COS433/Math 473: Cryptography

Mark Zhandry

Princeton University

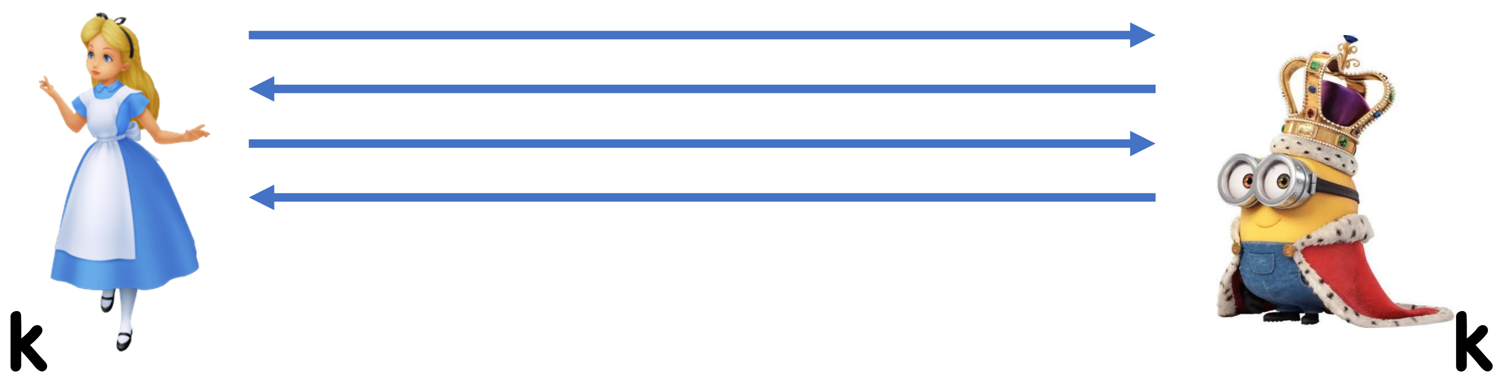
Spring 2017

Previously…

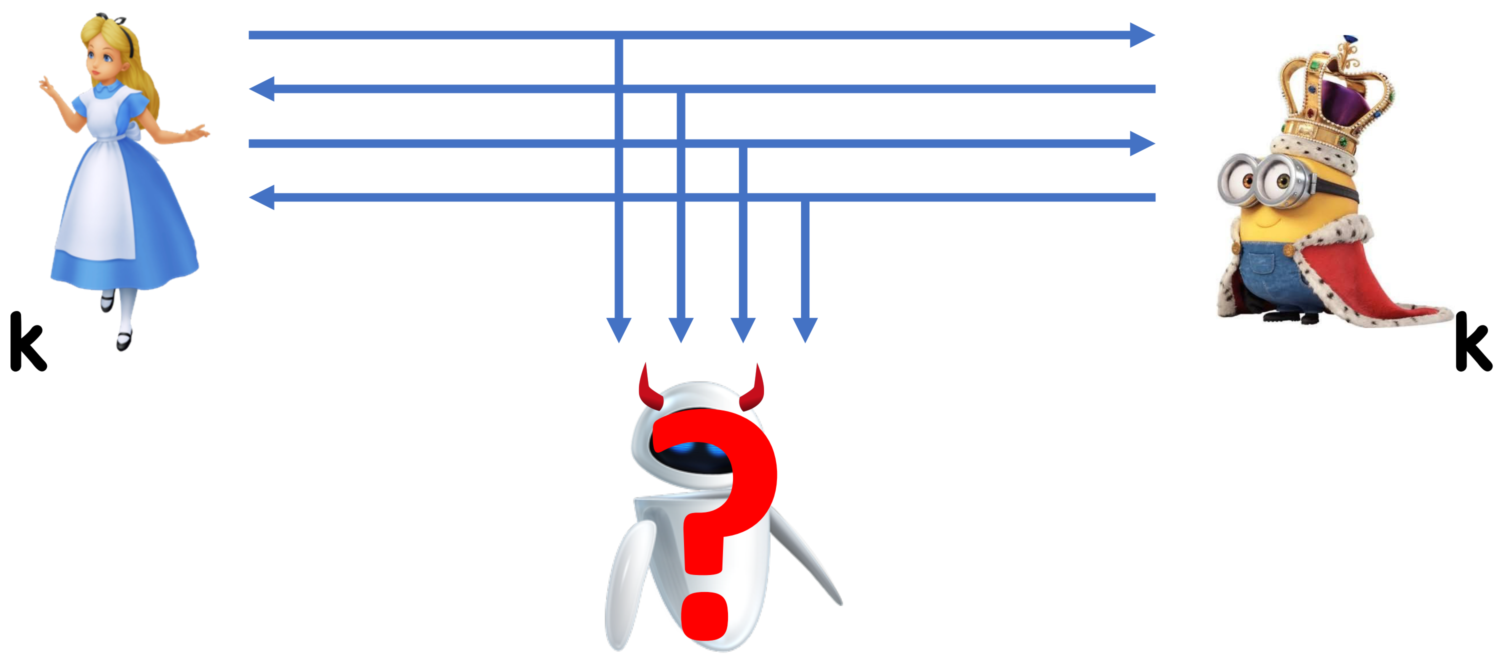
Public Key Distribution

Public Key Distribution

Public Key Distribution



Public Key Distribution



Public Key Distribution

Pair of interactive algorithms **A,B**

Correctness:

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

Shared key is $k := o_A=o_B$

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

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

Trapdoor Permutations

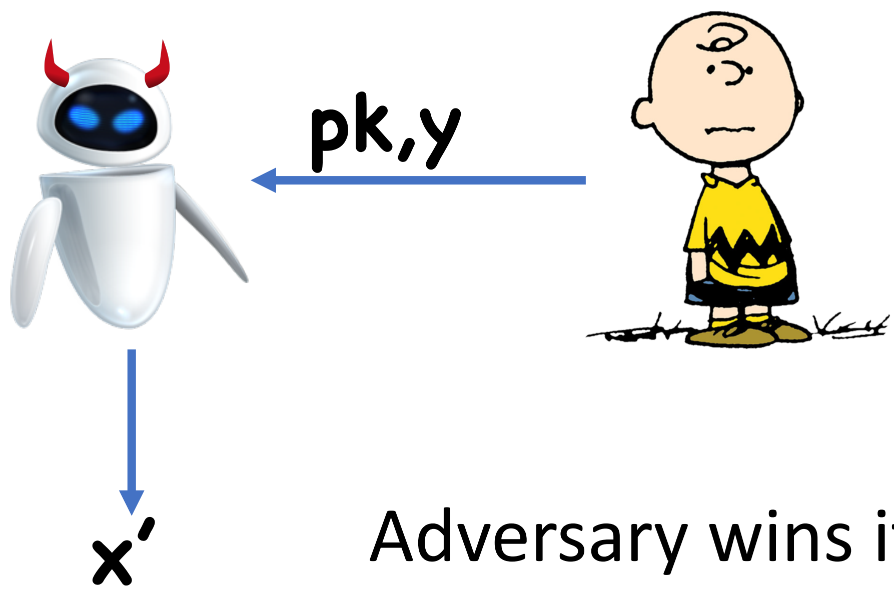
Domain $\mathbf{X}$

Gen(): outputs **(pk,sk)**
$F(pk, x \in X) = y \in X$
$F^{-1}(sk, y) = x$

Correctness:
$\Pr[\ F^{-1}(sk, F(pk, x)) = x : (pk,sk) \leftarrow Gen()\ ] = 1$

Correctness implies F, F^{-1} are deterministic, permutations

Trapdoor Permutation Security

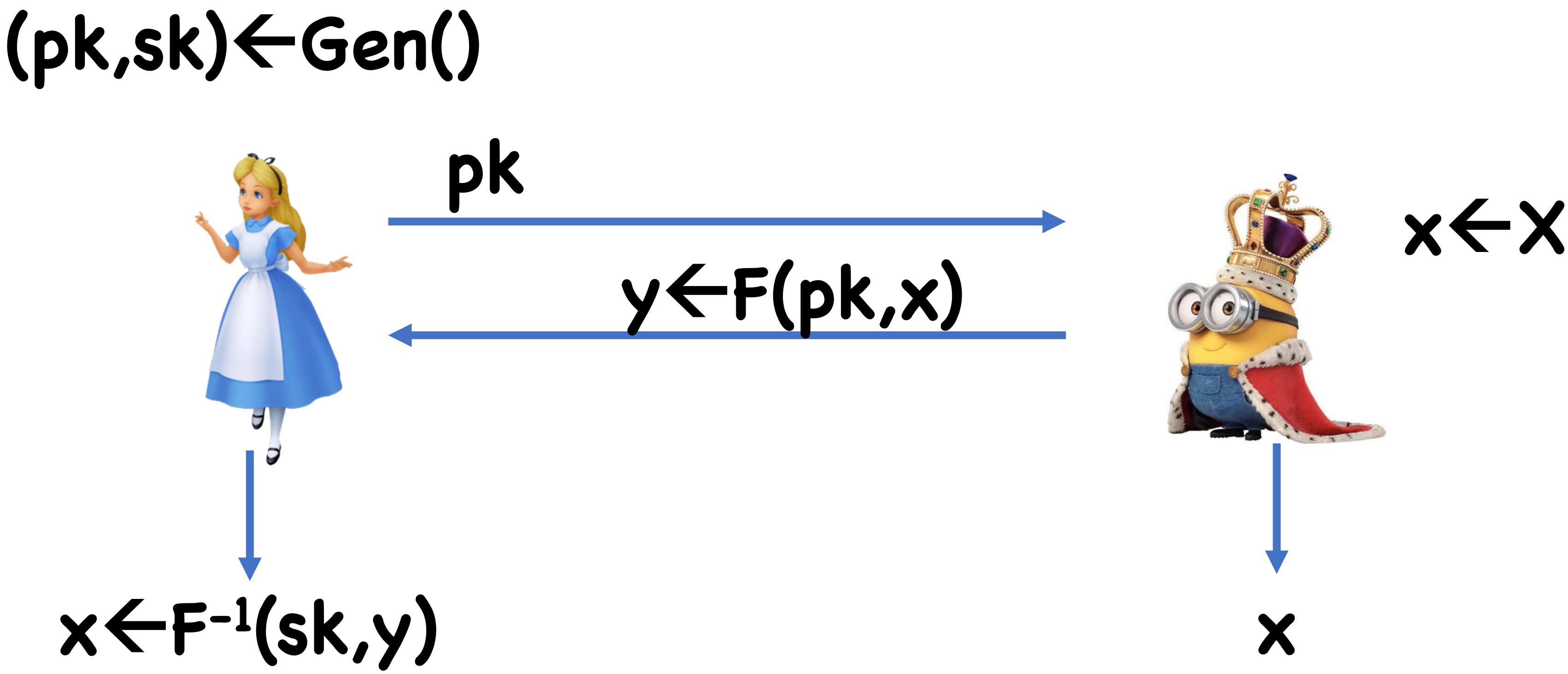


(sk,pk)←Gen()
x←X
y←F(pk,x)

Adversary wins if **x=x′**

In other words, **F(pk, ·)** is a one-way function

Key Distribution from TDPs



Key Distribution from TDPs

(pk,sk)←Gen()

pk

y←F(pk,x)

x←X

x←h(F^{-1}(sk,y))

h(x)

h a hardcore bit for **F(pk, ·)**

Trapdoor Permutations 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)**

F(pk,x): Output $y = x^e \bmod N$

F^{-1}(sk,c): Output $x = y^d \bmod N$

Caveats

RSA is not a true TDP as defined

- Why???
- What's the domain?

Nonetheless, distinction is not crucial to most applications

- In particular, works for key agreement protocol

Key Distribution from DH

Everyone agrees on group **G** of 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**

$a \leftarrow \mathbb{Z}_p$

$g^a \rightarrow$

$\leftarrow g^b$

$b \leftarrow \mathbb{Z}_p$

Key Distribution from DH

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

$a \leftarrow \mathbb{Z}_p$

$g^a \rightarrow$

$\leftarrow g^b$

$b \leftarrow \mathbb{Z}_p$

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

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

Key Distribution from DH

Theorem: If **(t,ε)–**DDH holds on **G**, then the Diffie-Hellman protocol is **(t,ε)–**secure

Proof:

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

What if only CDH holds, but DDH is easy?

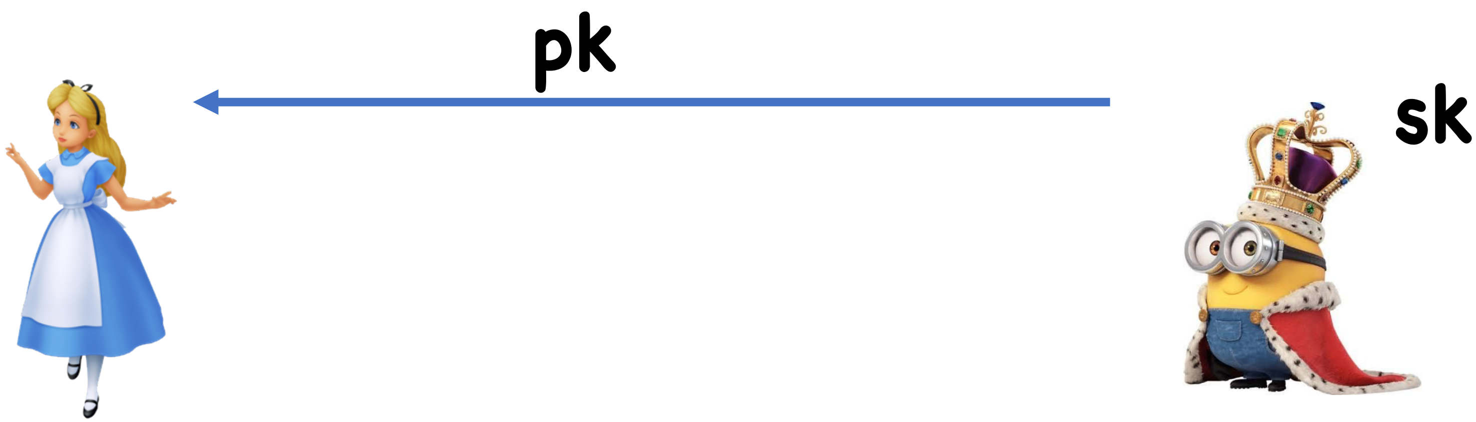
Today

Public key encryption

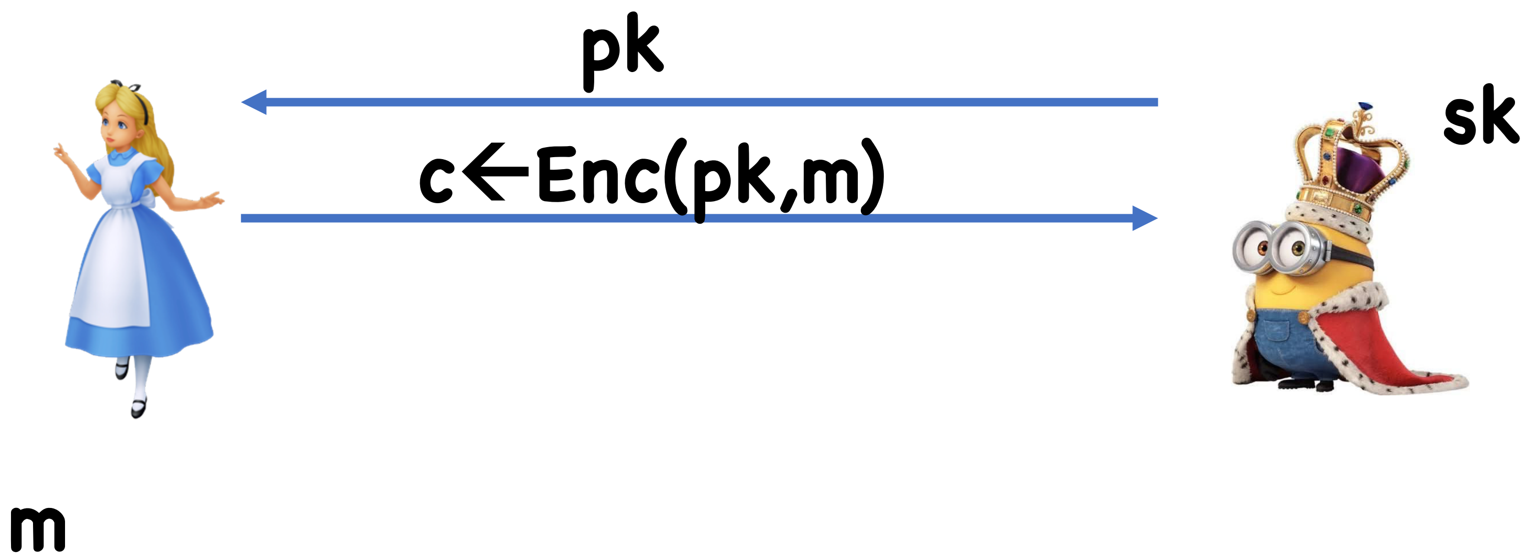
- Removes need to key exchange in the first place

Public Key Encryption

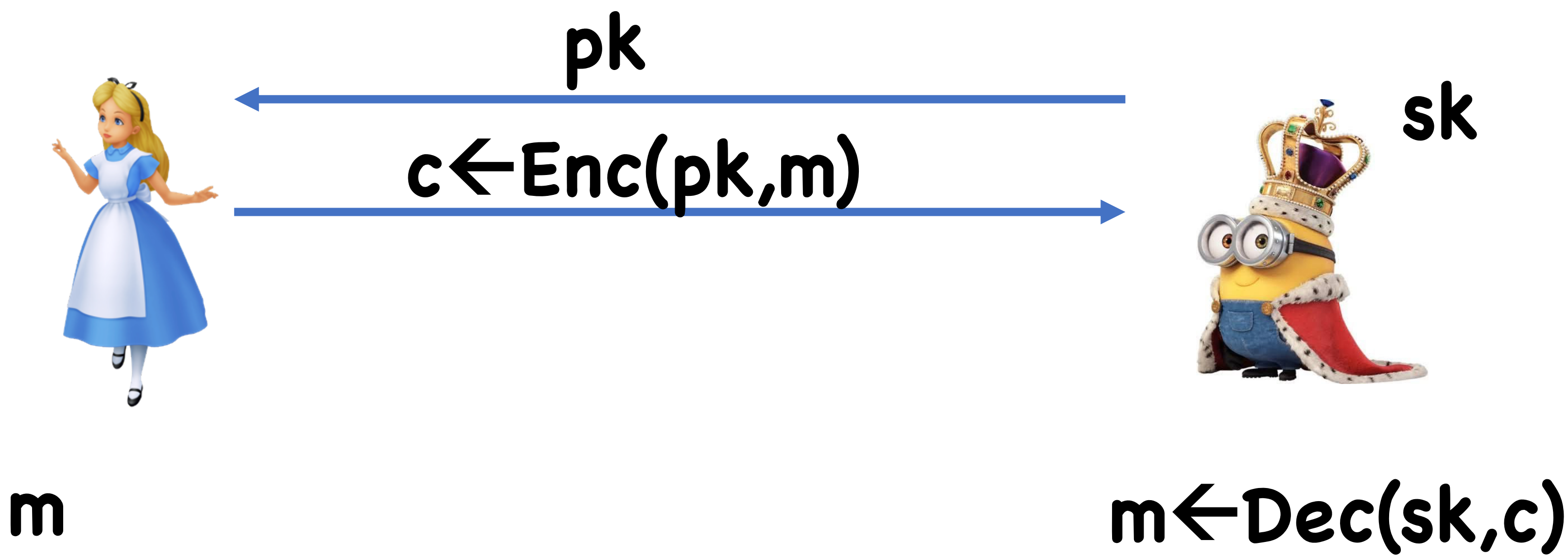
Public Key Encryption



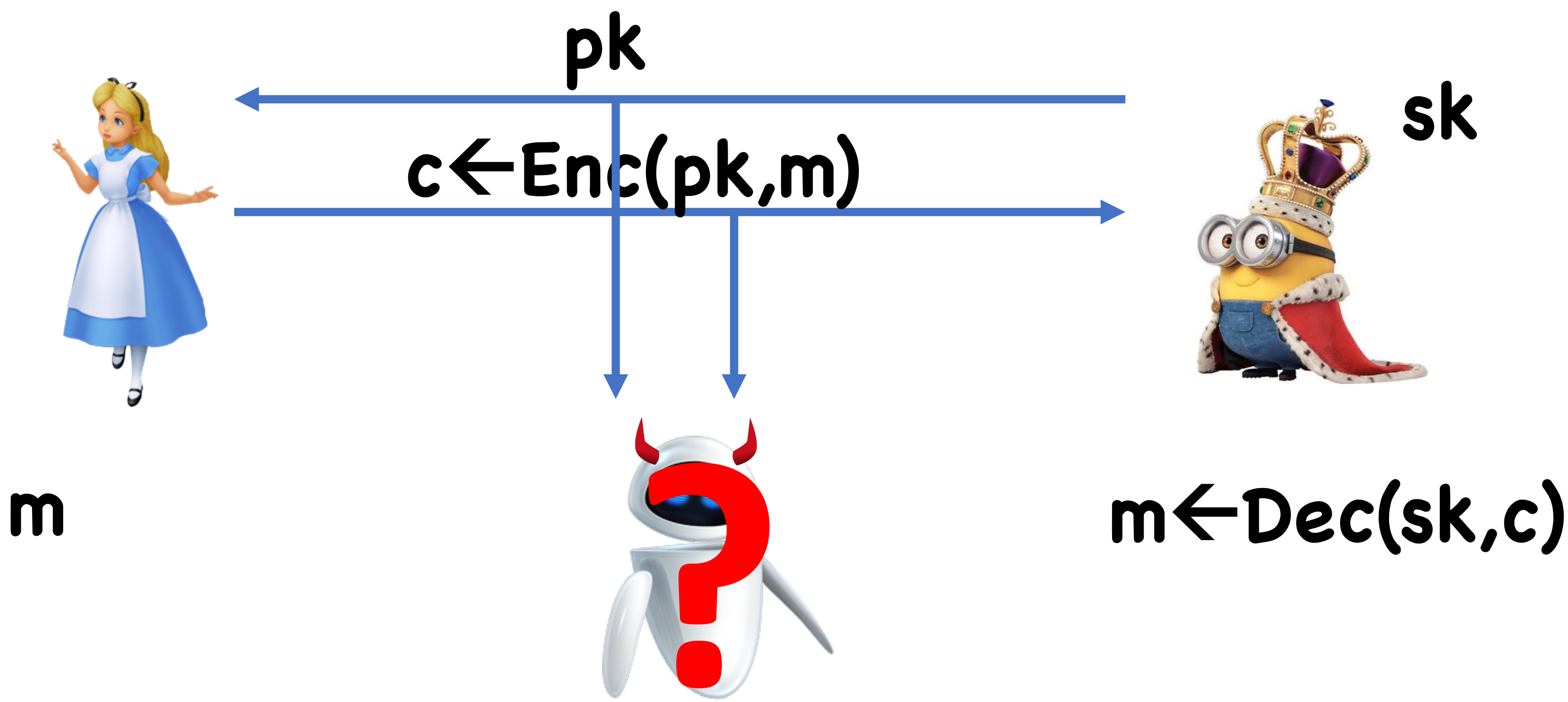
Public Key Encryption



Public Key Encryption



Public Key Encryption



PKE Syntax

Message space **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

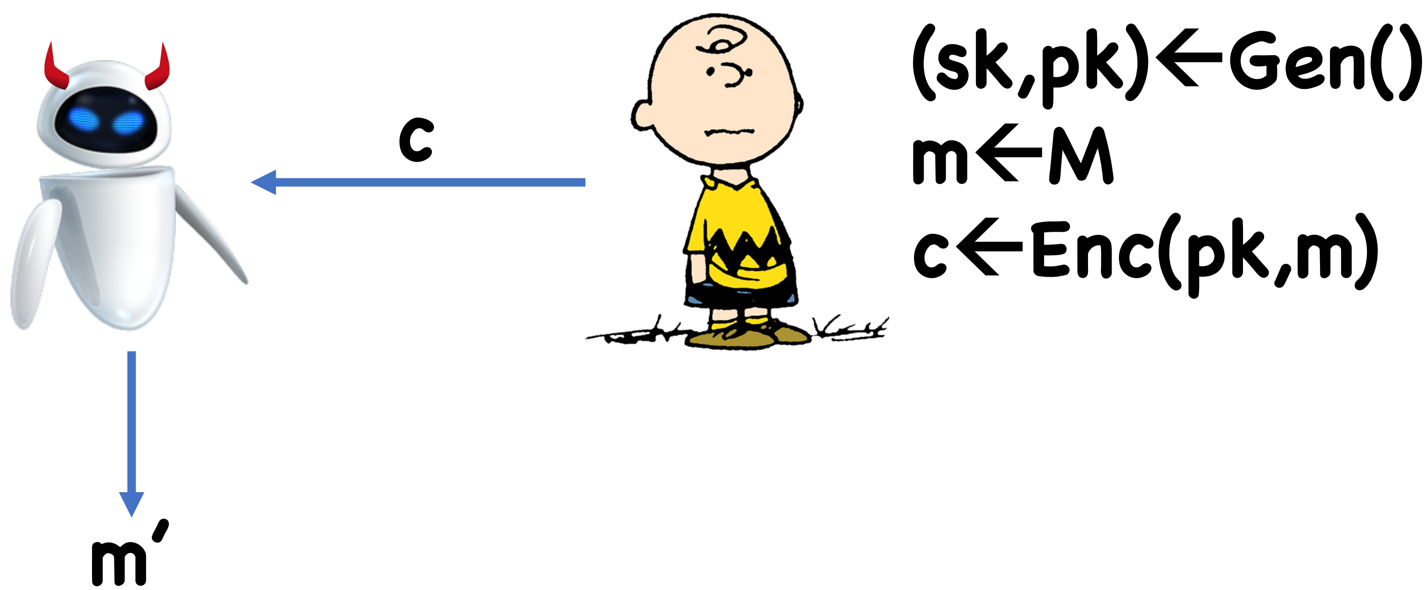
One-way security

Semantic Security

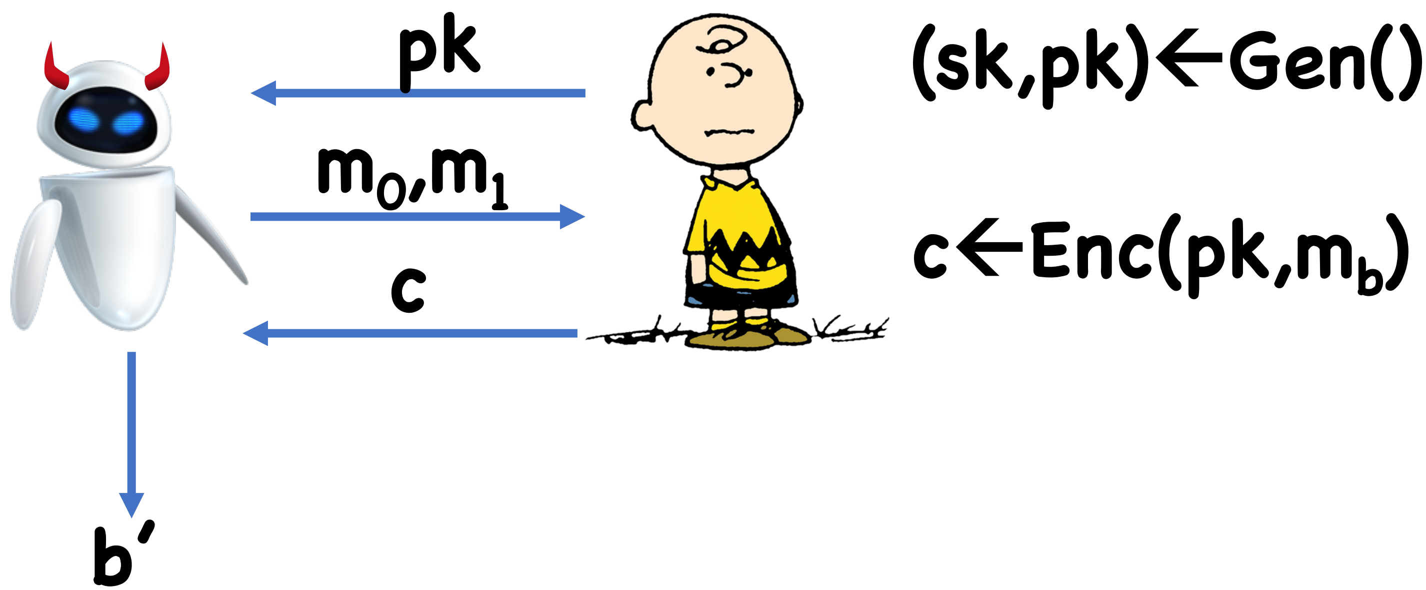
CPA security

CCA Security

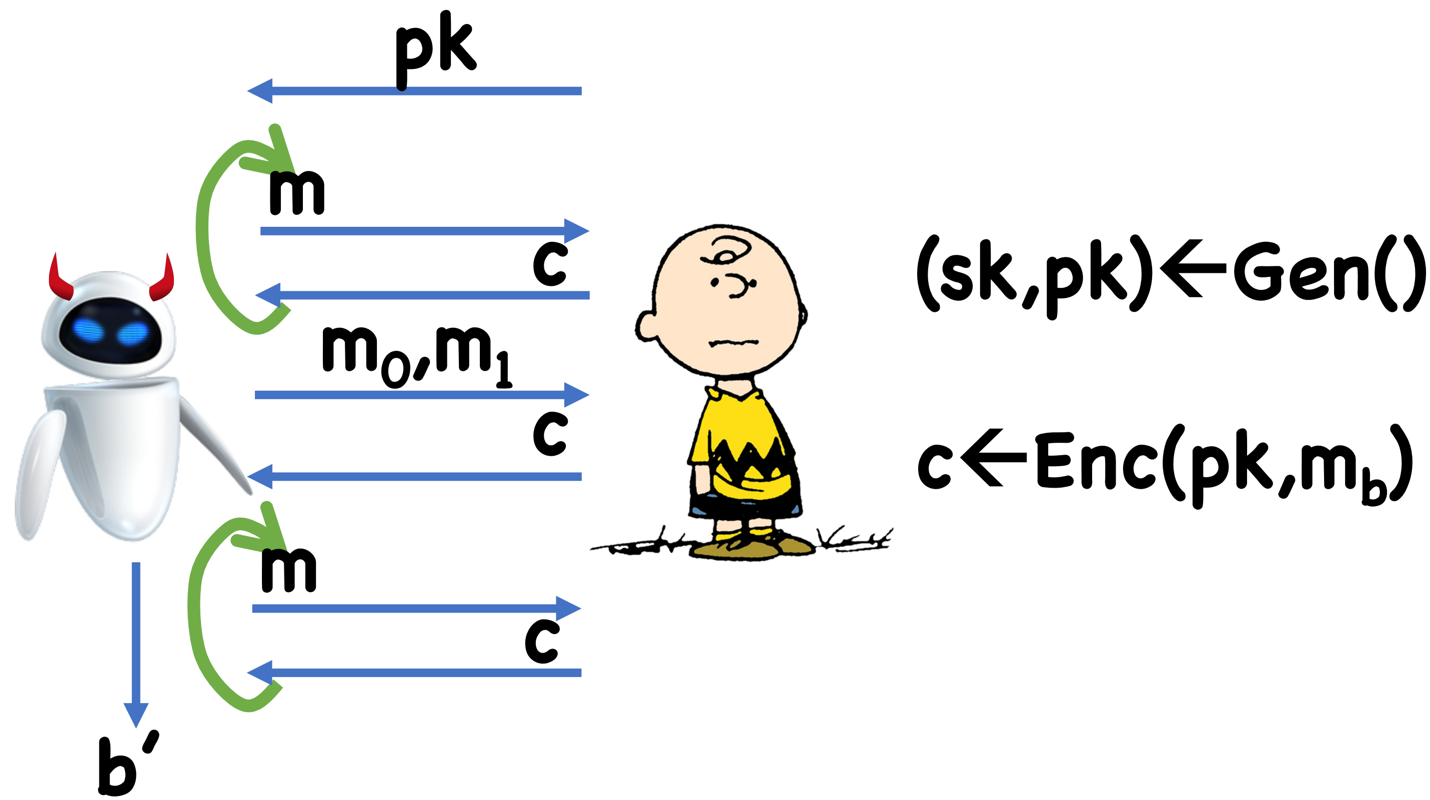
One-way Security



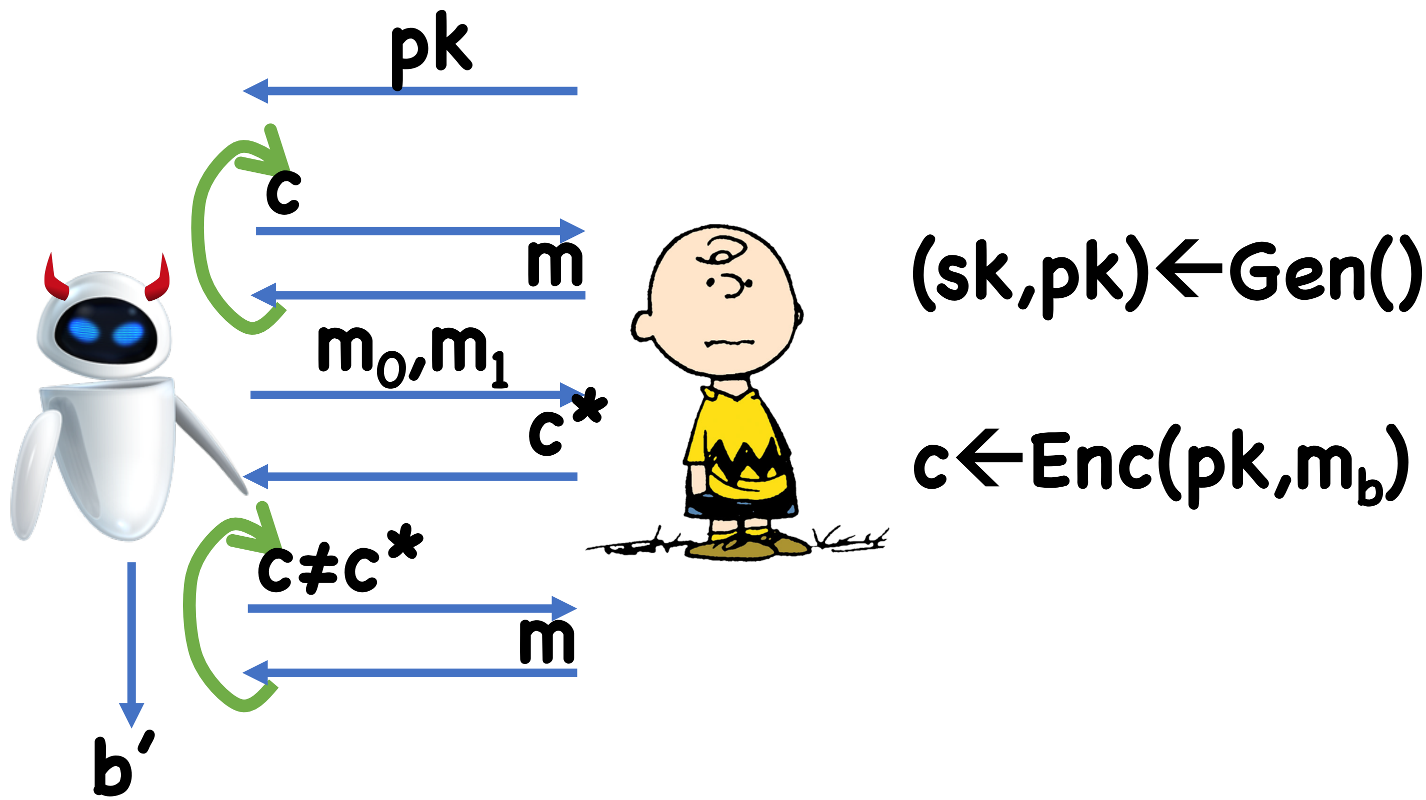
Semantic Security



CPA Security



CCA Security



Question: Which two notions are equivalent?

One-Way Encryption from TDPs

$\mathbf{Gen_E() = Gen_{TDP}()}$

Enc(pk,m): Output $\mathbf{c = F(pk,m)}$

Dec(sk,c): Output $\mathbf{m' = F^{-1}(sk,c)}$

Semantically Secure Encryption from TDPs

Ideas?

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:

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

Therefore, need **d** to be large, but ok taking **$e=3$**

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 m from $m_0,\ m_1$

Running time:

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

ElGamal

Group **G** of order **p**, generator **g**
Message space = **G**

Gen():

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

Enc(pk,m∈{0,1}):

- $r \leftarrow \mathbb{Z}_p$
- $c = (g^r, h^r \times m)$

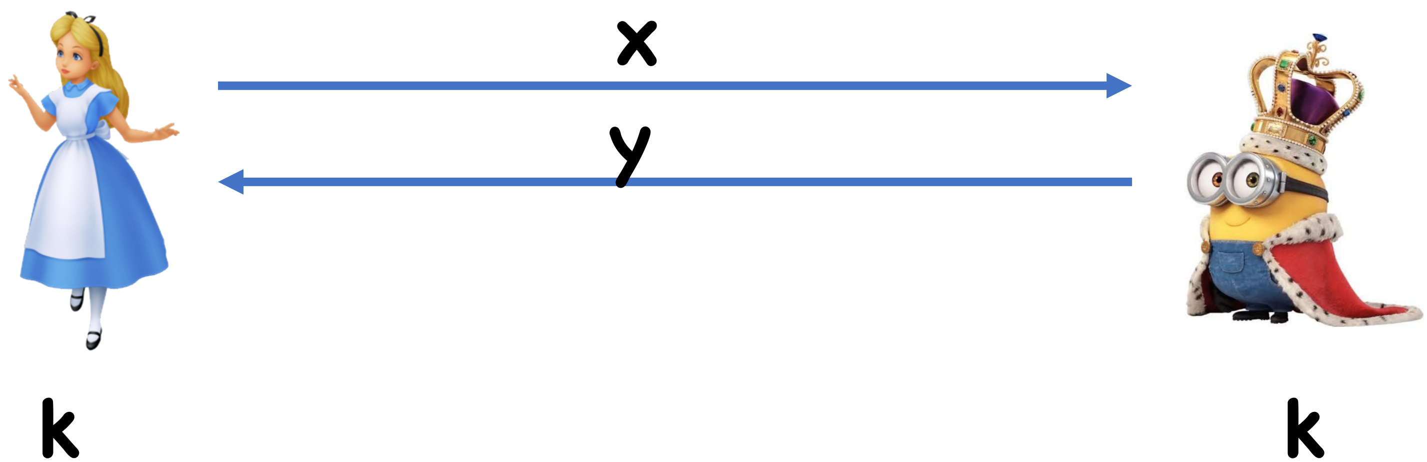
Dec?

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

$x = A()$ →

← $y = B(x)$

$k = A(state_A, x, y)$

$k = B(state_B, x)$

Here, $\mathbf{state_A}$, $\mathbf{state_B}$, 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

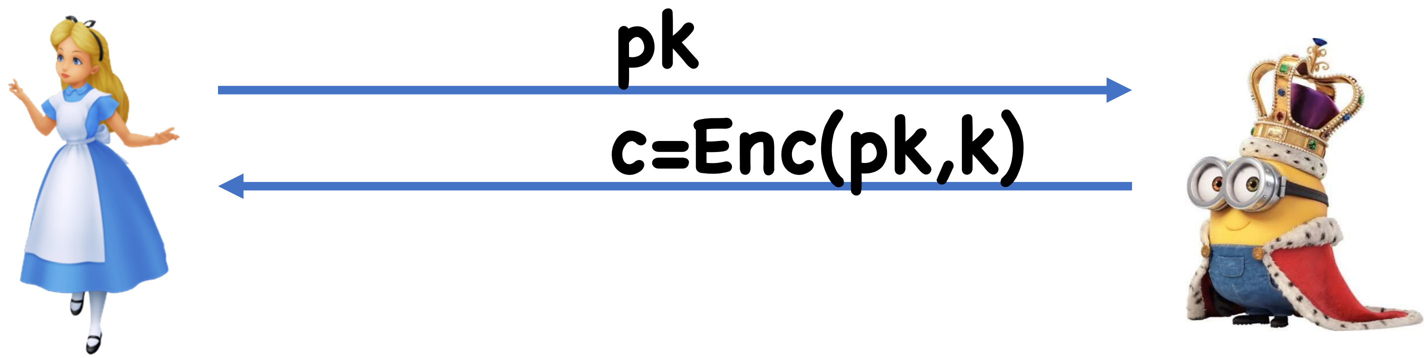
Theorem: If **(A,B)** is a **(t,ε)-**secure one-round key exchange protocol, then **(Gen,Enc,Dec)** is **(t-t',ε)-** Semantically Secure

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



k ← Dec(sk,c)   k←M

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

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

- **k←$\mathbf{Dec}_{PKE}$(sk,c_0)**
- **m←$\mathbf{Dec}_{SKE}$(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: $(\mathbf{Enc_{PKE}(pk,k)},\ \mathbf{Enc_{SKE}(k,m_1)})$

CCA-secure encryption

CCA Secure PKE from TDPs

Let $(\mathsf{Enc}_{\mathsf{SKE}},\mathsf{Dec}_{\mathsf{SKE}})$ be a CCA-secure secret key encryption scheme.

Let $(\mathsf{Gen},\mathsf{F},\mathsf{F}^{-1})$ be a TDP

Let H be a hash function (we'll pretend it's a random oracle)

CCA Secure PKE from TDPs

$\mathbf{Gen_{PKE}() = Gen()}$

$\mathbf{Enc_{PKE}(pk, m)}$:

- Choose random $\mathbf{r}$
- Let $\mathbf{c} \leftarrow \mathbf{F(pk,r)}$
- Let $\mathbf{d} \leftarrow \mathbf{Enc_{SKE}(H(r), m)}$
- Output $\mathbf{(c_0,c_1)}$

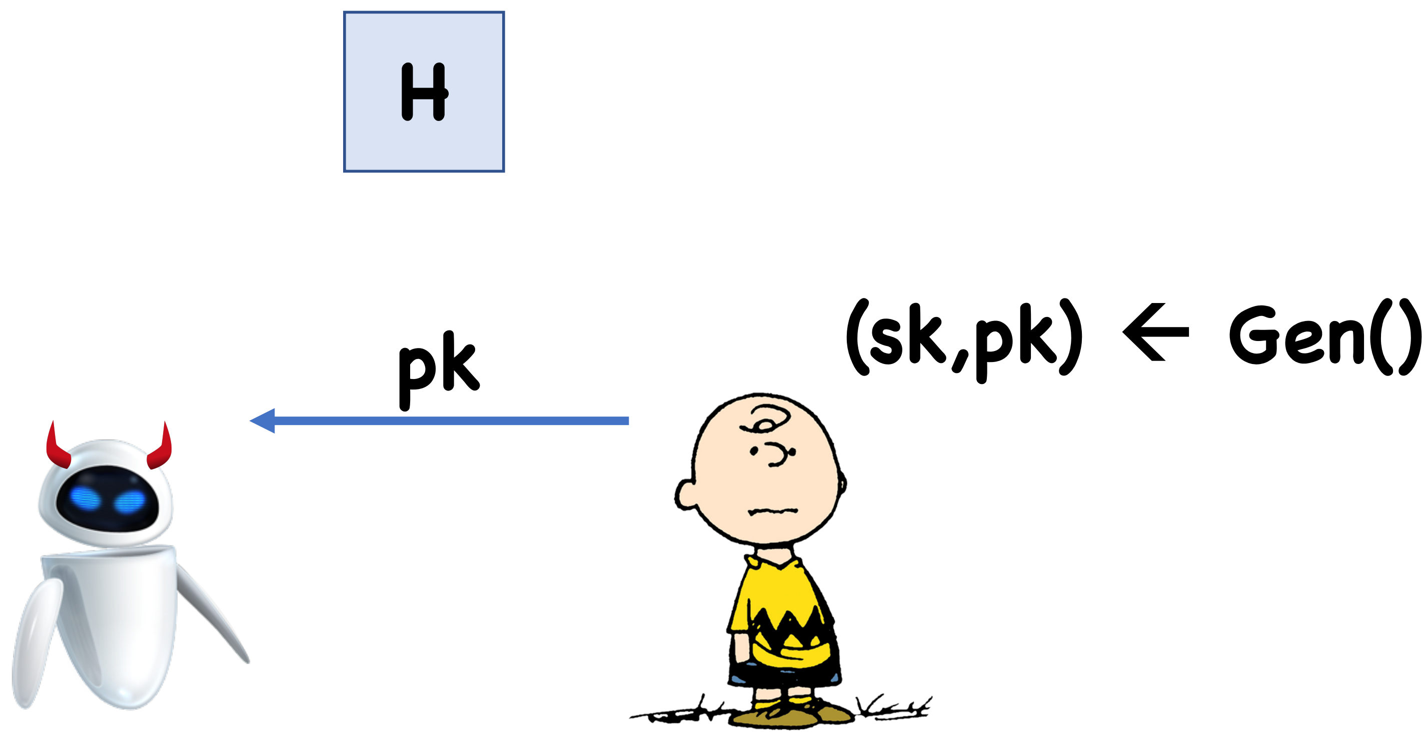
$\mathbf{Dec_{PKE}(sk, (c, d))}$:

- $\mathbf{Let\ r \leftarrow F^{-1}(sk, c)}$
- $\mathbf{Let\ m \leftarrow Dec_{SKE}(H(r), d)}$

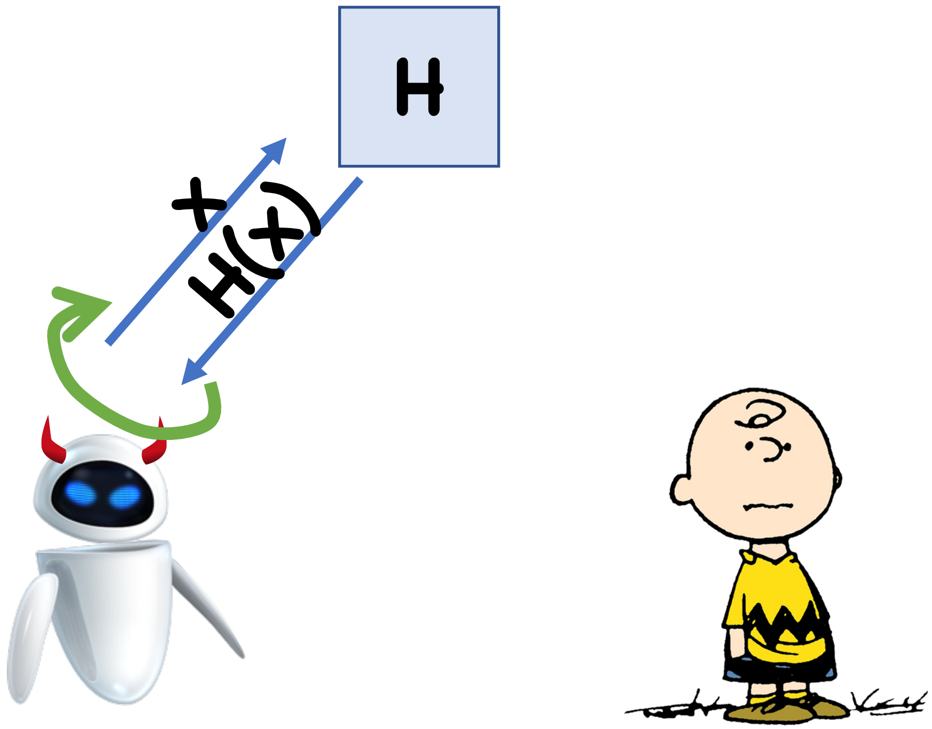
CCA Secure PKE from TDPs

Theorem: If $(\mathbf{Enc}_{SKE},\mathbf{Dec}_{SKE})$ is a CCA-secure secret key encryption scheme, $(\mathbf{Gen},\mathbf{F},\mathbf{F}^{-1})$ is a TDP, and $\mathbf{H}$ is modeled as a random oracle, then $(\mathbf{Gen}_{PKE},\mathbf{Enc}_{PKE},\mathbf{Dec}_{PKE})$ is a CCA secure public key encryption scheme

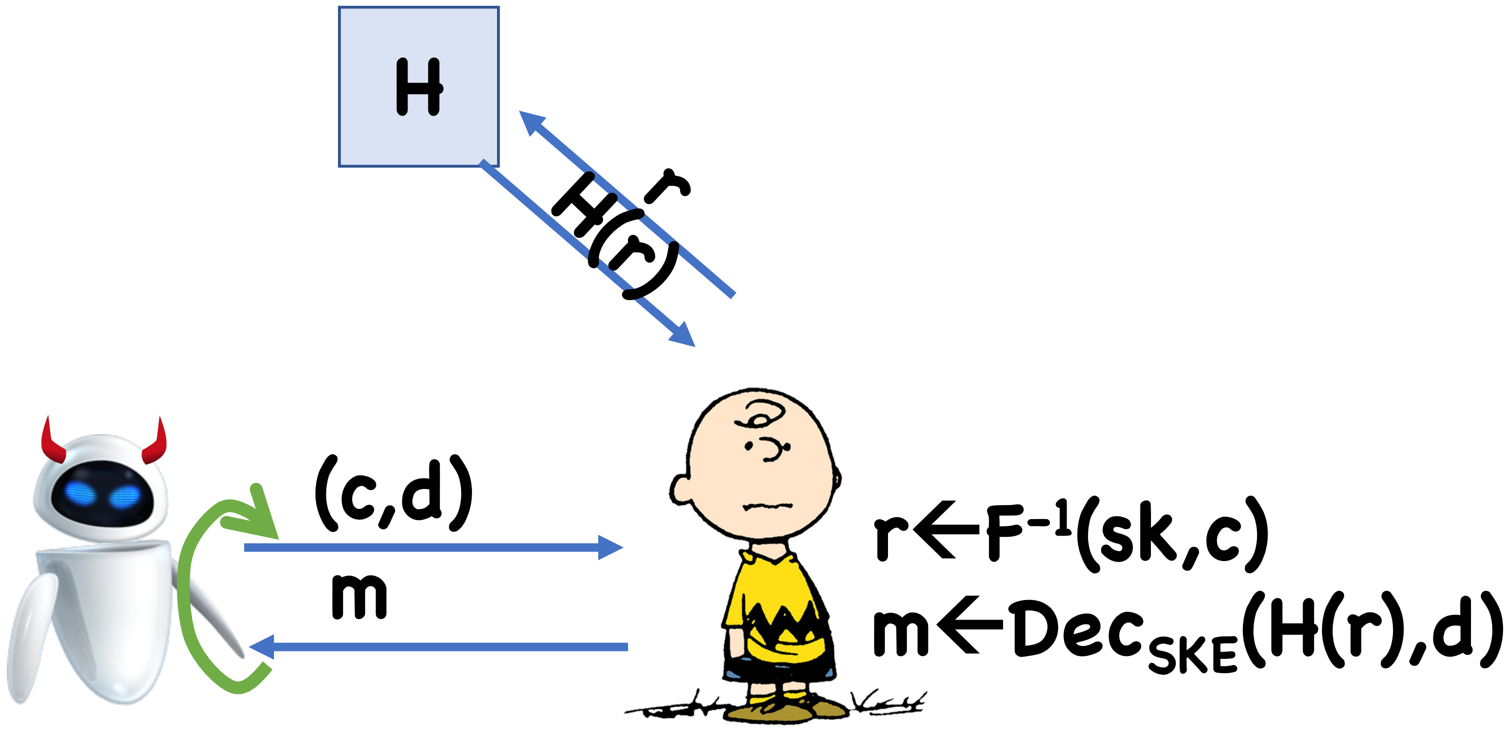
Proof



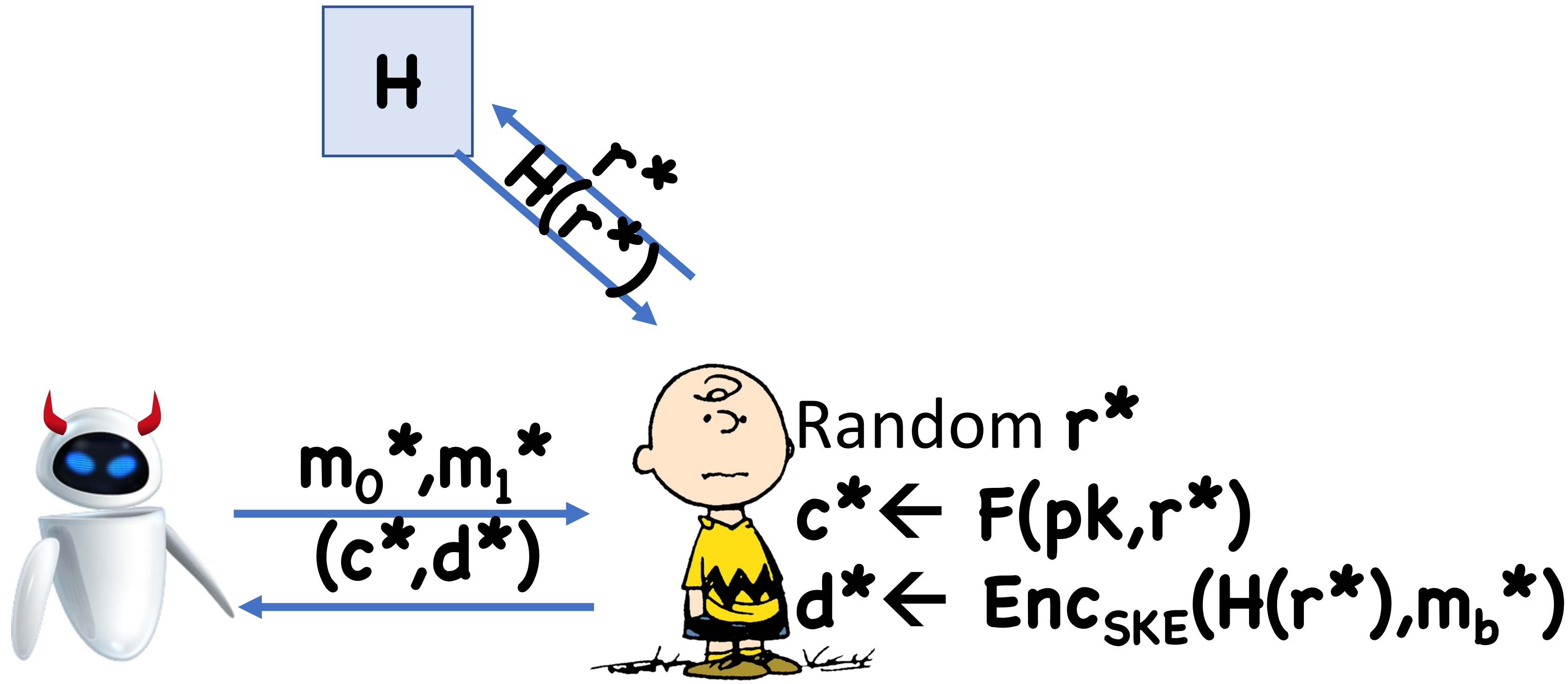
Proof



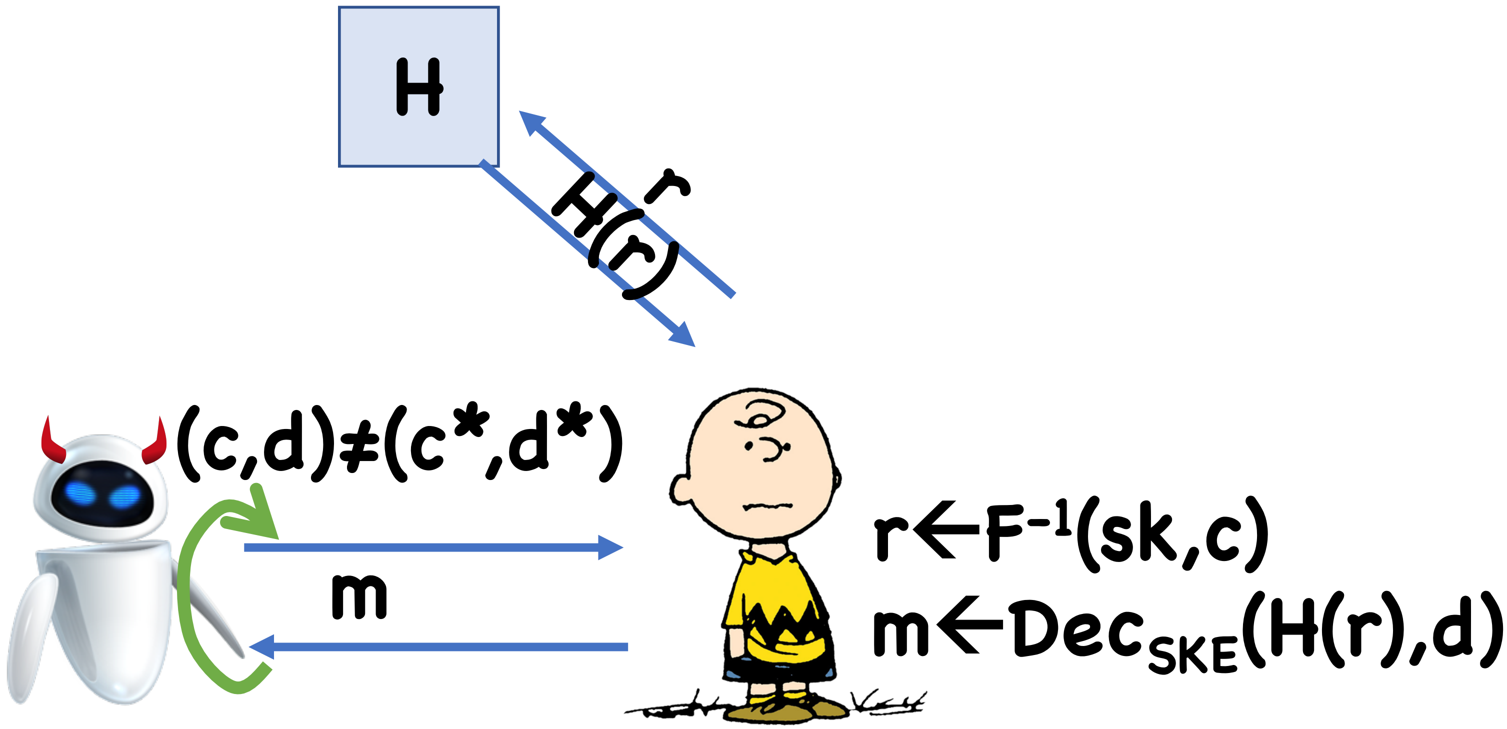
Proof



Proof



Proof



Proof

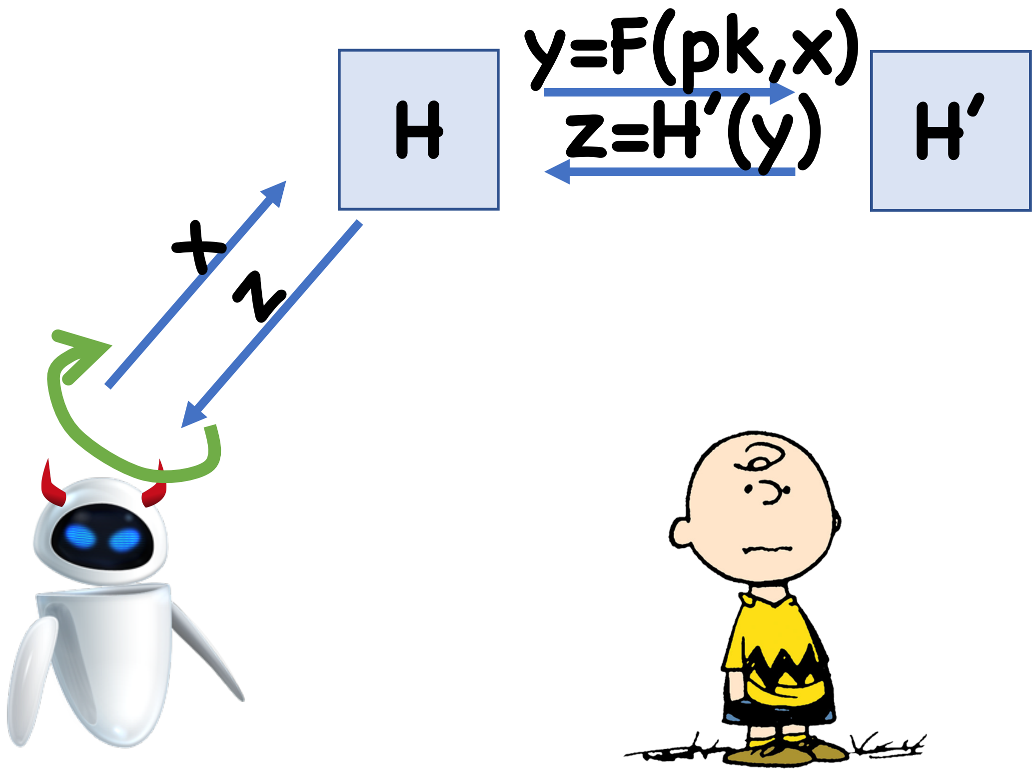
Step 1: sample **H** as follows:

- Choose a random function **H'**
- Let **H(x) = H'(F(pk, x))**

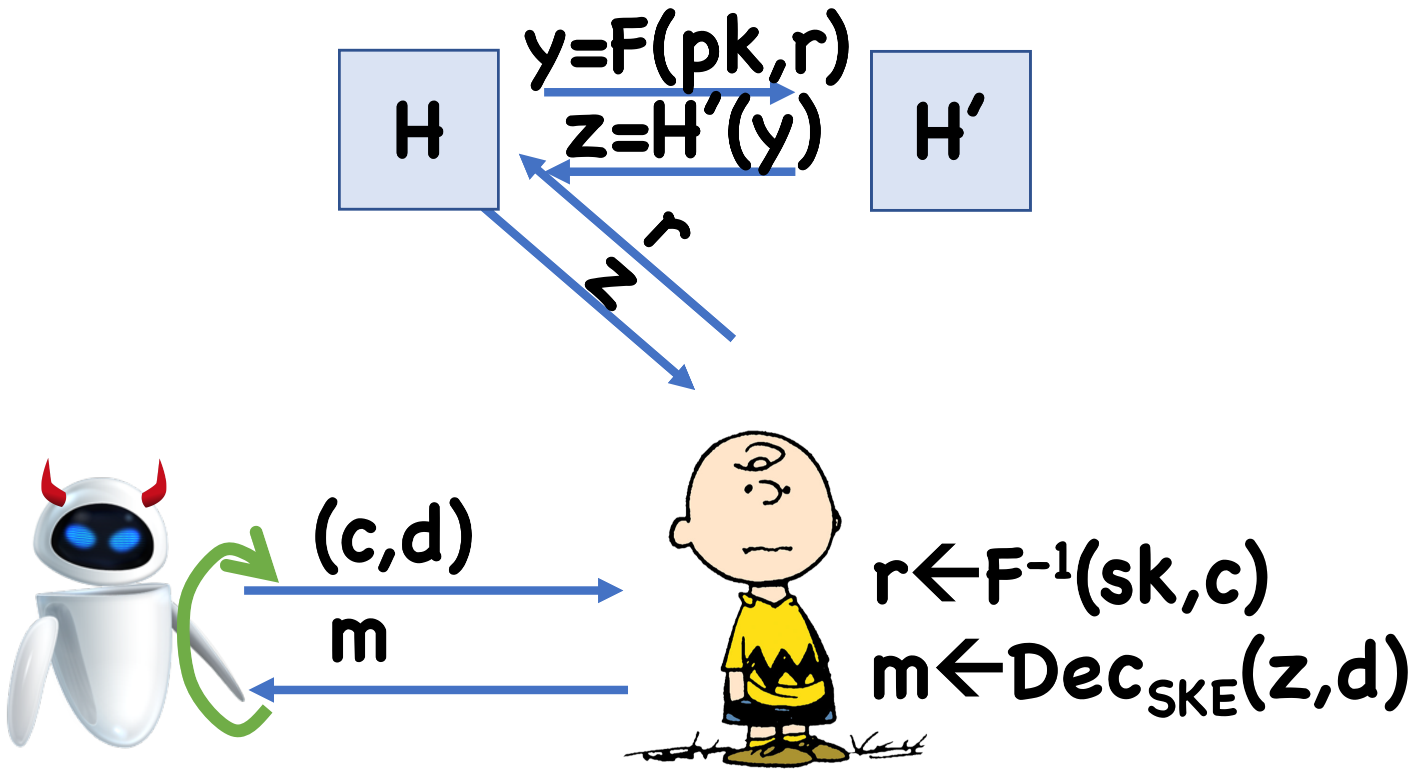
Since **F(pk, ·)** is a permutation, all outputs of **H(x)** are independent and uniform

Therefore, **H(x)** is still a random oracle

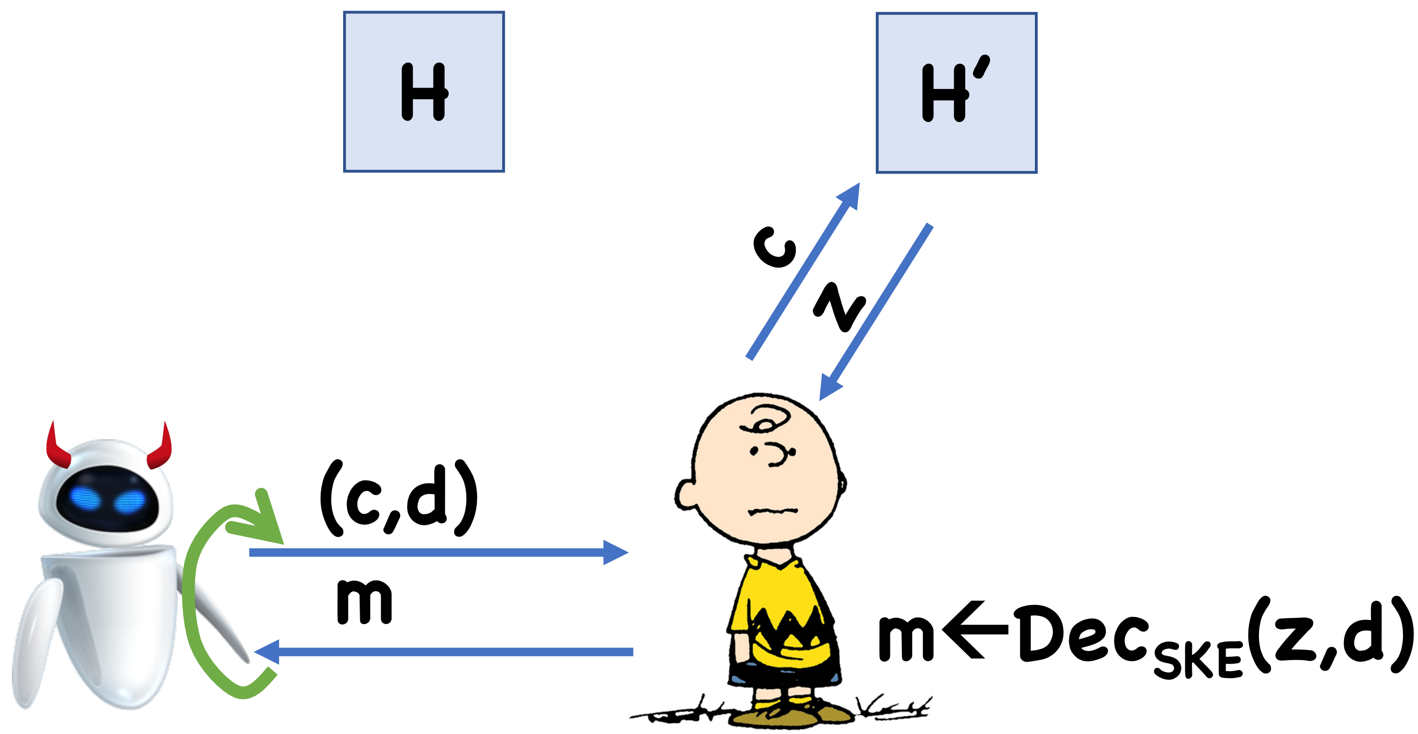
Proof



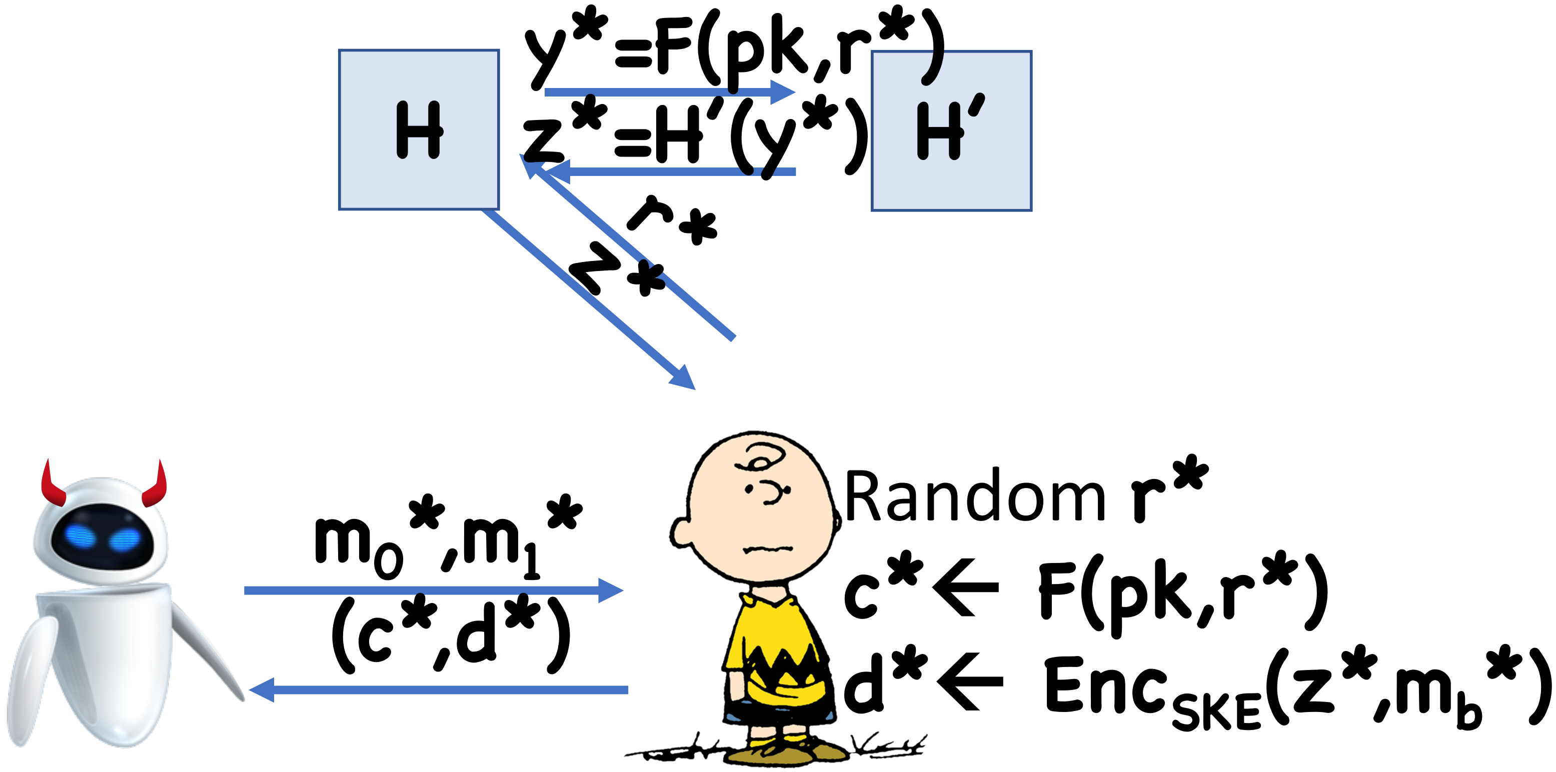
Proof



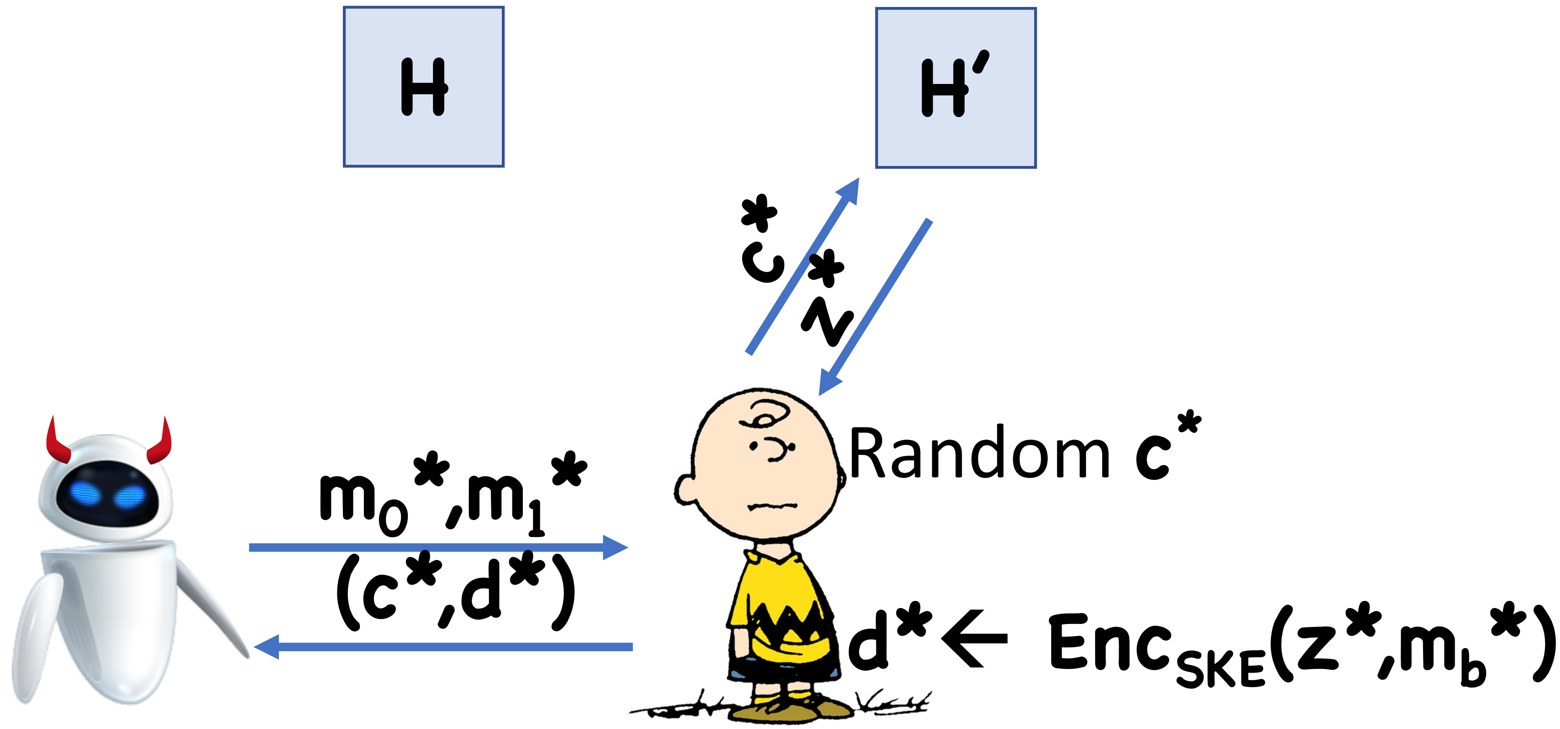
Proof



Proof



Proof



Observation: now Charlie doesn't need **sk** to run experiment

Proof

Consider two cases:

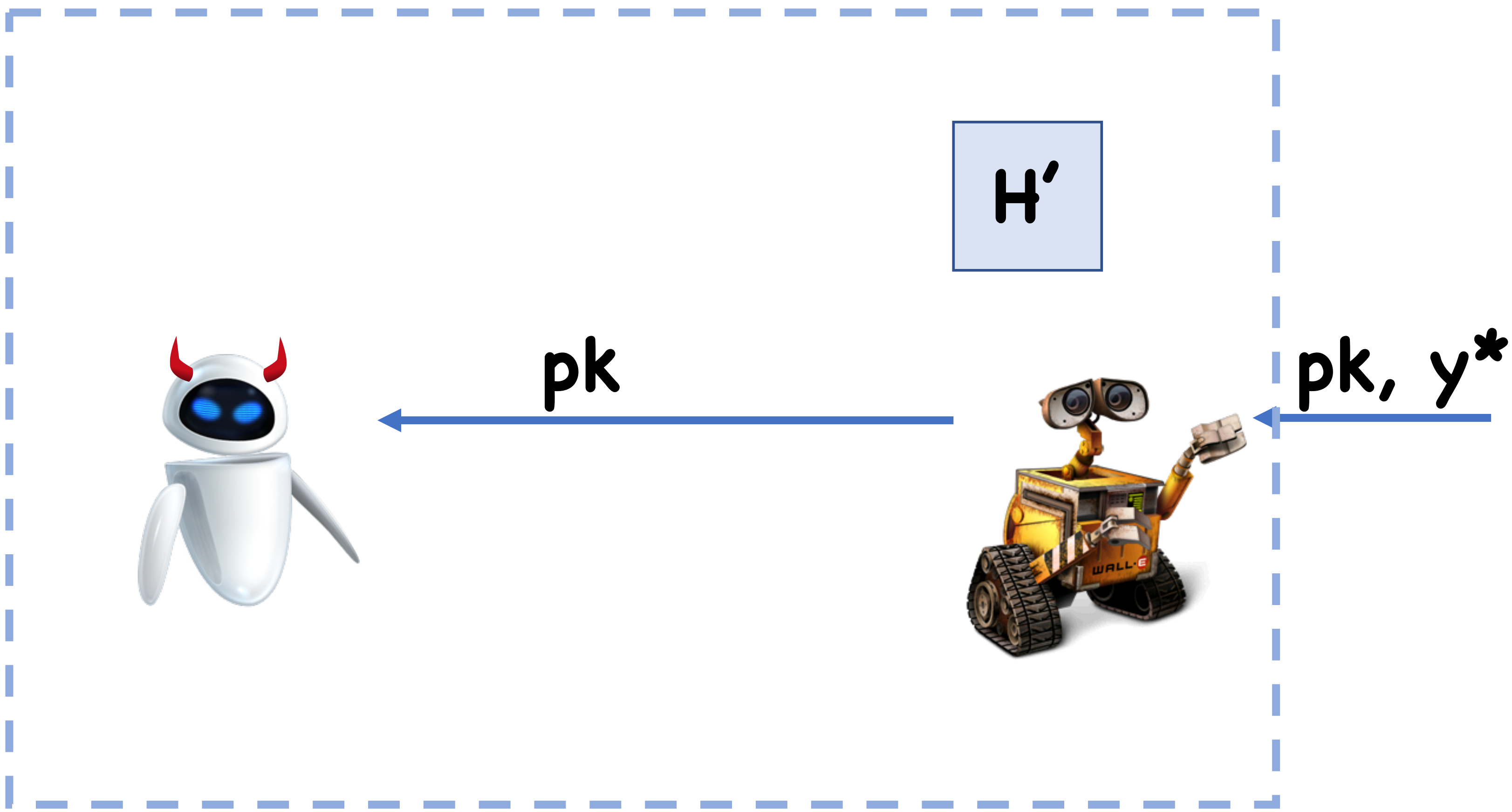
Case 1: adversary makes a RO query to **H** on

$$r^* = F^{-1}(sk,\ c^*)$$

Case 2: adversary never makes a RO query on r^*

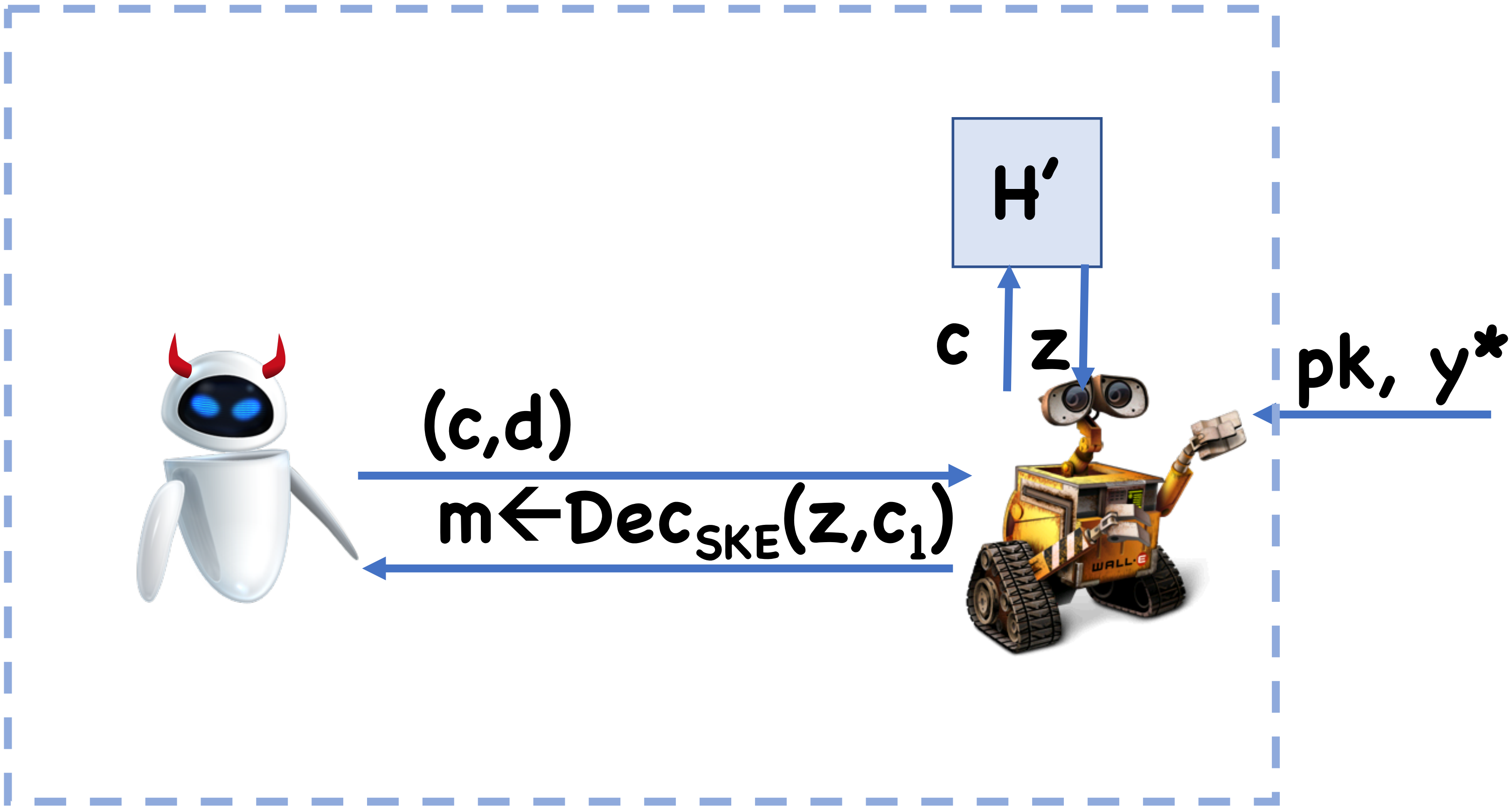
Proof

Case 1: construct TDP adversary 



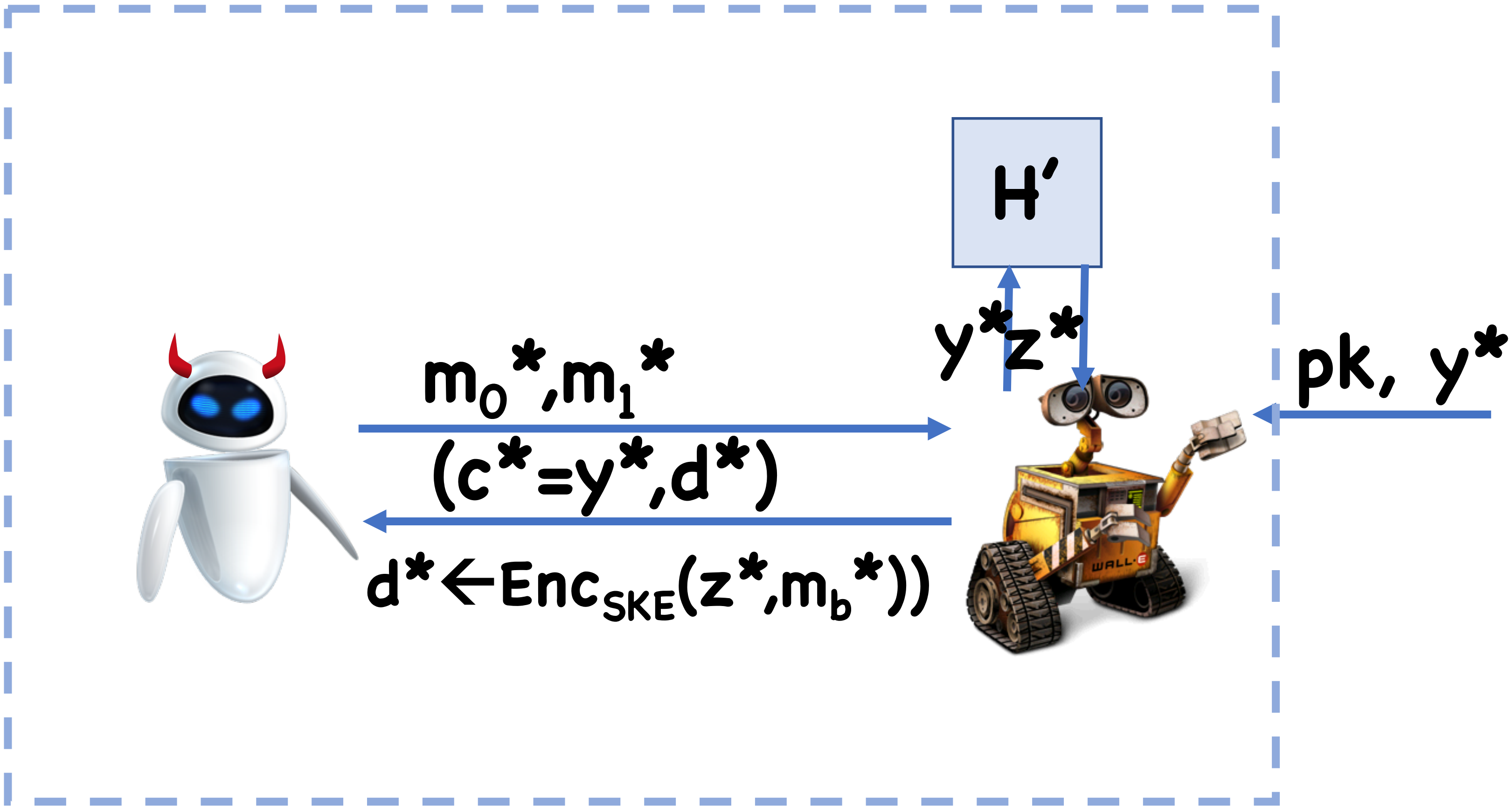
Proof

Case 1: construct TDP adversary 



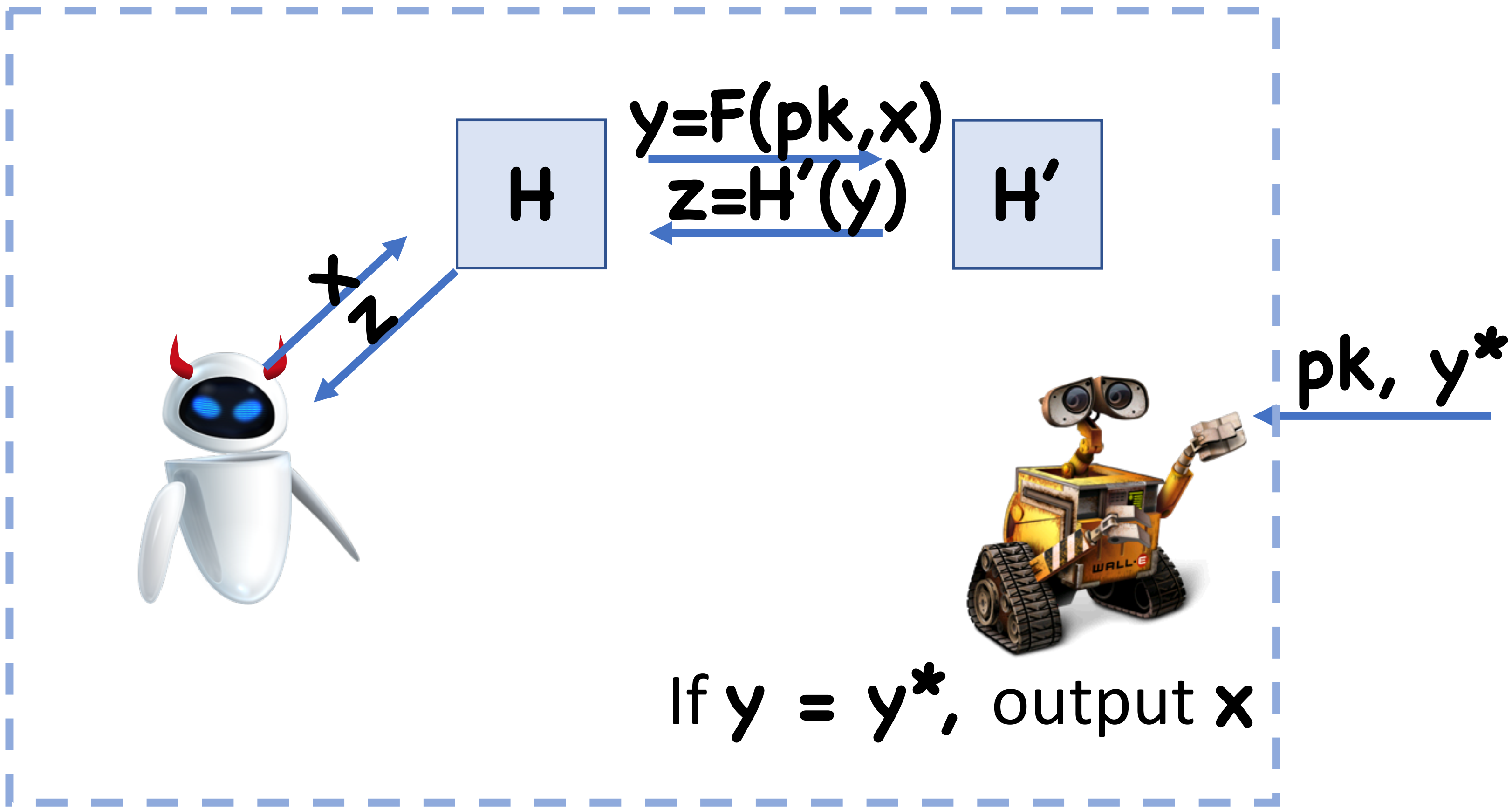
Proof

Case 1: construct TDP adversary 



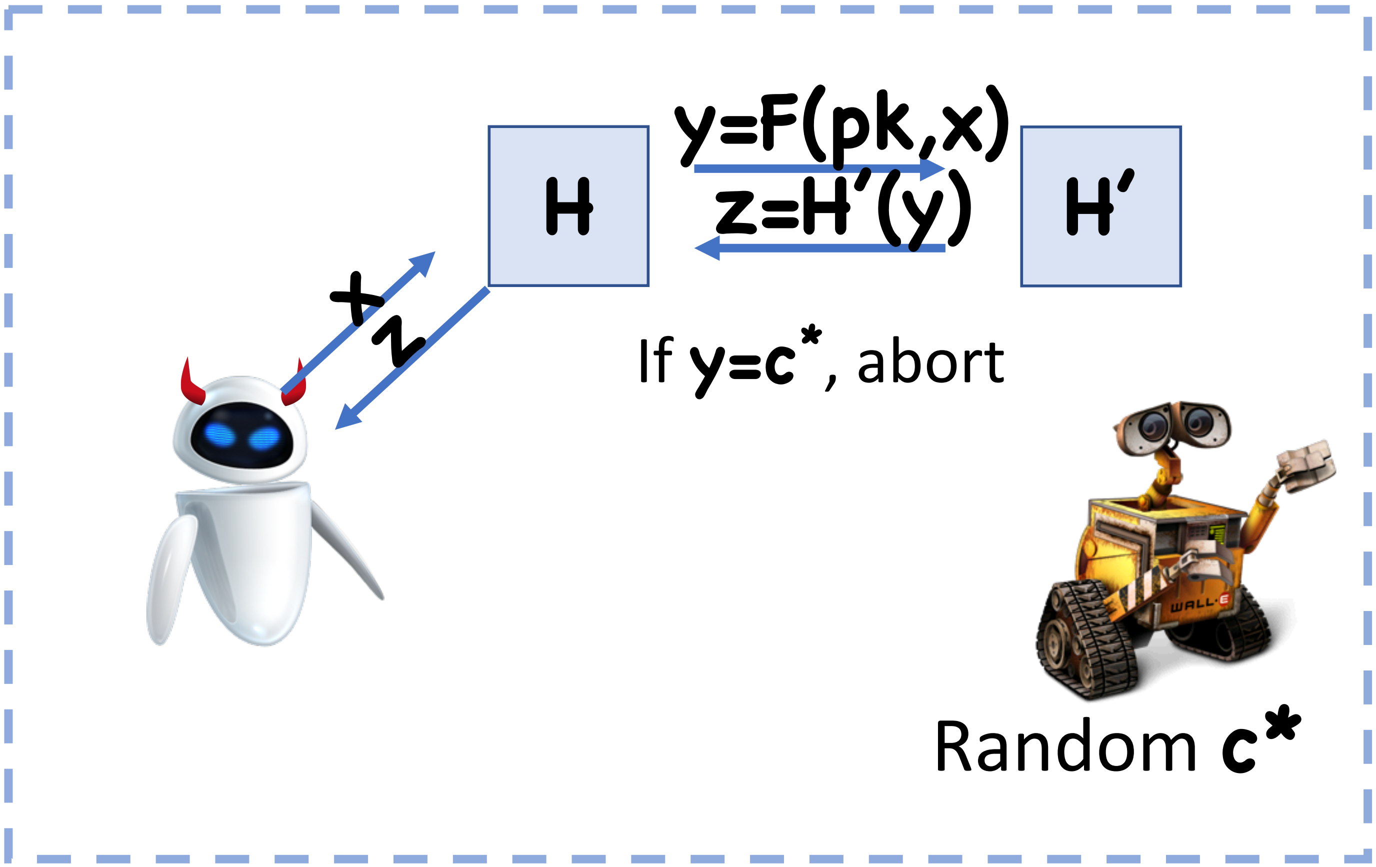
Proof

Case 1: construct TDP adversary 

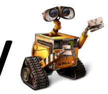


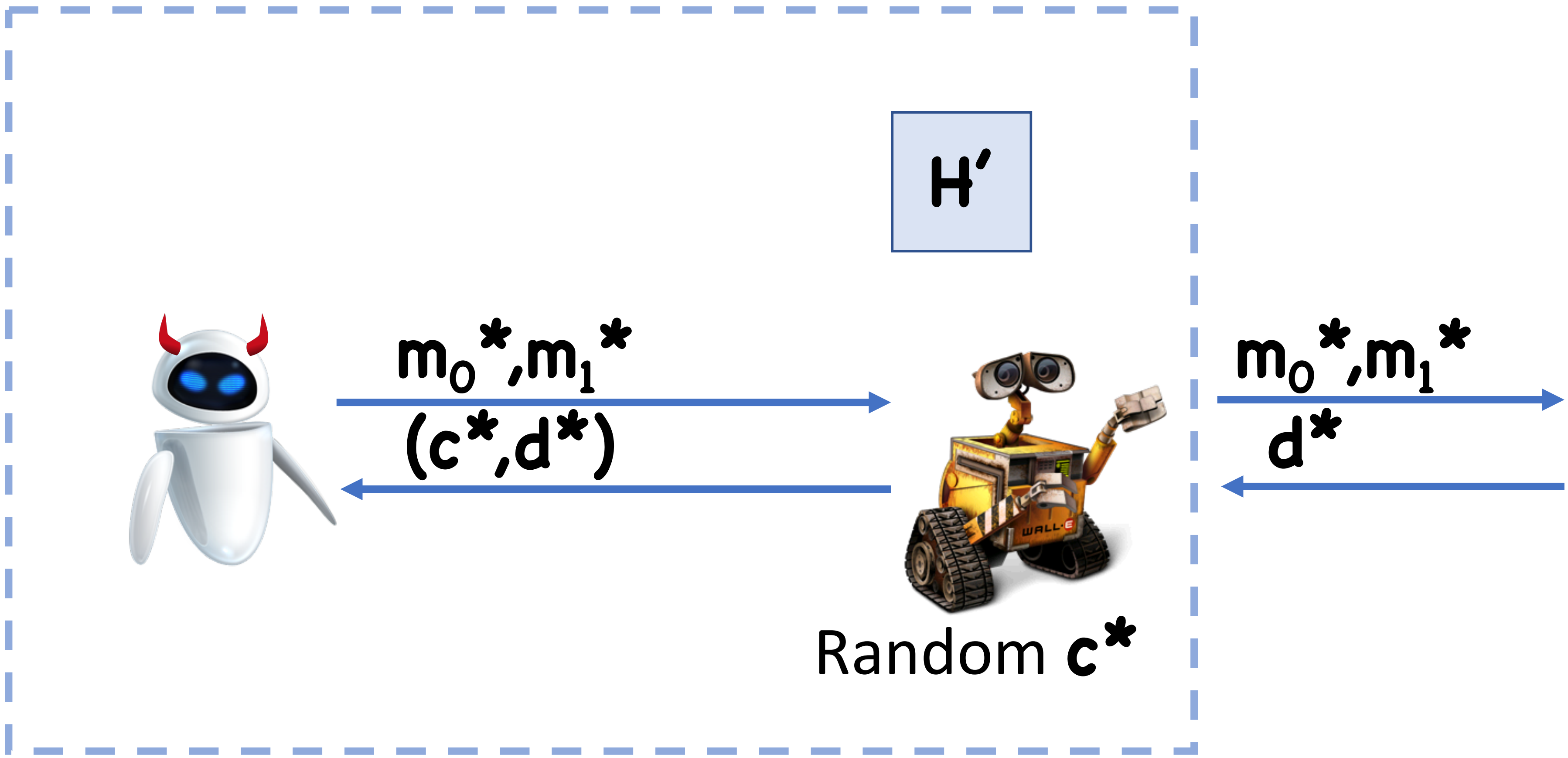
Proof

Case 2: construct $\textbf{Enc}_{\textbf{SKE}}$ adversary 



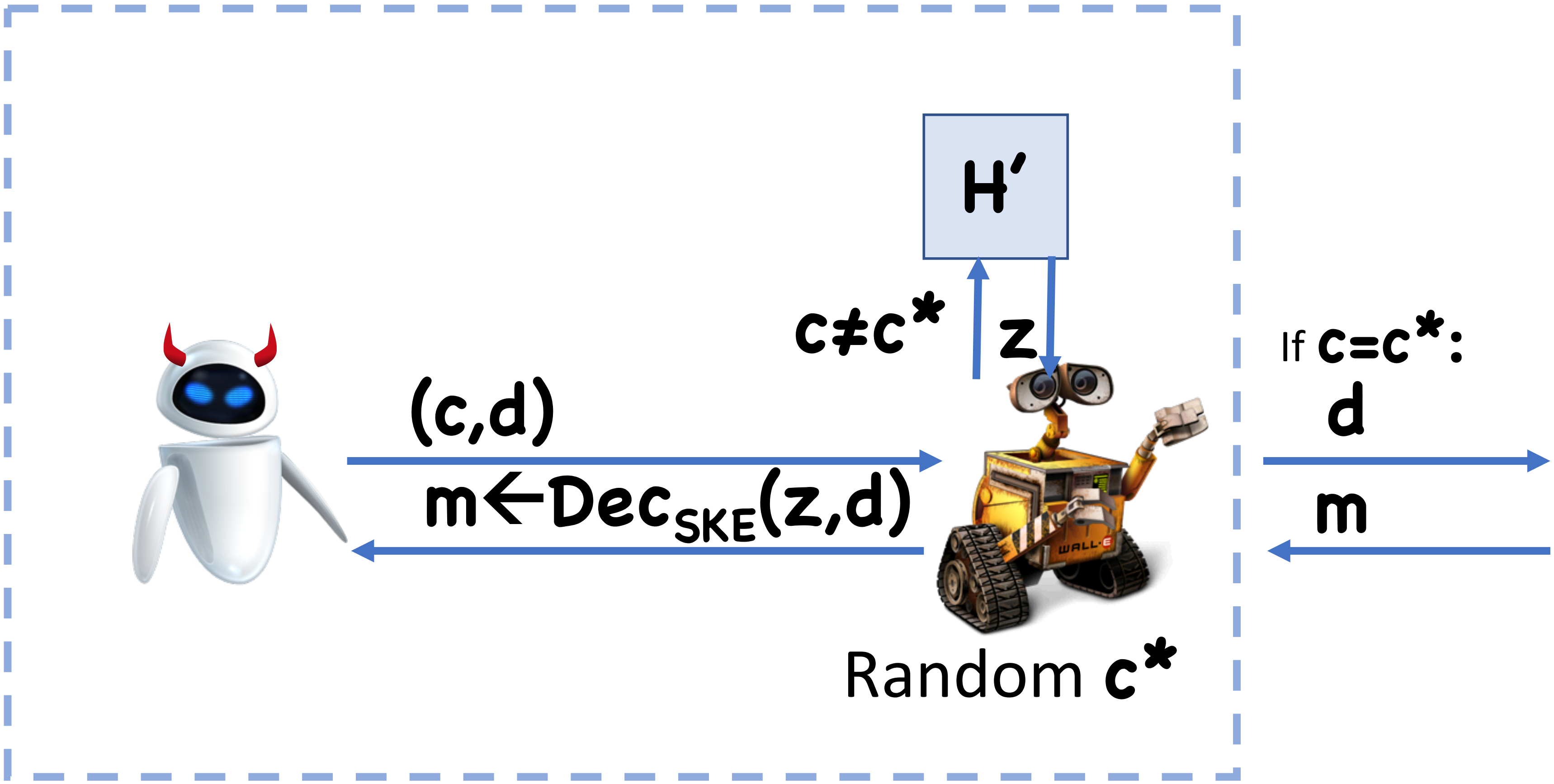
Proof

Case 2: construct $\mathbf{Enc_{SKE}}$ adversary 

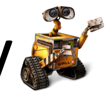


Proof

Case 2: construct **Enc**$_{\mathbf{SKE}}$ adversary



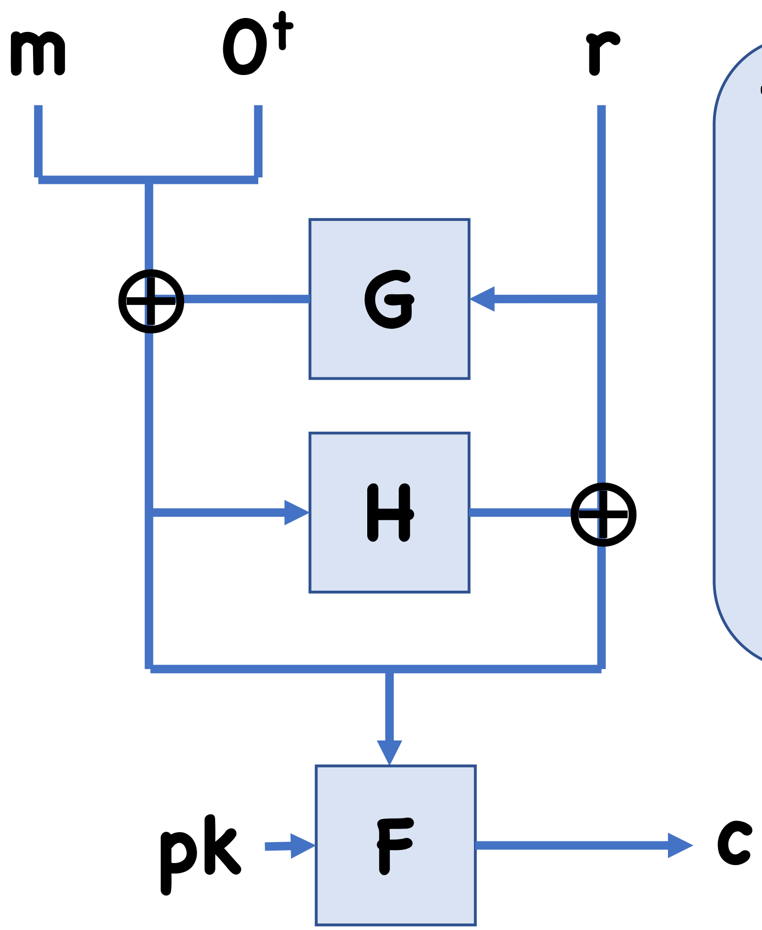
Proof

Case 2: construct $\mathbf{Enc_{SKE}}$ adversary 

Analysis:

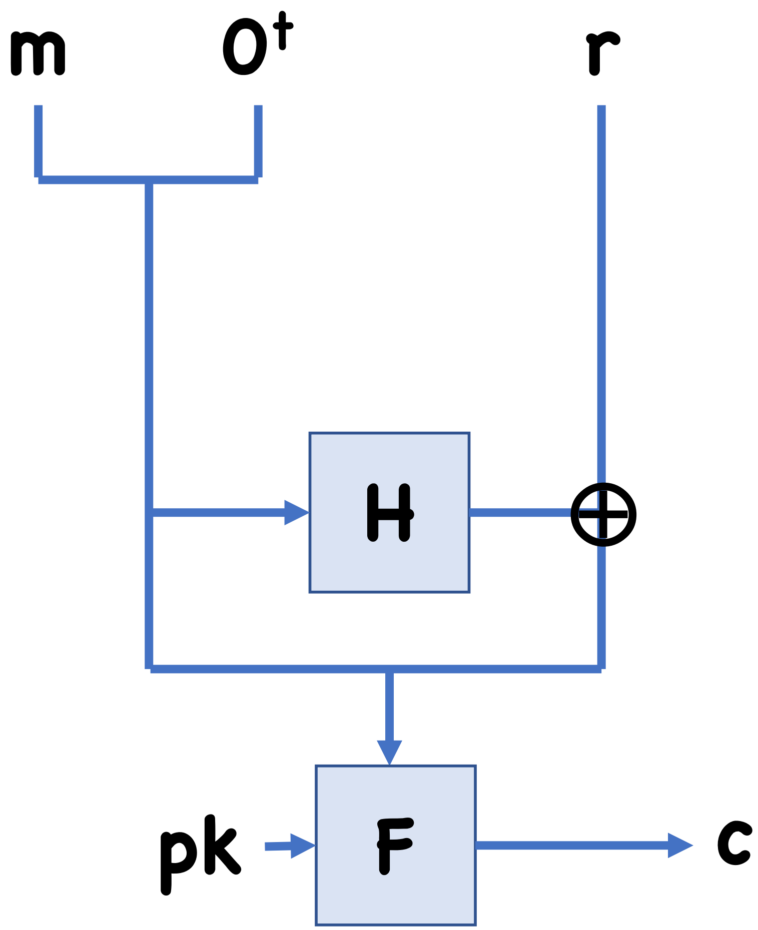
- Effectively set $\mathbf{H'(c_0^*) = k}$, where $\mathbf{k}$ is (unknown) challenger key
- Answers all queries correctly, provided adversary never queries RO on $\mathbf{c^*}$
- Therefore, breaks security of $\mathbf{Enc_{SKE}}$ in case 2

OAEP



Theorem: For RSA TDP, if **G,H** are modeled as a random oracles, then **(Gen_{PKE},Enc_{PKE},Dec_{PKE})** is a CCA secure public key encryption scheme

Insecure OAEP Variants

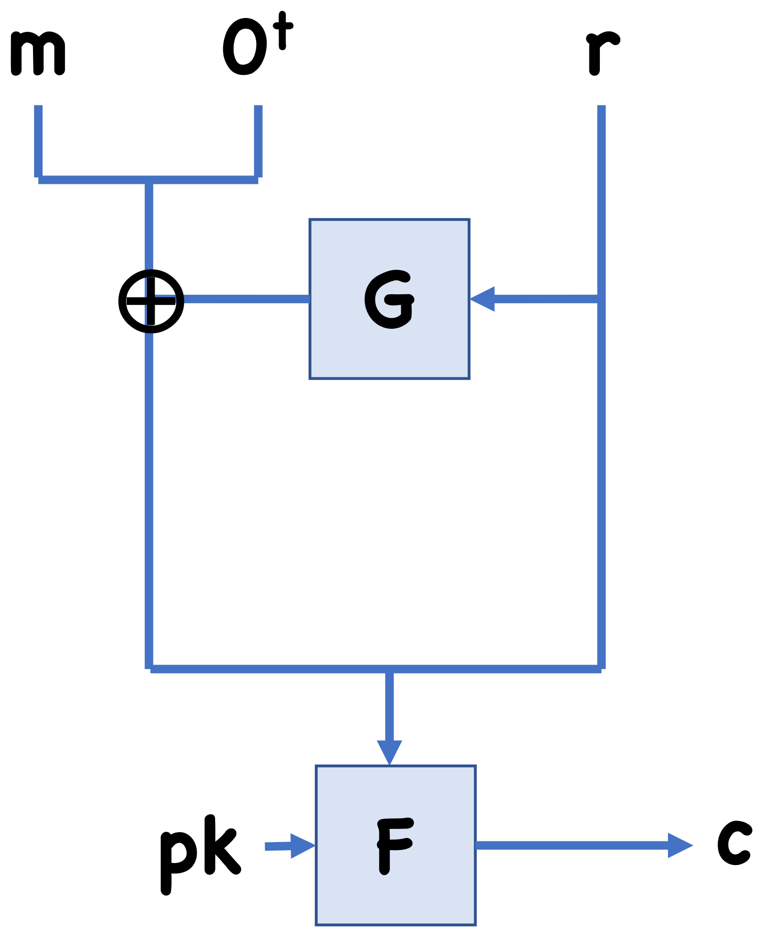


$c = F(pk, (m, 0^t, y))$

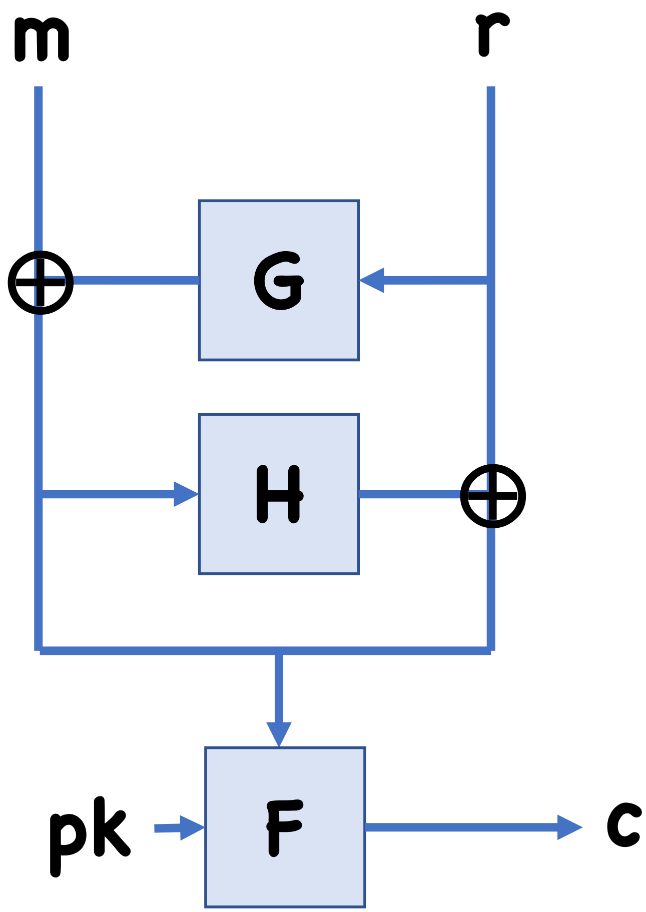
May contain **m** in the clear

- $F(pk, (m,x,y))$
  $= (m, F(pk, (x,y)))$

Insecure OAEP Variants



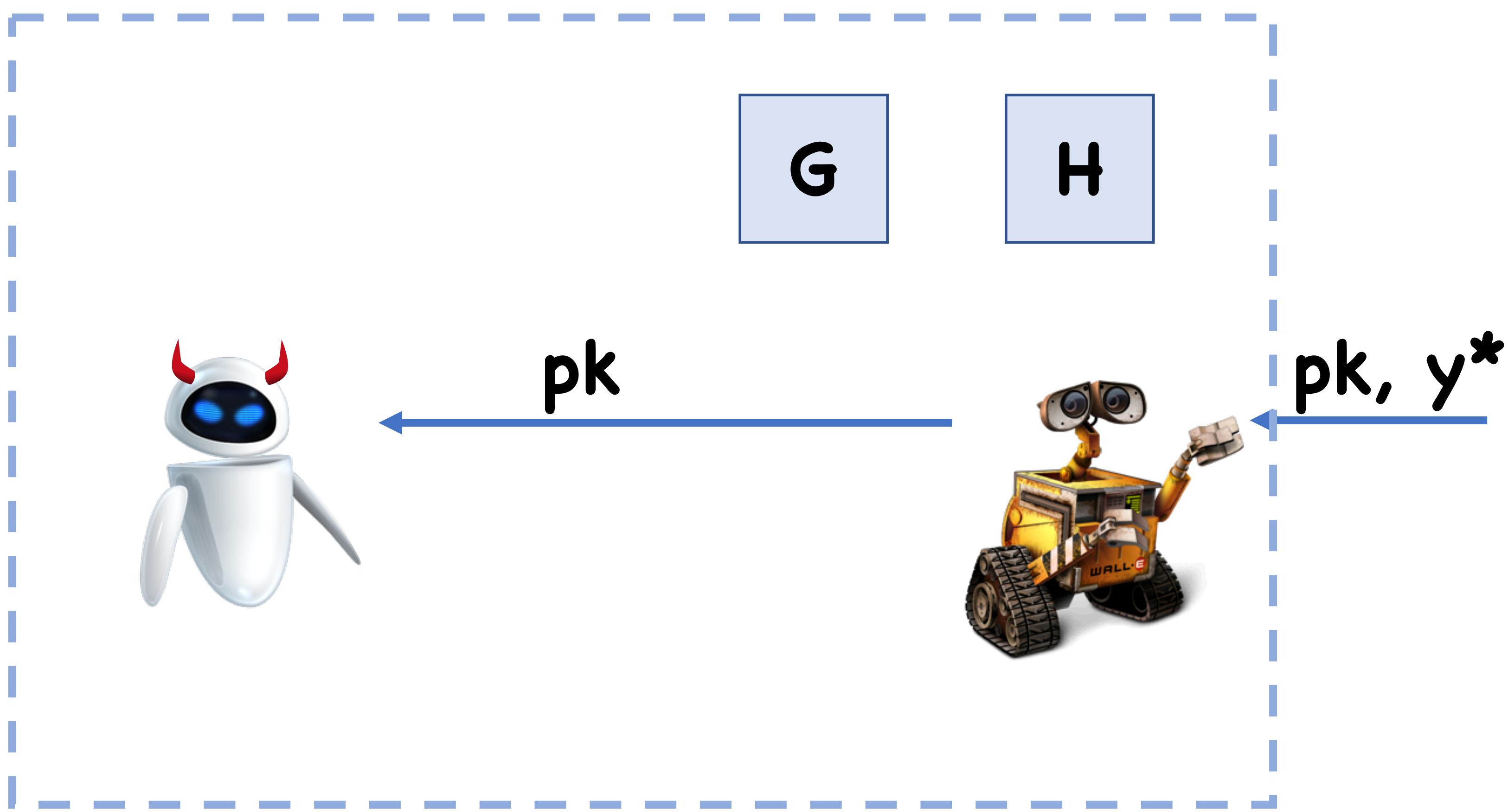
Why padding?



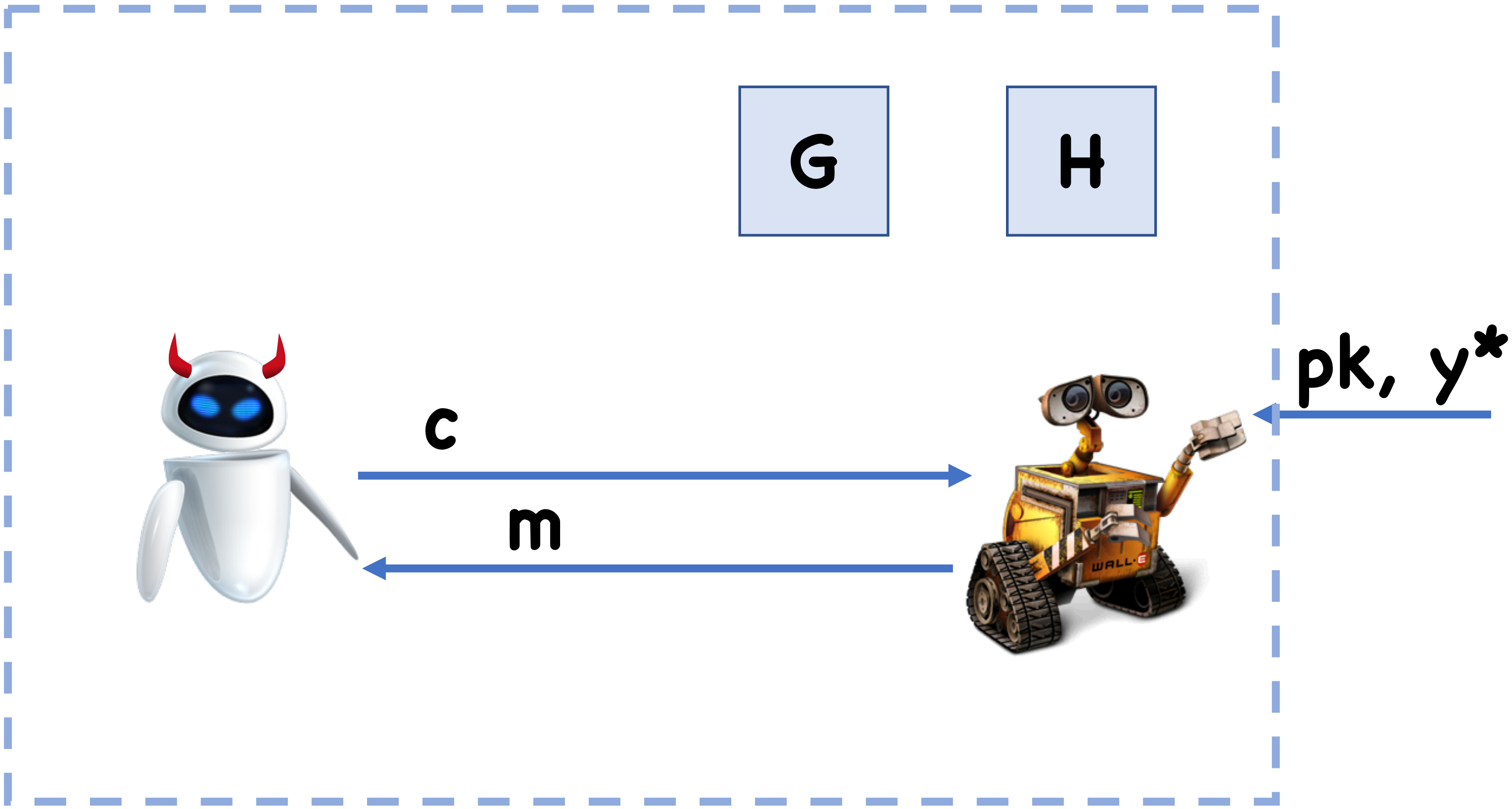
All ciphertexts decrypt to valid messages

- Makes it hard to argue security

High Level Proof Sketch



Claim: For any valid ctxt **c** queried by adv, adv must have created **c** by running **Enc(pk,m;r).** In this case, **m** can be decoded by looking at queries to **G,H**



Advantages of RSA-OAEP

RSA domain is at least 2048 bits

In hybrid encryption, ciphertext overhead =2048 bits

With OAEP (optimal asymmetric encryption padding), plaintext size can be, say 2048-256 bits with ciphertext size = 2048 bits

- Overhead = 256 bits

Reminders

Project Due Tomorrow

Homework 6 will be out today

Next Time

Digital Signatures (aka public key MACs)