g^a}$	
Compute key $\mathsf{k} = m_B^a$		Compute key $\mathsf{k} = m_A^b$

- Suppose we have a private-key encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. Without loss of generality, we can assume that $\mathsf{Gen}()$ outputs a uniformly random key sk from a set S. Recall that a private-key encryption scheme looks as follows



Second Component: Private-key Encryption II

- Consider, for example, the one-time pad encryption scheme

Gen ()

sk   sk

$$\xrightarrow{c = m \circ \mathsf{sk}}$$

$$\widetilde{m} = c \circ \mathrm{inv}(\mathsf{sk})$$

Combining to obtain a Public-key Encryption Scheme I

If the key of the first component is random over the set S (from which the private-key of the second-component is chosen) then we can stick together these two protocols as follows

Alice		Bob
	$\xleftarrow{\quad m_B \quad}$	
	$\xrightarrow{\quad m_A \quad}$	
Compute key k		Compute key k
Set $\mathsf{sk} = \mathsf{k}$		Set $\mathsf{sk} = \mathsf{k}$
	$\xrightarrow{c = \mathsf{Enc}_{\mathsf{sk}}(m)}$	
		$\widetilde{m} = \mathsf{Dec}_{\mathsf{sk}}(c)$

Combining to obtain a Public-key Encryption Scheme II

We can merge the message m_A and c into one-single message. And we get the following scheme.

Alice		Bob
	$\xleftarrow{\quad m_B \quad}$	
Compute key k		
- - - - - - - -	- - - - - - - -	- - - - - - - -
Set sk $=$ k		
	$\xrightarrow{\quad m_A, c = \mathsf{Enc}_{\mathsf{sk}}(m) \quad}$	
		Compute key k
		Set sk $=$ k
		$\widetilde{m} = \mathsf{Dec}_{\mathsf{sk}}(c)$

Combining to obtain a Public-key Encryption Scheme III

Every time we want to encrypt a message m, we calculate a fresh key k. And we get the following scheme.

<u>Alice</u>		<u>Bob</u>
	$\xleftarrow{\quad m_B \quad}$	
Compute key k Set $\mathsf{sk} = \mathsf{k}$		
	$\xrightarrow{\quad m_A,\, c = \mathsf{Enc}_{\mathsf{sk}}(m) \quad}$	
		Compute key k Set $\mathsf{sk} = \mathsf{k}$ $\widetilde{m} = \mathsf{Dec}_{\mathsf{sk}}(c)$

Combining to obtain a Public-key Encryption Scheme IV

Finally, we interpret the message m_B as the public-key for Bob. And the messages (m_A, c) as the encryption of the message m. This gives us our public-key encryption scheme!

Alice		Bob
	$\xleftarrow{\mathsf{pk} = m_B}$	
Compute key k		
Set sk = k		
	$\xrightarrow[\text{where } c' = \mathsf{Enc}_{\mathsf{sk}}(m)]{c = (m_A, c'),}$	
		Compute key k
		Set sk = k
		$\widetilde{m} = \mathsf{Dec}_{\mathsf{sk}}(c)$

Example I

- Suppose our first component is Diffie-Hellman key-agreement protocol and the second component is one-time pad. Then we get the following public-key encryption scheme.

Alice		Bob
	$\xleftarrow{\mathsf{pk} = g^{b}}$	
Compute key $\mathsf{k} = g^{ab}$ Set $\mathsf{sk} = \mathsf{k}$		
	$\xrightarrow{c = (m_A = g^{a},\, c' = m \cdot g^{ab})}$	
		Compute key $\mathsf{k} = g^{ab}$ Set $\mathsf{sk} = \mathsf{k}$ $\widetilde{m} = c' \cdot \mathsf{inv}(g^{ab})$

Example II

This is the ElGamal public-key encryption scheme!

Summary: ElGamal Encryption

- Let us summarize the ElGamal Public-key Encryption as an instantiation of 2-round Diffie-Hellman key-agreement protocol and the one-time pad private-key encryption scheme
- Recall that to describe a private-key encryption scheme we had to provide the algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. Similarly, to describe a public-key encryption scheme, we will have to provide the $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ algorithms
- Assume that the DDH Assumption holds for the group $(G, \circ)$ of size N, and the group G has a generator g
- For perspective, N is large and is in the order of 2^n, where $n = 1024$. Our algorithms have to be polynomial in n and the adversary, to break the scheme, has to invest roughly $2^{\text{constant} \cdot n}$ effort

Generation Algorithm.

- Recall that in the private-key encryption scheme the generation algorithm Gen() outputs the secret-key for the encryption scheme. In public-key encryption, the generation algorithm has to output the public-key pk for the scheme. Additionally, it has to output the "trapdoor" trap that assists the receiver to decrypt the cipher-text. If such a trapdoor does not exist, then Bob gets no additional advantage over an eavesdropper to decrypt the cipher-text.

Gen():

1. Sample $b \xleftarrow{\$} \{0, 1, 2, \ldots, N-1\}$
2. Compute $B = g^b$ (using repeated squaring technique)
3. Return $(\mathrm{pk} = B, \mathrm{trap} = b)$

Now, the receiver can broadcast the pk to everyone and keep trap secret with herself to assist in the decryption algorithm

Encryption Algorithm.

- Recall that in the private-key encryption scheme the encryption algorithm takes two inputs (the secret-key and the message) $\mathsf{Enc}_{\mathsf{sk}}(m)$ and outputs the cipher-text. In the public-key encryption, it will take the public-key and the message as input and output the cipher-text.

$\mathsf{Enc}_{\mathsf{pk}}(m)$:

1. Sample $a \xleftarrow{\$} \{0, 1, 2, \ldots, N-1\}$
2. Compute $A = g^a$ (using repeated squaring technique)
3. Compute $\mathsf{mask} = \mathsf{pk}^a$ (using repeated squaring technique)
4. Return the cipher-text $c = (A, m \circ \mathsf{mask})$

In the ElGamal encryption scheme $\mathsf{pk} = B$. Note that each time the encryption algorithm is invoked, it will create a new random mask. If the same mask is generated in two different invocations of the encryption algorithm, then it must be the case that the same A was generate in those two invocations. That implies that the same a was generate in those two invocations, which has probability $\sqrt{2^{-n}} = 2^{-n/2}$ by the birthday bound)

Decryption Algorithm.

- Recall that in the private-key encryption scheme the decryption algorithm takes two inputs (the secret-key and the cipher-text) $\mathrm{Dec}_{\mathsf{sk}}(c)$. In the public-key encryption, it will take the cipher-text and the trapdoor generated during the generation procedure as input.

$\mathrm{Dec}_{\mathsf{trap}}(\widetilde{A}, \widetilde{c})$:

1. Compute $\widetilde{\mathsf{mask}}(\widetilde{A})^{\mathsf{trap}}$
2. Return $\widetilde{c} \circ \mathsf{inv}(\widetilde{\mathsf{mask}})$

Recall that $\mathsf{trap} = b$. If $\widetilde{A} = g^a$, then $\widetilde{\mathsf{mask}} = g^{ab}$.

Hybrid Encryption I

- We will combine <u>any</u> public-key encryption scheme with <u>any</u> private-key encryption scheme to create a new public-key encryption (called, the hybrid-encryption scheme)
- We emphasize that any public-key encryption scheme can be used. It need not be the ElGamal Scheme. You can choose any encryption scheme that you prefer.
- The benefit of hybrid-encryption is that it allows us to combine two encryption scheme in a modular fashion.
- Suppose the public-key encryption scheme is provided by the triplet of algorithms

$$\left(\mathsf{Gen}^{(\mathsf{pub})}, \mathsf{Enc}^{(\mathsf{pub})}, \mathsf{Dec}^{(\mathsf{pub})}\right)$$

Hybrid Encryption II

- Suppose the private-key encryption scheme is provided by the triplet of algorithms

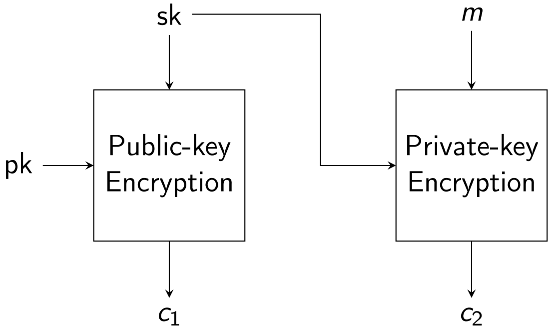
$$(\mathsf{Gen}^{(\mathsf{priv})}, \mathsf{Enc}^{(\mathsf{priv})}, \mathsf{Dec}^{(\mathsf{priv})})$$

- Now, we need to describe the hybrid-encryption scheme algorithms

$$(\mathsf{Gen}^{(\mathsf{hyb})}, \mathsf{Enc}^{(\mathsf{hyb})}, \mathsf{Dec}^{(\mathsf{hyb})})$$

Hybrid Encryption III

Let us first draw a block-diagram for intuition purpose



- The secret-key sk will be encrypted by the public-key encryption
- The secret-key sk will be used to encryption the actual message m using the private-key encryption

Generation Algorithm for Hybrid-Encryption.

$\mathsf{Gen}^{(\mathsf{hyb})}()$:

1. Return $(\mathsf{pk}, \mathsf{trap}) = \mathsf{Gen}^{(\mathsf{pub})}()$

The receiver broadcasts pk and keeps trap safe with herself

Encryption Algorithm for Hybrid-Encryption.

$\mathsf{Enc}_{\mathsf{pk}}(m)$:

1. Generate $\mathsf{sk} = \mathsf{Gen}^{(\mathsf{priv})}()$
2. Encrypt the secret-key $c_1 = \mathsf{Enc}_{\mathsf{pk}}^{(\mathsf{pub})}(\mathsf{sk})$
3. Encrypt the message $c_2 = \mathsf{Enc}_{\mathsf{sk}}^{(\mathsf{priv})}(m)$
4. Return the cipher-test (c_1, c_2)

Hybrid Encryption VI

Decryption Algorithm for Hybrid-Encryption.

$\mathrm{Dec}_{\mathrm{trap}}(\widetilde{c}_1, \widetilde{c}_2)$:

1. Decrypt the secret-key $\widetilde{\mathrm{sk}} = \mathrm{Dec}^{(\mathrm{pub})}_{\mathrm{trap}}(\widetilde{c}_1)$
2. Return the decrypted the message $\widetilde{m} = \mathrm{Dec}^{(\mathrm{priv})}_{\widetilde{\mathrm{sk}}}(\widetilde{c}_2)$