COS433/Math 473: Cryptography

Mark Zhandry

Princeton University

Spring 2017

Last Time

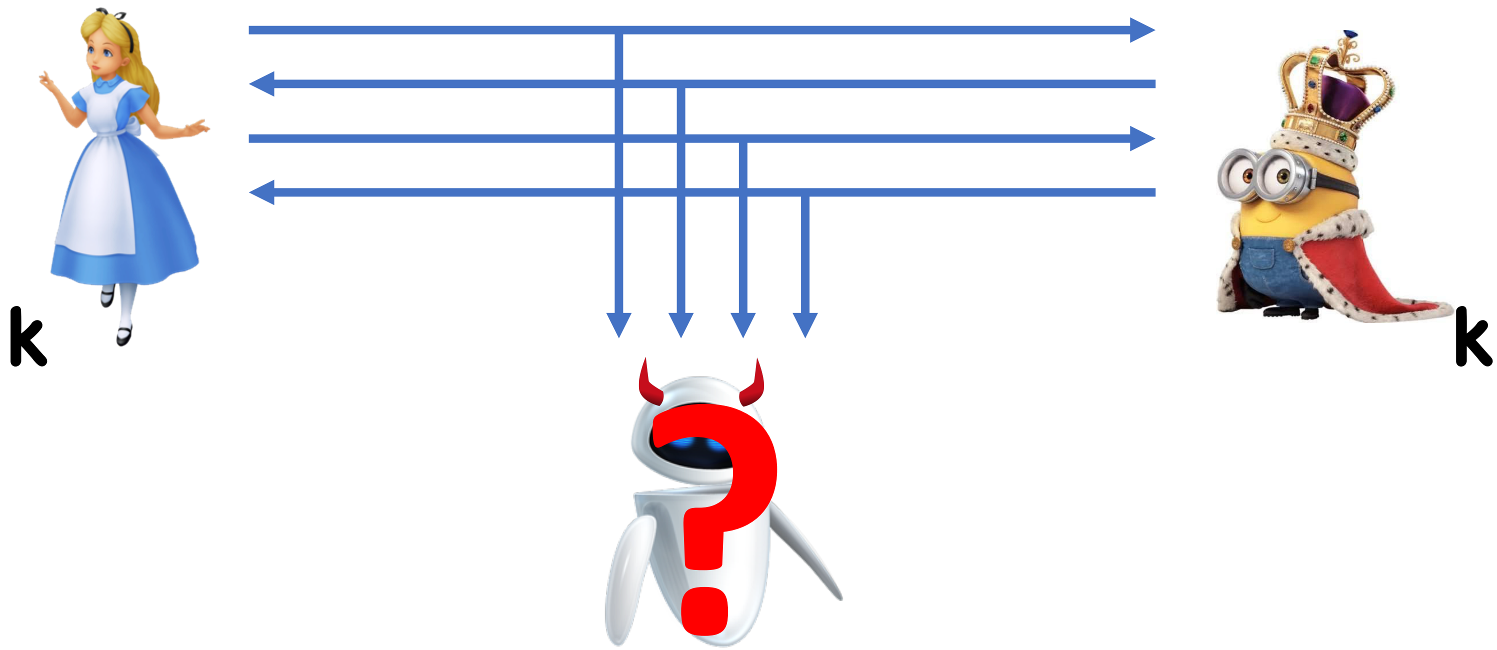
Exchanging Keys

Public Key Distribution

Public Key Distribution

Public Key Distribution

Public Key Distribution



Public Key Distribution

Pair of interactive algorithms $\mathbf{A(\lambda),B(\lambda)}$

Correctness:

$$\Pr[o_A=o_B:\ (\text{Trans},o_A,o_B)\leftarrow(A,B)(\lambda,\ \lambda)] = 1$$

Shared key is $\mathbf{k} := o_A=o_B$

- Define $\mathbf{(Trans,k)\leftarrow(A,B)(\lambda)}$

Security: $\mathbf{(Trans,k)}$ is computationally indistinguishable from $\mathbf{(Trans,k')}$ where $\mathbf{k'\leftarrow K}$

Key Distribution from RSA

p,q random primes

N=pq

N

Key Distribution from RSA

p,q random primes

N=pq

N

y

$x \leftarrow \mathbb{Z}_N^*$

$y \leftarrow x^3 \bmod N$

x

Key Distribution from RSA

p,q random primes

N=pq

N

y

$x \leftarrow \mathbb{Z}_N^*$

$y \leftarrow x^3 \bmod N$

$x = y^{1/3} \bmod N$

x

Analysis

$\mathbf{x}$ uniquely defined as long as $\mathbf{GCD(3,\Phi(N)) = 1}$

- $\mathbf{3}$ is not a factor of $\mathbf{(p-1)}$ or $\mathbf{(q-1)}$

How does Alice compute $\mathbf{x = y^{1/3} \bmod N}$?

Security:

- Computing cube roots is hard (assuming RSA)
- Adversary cannot compute $\mathbf{x}$
- However, $\mathbf{x}$ is distinguishable from a random key

Key Distribution from DH

Everyone agrees on group **G** or prime order **p**

$a \leftarrow \mathbb{Z}_p$

$b \leftarrow \mathbb{Z}_p$

Key Distribution from DH

Everyone agrees on group **G** or prime order **p**

$a \leftarrow \mathbb{Z}_p$

g^a

g^b

$b \leftarrow \mathbb{Z}_p$

Key Distribution from DH

Everyone agrees on group **G** or prime order **p**

$\mathbf{a} \leftarrow \mathbb{Z}_p$

$g^a \longrightarrow$

$\longleftarrow g^b$

$\mathbf{b} \leftarrow \mathbb{Z}_p$

$k = (g^b)^a = g^{ab}$

$k = (g^a)^b = g^{ab}$

Key Distribution from DH

Theorem: If DDH holds on $\mathbb{G}$, then the Diffie-Hellman protocol is secure

Proof:

- $\mathbf{(Trans,k) = (\ (g^a,g^b),\ g^{ab})}$
- DDH means indistinguishable from $\mathbf{(\ (g^a,g^b),\ g^c)}$

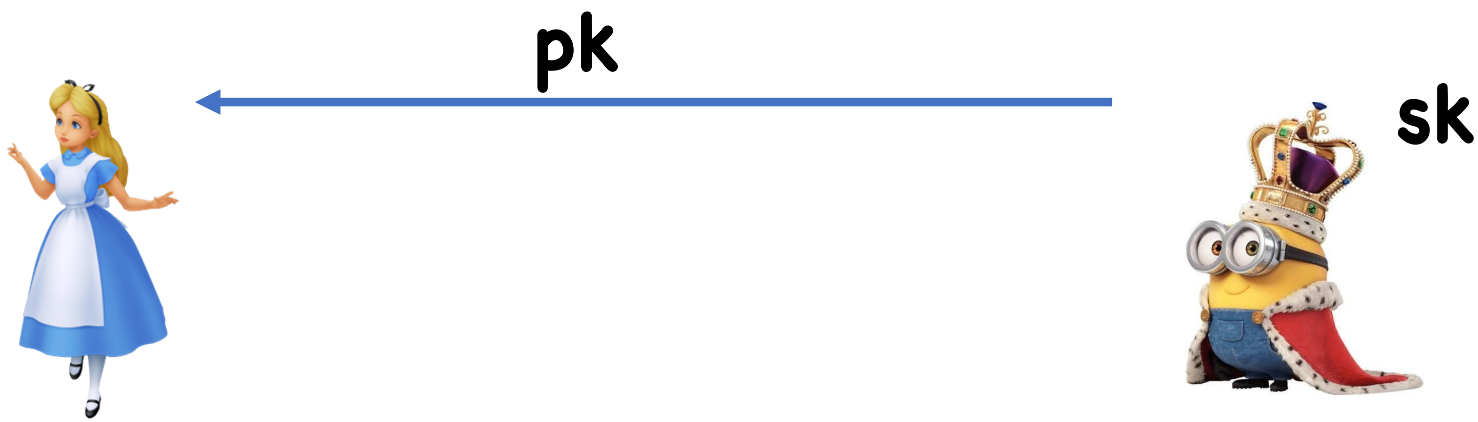
What if only CDH holds, but DDH is easy?

Today

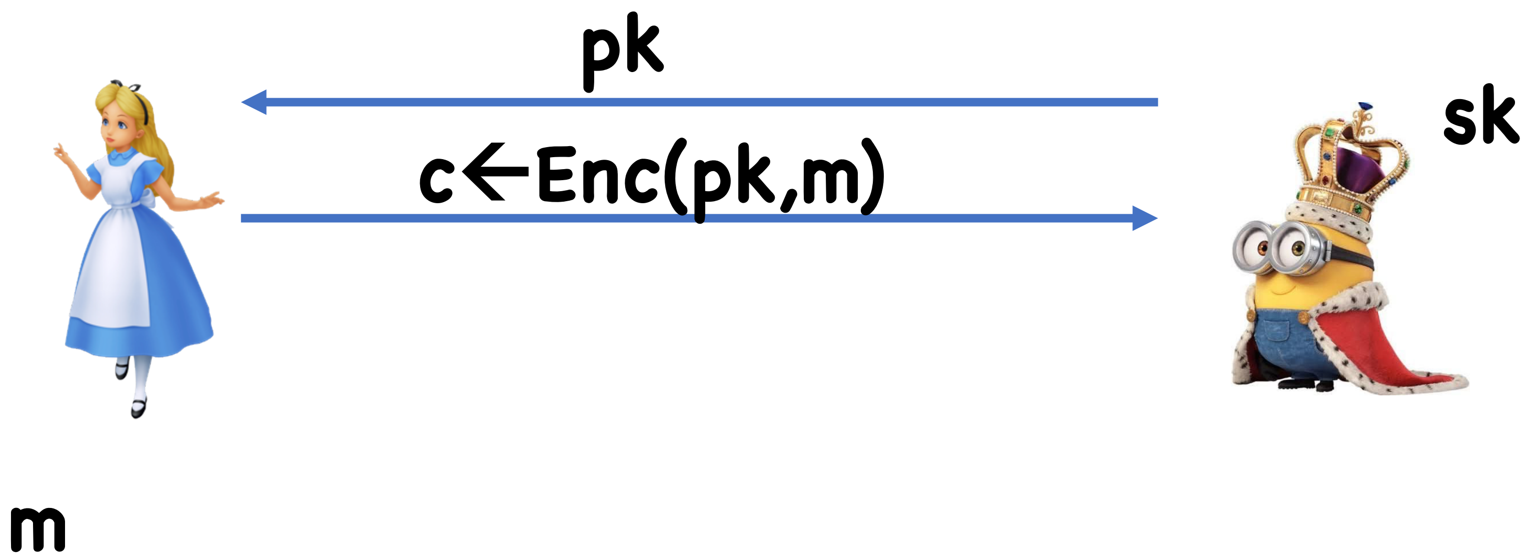
Public key encryption

Public Key Encryption

Public Key Encryption



Public Key Encryption



Public Key Encryption

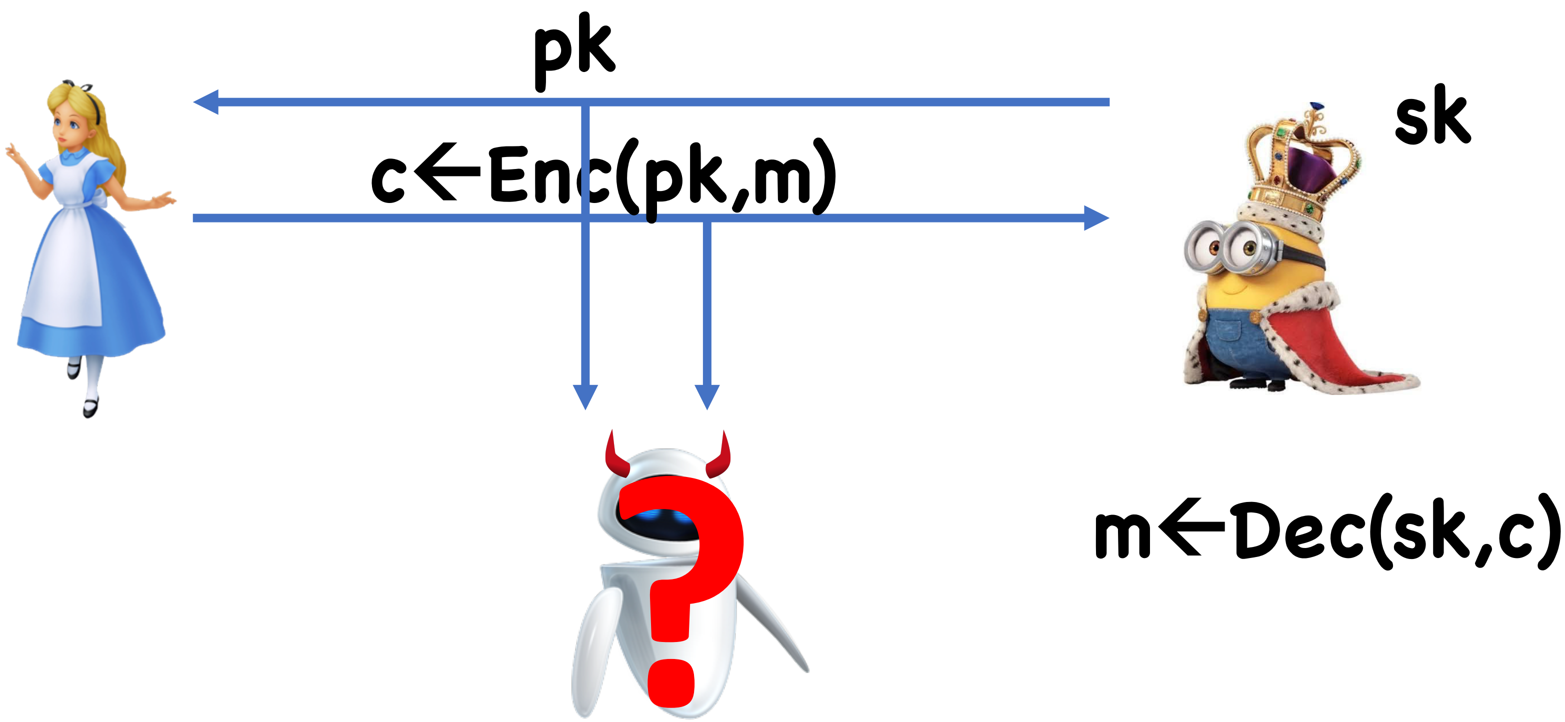
pk

c←Enc(pk,m)

sk

m←Dec(sk,c)

Public Key Encryption



PKE vs Key Agreement

Key agreement:

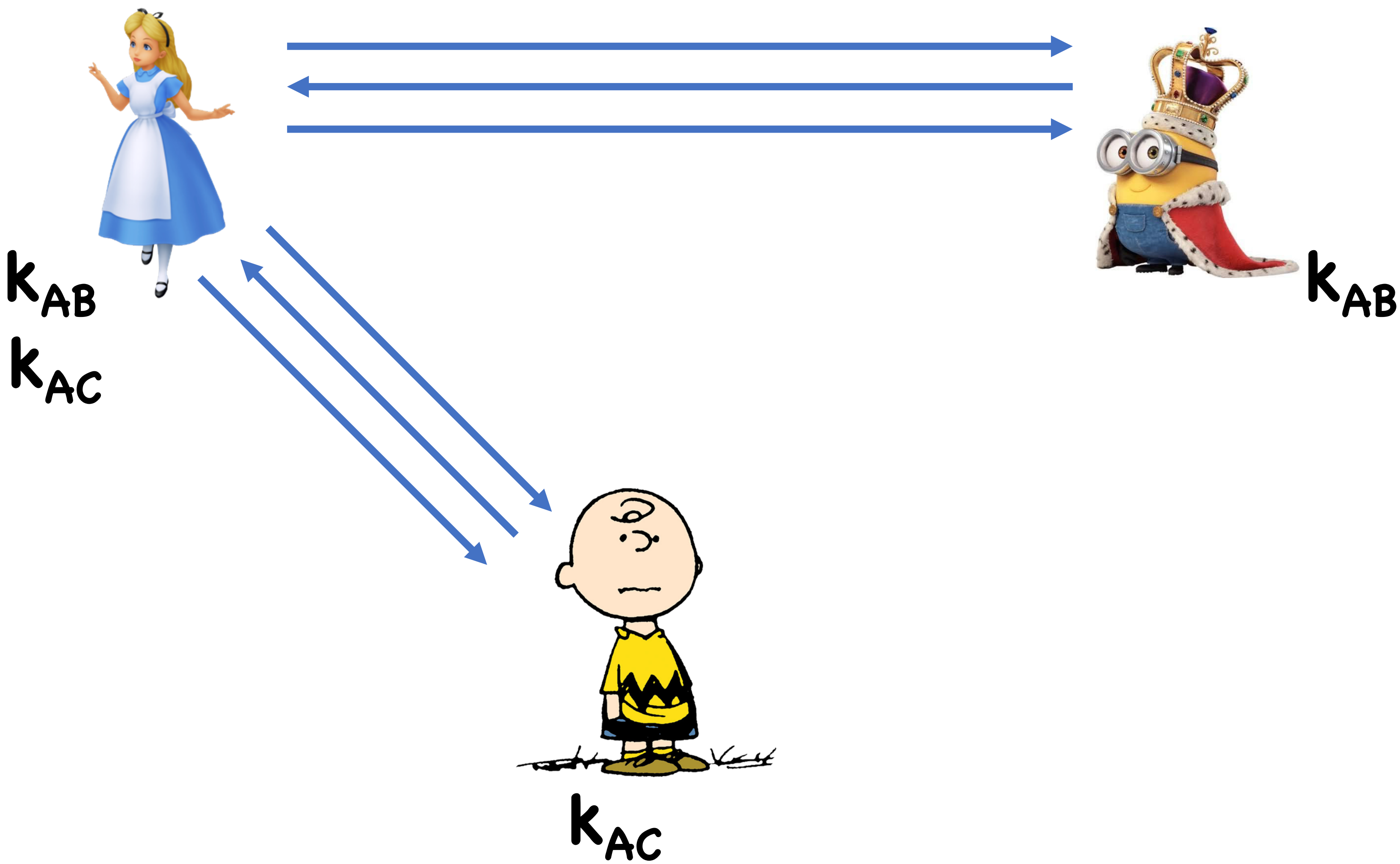
PKE vs Key Agreement

Key agreement:

k_{AB} k_{AB}

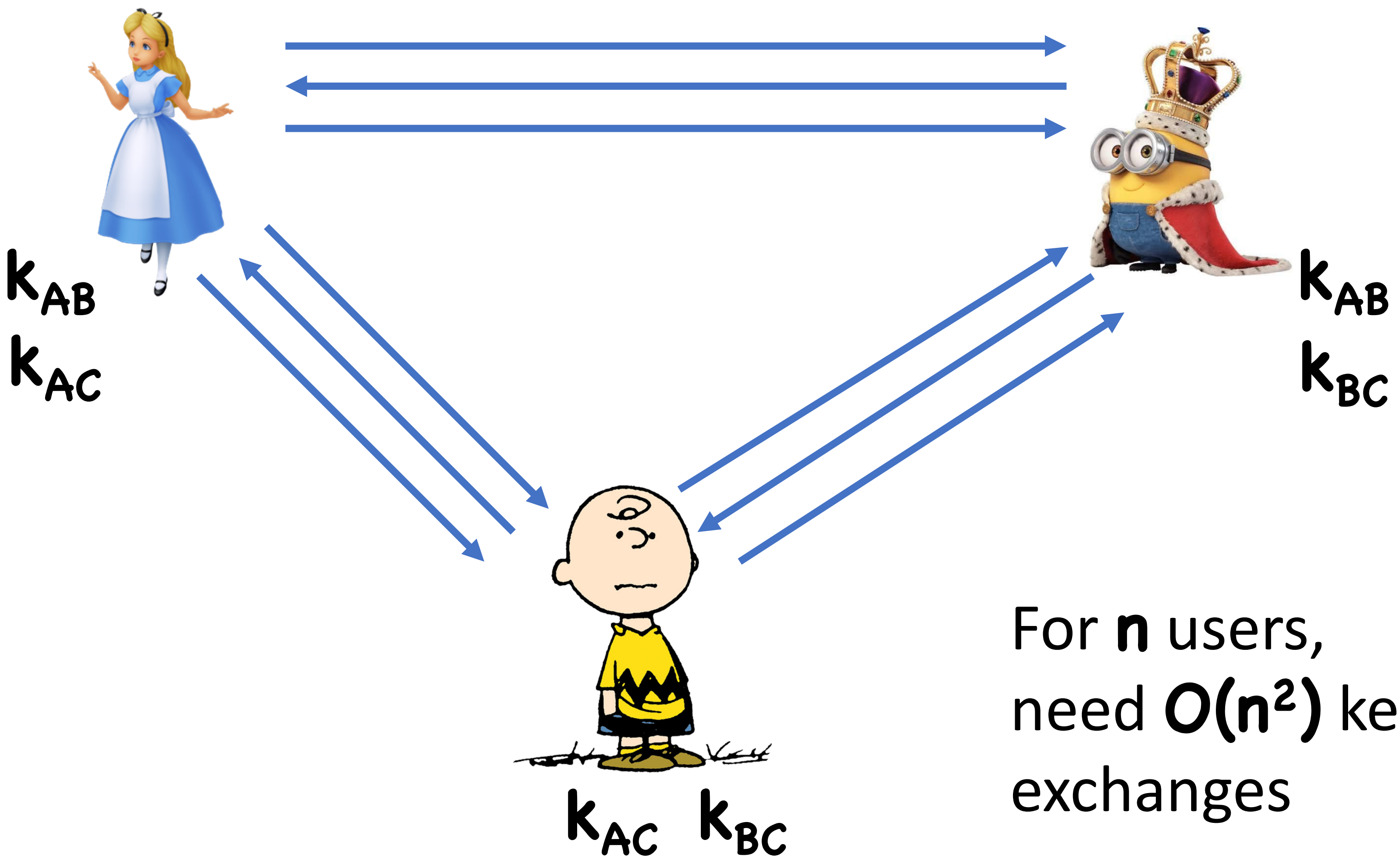
PKE vs Key Agreement

Key agreement:



PKE vs Key Agreement

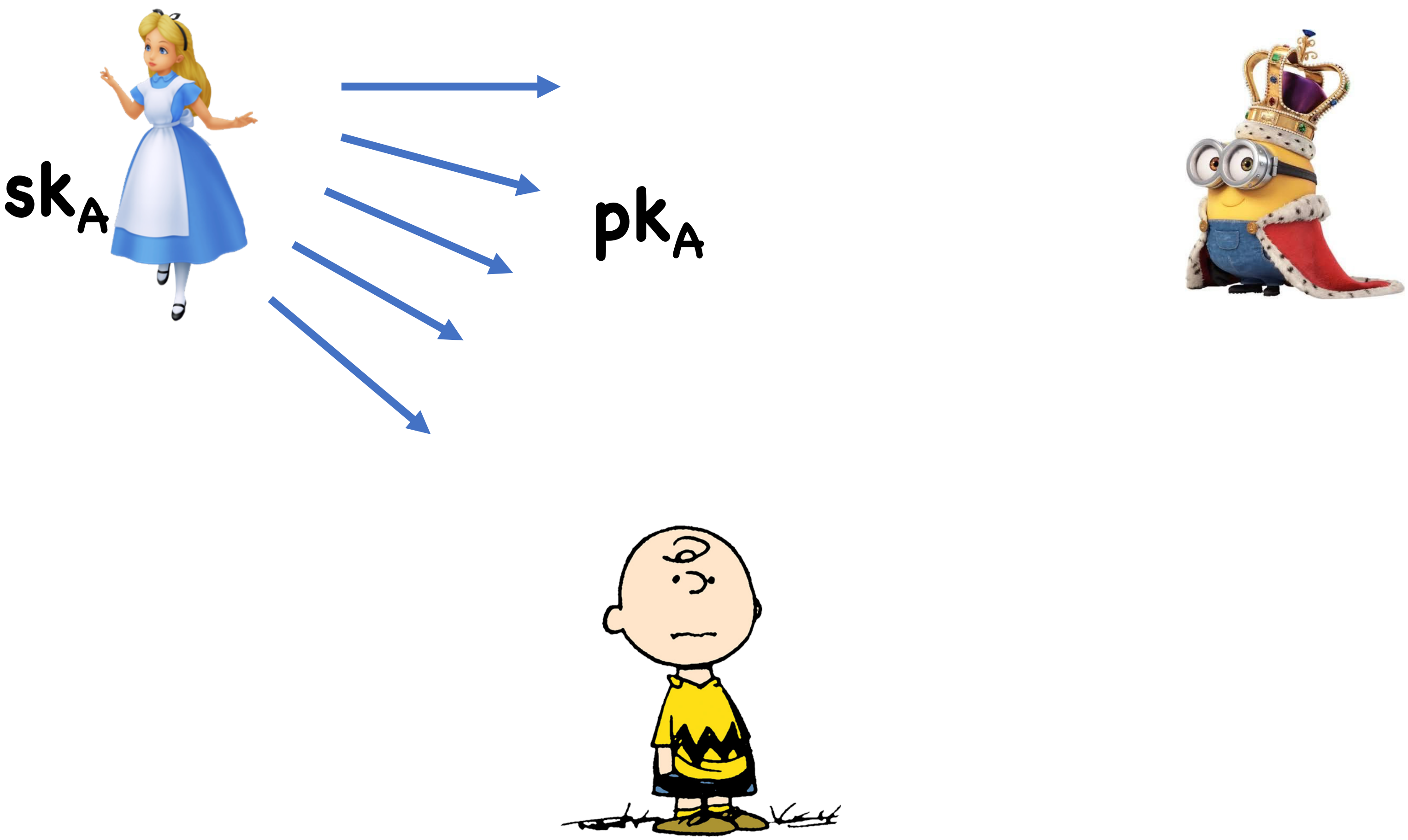
Key agreement:



For **n** users, need **O(n²)** key exchanges

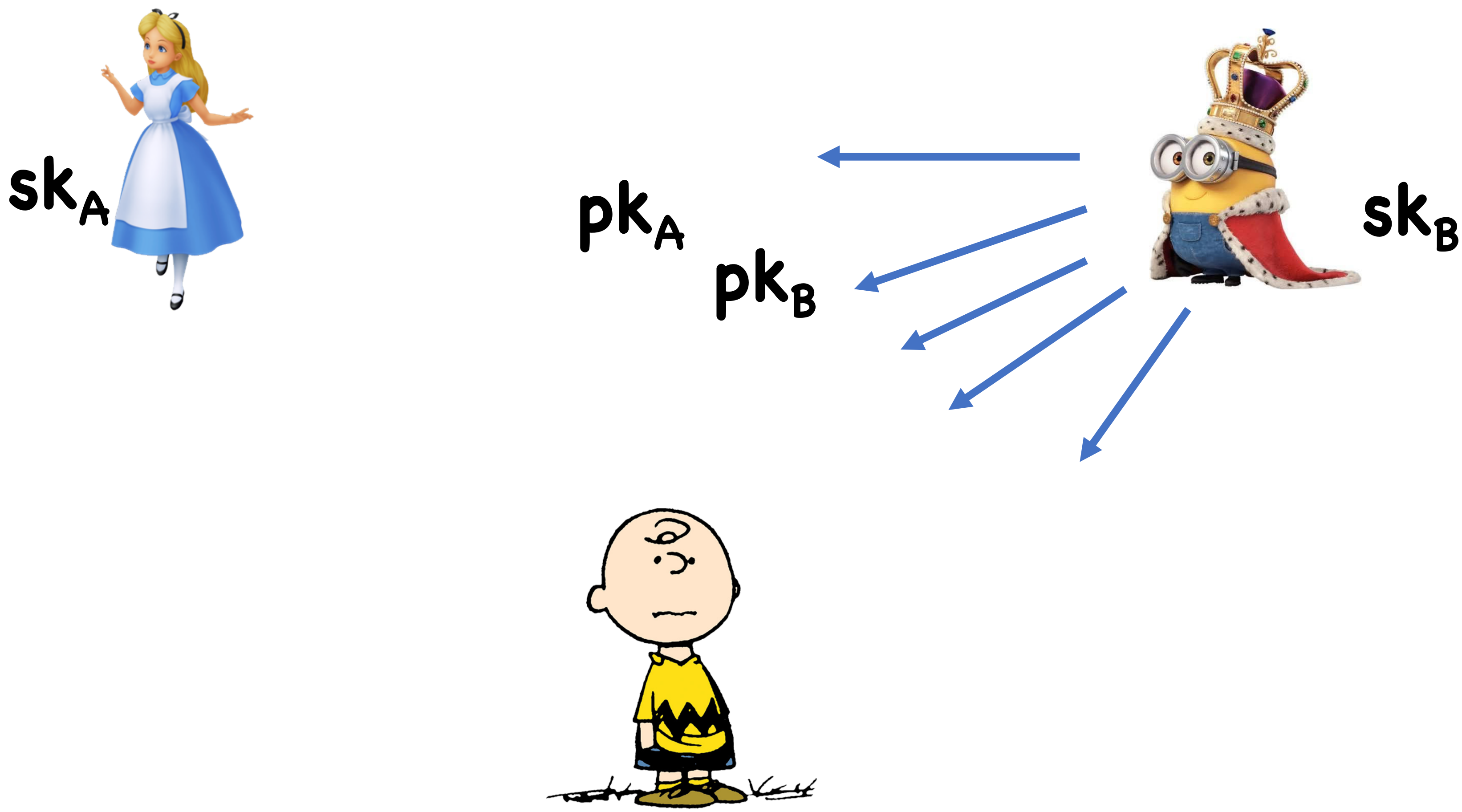
PKE vs Key Agreement

PKE:



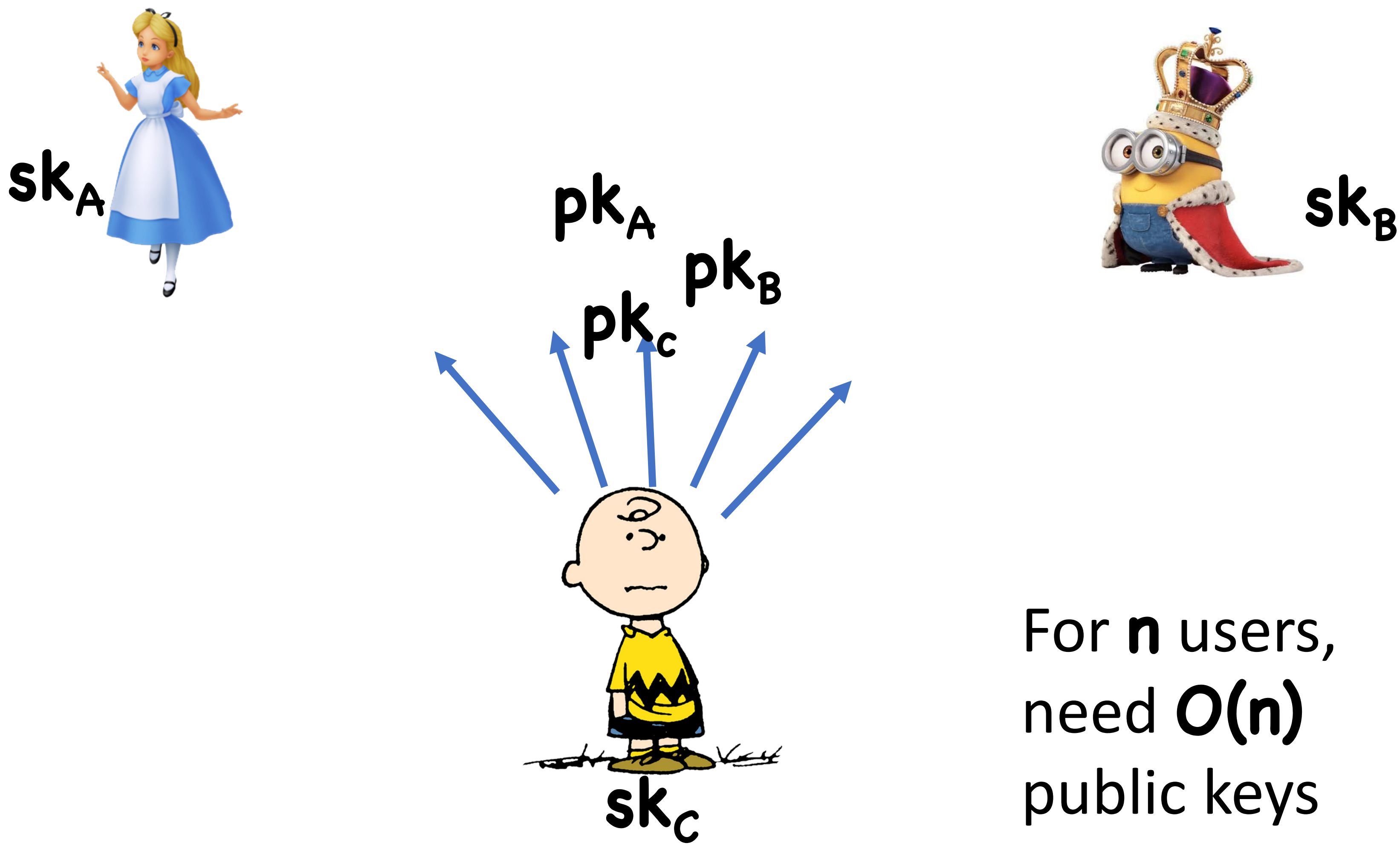
PKE vs Key Agreement

PKE:



PKE vs Key Agreement

PKE:



For **n** users, need **O(n)** public keys

PKE Syntax

Message space $\mathbf{M}$

Algorithms:

- **(sk,pk)←Gen(λ)**
- **Enc(pk,m)**
- **Dec(sk,m)**

Correctness:

Pr[Dec(sk,Enc(pk,m)) = m: (sk,pk)←Gen(λ)] = 1

Security

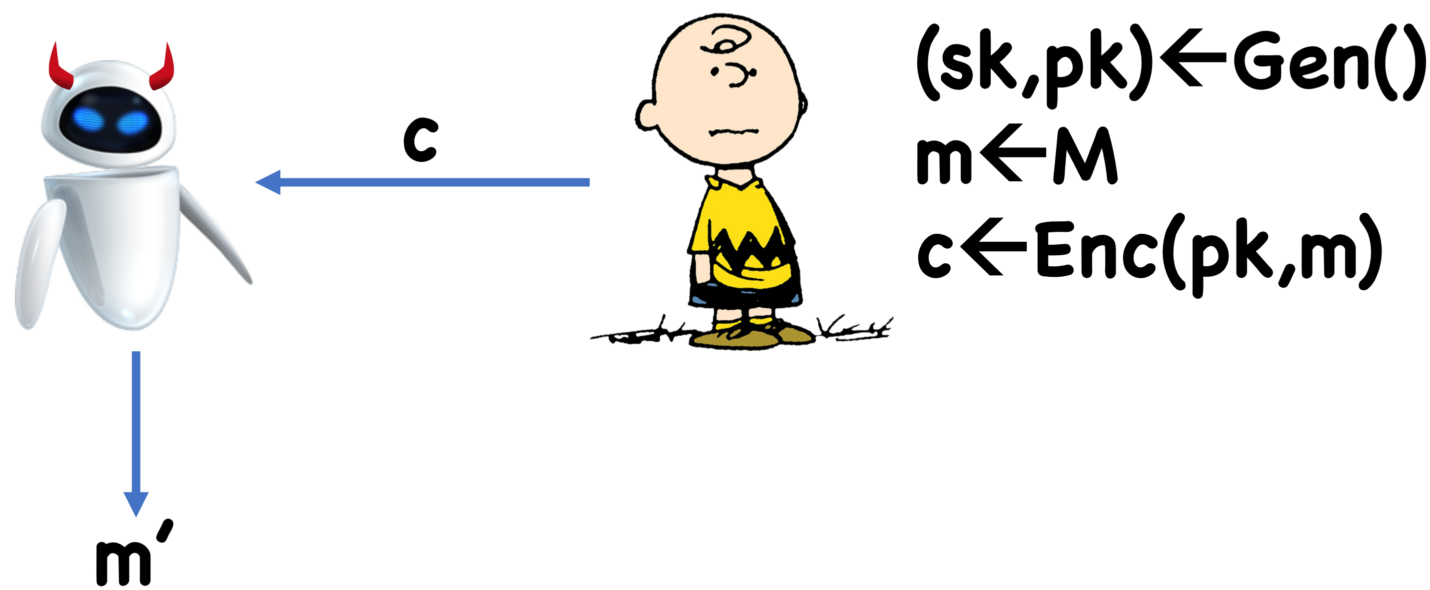
One-way security

Semantic Security

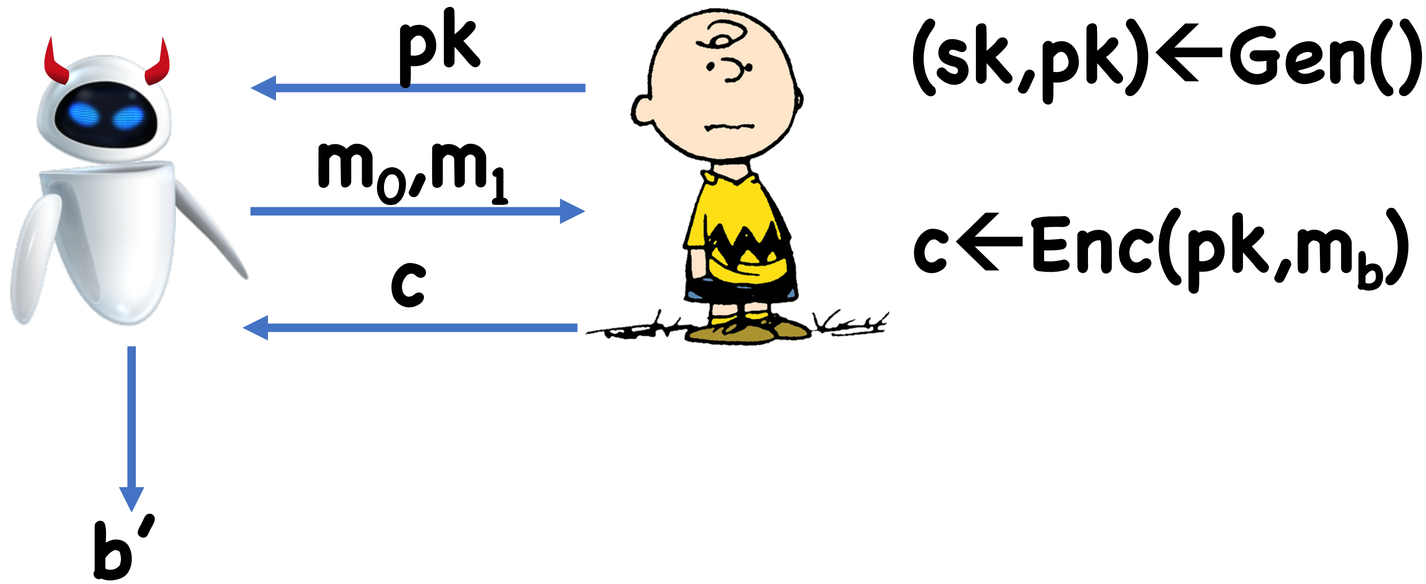
CPA security

CCA Security

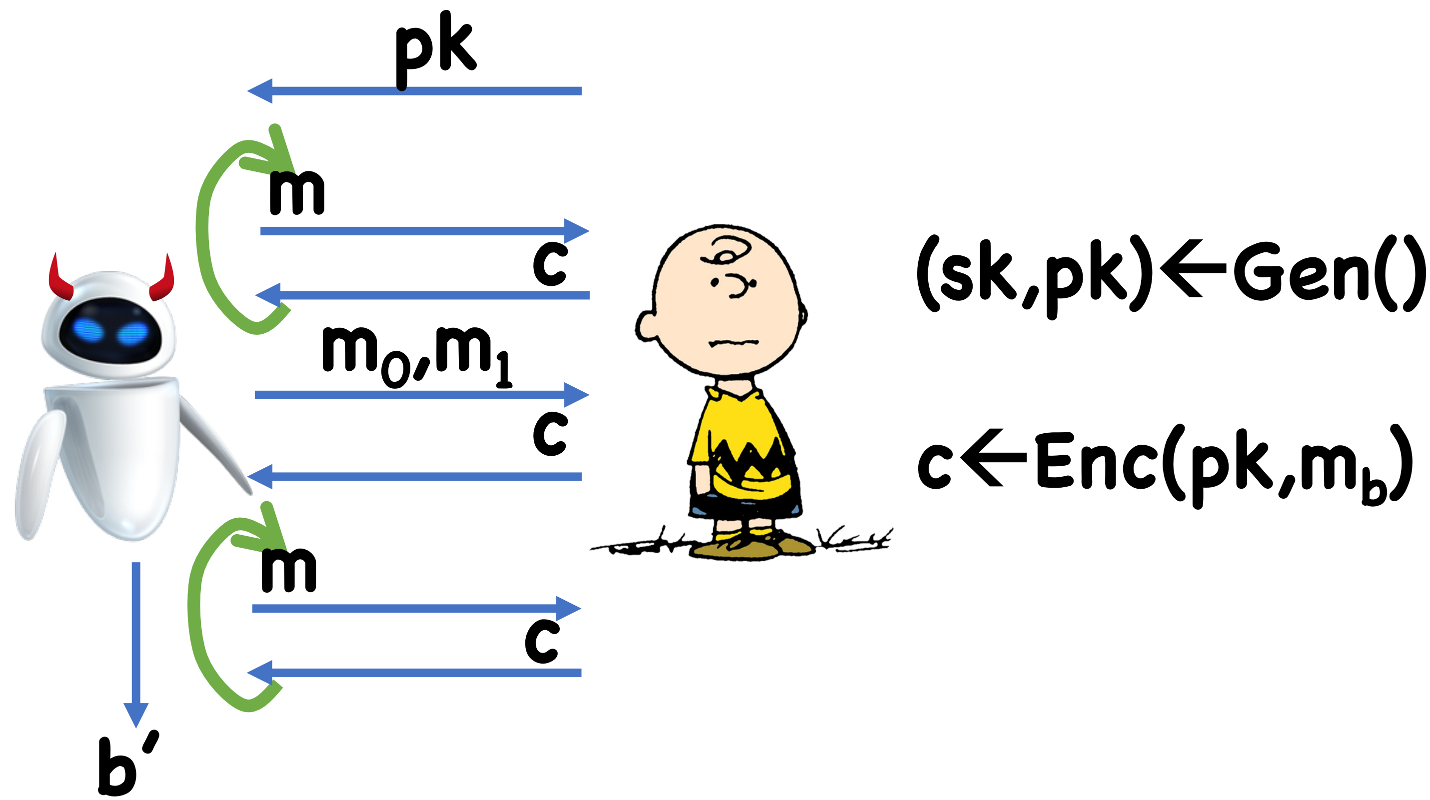
One-way Security



Semantic Security

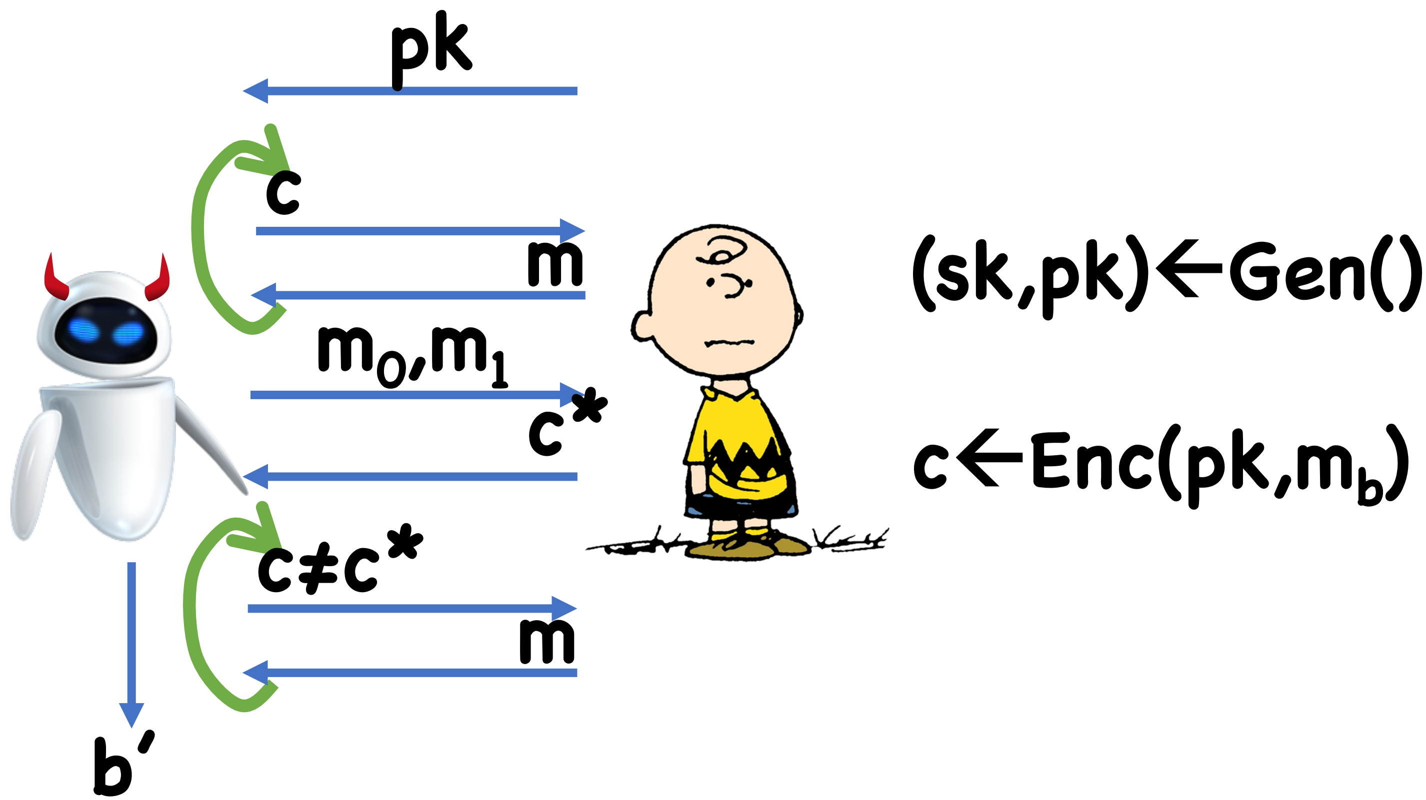


CPA Security



Theorem: An encryption scheme (Gen,Enc,Dec) is semantically secure if and only if it is CPA secure

CCA Security



One-way Encryption from RSA

Gen():

- Choose random primes **p,q**
- Let **N=pq**
- Choose **e,d** .s.t **ed=1 mod (p-1)(q-1)**
- Output **pk=(N,e), sk=(N,d)**

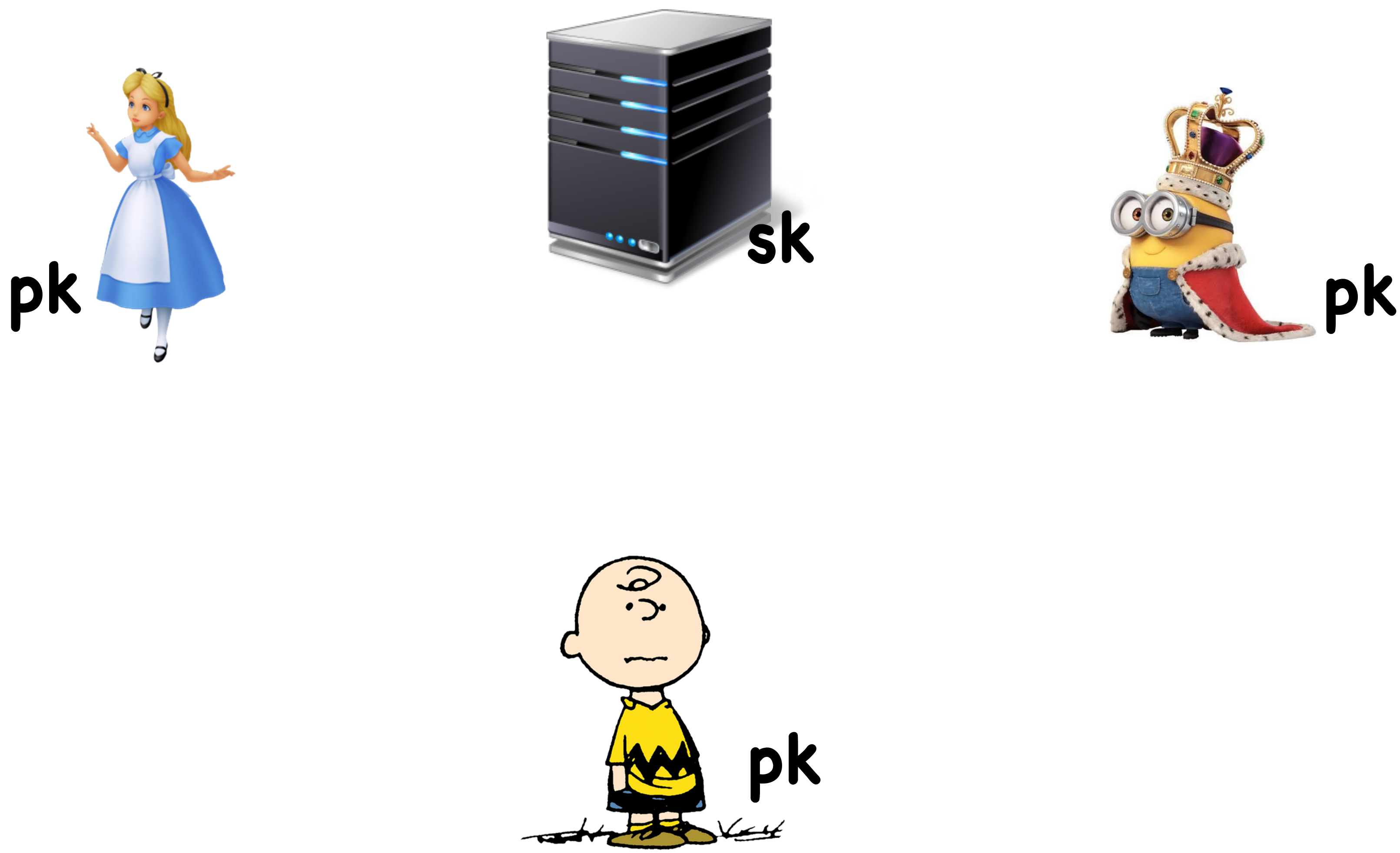
Enc(pk,m): Output $c = m^e \bmod N$

Dec(sk,c): Output $m' = c^d \bmod N$

Theorem: If the RSA one-way function is secure for ***e***, then the RSA encryption is one-way secure

Proof: adversary sees exactly output of one-way function and is asked to invert

Considerations



Considerations

A single server often has to decrypt many ciphertexts, whereas each user only encrypts a few messages

Therefore, would like to make decryption fast

Considerations

Encryption running time:

- $O(\log e)$ multiplications, each taking $O(\log^2 N)$
- Overall $O(\log e \log^2 N)$

Decryption running time:

- $O(\log d \log^2 N)$

(Note that $ed \geq \Phi(N) \approx N$)

Considerations

Possibilities:

- $\mathbf{e}$ tiny (e.g. $\mathbf{3}$): fast encryption, slow decryption
- $\mathbf{d}$ tiny (e.g. $\mathbf{3}$): fast decryption, slow encryption
  - Problem?
- $\mathbf{d}$ relatively small (e.g. $\mathbf{d \approx N^{0.1}}$)
  - Turns out, there is an attack that works whenever $\mathbf{d < N^{.292}}$

Therefore, need $\mathbf{d}$ to be large, but ok taking $\mathbf{e=3}$

Considerations

Chinese remaindering to speed up decryption:

- Let $\mathbf{sk=(d_0,d_1)}$ where

$$d_0 = d \bmod (p-1),\ d_1 = d \bmod (q-1)$$

- Let $c_0 = c \bmod p,\ c_1 = c \bmod q$
- Compute $m_0 = c^{d0} \bmod p,\ m_1 = c^{d1} \bmod q$
- Reconstruct $\mathbf{m}$ from $m_0,\ m_1$

Running time:

- $r \log^3 p + r \log^3 q + O(\log^2 N) \approx r(\log^3 N)/4$

CPA security from RSA?

Use hardcore bit for RSA func:

- **Enc(pk,m):** $r \leftarrow \mathbb{Z}_N^*$, $c=(r^e \bmod N,\ h(r)\oplus m)$

> **Theorem:** If RSA is one-way and h is hardcore for RSA, then this encryption scheme is CPA secure

Proof:

$$(r^e \bmod N,\ h(r)\oplus m_0) \approx (r^e \bmod N,\ b\oplus m_0)$$
$$\approx (r^e \bmod N,\ b\oplus m_1) \approx (r^e \bmod N,\ h(r)\oplus m_1)$$

Goldwasser-Micali

Gen():

- Choose random primes **p,q**
- Let **N=pq**
- Choose **x** a quadratic non-residue mod **p** and **q**
- Output **pk=(N,x), sk=(p,q)**

Enc(pk,m$\in${0,1}): $r \leftarrow \mathbb{Z}_N^*$, $c \leftarrow x^m r^2 \bmod N$

- If **m=0**, then c is a quadratic residue
- If **m=1**, then c is a non-residue

Determining Residues

Let $\mathbf{c} \in \mathbb{Z}_\mathbf{p}^*$ for a prime $\mathbf{p}$

How to test if $\mathbf{c}$ is a quadratic residue?

Let $\mathbf{c} \in \mathbb{Z}_\mathbf{N}^*$ for $\mathbf{N=pq}$

- If you know $\mathbf{p}$ and $\mathbf{q}$, test for residuosity mod $\mathbf{p}$ and $\mathbf{q}$ $\Rightarrow$ QR mod $\mathbf{N}$ iff QR mod both $\mathbf{p}$ and $\mathbf{q}$
- If you don't know factors, presumed hard

Definition: The Quadratic Residuosity problem mod **N=pq** is to distinguish a random QR from a random x that is not a QR mod **p** or mod **q**

Theorem: If the QR problem is hard, then Goldwasser Micali is CPA secure

Theorem: If the QR problem is hard, then Goldwasser Micali is CPA secure

Proof:

- Hybrid 0: **pk** is honestly generated, encrypt $\mathbf{m_0}$
- Hybrid 1: **pk** is a random QR, encrypt $\mathbf{m_0}$
- Hybrid 2: **pk** is a random QR, encrypt $\mathbf{m_1}$
- Hybrid 3: **pk** is honestly generated, encrypt $\mathbf{m_1}$

Bit encryption $\Rightarrow$ Multi-bit

Let **Gen,Enc,Dec** be an encryption scheme for bits

Gen'() = Gen()
Enc'(pk, (m_1,…,m_n)) = (Enc(pk,m_1),…,Enc(pk,m_n))
Dec'(sk, (c_1,…,c_n)) = (Dec(sk,c_1),…,Dec(sk,c_n))

Theorem: If **(Gen,Enc,Dec)** is CPA secure, then so is **(Gen',Enc',Dec')**

ElGamal

Group $\mathbf{G}$ of order $\mathbf{p}$, generator $\mathbf{g}$
Message space = $\mathbf{G}$

Gen():
- Choose random $\mathbf{a} \leftarrow \mathbb{Z}_p^*$, let $\mathbf{h} \leftarrow \mathbf{g}^{\mathbf{a}}$
- $\mathbf{pk=h,\ sk=a}$

Enc(pk,m$\in${0,1}):
- $\mathbf{r} \leftarrow \mathbb{Z}_p$
- $\mathbf{c} = (\mathbf{g}^{\mathbf{r}}, \mathbf{h}^{\mathbf{r}} \times \mathbf{m})$

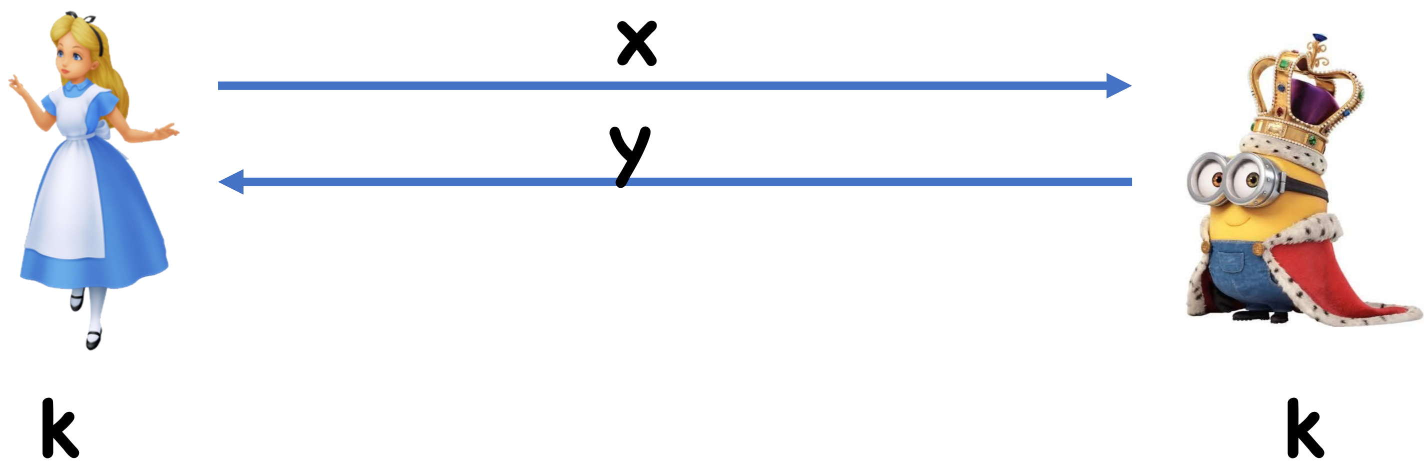
Dec?

Theorem: If DDH is hard in $\mathbf{G}$, then ElGamal is CPA secure

Proof:

- Adversary sees $\mathbf{h=g^a, g^r, g^{ar}\times m_0}$
- DDH: indistinguishable from $\mathbf{g^a, g^r, g^c\times m_0}$
- Same as $\mathbf{g^a, g^r, g^c\times m_1}$
- DDH again: indistinguishable from $\mathbf{g^a, g^r, g^{ar}\times m_0}$

PKE from One-Round Key Exchange



PKE from One-Round Key Exchange

$x = A()$ →

← $y = B(x)$

$k = A(\text{state}_A, x, y)$

$k = B(\text{state}_B, x)$

Here, $\textbf{state}_A$, $\textbf{state}_B$, are the internal states of **A,B** after first message

PKE from One-Round Key Exchange

Gen(): Run **A()**, getting **x**, and $\mathbf{state_A}$

- $\mathbf{sk = (x, state_A),\ pk = x}$

Enc(pk,m):

- Run **B(x)** to get **y** and $\mathbf{state_B}$,
- Run $\mathbf{B(state_B,\ x)}$ to get **k**
- $\mathbf{c = (y,\ k \oplus m)}$

Dec(sk, (y,d)):

- Run $\mathbf{A(state_A,\ x,\ y)}$ to get **k**
- $\mathbf{m \leftarrow d \oplus k}$

PKE from One-Round Key Exchange

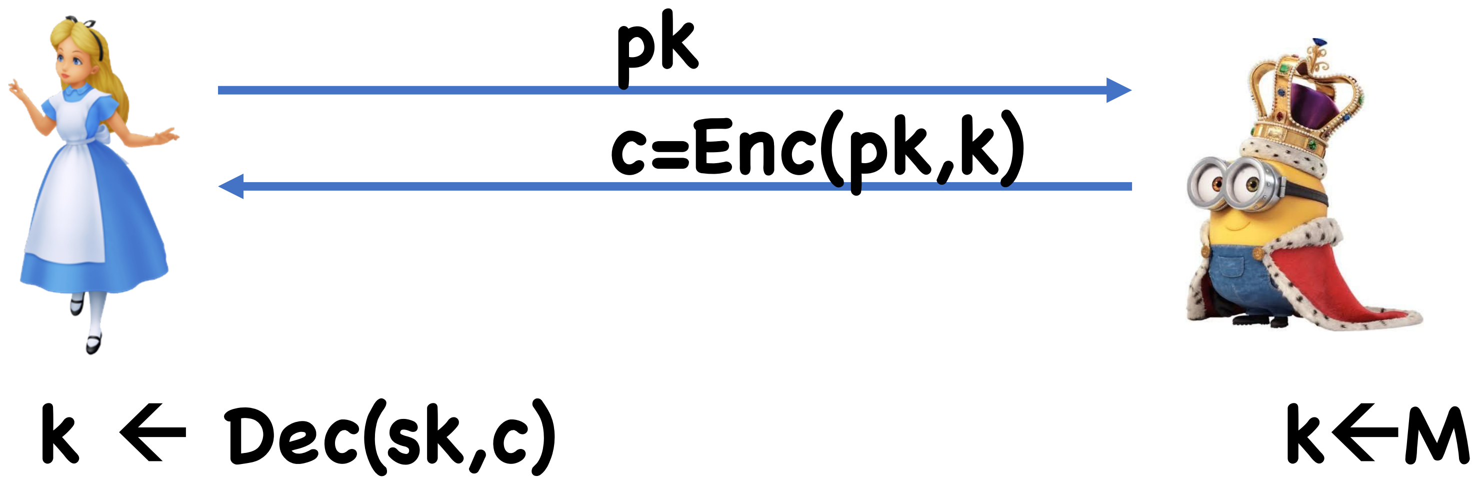
Theorem: If (A,B) is a secure one-round key exchange protocol, then **(Gen,Enc,Dec)** is CPA secure

Proof:

(pk, c) = (x,y,d) is exactly what the adversary would see if:

- Run key agreement protocol to get **k**
- Encrypt **m** using **k** as OTP

One-Round Key Exchange from PKE



Black Box Separations

Recall: hard to build key agreement from one-way functions

Therefore, also hard to build PKE from one-way functions

Appears we must rely on number theory for PKE

Practical Considerations

Number theory is computationally expensive
- Need big number arithmetic

Symmetric crypto (e.g. block ciphers) much faster

Want to minimize use of number theory, and rely mostly on symmetric crypto

Hybrid Encryption

Let $(\mathbf{Gen}_{PKE}, \mathbf{Enc}_{PKE}, \mathbf{Dec}_{PKE})$ be a PKE scheme, $(\mathbf{Enc}_{SKE}, \mathbf{Dec}_{SKE})$ a SKE scheme

$\mathbf{Gen}() = \mathbf{Gen}_{PKE}()$
$\mathbf{Enc}(pk, m)$: $k \leftarrow K$, $c = (\mathbf{Enc}_{PKE}(pk,k), \mathbf{Enc}_{SKE}(k,m))$
$\mathbf{Dec}(sk, (c_0, c_1))$:

- $k \leftarrow \mathbf{Dec}_{PKE}(sk, c_0)$
- $m \leftarrow \mathbf{Dec}_{SKE}(k, c_1)$

Now PKE used to encrypt something small (e.g. 128 bits), SKE used to encrypt actual message (say, GB's)

Hybrid Encryption

Theorem: If $(\mathbf{Gen}_{PKE}, \mathbf{Enc}_{PKE}, \mathbf{Dec}_{PKE})$ is CPA secure and $(\mathbf{Enc}_{SKE}, \mathbf{Dec}_{SKE})$ is one-time secure, then $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ is CPA secure

Hybrid 0: $(\mathbf{Enc}_{PKE}(pk,k),\ \mathbf{Enc}_{SKE}(k,m_0))$
Hybrid 1: $(\mathbf{Enc}_{PKE}(pk,k'),\ \mathbf{Enc}_{SKE}(k,m_0))$
Hybrid 2: $(\mathbf{Enc}_{PKE}(pk,k'),\ \mathbf{Enc}_{SKE}(k,m_1))$
Hybrid 3: $(\mathbf{Enc}_{PKE}(pk,k),\ \mathbf{Enc}_{SKE}(k,m_1))$

Next Time

Trapdoor Permutations

Begin: digital signatures (aka public key MACs)