

# Defining Encryption

## Lecture 2

# Defining Encryption

## Lecture 2

Secrecy when Computationally Bounded

# Roadmap

# Roadmap

- First, Symmetric Key Encryption

# Roadmap

- First, Symmetric Key Encryption
- Defining the problem
  - We'll do it elaborately, so that it will be easy to see different levels of security

# Roadmap

- First, Symmetric Key Encryption
- Defining the problem
  - We'll do it elaborately, so that it will be easy to see different levels of security
- Solving the problem
  - In theory and in practice



# Roadmap

- First, Symmetric Key Encryption
- Defining the problem
  - We'll do it elaborately, so that it will be easy to see different levels of security
- Solving the problem
  - In theory and in practice
- Today: defining encryption

# Building the Model



# Building the Model

- Alice, Bob and Eve. Alice and Bob share a key (a bit string)



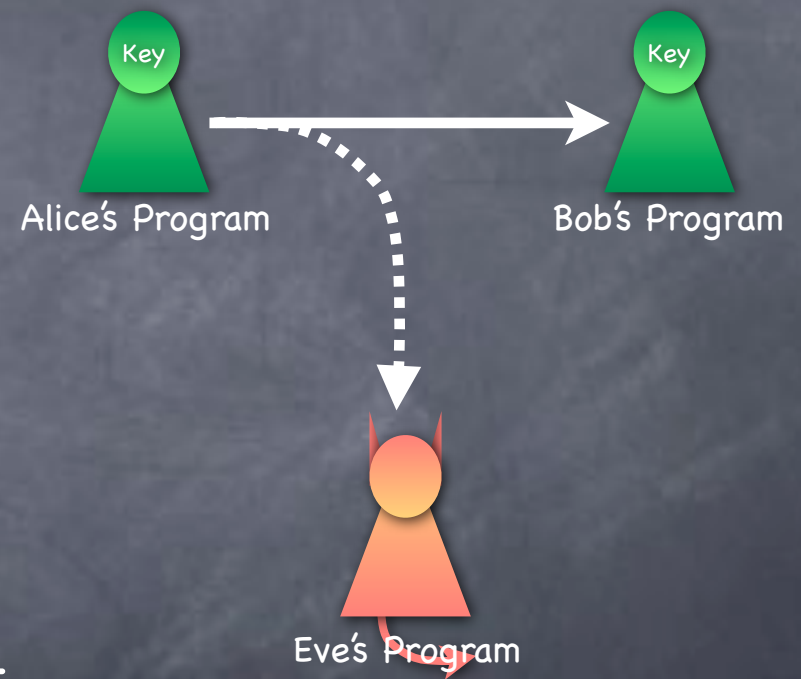
# Building the Model

- Alice, Bob and Eve. Alice and Bob share a key (a bit string)
- Alice wants Bob to learn a message, “without Eve learning it”

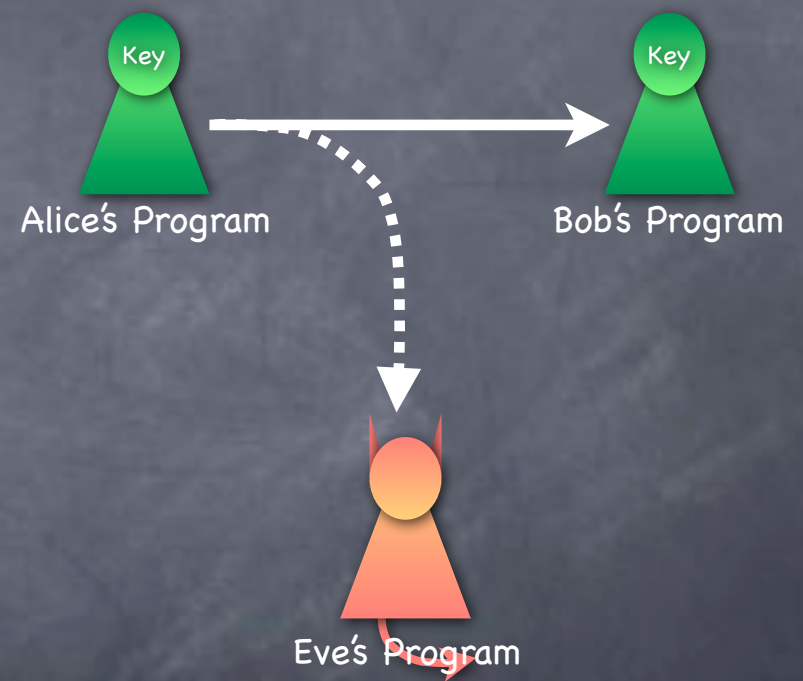


# Building the Model

- Alice, Bob and Eve. Alice and Bob share a key (a bit string)
- Alice wants Bob to learn a message, “without Eve learning it”
- Alice can send out a bit string on the channel. Bob and Eve both get it

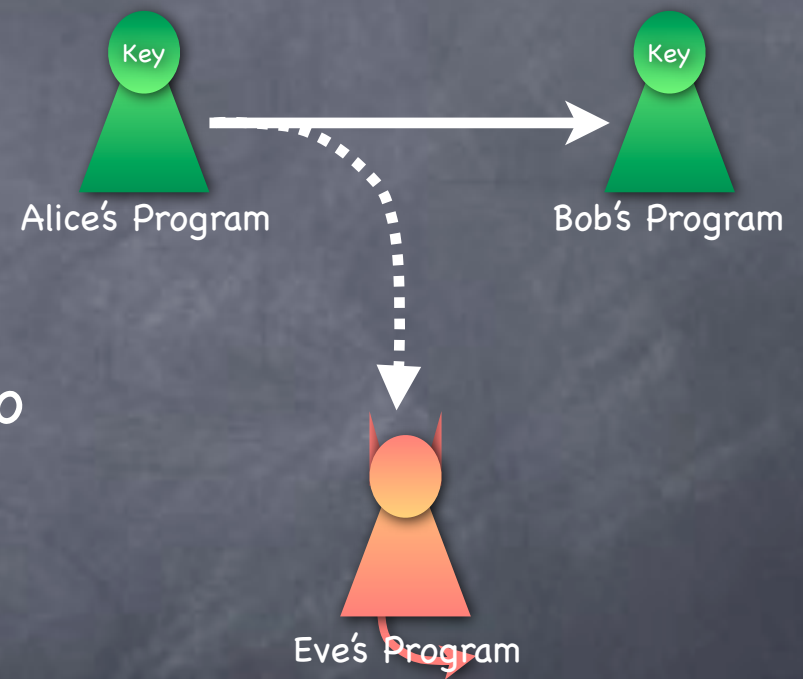


# Encryption: Syntax



# Encryption: Syntax

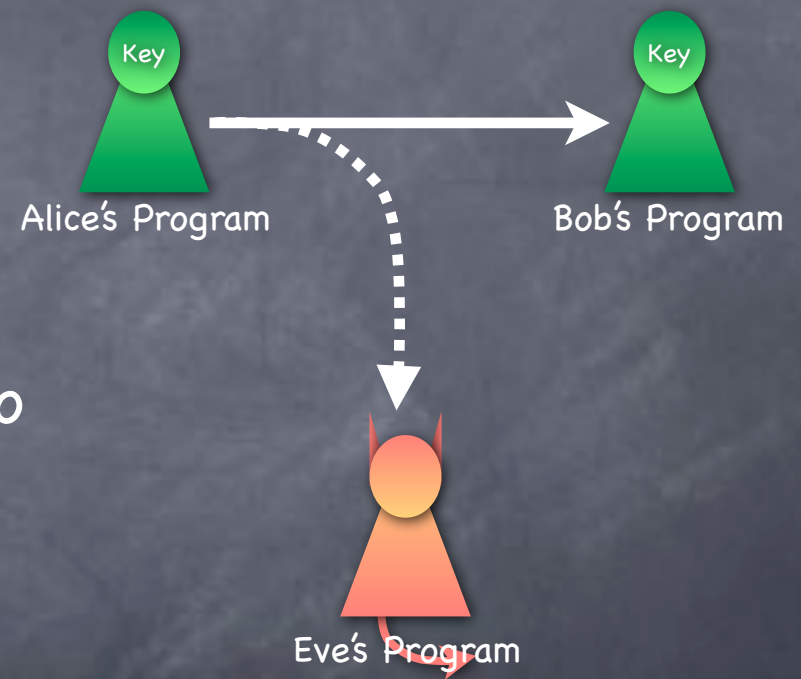
- Three algorithms
  - **Key Generation:** What Alice and Bob do a priori, for creating the shared secret key
  - **Encryption:** What Alice does with the message and the key to obtain a “ciphertext”
  - **Decryption:** What Bob does with the ciphertext and the key to get the message out of it



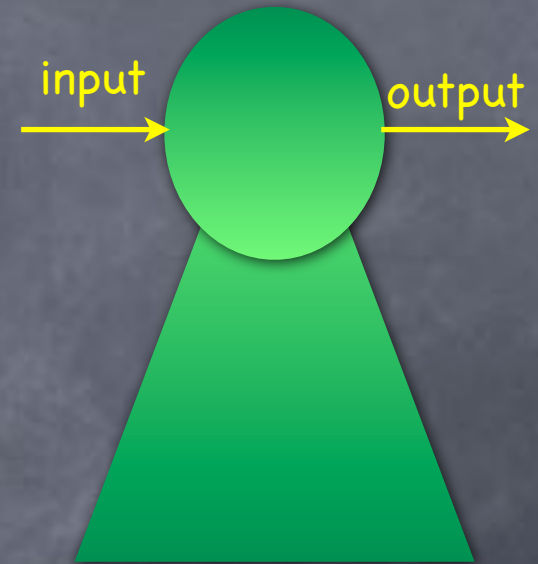


# Encryption: Syntax

- Three algorithms
  - **Key Generation:** What Alice and Bob do a priori, for creating the shared secret key
  - **Encryption:** What Alice does with the message and the key to obtain a “ciphertext”
  - **Decryption:** What Bob does with the ciphertext and the key to get the message out of it
- All of these are (probabilistic) computations



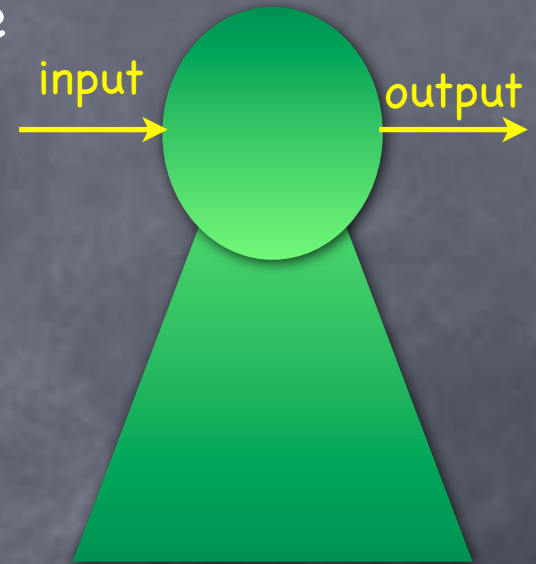
# Modeling Computation





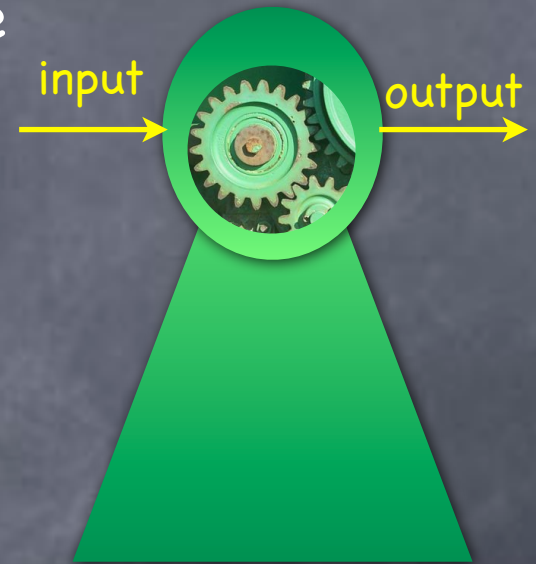
# Modeling Computation

- In our model (standard model) parties are programs (computations, say Turing Machines)



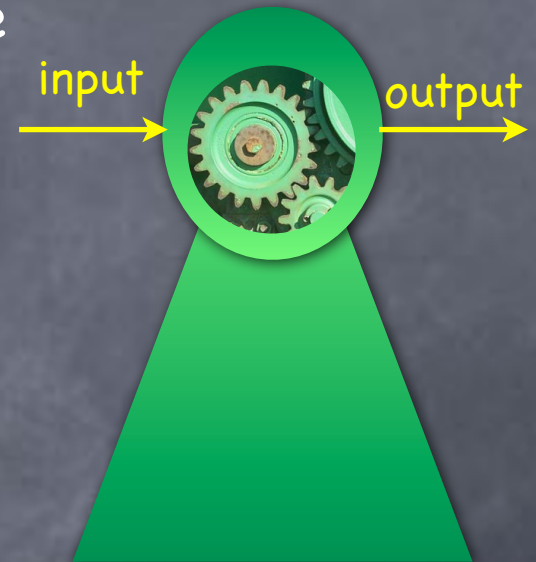
# Modeling Computation

- In our model (standard model) parties are programs (computations, say Turing Machines)



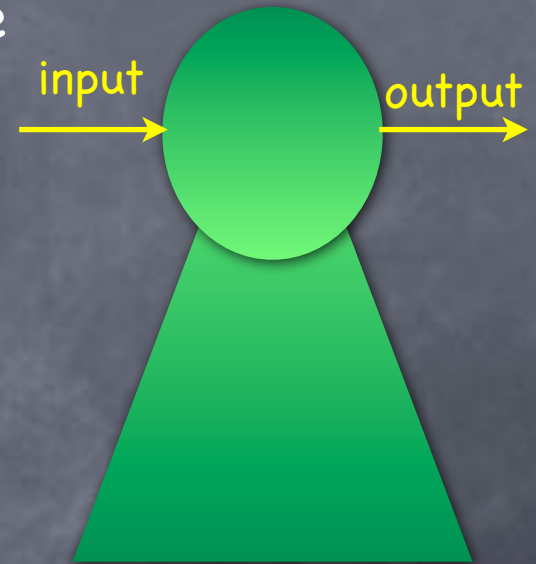
# Modeling Computation

- In our model (standard model) parties are programs (computations, say Turing Machines)
- Effect of computation limited to be in a blackbox manner (only through input/output functionality)



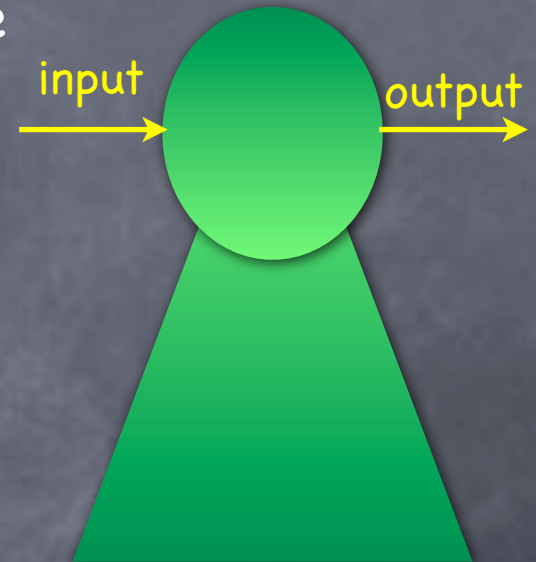
# Modeling Computation

- In our model (standard model) parties are programs (computations, say Turing Machines)
- Effect of computation limited to be in a blackbox manner (only through input/output functionality)



# Modeling Computation

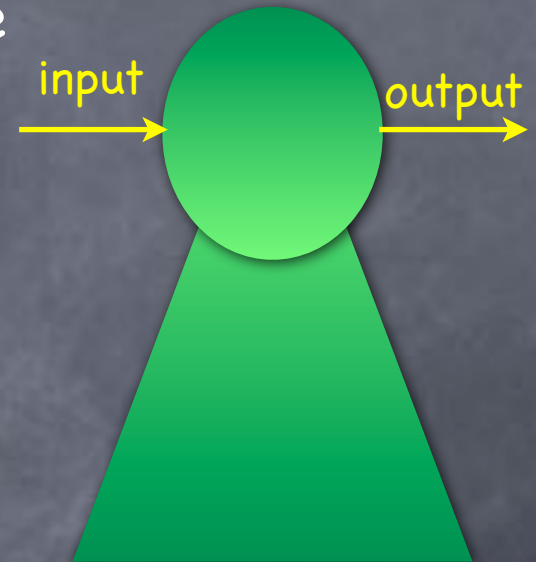
- In our model (standard model) parties are programs (computations, say Turing Machines)
- Effect of computation limited to be in a blackbox manner (only through input/output functionality)
- No side-information (timing, electric signals, ...) unless explicitly modeled





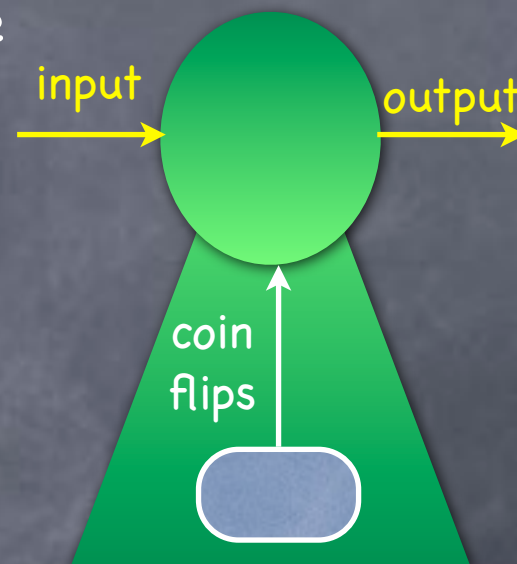
# Modeling Computation

- In our model (standard model) parties are programs (computations, say Turing Machines)
- Effect of computation limited to be in a blackbox manner (only through input/output functionality)
  - No side-information (timing, electric signals, ...) unless explicitly modeled
  - Can be probabilistic



# Modeling Computation

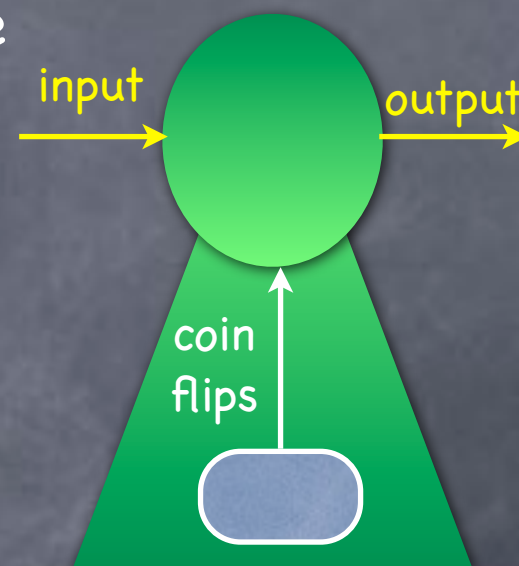
- In our model (standard model) parties are programs (computations, say Turing Machines)
- Effect of computation limited to be in a blackbox manner (only through input/output functionality)
  - No side-information (timing, electric signals, ...) unless explicitly modeled
  - Can be probabilistic





# Modeling Computation

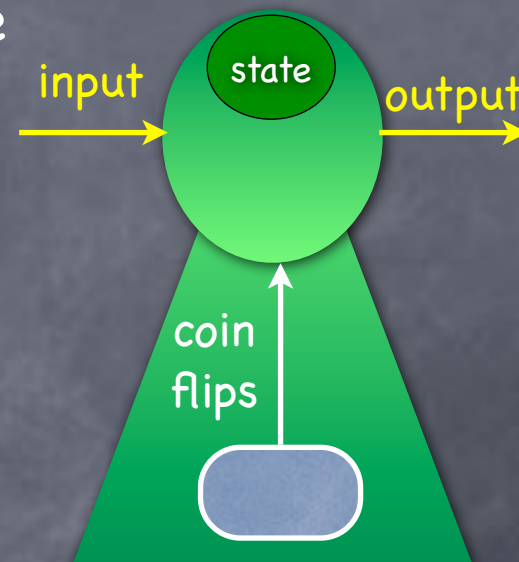
- In our model (standard model) parties are programs (computations, say Turing Machines)
- Effect of computation limited to be in a blackbox manner (only through input/output functionality)
  - No side-information (timing, electric signals, ...) unless explicitly modeled
  - Can be probabilistic



Ideal coin flips: If  $n$  coins flipped, each outcome has probability  $2^{-n}$

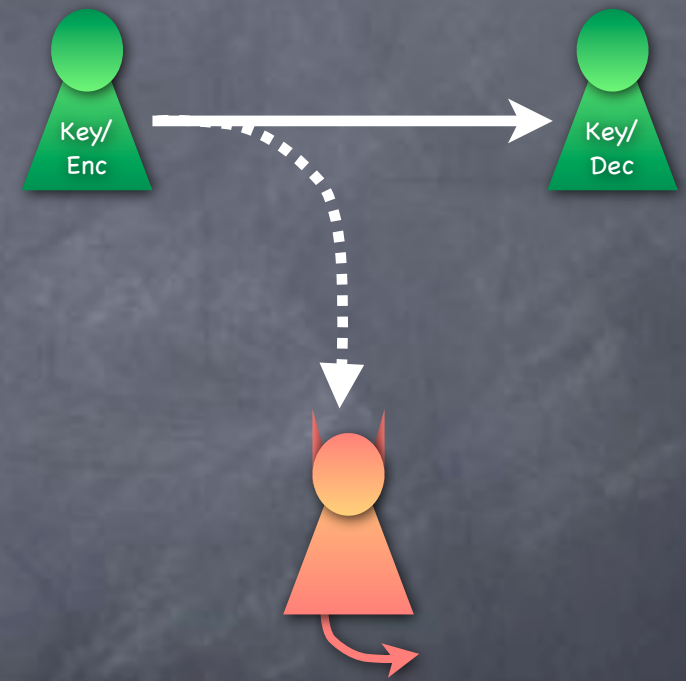
# Modeling Computation

- In our model (standard model) parties are programs (computations, say Turing Machines)
- Effect of computation limited to be in a blackbox manner (only through input/output functionality)
  - No side-information (timing, electric signals, ...) unless explicitly modeled
  - Can be probabilistic
  - Sometimes stateful



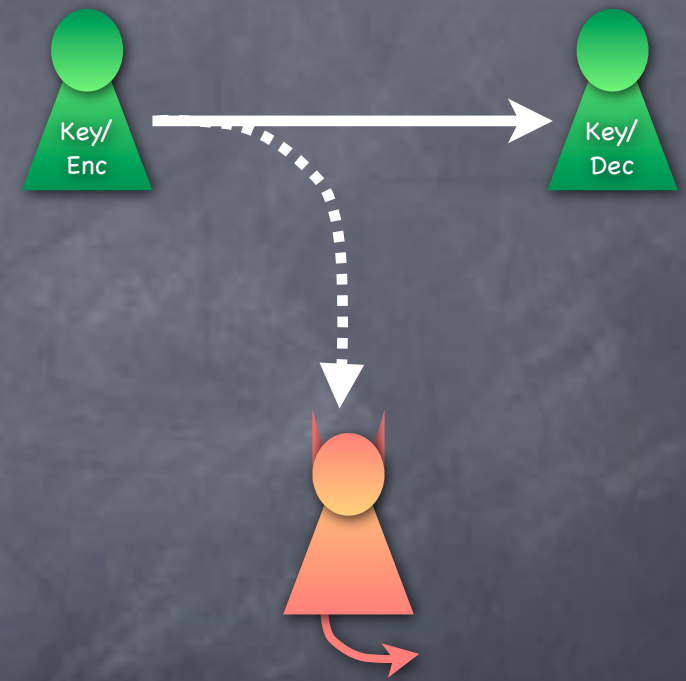
Ideal coin flips: If  $n$  coins flipped, each outcome has probability  $2^{-n}$

# The Environment



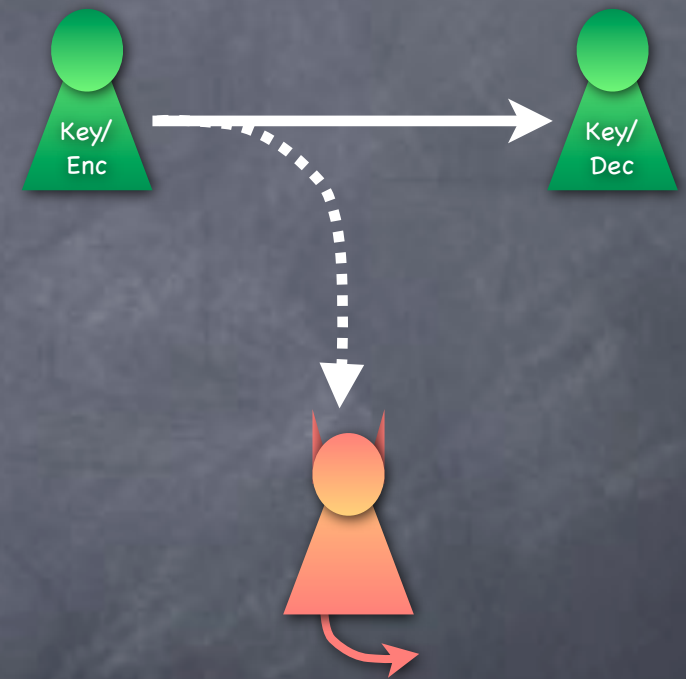
# The Environment

- Where does the message come from?



# The Environment

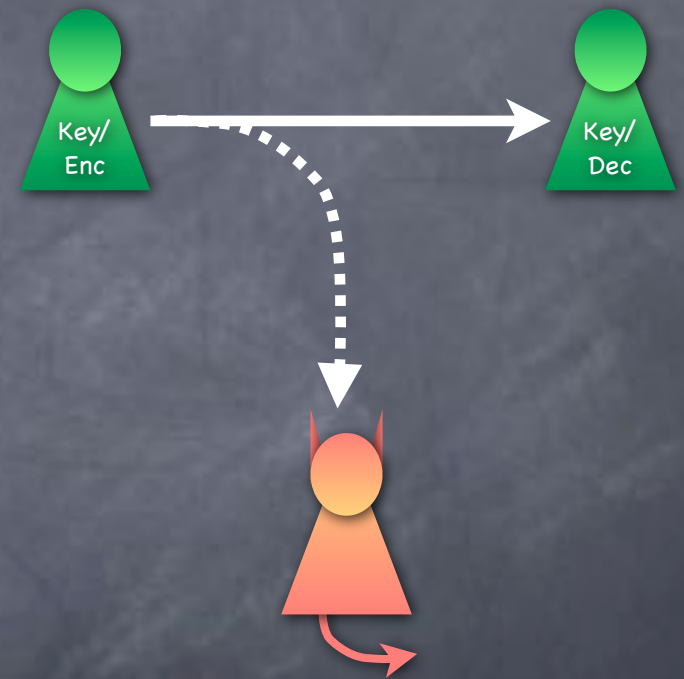
- Where does the message come from?
- Eve might already have partial information about the message, or might receive such information later





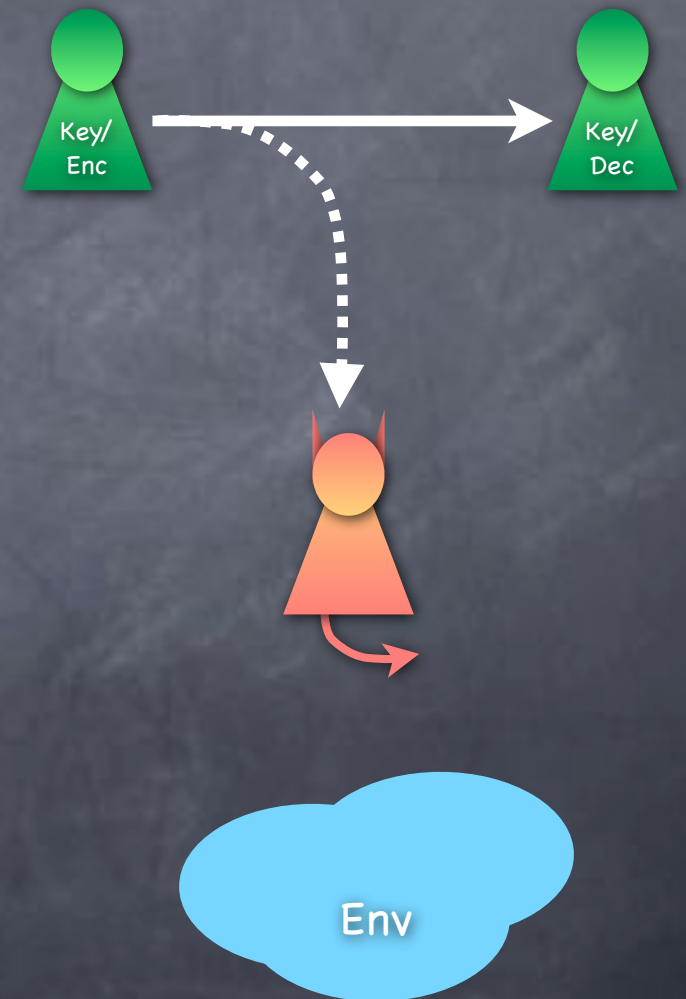
# The Environment

- Where does the message come from?
- Eve might already have partial information about the message, or might receive such information later
- In fact, Eve might influence the choice of the message



# The Environment

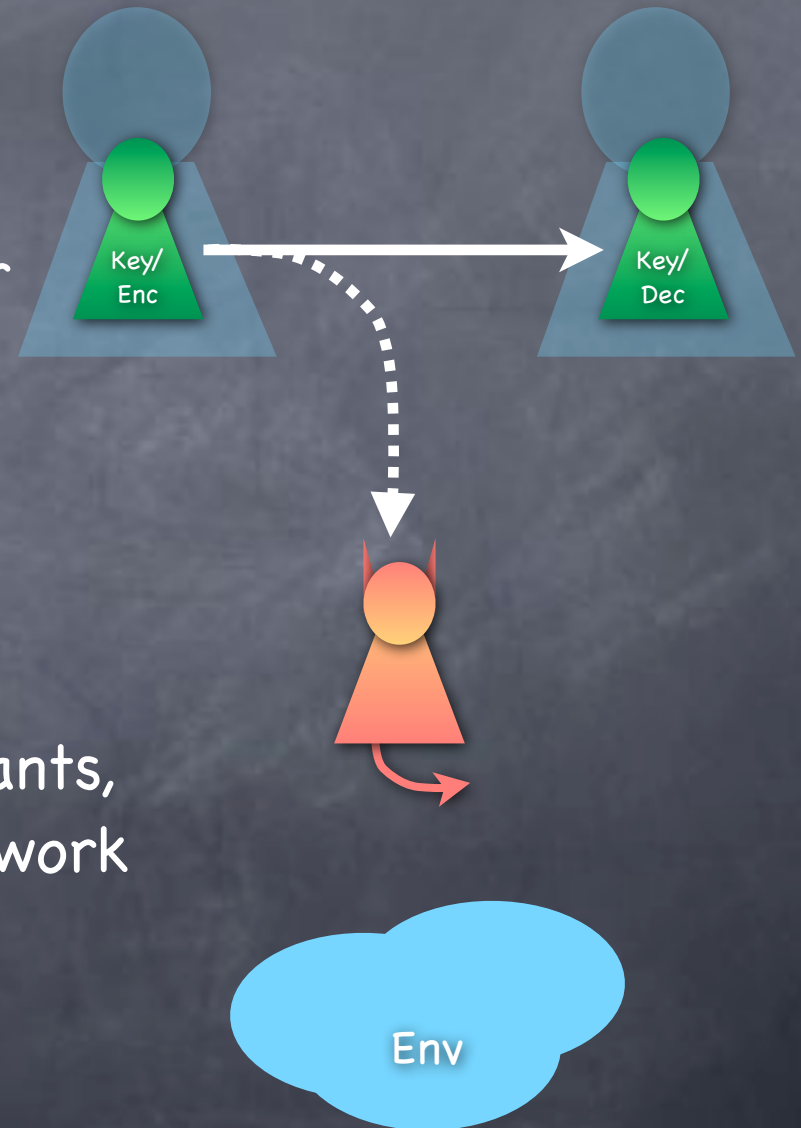
- Where does the message come from?
- Eve might already have partial information about the message, or might receive such information later
- In fact, Eve might influence the choice of the message
- The environment





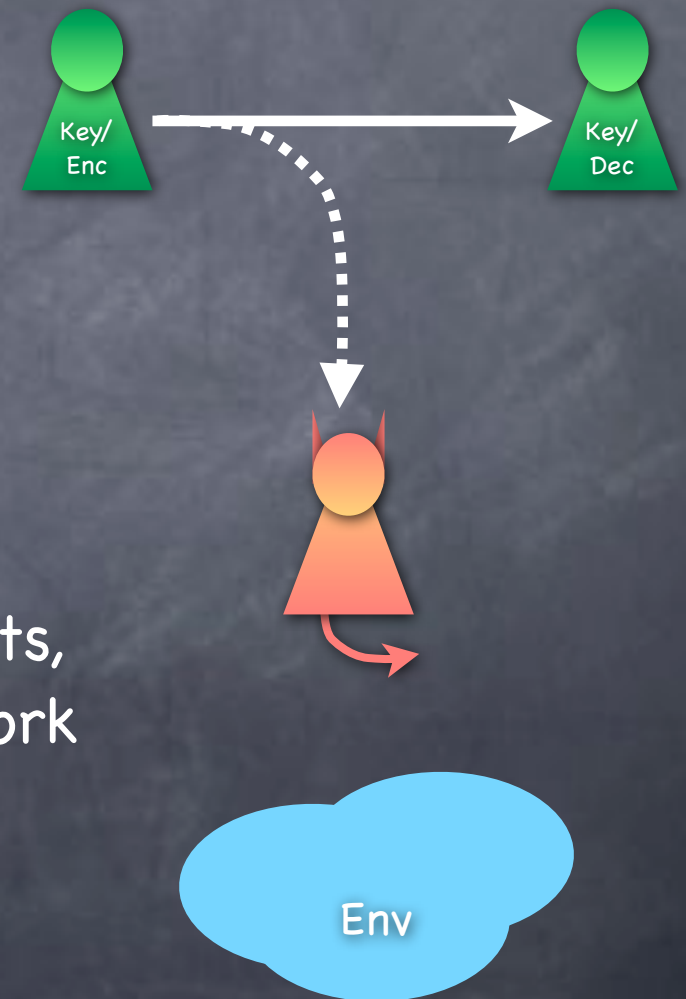
# The Environment

- Where does the message come from?
  - Eve might already have partial information about the message, or might receive such information later
  - In fact, Eve might influence the choice of the message
- The environment
  - Includes the operating systems and other programs run by the participants, as well as other parties, if in a network



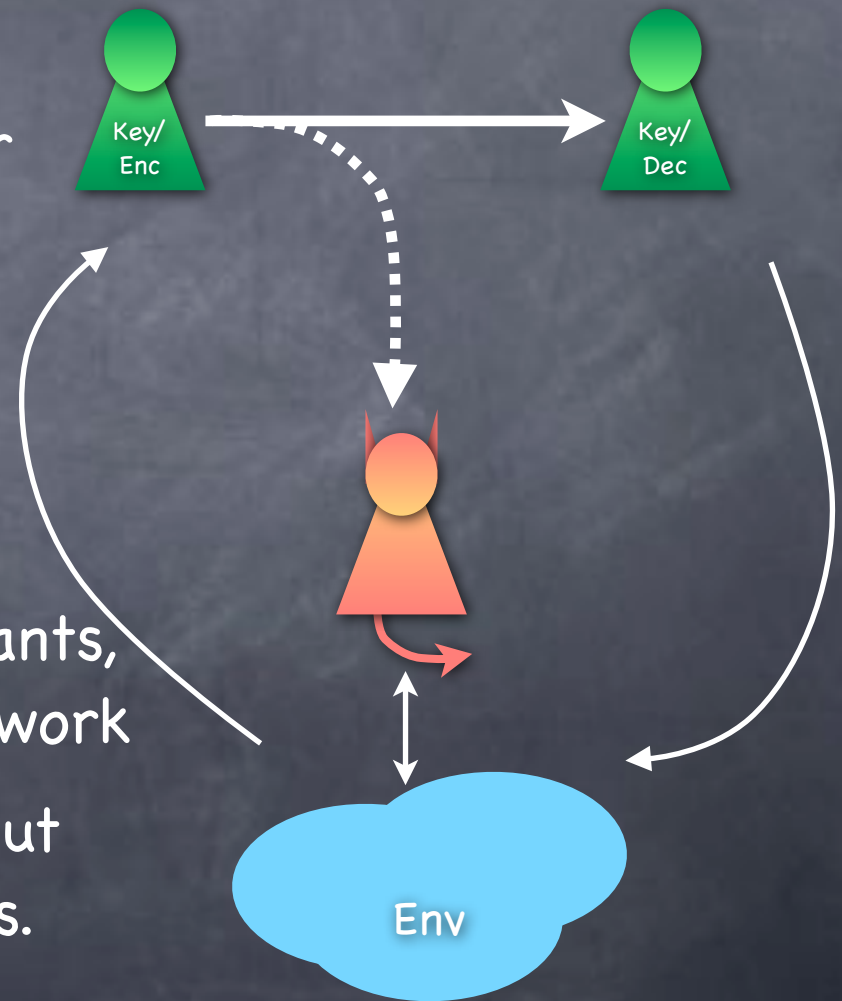
# The Environment

- Where does the message come from?
  - Eve might already have partial information about the message, or might receive such information later
  - In fact, Eve might influence the choice of the message
- The environment
  - Includes the operating systems and other programs run by the participants, as well as other parties, if in a network

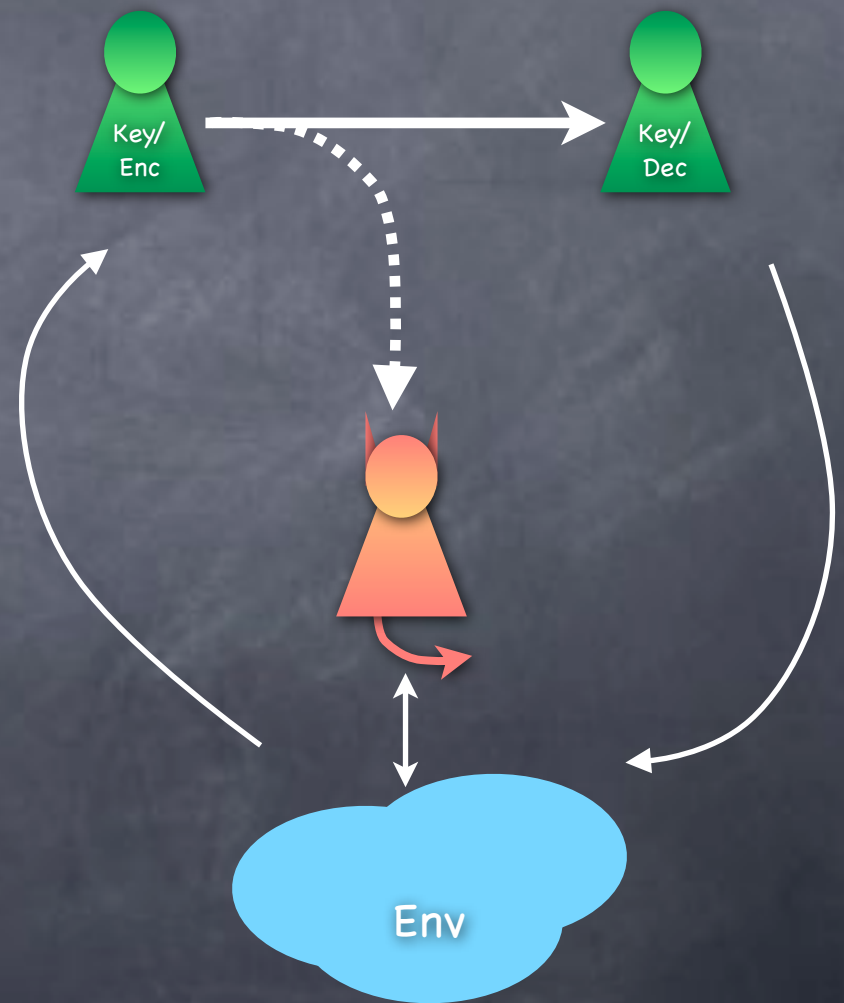


# The Environment

- Where does the message come from?
  - Eve might already have partial information about the message, or might receive such information later
  - In fact, Eve might influence the choice of the message
- The environment
  - Includes the operating systems and other programs run by the participants, as well as other parties, if in a network
  - Abstract entity from which the input comes and to which the output goes. Arbitrarily influenced by Eve

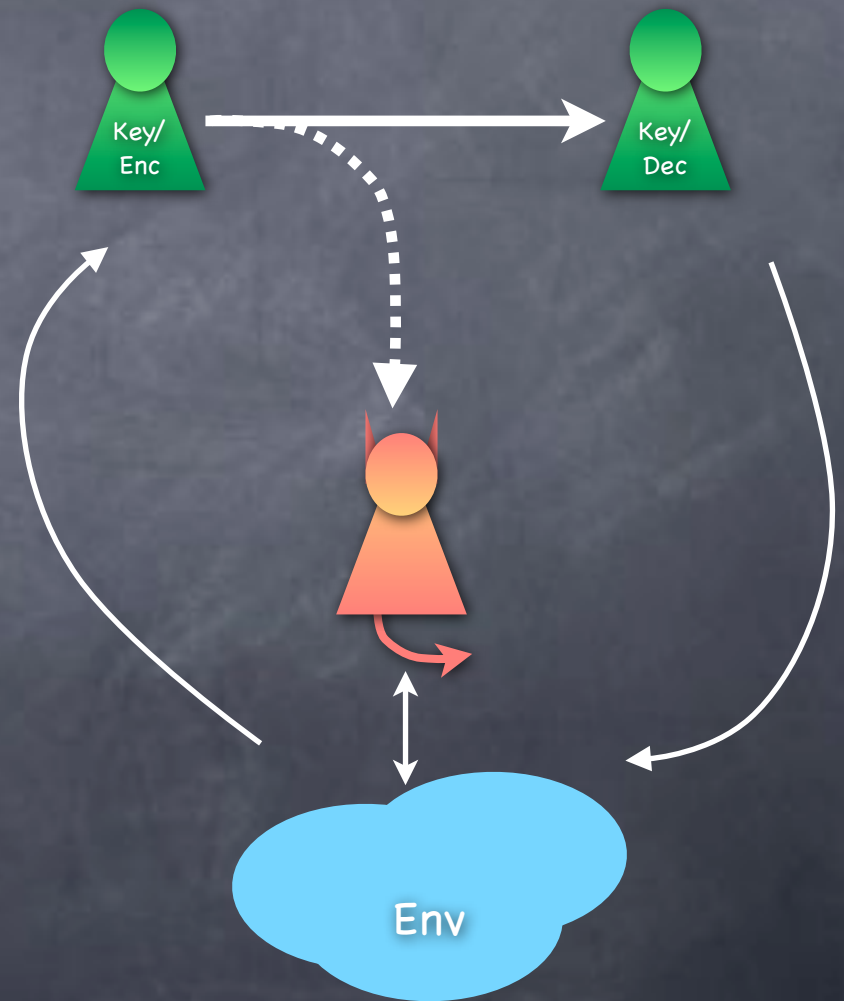


# Defining Security



# Defining Security

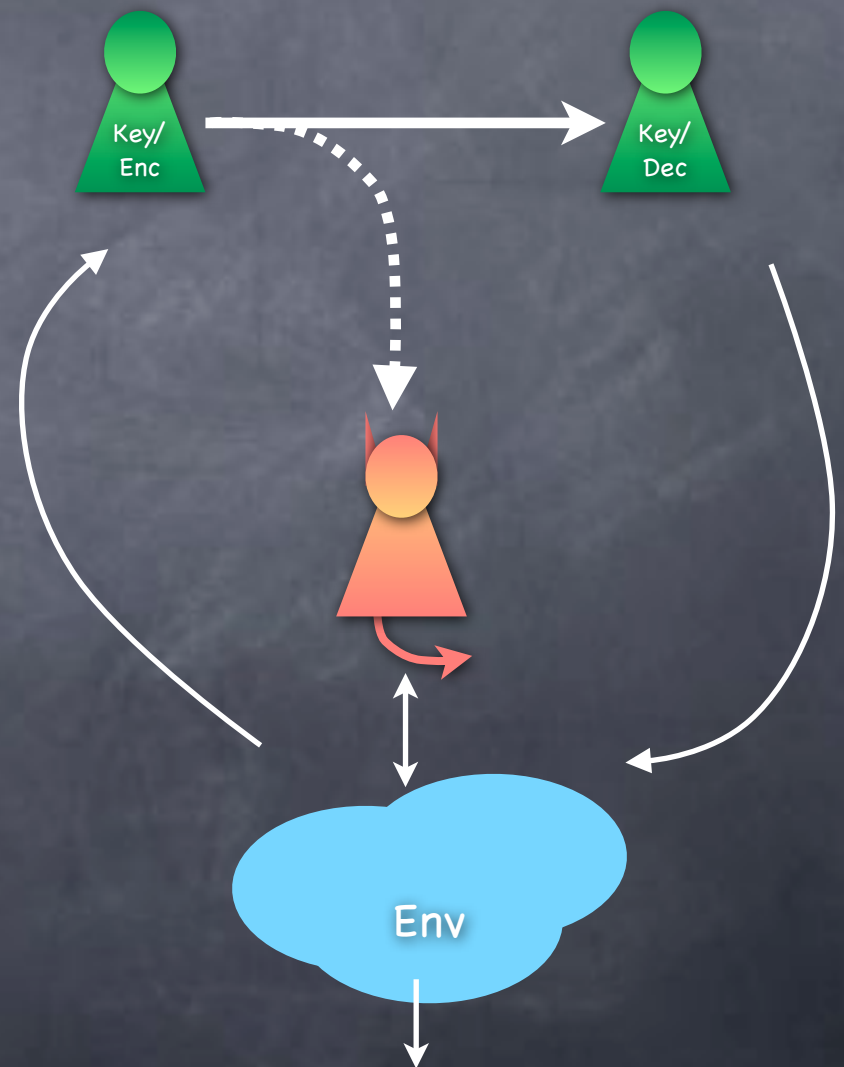
- Eve shouldn't be able to produce any "bad effects" in any environment





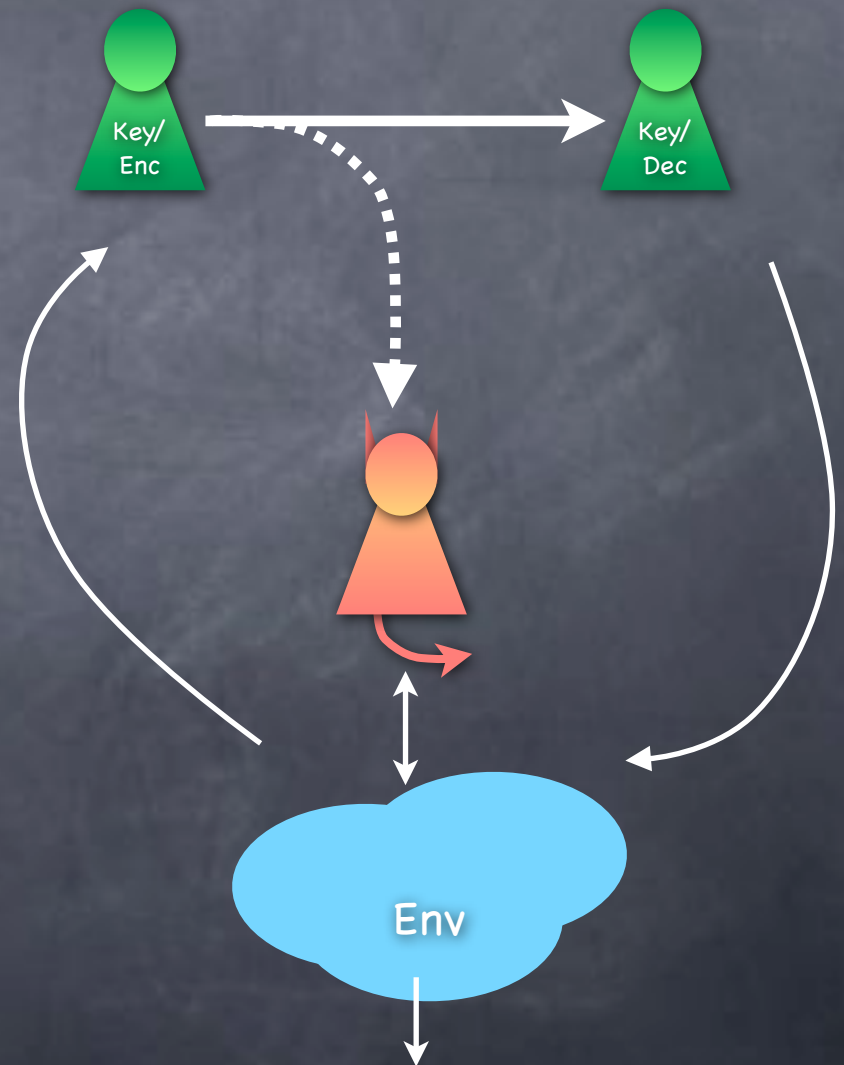
# Defining Security

- Eve shouldn't be able to produce any "bad effects" in any environment
- Effects in the environment: modeled as a bit in the environment (called the output bit)



# Defining Security

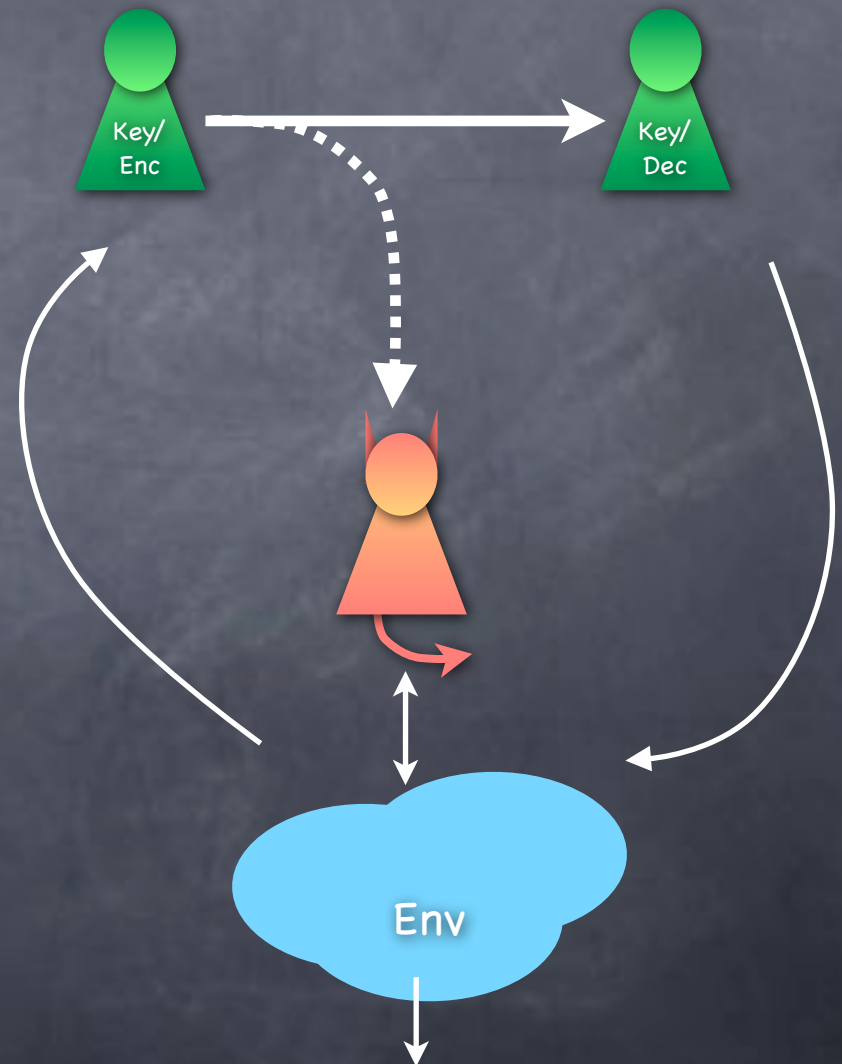
- Eve shouldn't be able to produce any "bad effects" in any environment
- Effects in the environment: modeled as a bit in the environment (called the output bit)
- What is bad?





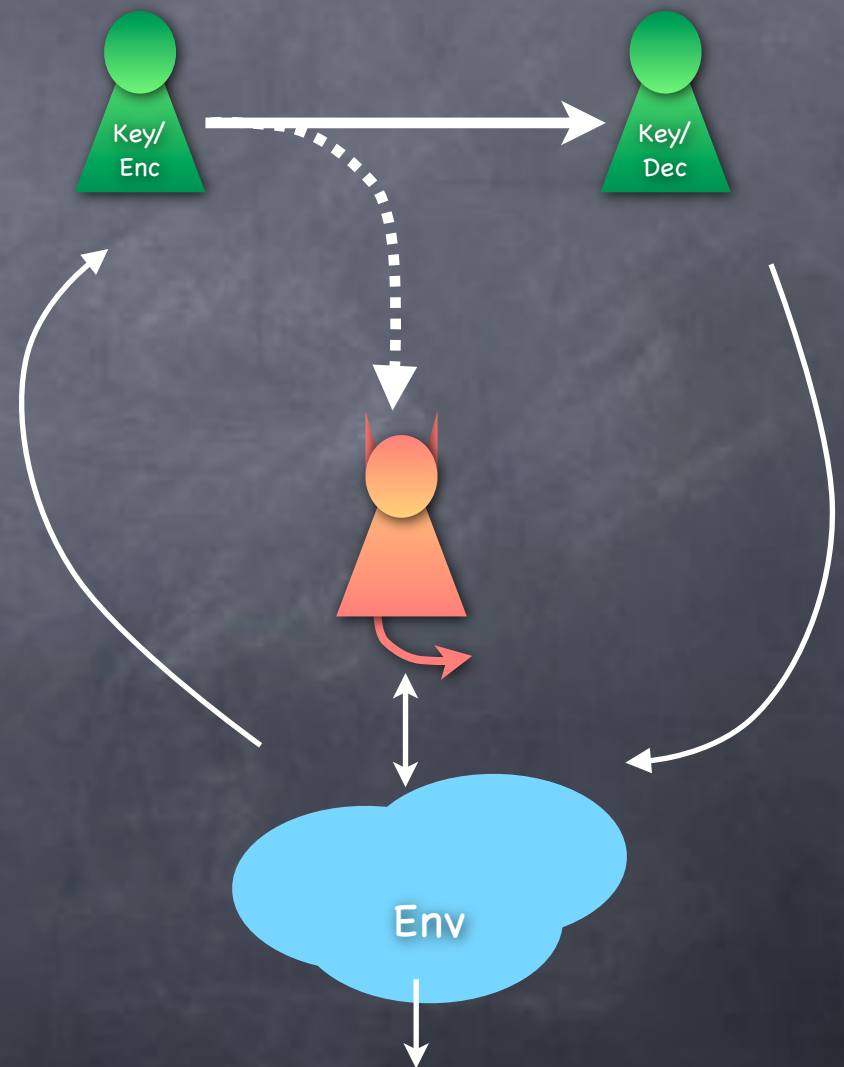
# Defining Security

- Eve shouldn't be able to produce any "bad effects" in any environment
- Effects in the environment: modeled as a bit in the environment (called the output bit)
- What is bad?
  - Anything that Eve couldn't have caused if an "ideal channel" was used



# Defining Security

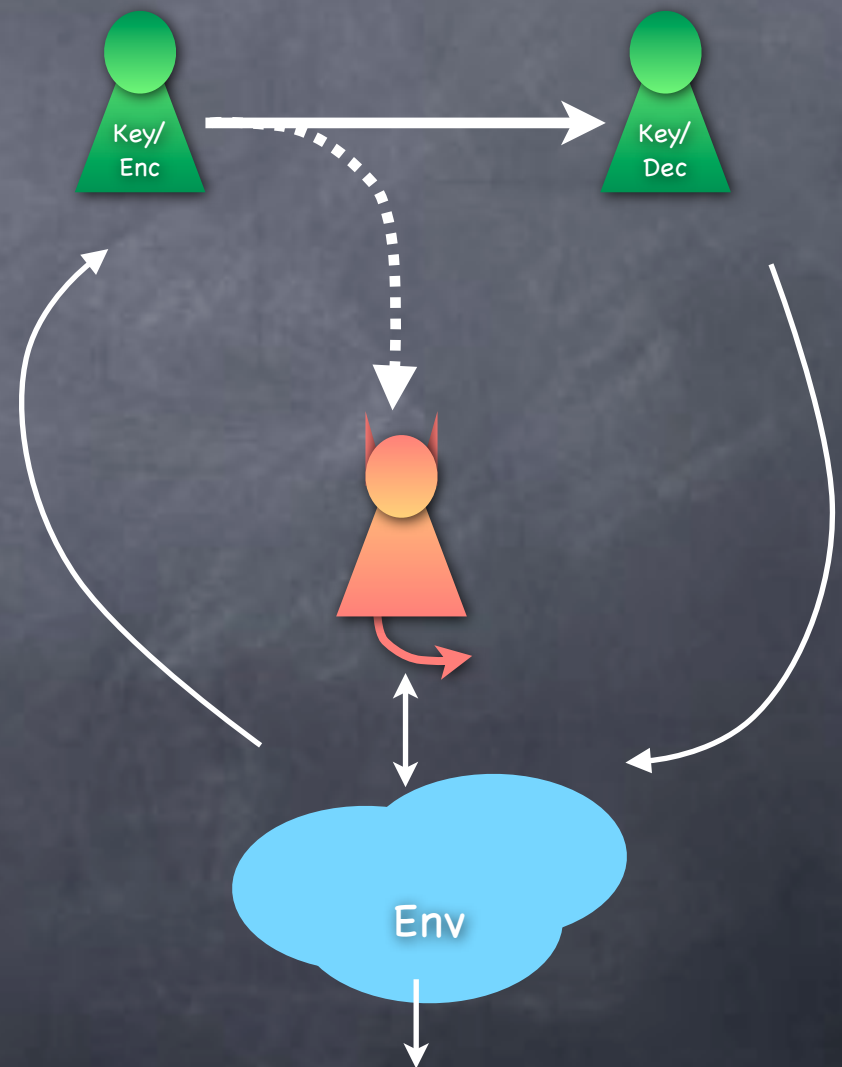
## The REAL/IDEAL Paradigm



# Defining Security

## The REAL/IDEAL Paradigm

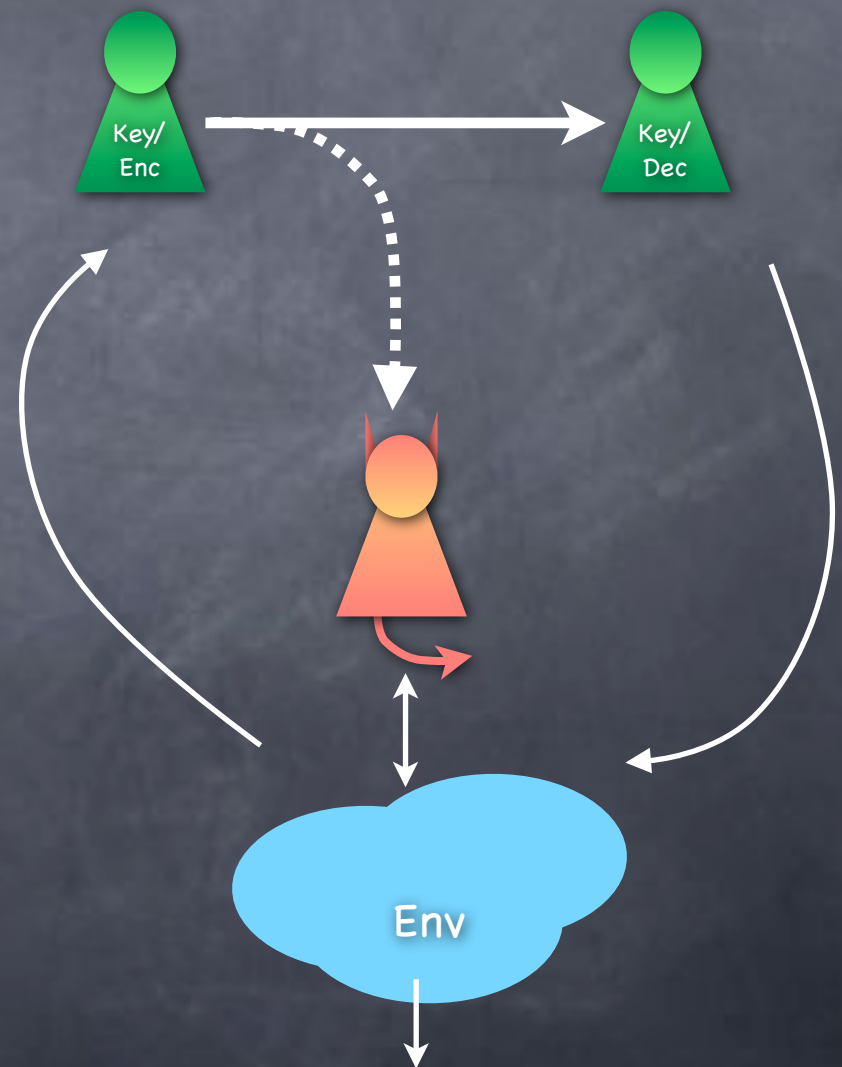
- Eve shouldn't produce any more effects than she could have in the ideal world



# Defining Security

## The REAL/IDEAL Paradigm

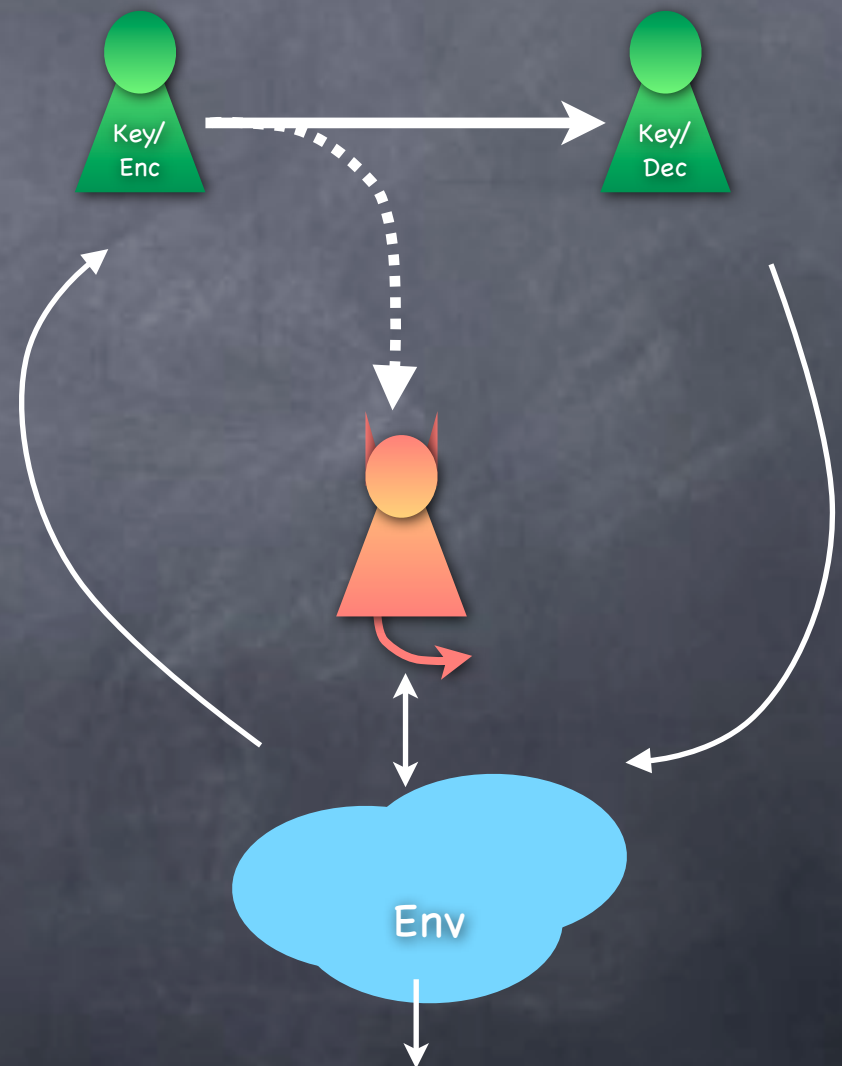
- Eve shouldn't produce any more effects than she could have in the ideal world
- **IDEAL world:** Message sent over a (physically) secure channel. No encryption in this world.



# Defining Security

## The REAL/IDEAL Paradigm

- Eve shouldn't produce any more effects than she could have in the ideal world
- **IDEAL world:** Message sent over a (physically) secure channel. No encryption in this world.
- **REAL world:** Using encryption

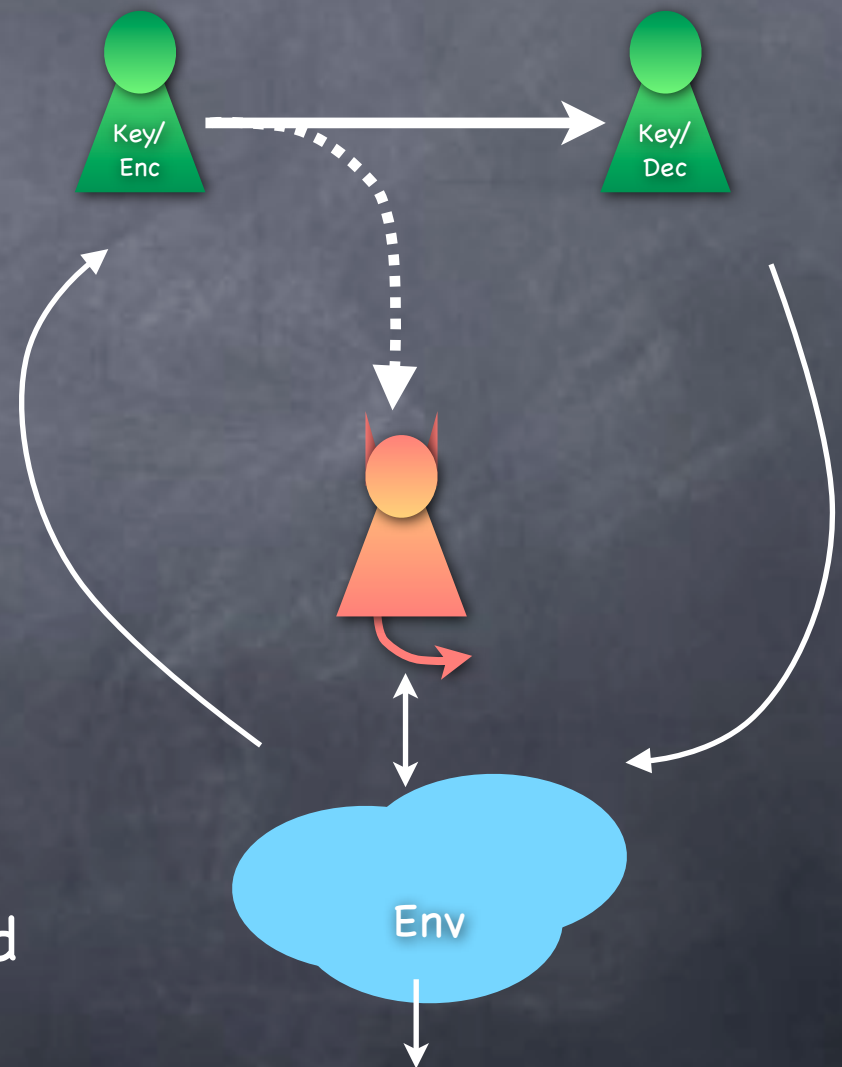




# Defining Security

## The REAL/IDEAL Paradigm

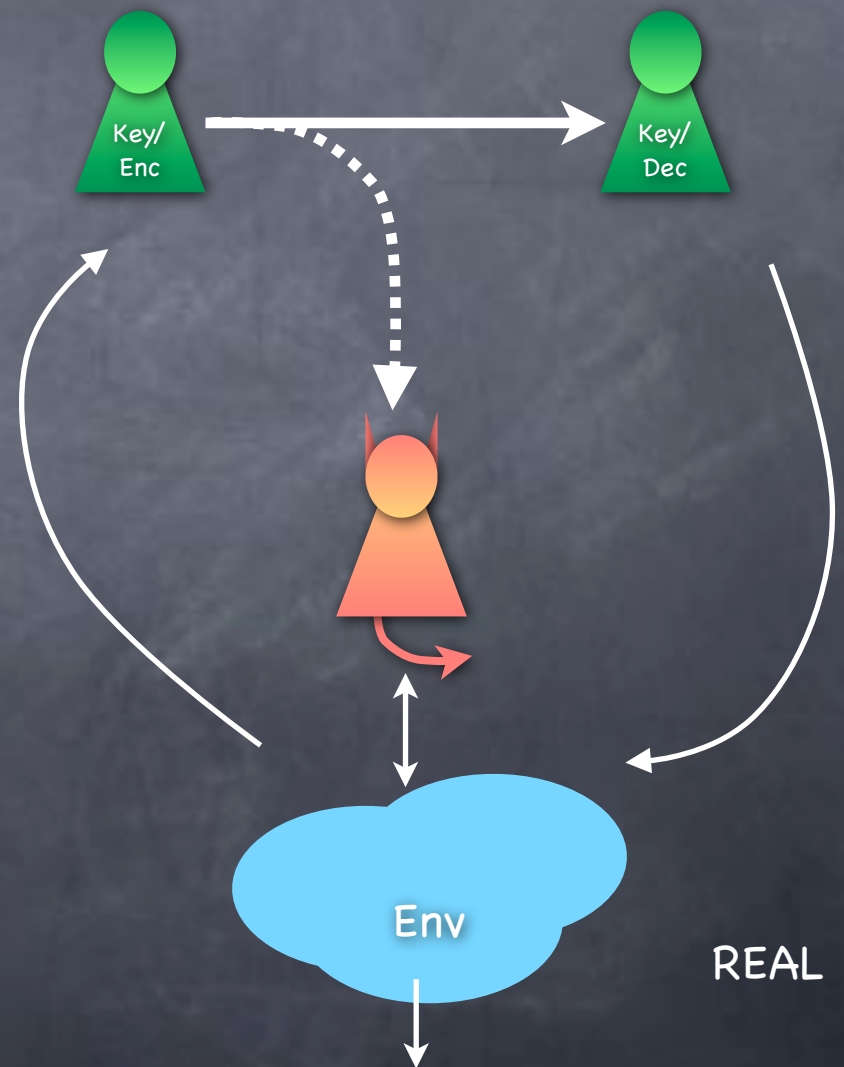
- Eve shouldn't produce any more effects than she could have in the ideal world
  - **IDEAL world:** Message sent over a (physically) secure channel. No encryption in this world.
  - **REAL world:** Using encryption
  - Encryption is **secure if** whatever an Eve can do in the REAL world, an Eve' can do in the IDEAL world





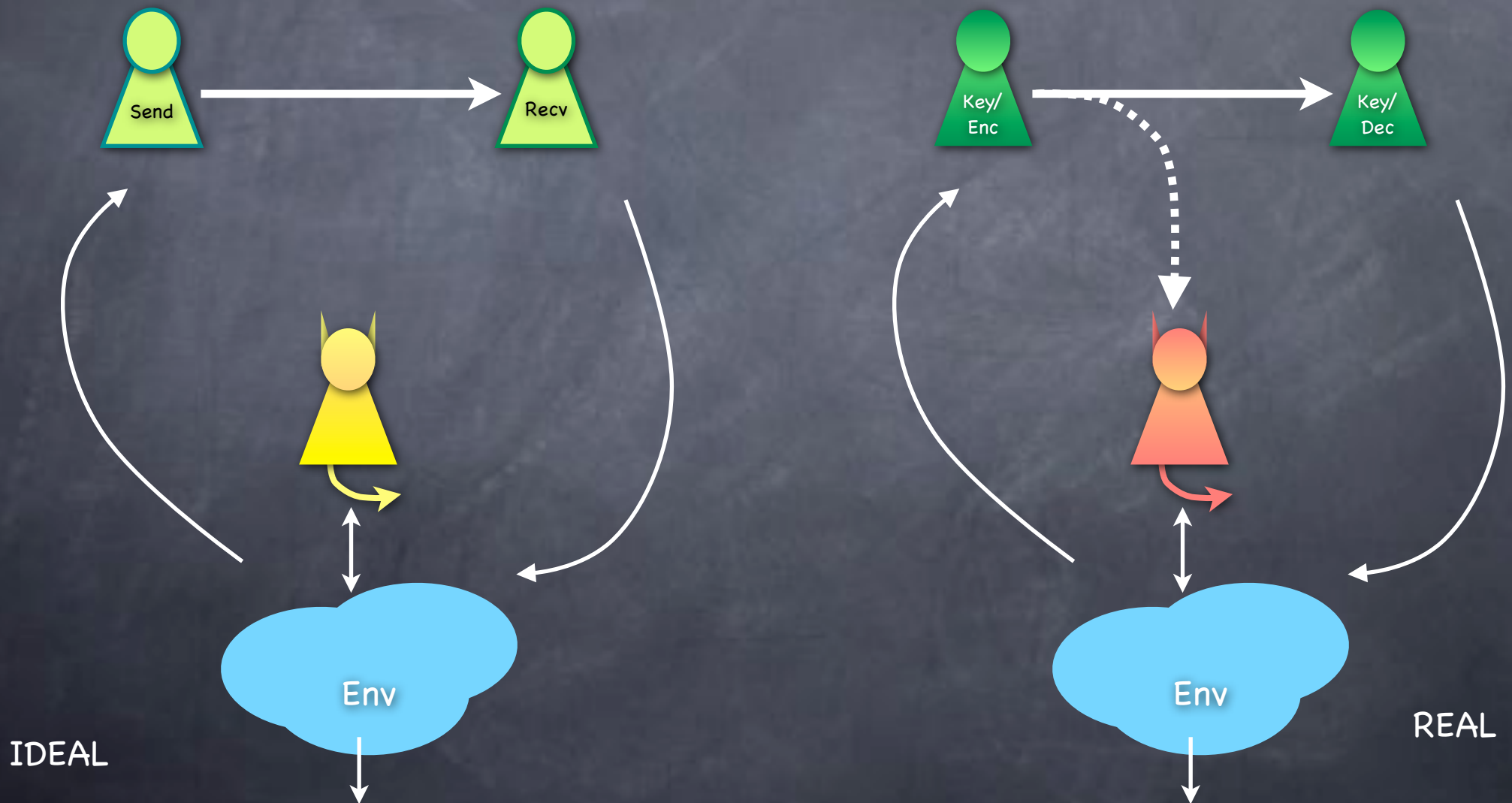
# Defining Security

## The REAL/IDEAL Paradigm



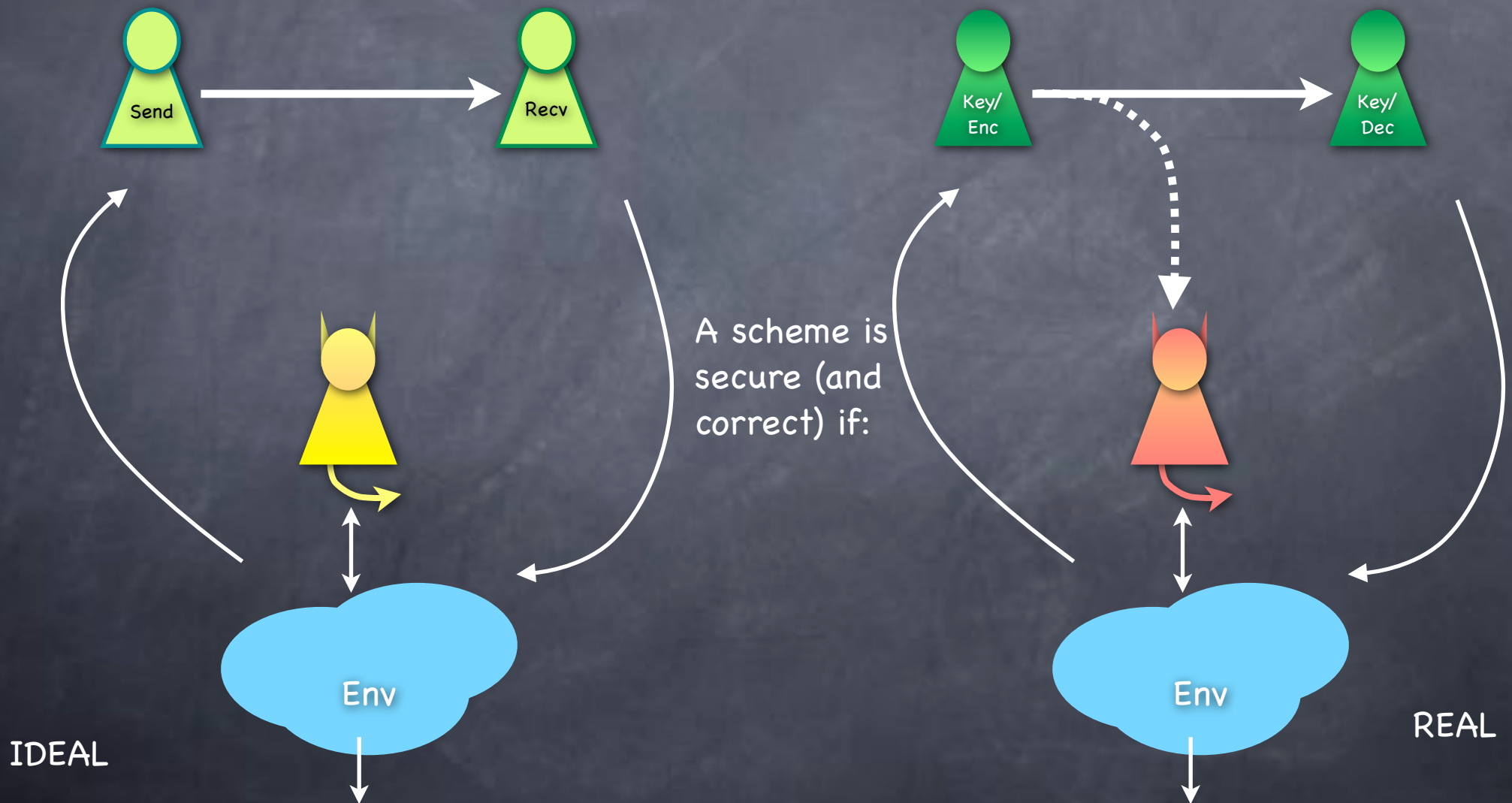
# Defining Security

## The REAL/IDEAL Paradigm



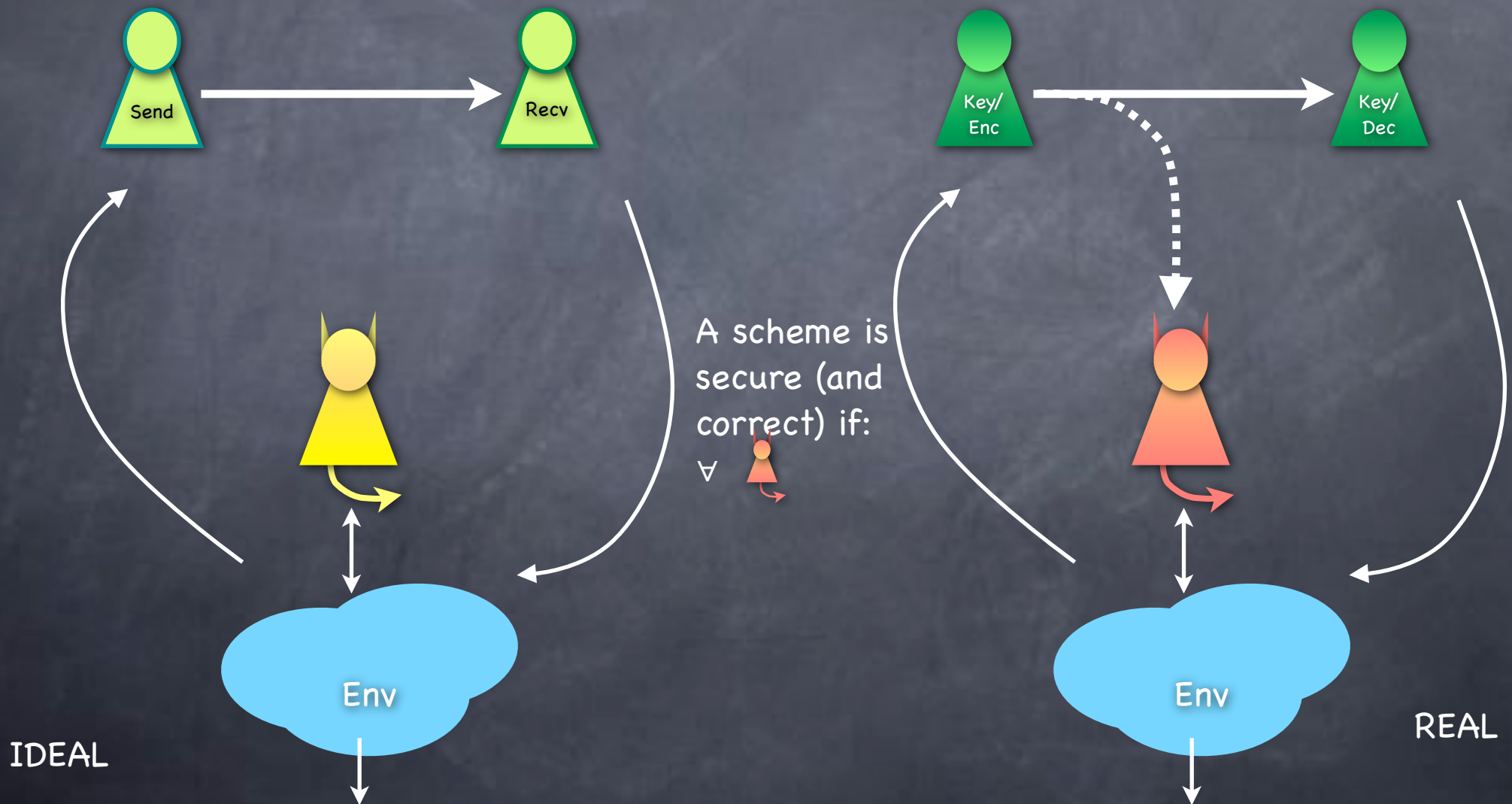
# Defining Security

## The REAL/IDEAL Paradigm



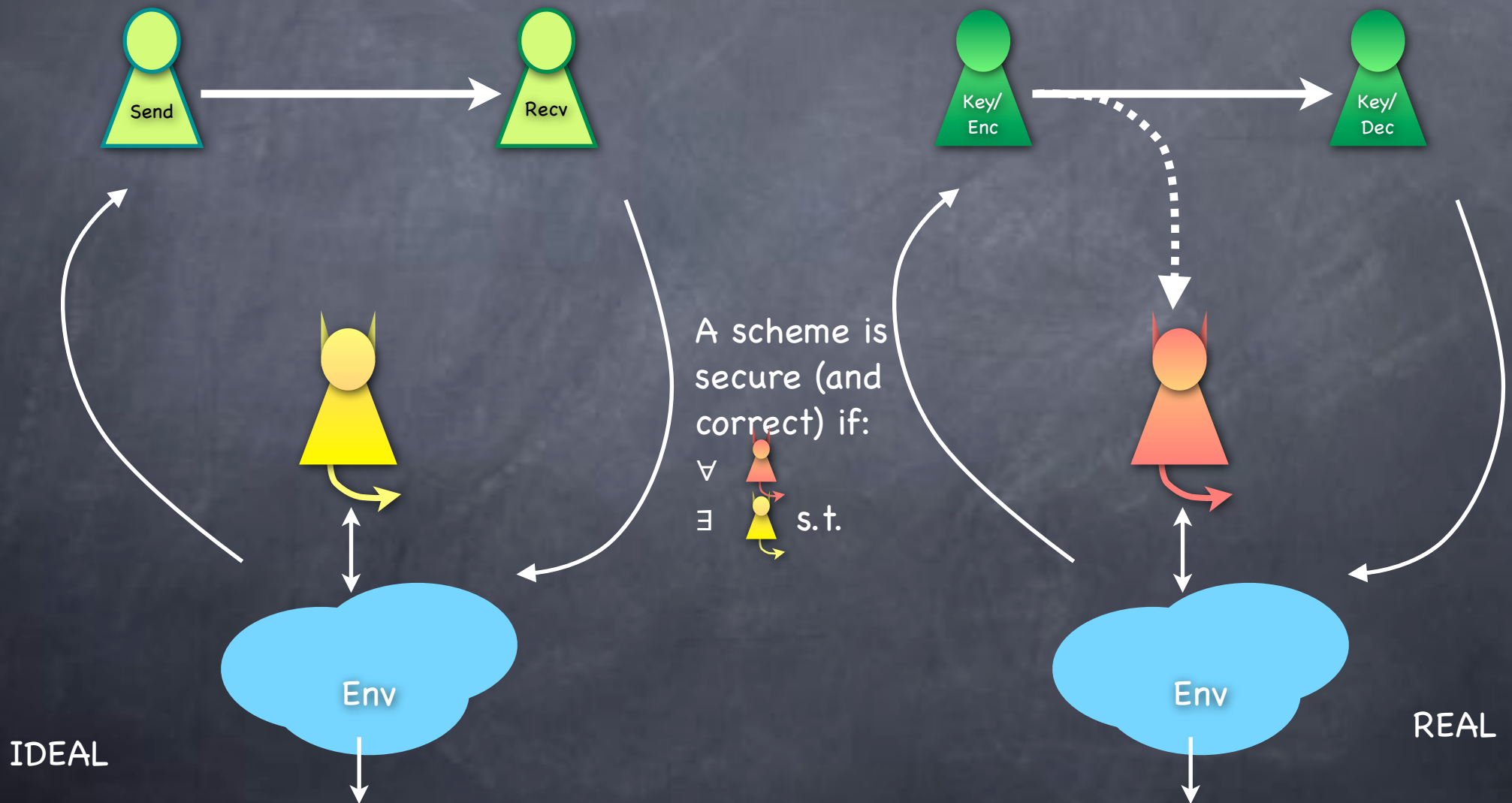
# Defining Security

## The REAL/IDEAL Paradigm



# Defining Security

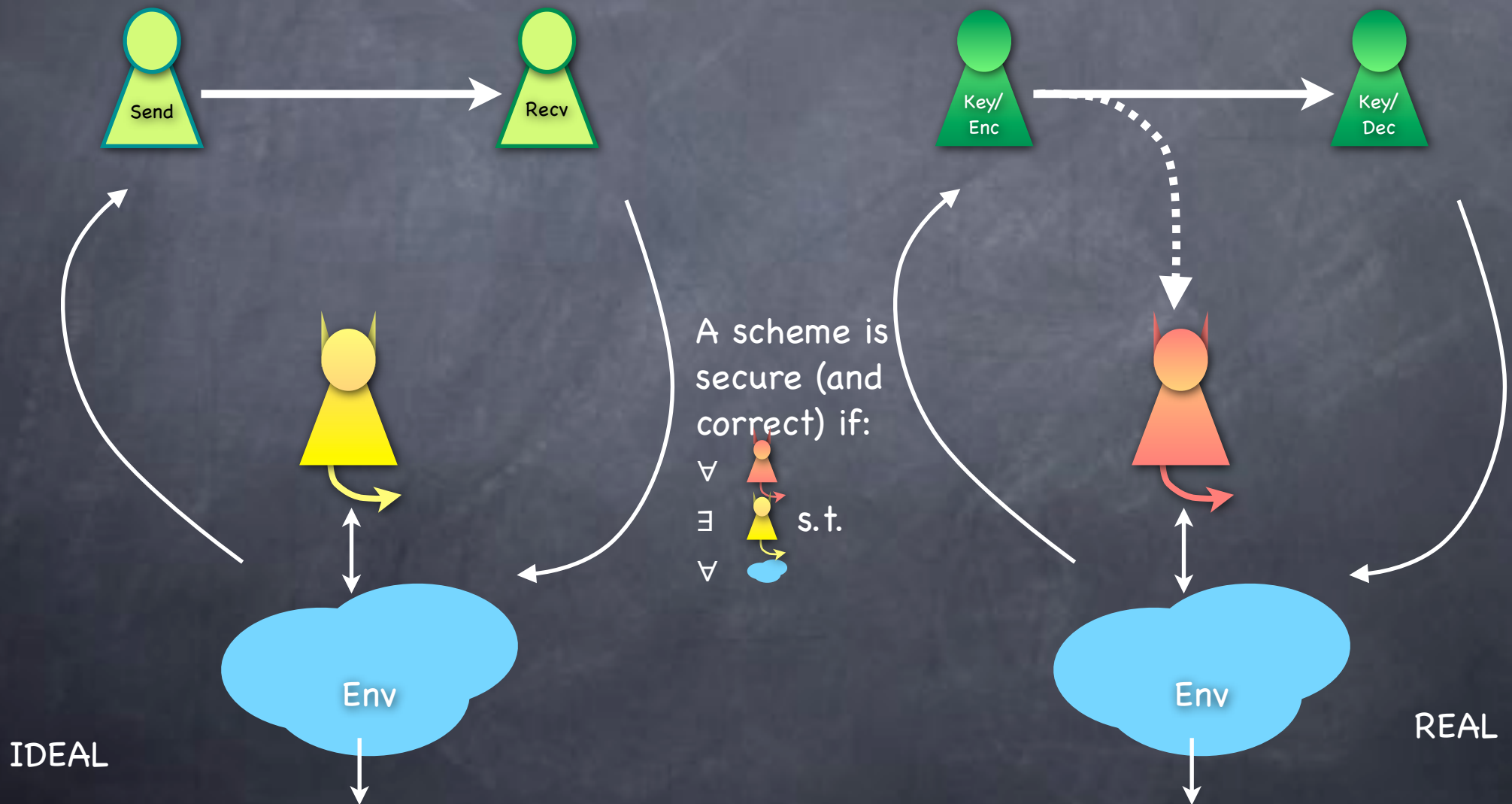
## The REAL/IDEAL Paradigm





# Defining Security

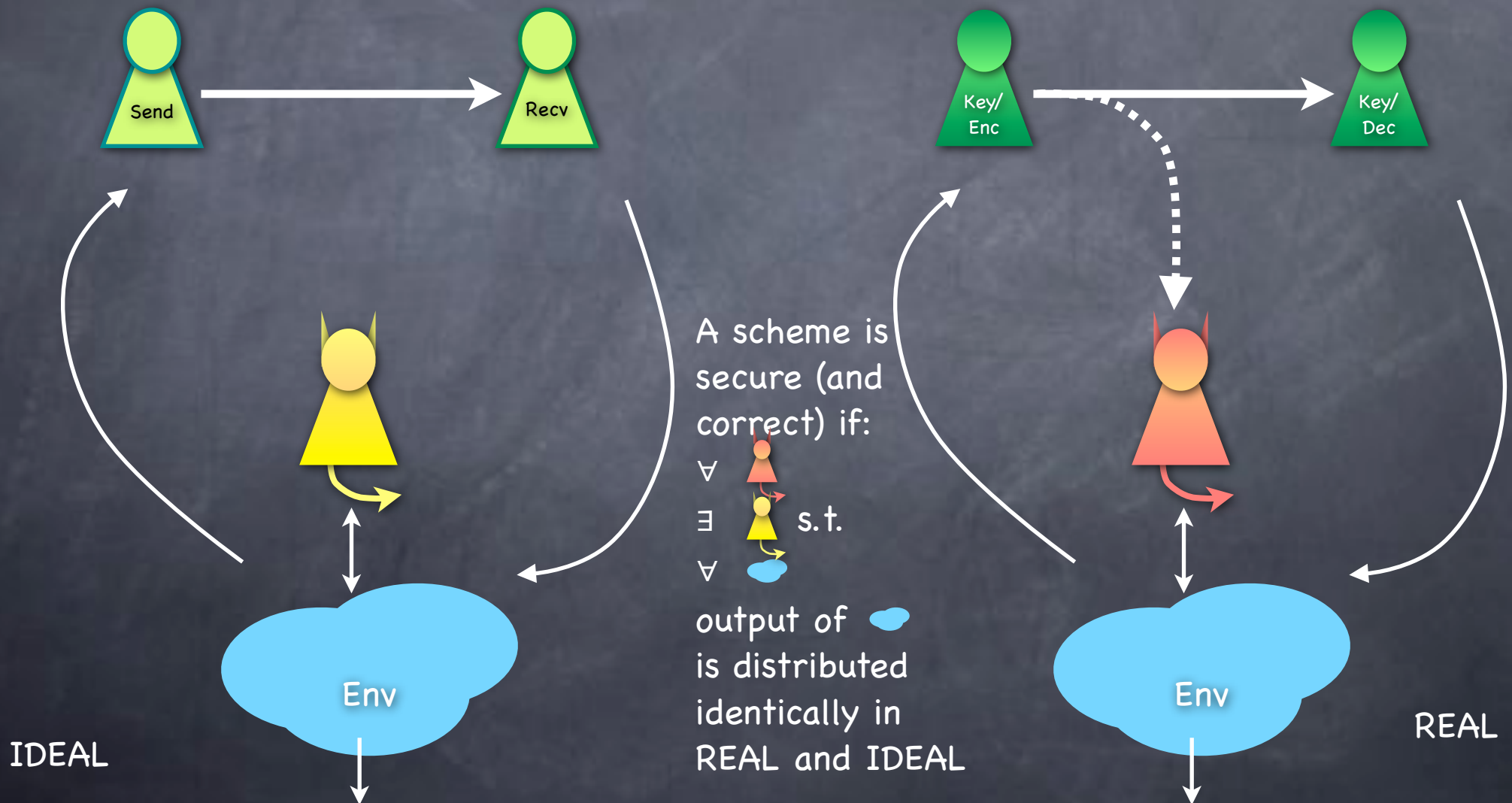
## The REAL/IDEAL Paradigm





# Defining Security

## The REAL/IDEAL Paradigm



Ready to go...

# Ready to go...

- REAL/IDEAL (a.k.a simulation-based) security forms the basic template for a large variety of security definitions

# Ready to go...

- REAL/IDEAL (a.k.a simulation-based) security forms the basic template for a large variety of security definitions
- We will see three definitions of encryption

# Ready to go...

- REAL/IDEAL (a.k.a simulation-based) security forms the basic template for a large variety of security definitions
- We will see three definitions of encryption
  - Security of “one-time encryption”

# Ready to go...

- REAL/IDEAL (a.k.a simulation-based) security forms the basic template for a large variety of security definitions
- We will see three definitions of encryption
  - Security of “one-time encryption”
  - Security of (muti-message) encryption



# Ready to go...

- REAL/IDEAL (a.k.a simulation-based) security forms the basic template for a large variety of security definitions
- We will see three definitions of encryption
  - Security of “one-time encryption”
  - Security of (muti-message) encryption
  - Security against “active attacks”

# Ready to go...

- REAL/IDEAL (a.k.a simulation-based) security forms the basic template for a large variety of security definitions
- We will see three definitions of encryption
  - Security of “one-time encryption”
  - Security of (muti-message) encryption
  - Security against “active attacks”
- Will also see alternate (but essentially equivalent) security definitions

# Onetime Encryption

# Onetime Encryption

## The Syntax

- Shared-key (Private-key) Encryption
  - **Key Generation:** Randomized
    - $K \leftarrow \mathcal{K}$ , uniformly randomly drawn from the key-space (or according to a key-distribution)
  - **Encryption:** Deterministic
    - $\text{Enc}: \mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$
  - **Decryption:** Deterministic
    - $\text{Dec}: \mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$

# Onetime Encryption

## Perfect Secrecy

# Onetime Encryption

## Perfect Secrecy

- For all messages  $m, m'$  in  $\mathcal{M}$ 
  - $\{\text{Enc}(m, K)\}_{K \leftarrow \text{KeyGen}} = \{\text{Enc}(m', K)\}_{K \leftarrow \text{KeyGen}}$



# Onetime Encryption

## Perfect Secrecy

- For all messages  $m, m'$  in  $\mathcal{M}$ 
  - $\{\text{Enc}(m, K)\}_{K \leftarrow \text{KeyGen}} = \{\text{Enc}(m', K)\}_{K \leftarrow \text{KeyGen}}$

| $\mathcal{M} \backslash \mathcal{K}$ | 0 | 1 | 2 | 3 |
|--------------------------------------|---|---|---|---|
| a                                    | x | y | y | z |
| b                                    | y | x | z | y |

# Onetime Encryption

## Perfect Secrecy

- For all messages  $m, m'$  in  $\mathcal{M}$ 
  - $\{\text{Enc}(m, K)\}_{K \leftarrow \text{KeyGen}} = \{\text{Enc}(m', K)\}_{K \leftarrow \text{KeyGen}}$
- Distribution of ciphertext is defined by the randomness in the key

| $\mathcal{M} \backslash \mathcal{K}$ | 0 | 1 | 2 | 3 |
|--------------------------------------|---|---|---|---|
| a                                    | x | y | y | z |
| b                                    | y | x | z | y |

# Onetime Encryption

## Perfect Secrecy

- For all messages  $m, m'$  in  $\mathcal{M}$ 
  - $\{\text{Enc}(m, K)\}_{K \leftarrow \text{KeyGen}} = \{\text{Enc}(m', K)\}_{K \leftarrow \text{KeyGen}}$
- Distribution of ciphertext is defined by the randomness in the key
- In addition, require correctness
  - $\forall m, K, \text{Dec}(\text{Enc}(m, K), K) = m$

| $\mathcal{M} \backslash \mathcal{K}$ | 0 | 1 | 2 | 3 |
|--------------------------------------|---|---|---|---|
| a                                    | x | y | y | z |
| b                                    | y | x | z | y |

# Onetime Encryption

## Perfect Secrecy

- For all messages  $m, m'$  in  $\mathcal{M}$ 
  - $\{\text{Enc}(m, K)\}_{K \leftarrow \text{KeyGen}} = \{\text{Enc}(m', K)\}_{K \leftarrow \text{KeyGen}}$
- Distribution of ciphertext is defined by the randomness in the key
- In addition, require correctness
  - $\forall m, K, \text{Dec}(\text{Enc}(m, K), K) = m$
- E.g. One-time pad:  $\mathcal{M} = \mathcal{K} = \mathcal{C} = \{0, 1\}^n$  and  $\text{Enc}(m, K) = m \oplus K, \text{Dec}(c, K) = c \oplus K$

| $\mathcal{M} \backslash \mathcal{K}$ | 0 | 1 | 2 | 3 |
|--------------------------------------|---|---|---|---|
| a                                    | x | y | y | z |
| b                                    | y | x | z | y |

# Onetime Encryption

## Perfect Secrecy

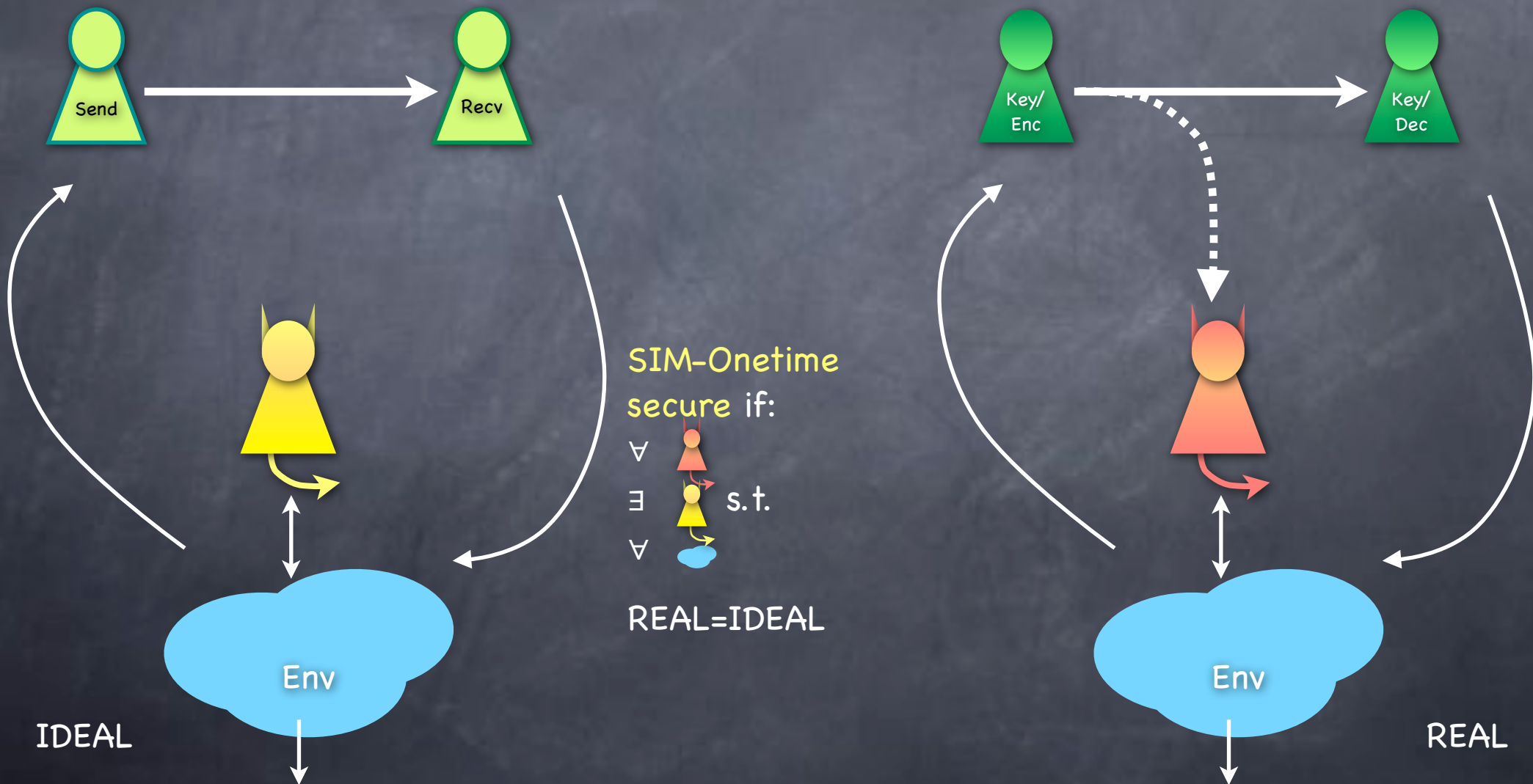
- For all messages  $m, m'$  in  $\mathcal{M}$ 
  - $\{\text{Enc}(m, K)\}_{K \leftarrow \text{KeyGen}} = \{\text{Enc}(m', K)\}_{K \leftarrow \text{KeyGen}}$
- Distribution of ciphertext is defined by the randomness in the key
- In addition, require correctness
  - $\forall m, K, \text{Dec}(\text{Enc}(m, K), K) = m$
- E.g. One-time pad:  $\mathcal{M} = \mathcal{K} = \mathcal{C} = \{0,1\}^n$  and  $\text{Enc}(m, K) = m \oplus K, \text{Dec}(c, K) = c \oplus K$ 
  - More generally  $\mathcal{M} = \mathcal{K} = \mathcal{C} = \mathcal{G}$  (a finite group) and  $\text{Enc}(m, K) = m + K, \text{Dec}(c, K) = c - K$

| $\mathcal{M} \backslash \mathcal{K}$ | 0 | 1 | 2 | 3 |
|--------------------------------------|---|---|---|---|
| a                                    | x | y | y | z |
| b                                    | y | x | z | y |



# Onetime Encryption

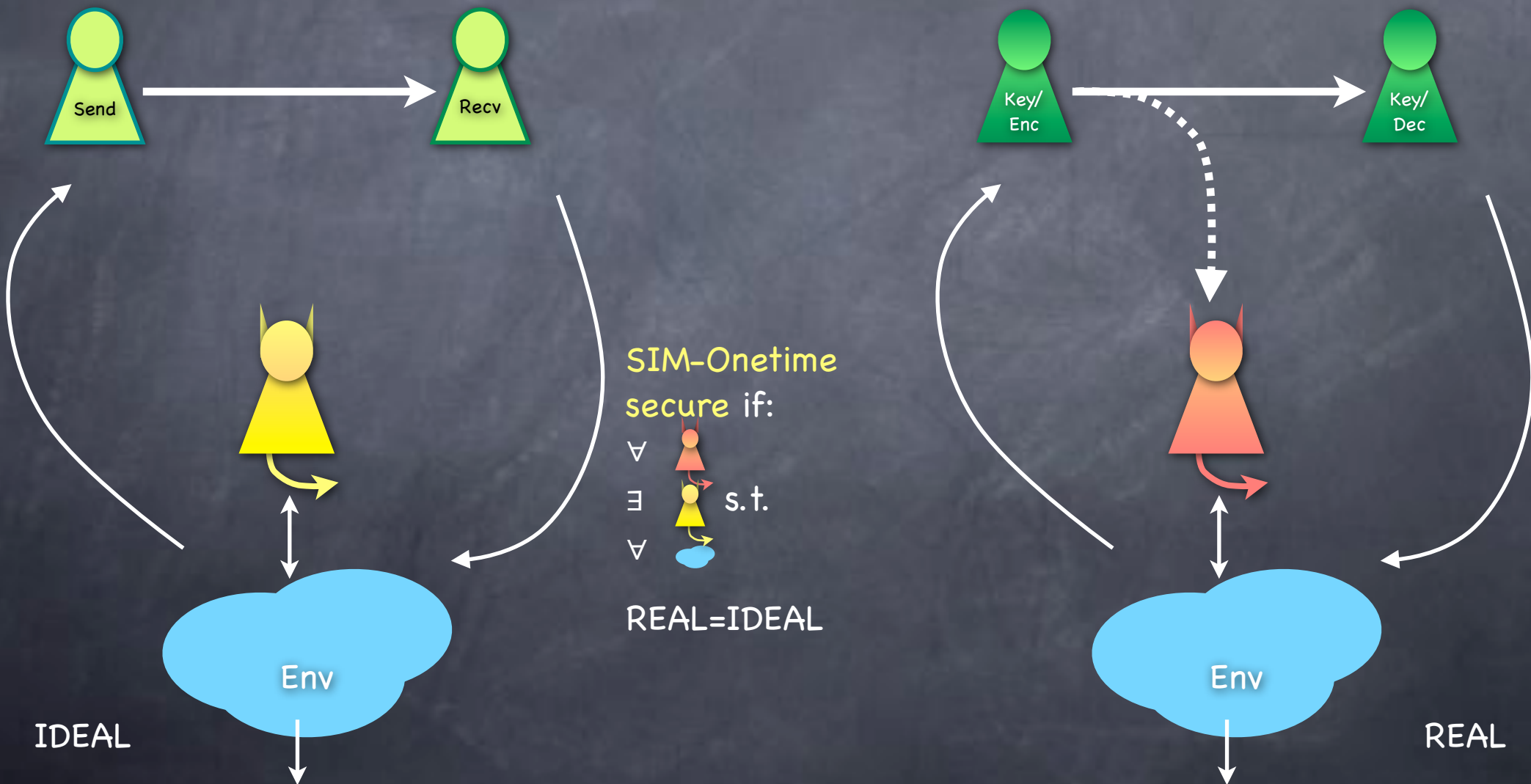
## SIM-Onetime Security



# Onetime Encryption

## SIM-Onetime Security

- Class of environments which send only one message

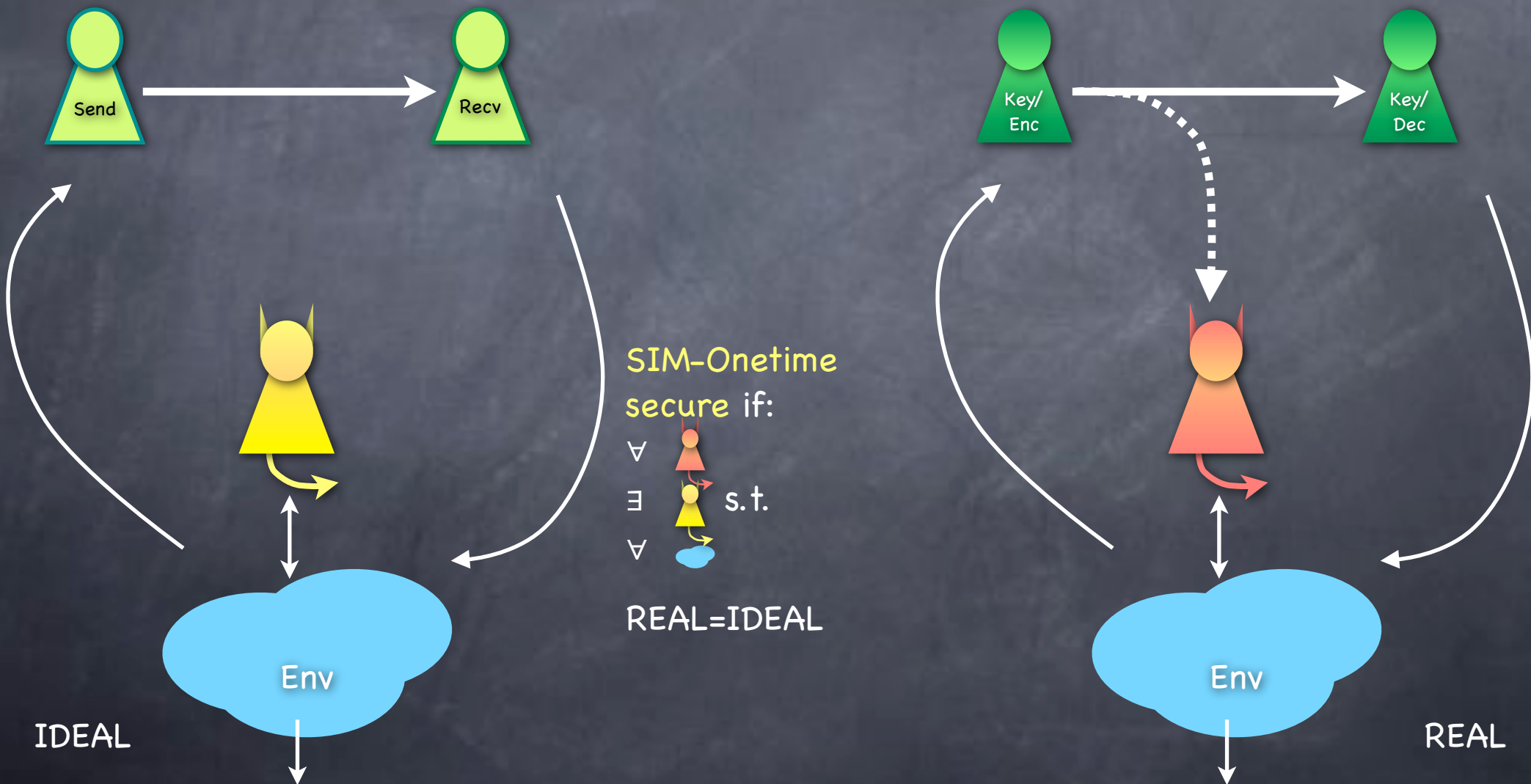


# Onetime Encryption

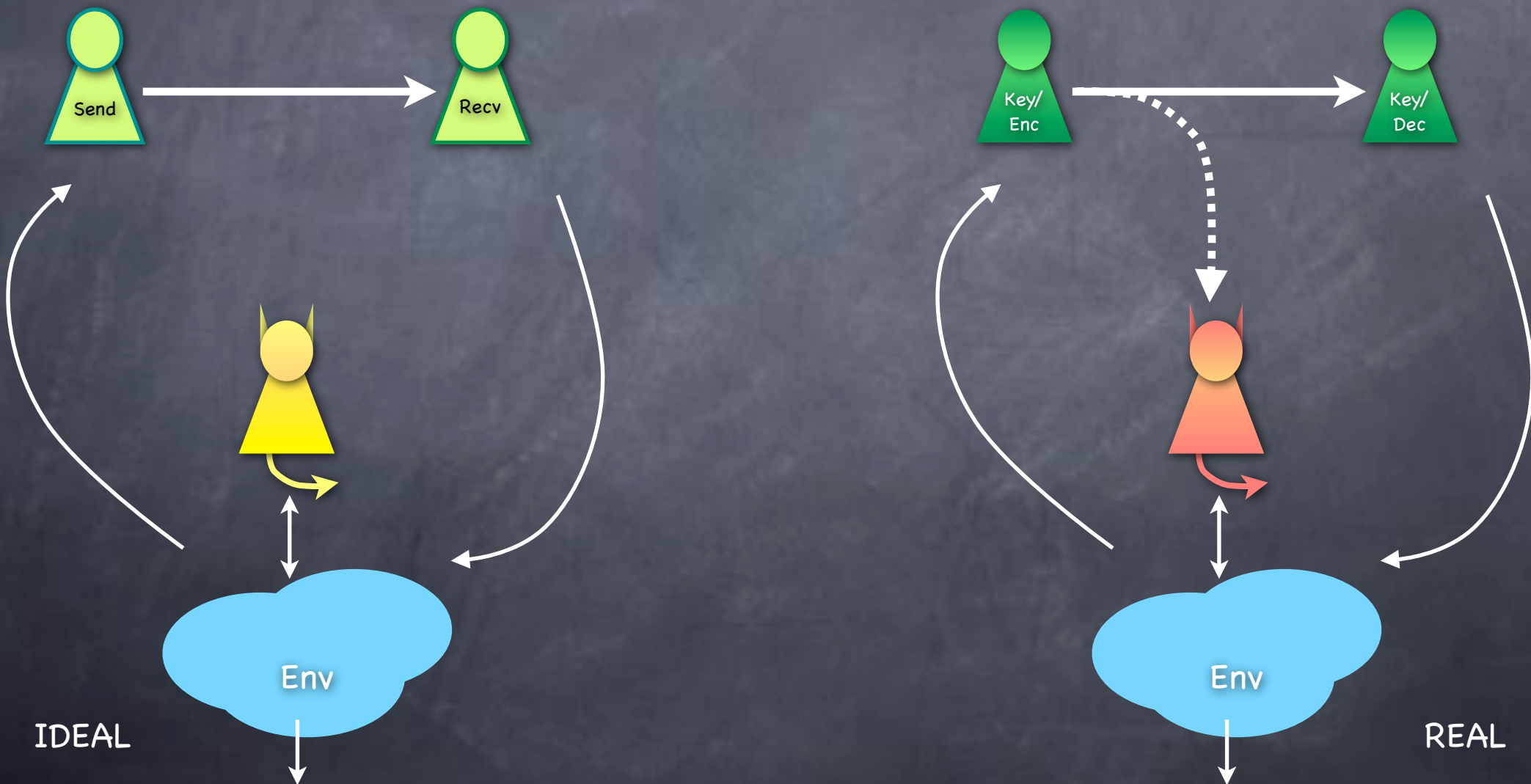
## SIM-Onetime Security

Equivalent to  
perfect secrecy  
+ correctness

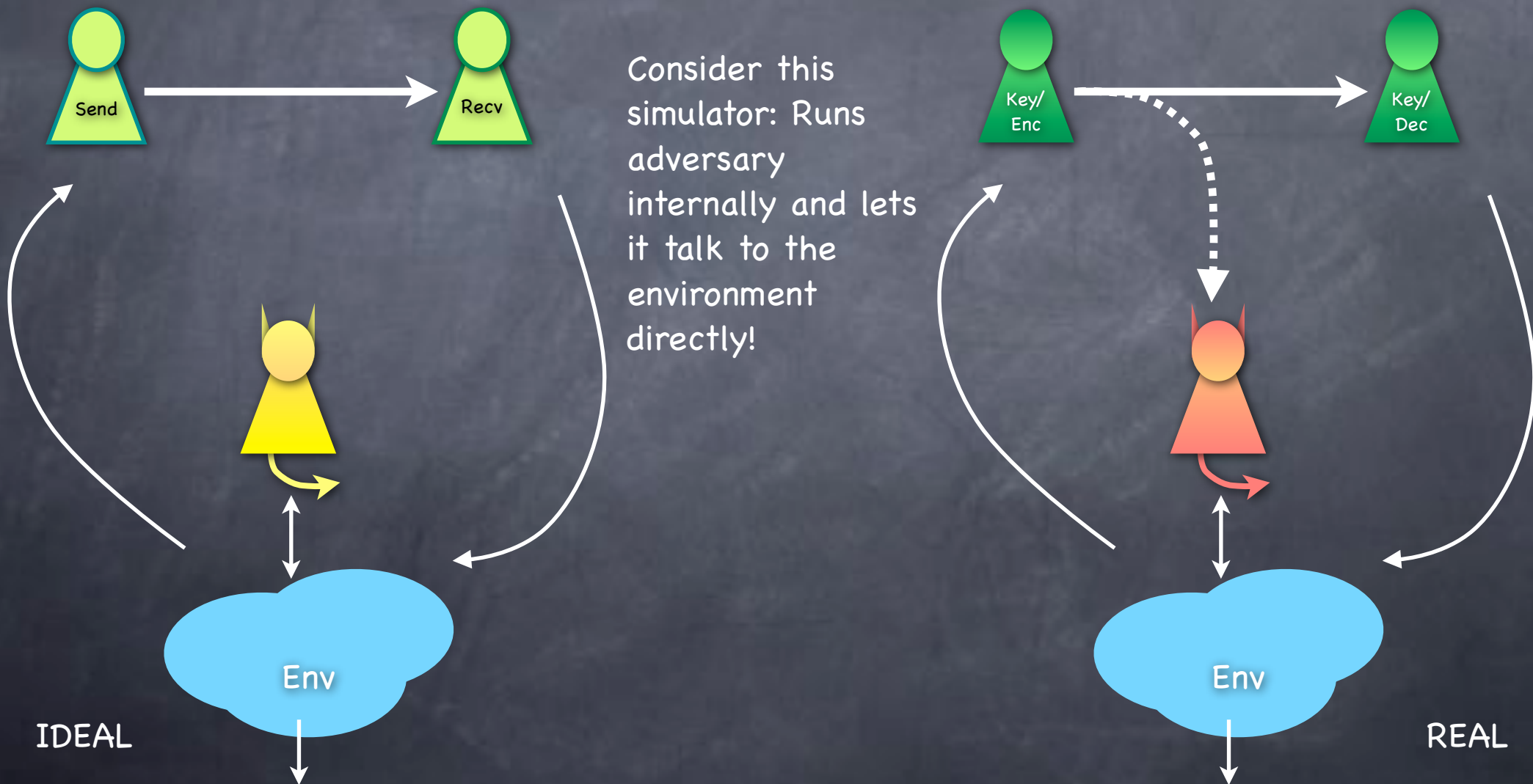
- Class of environments which send only one message



# Perfect Secrecy + Correctness $\Rightarrow$ SIM-Onetime Security

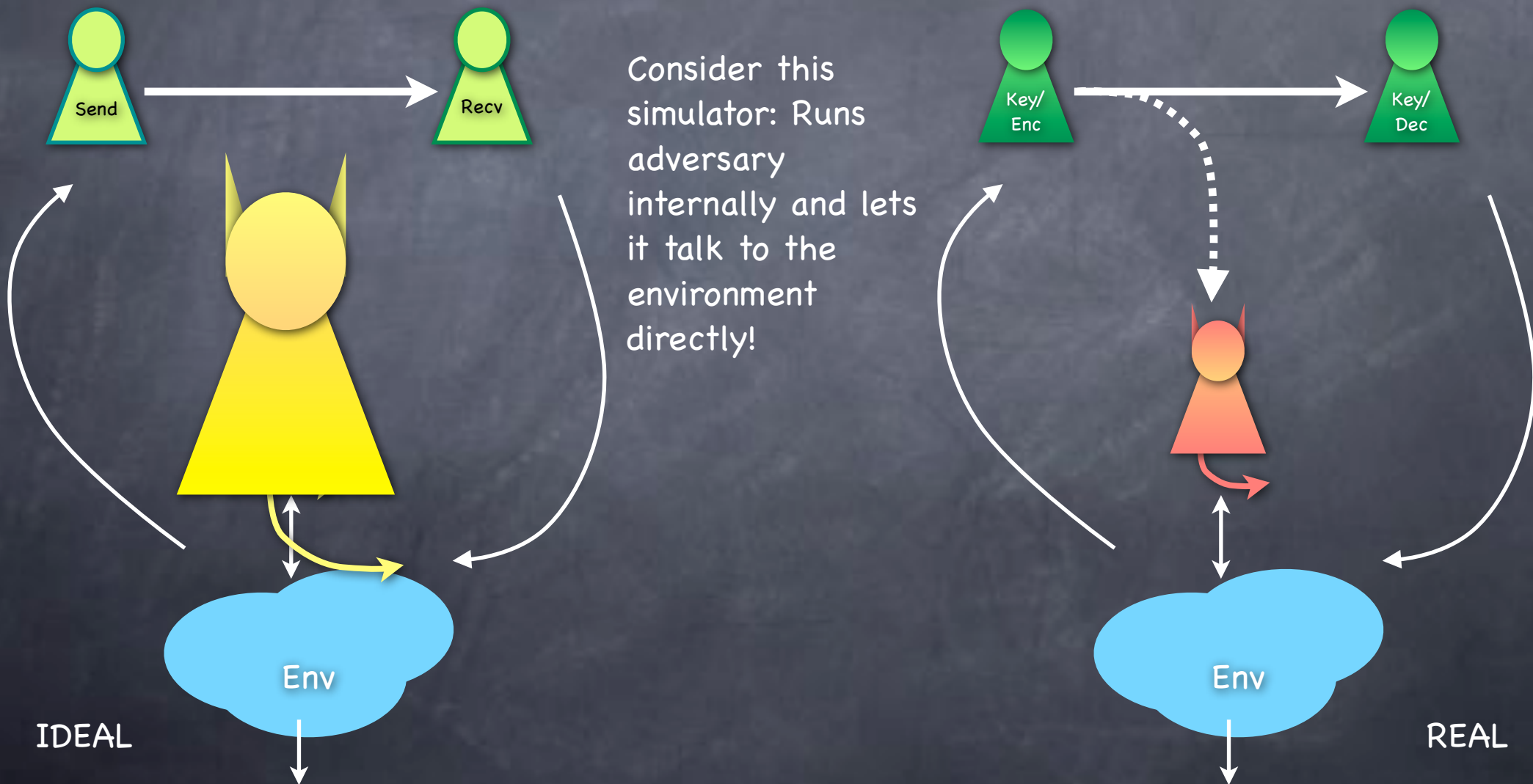


# Perfect Secrecy + Correctness $\Rightarrow$ SIM-Onetime Security



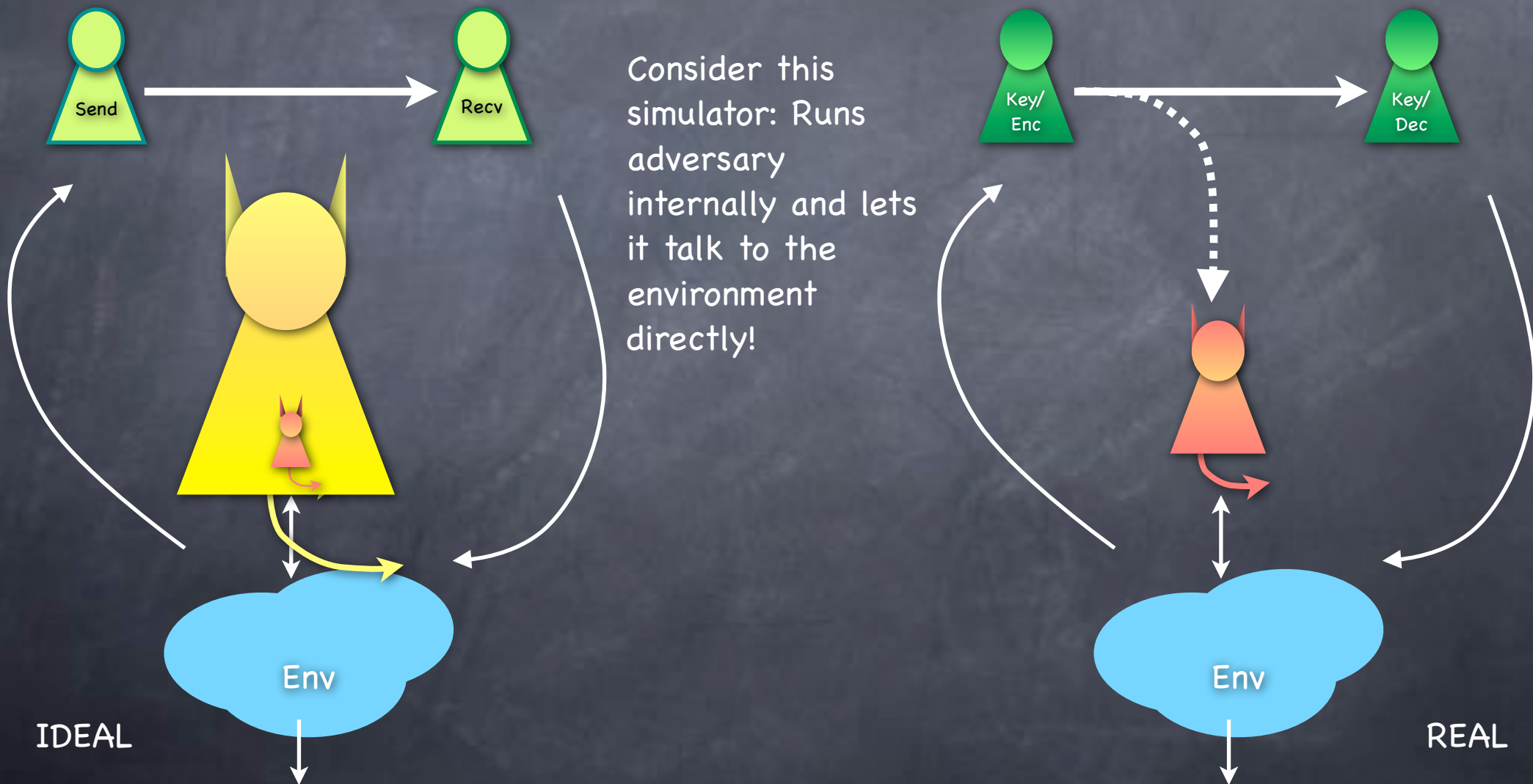


# Perfect Secrecy + Correctness $\Rightarrow$ SIM-Onetime Security

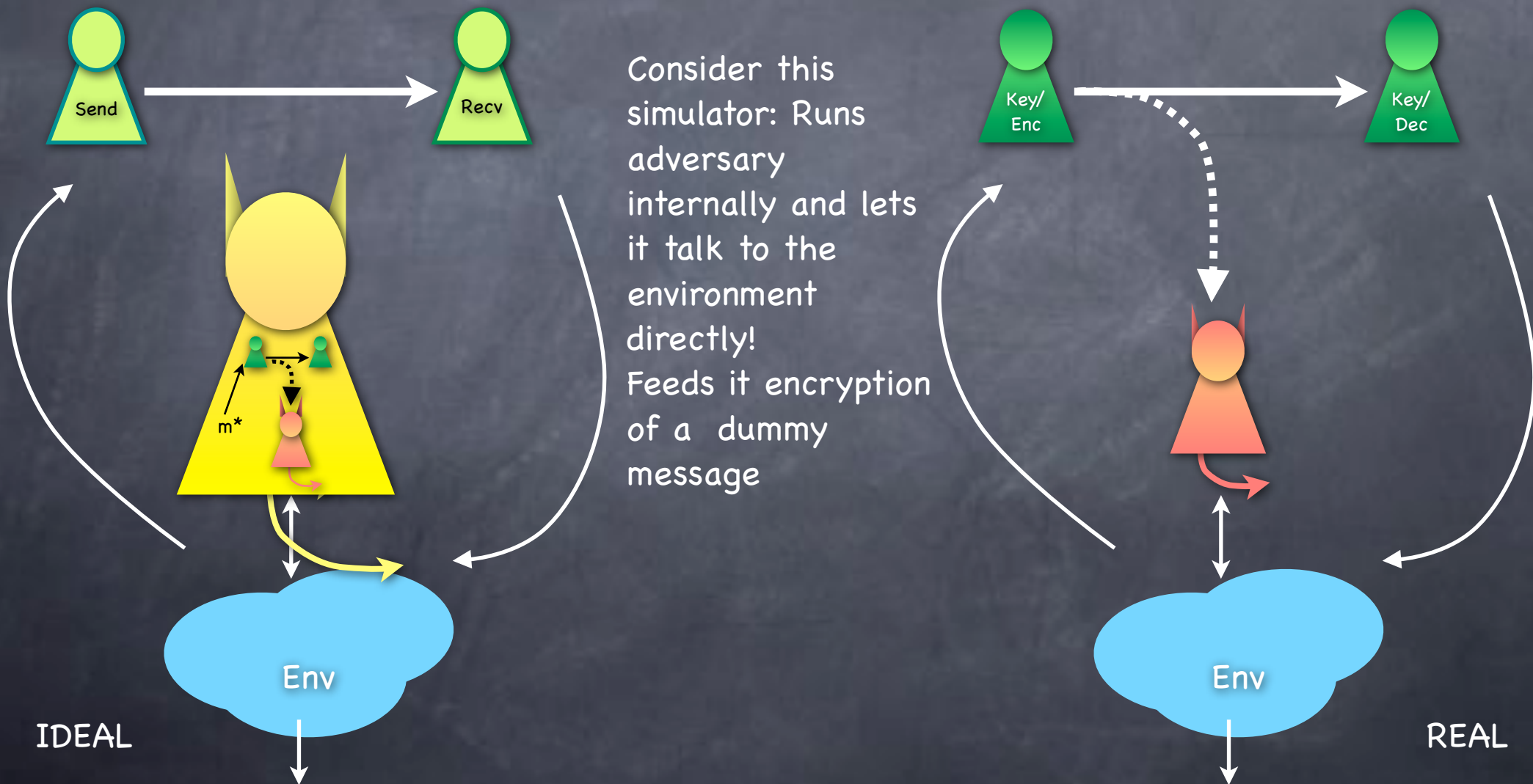




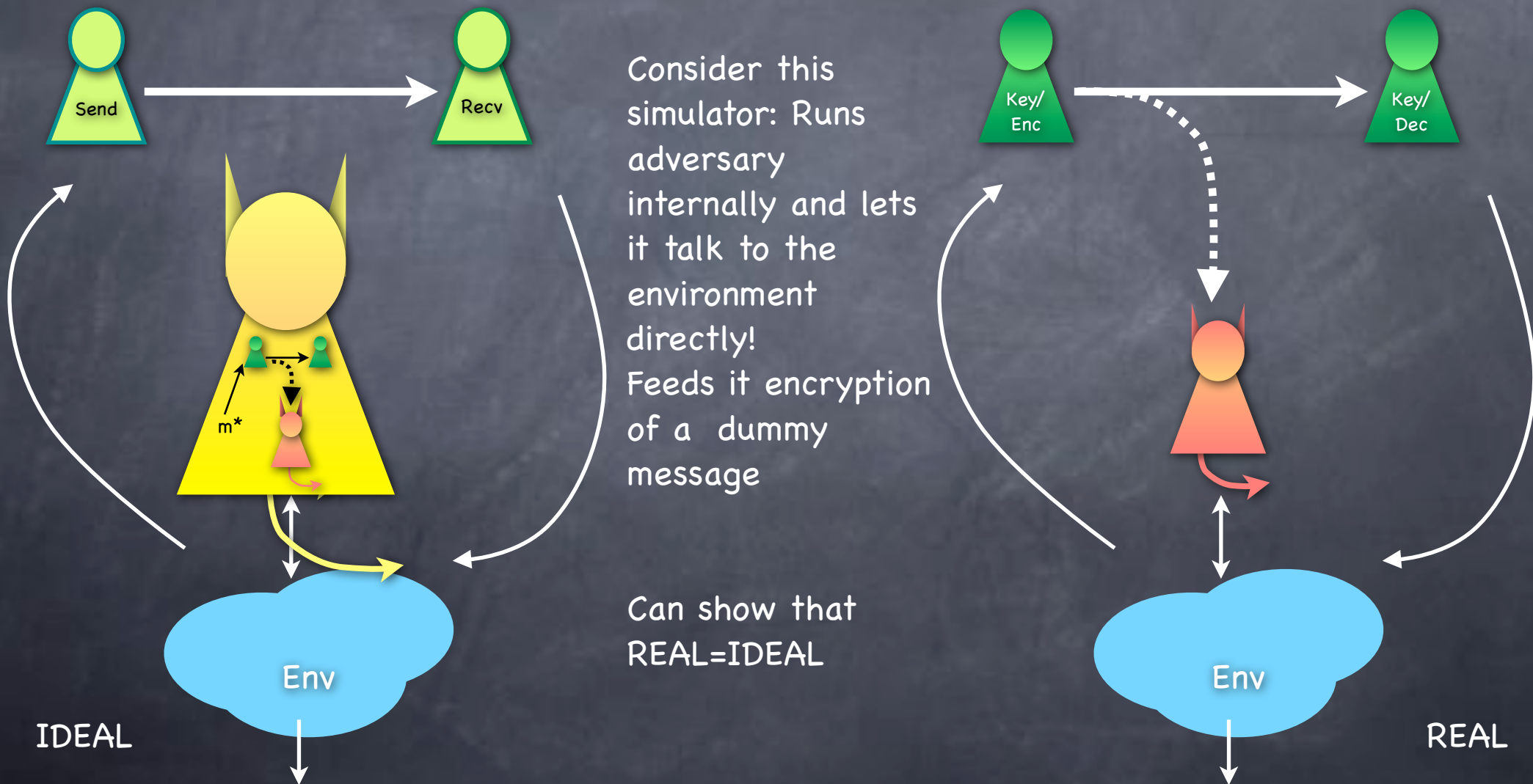
# Perfect Secrecy + Correctness $\Rightarrow$ SIM-Onetime Security



# Perfect Secrecy + Correctness $\Rightarrow$ SIM-Onetime Security



# Perfect Secrecy + Correctness $\Rightarrow$ SIM-Onetime Security



# Implicit Details

# Implicit Details

- Random coins used by the encryption scheme is kept private within the programs of the scheme (KeyGen, Enc, Dec)



# Implicit Details

- Random coins used by the encryption scheme is kept private within the programs of the scheme (KeyGen, Enc, Dec)
- If key is used for anything else (i.e., leaked to the environment) no more guarantees



# Implicit Details

- Random coins used by the encryption scheme is kept private within the programs of the scheme (KeyGen, Enc, Dec)
  - If key is used for anything else (i.e., leaked to the environment) no more guarantees
- In REAL, Eve only sees the ciphertext from Alice to Bob

# Implicit Details

- Random coins used by the encryption scheme is kept private within the programs of the scheme (KeyGen, Enc, Dec)
  - If key is used for anything else (i.e., leaked to the environment) no more guarantees
- In REAL, Eve only sees the ciphertext from Alice to Bob
  - In particular no timing attacks

# Implicit Details

- Random coins used by the encryption scheme is kept private within the programs of the scheme (KeyGen, Enc, Dec)
  - If key is used for anything else (i.e., leaked to the environment) no more guarantees
- In REAL, Eve only sees the ciphertext from Alice to Bob
  - In particular no timing attacks
- Message space is finite and known to Eve (and Eve')

# Implicit Details

- Random coins used by the encryption scheme is kept private within the programs of the scheme (KeyGen, Enc, Dec)
  - If key is used for anything else (i.e., leaked to the environment) no more guarantees
- In REAL, Eve only sees the ciphertext from Alice to Bob
  - In particular no timing attacks
- Message space is finite and known to Eve (and Eve')
  - Alternately, if message length is variable, it is given out to Eve' in IDEAL as well

# Implicit Details

- Random coins used by the encryption scheme is kept private within the programs of the scheme (KeyGen, Enc, Dec)
  - If key is used for anything else (i.e., leaked to the environment) no more guarantees
- In REAL, Eve only sees the ciphertext from Alice to Bob
  - In particular no timing attacks
- Message space is finite and known to Eve (and Eve')
  - Alternately, if message length is variable, it is given out to Eve' in IDEAL as well
  - Also, Eve' allowed to learn when a message is sent



# Onetime Encryption

IND-Onetime Security



# Onetime Encryption

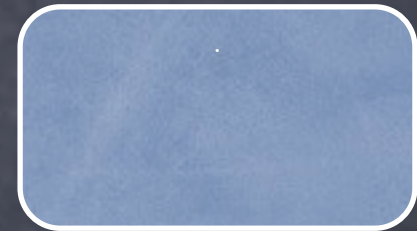
## IND-Onetime Security

- IND-Onetime Experiment

# Onetime Encryption

## IND-Onetime Security

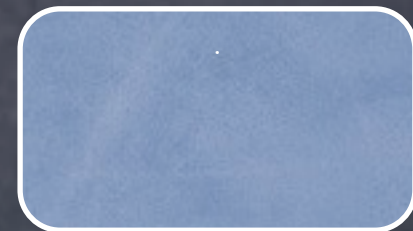
- IND-Onetime Experiment



# Onetime Encryption

## IND-Onetime Security

- IND-Onetime Experiment



# Onetime Encryption

## IND-Onetime Security

- IND-Onetime Experiment
  - Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$



$b \leftarrow \{0,1\}$

# Onetime Encryption

## IND-Onetime Security

- IND-Onetime Experiment

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$



$b \leftarrow \{0,1\}$



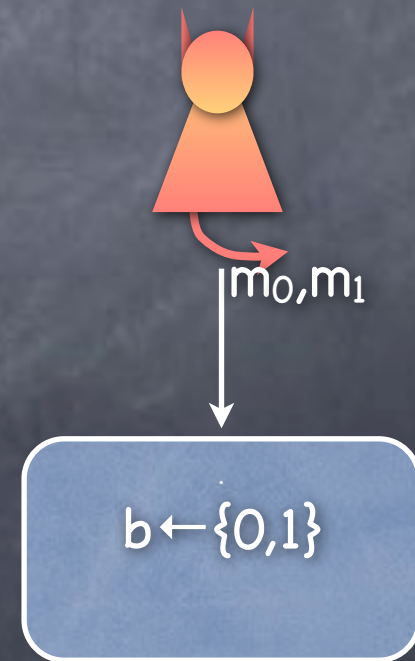
# Onetime Encryption

## IND-Onetime Security

- IND-Onetime Experiment

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$

- Adversary sends two messages  $m_0, m_1$  to the experiment

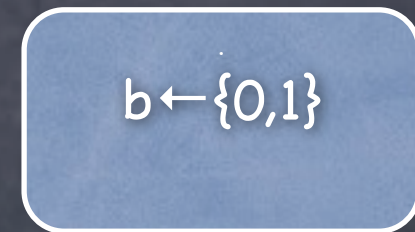


# Onetime Encryption

## IND-Onetime Security

- IND-Onetime Experiment

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$
- Adversary sends two messages  $m_0, m_1$  to the experiment
- Experiment replies with  $\text{Enc}(m_b, K)$

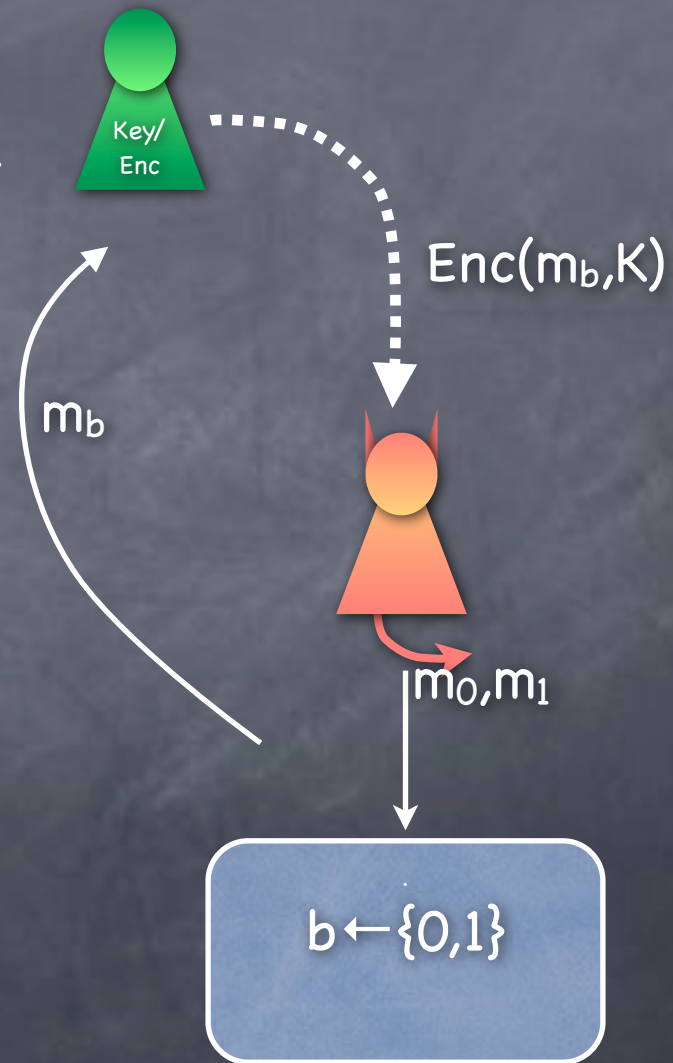


# Onetime Encryption

## IND-Onetime Security

- IND-Onetime Experiment

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$
- Adversary sends two messages  $m_0, m_1$  to the experiment
- Experiment replies with  $\text{Enc}(m_b, K)$

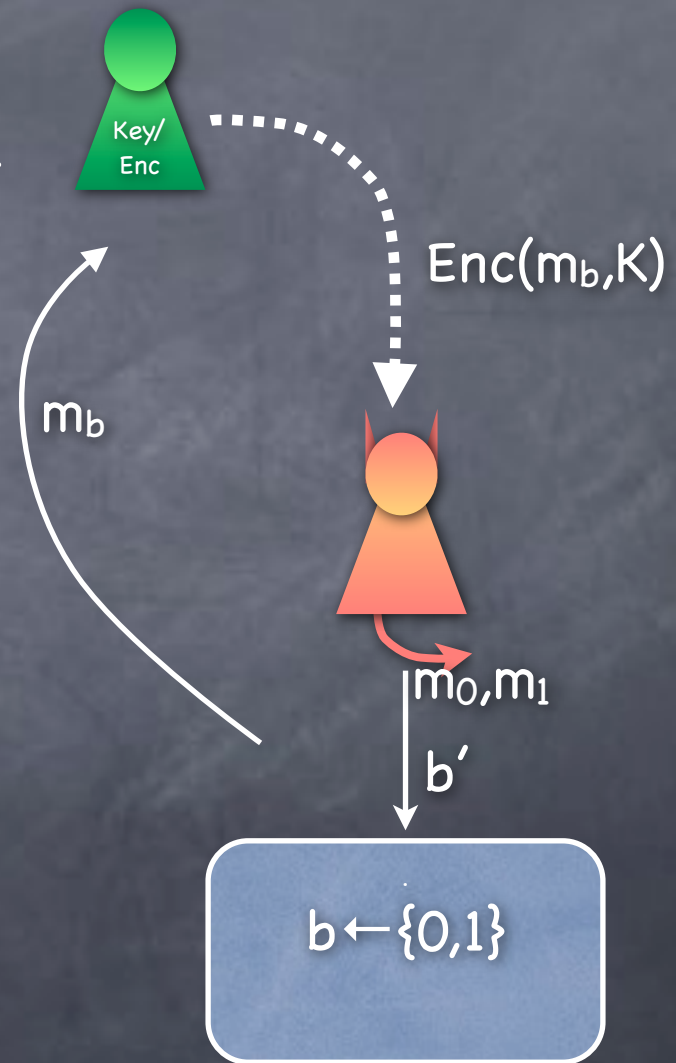


# Onetime Encryption

## IND-Onetime Security

- IND-Onetime Experiment

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$
- Adversary sends two messages  $m_0, m_1$  to the experiment
- Experiment replies with  $\text{Enc}(m_b, K)$
- Adversary returns a guess  $b'$

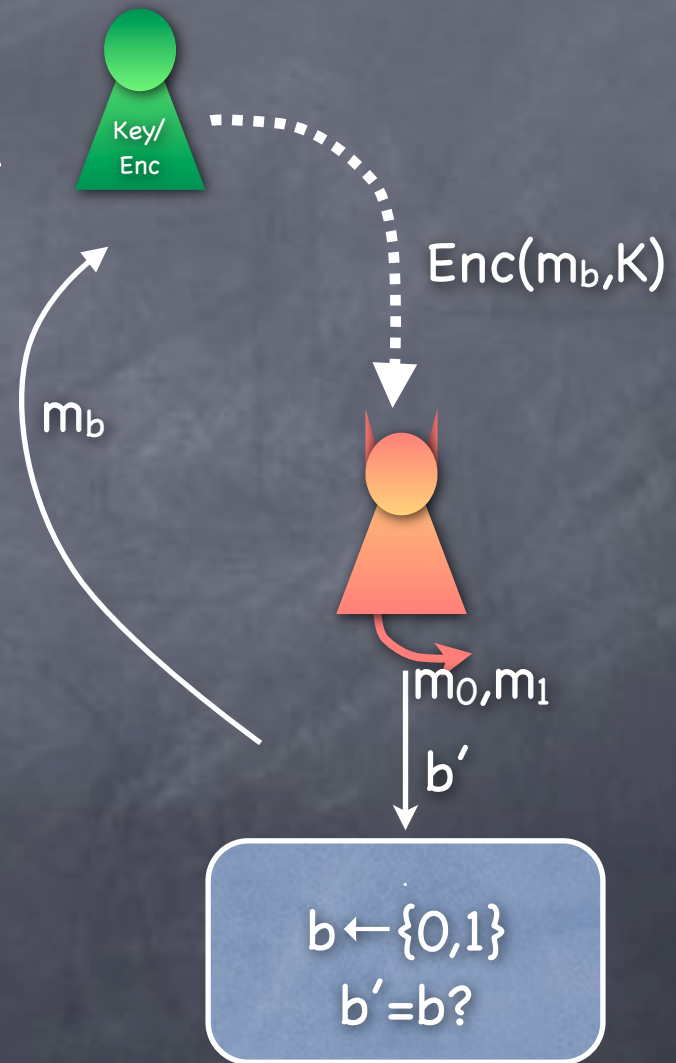


# Onetime Encryption

## IND-Onetime Security

- IND-Onetime Experiment

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$
- Adversary sends two messages  $m_0, m_1$  to the experiment
- Experiment replies with  $\text{Enc}(m_b, K)$
- Adversary returns a guess  $b'$



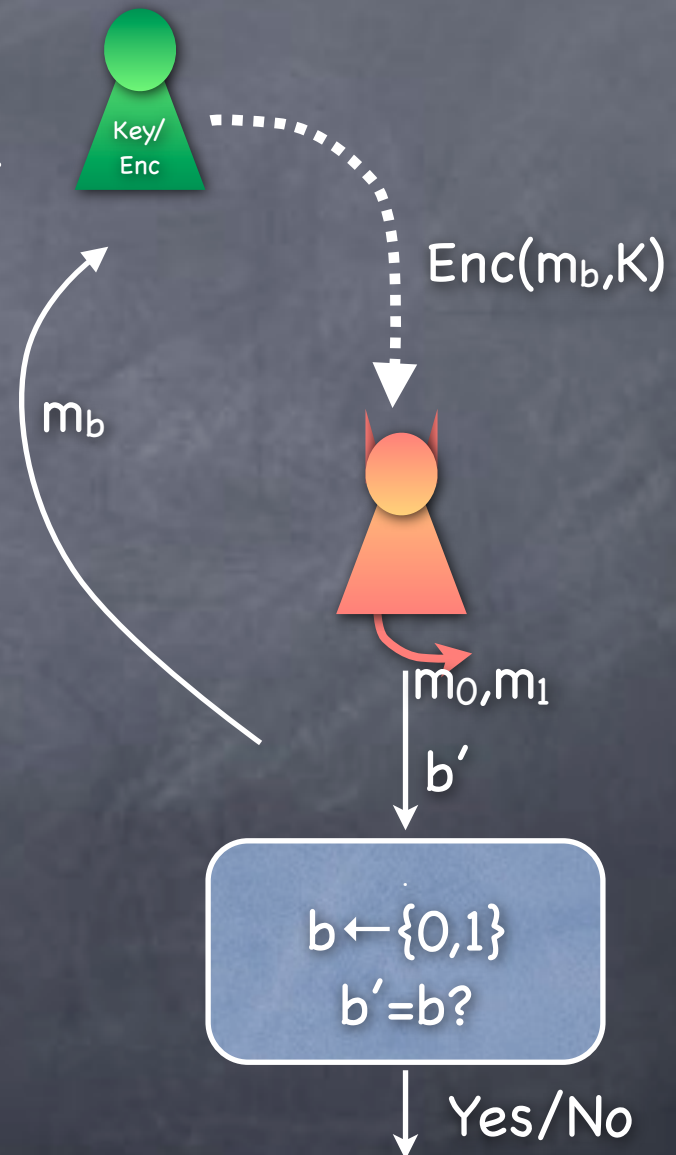


# Onetime Encryption

## IND-Onetime Security

### IND-Onetime Experiment

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$
- Adversary sends two messages  $m_0, m_1$  to the experiment
- Experiment replies with  $\text{Enc}(m_b, K)$
- Adversary returns a guess  $b'$
- Experiments outputs 1 iff  $b' = b$

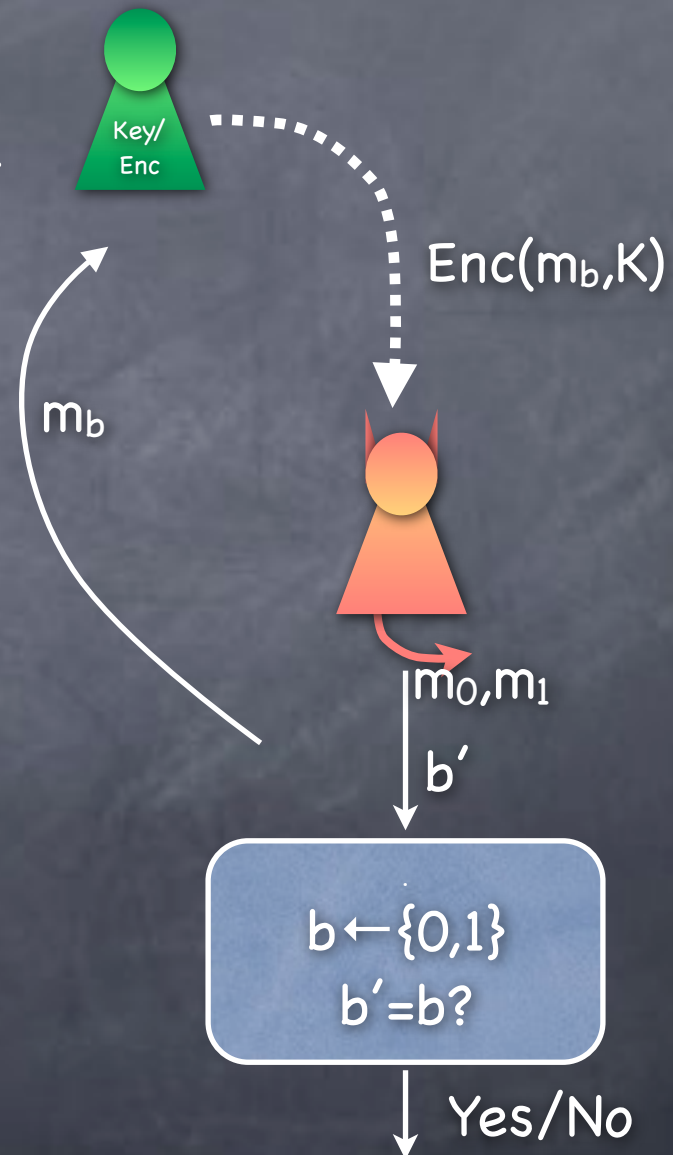


# Onetime Encryption

## IND-Onetime Security

- IND-Onetime Experiment

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$
- Adversary sends two messages  $m_0, m_1$  to the experiment
- Experiment replies with  $\text{Enc}(m_b, K)$
- Adversary returns a guess  $b'$
- Experiments outputs 1 iff  $b' = b$
- IND-Onetime secure if for every adversary,  $\Pr[b = b'] = 1/2$



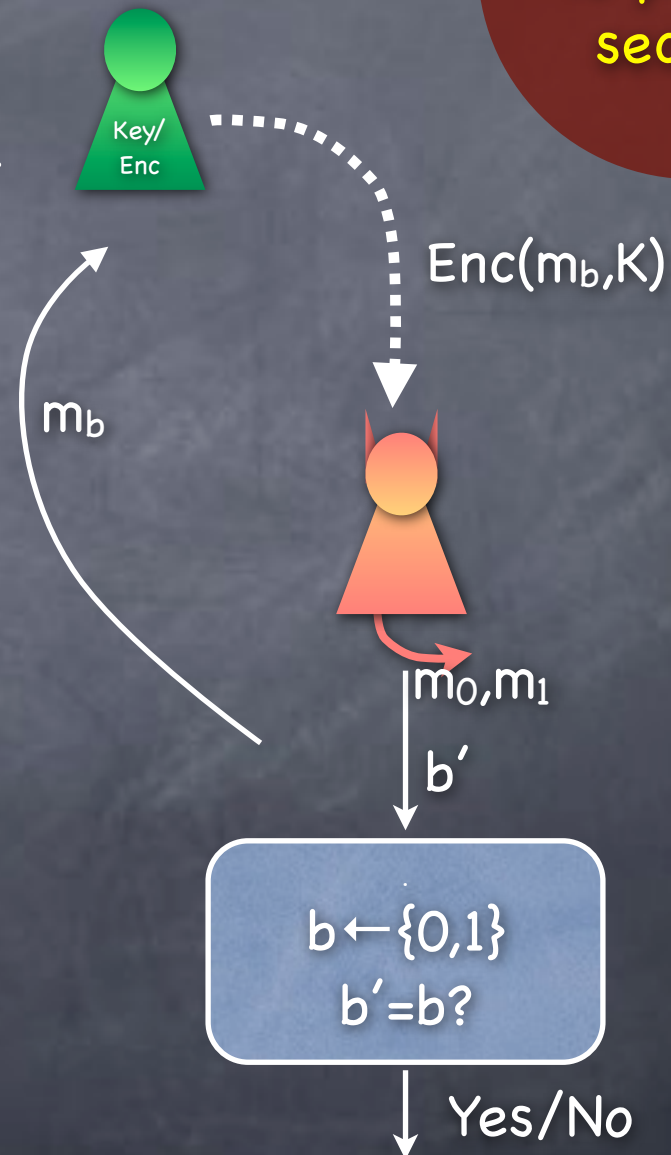
# Onetime Encryption

## IND-Onetime Security

Equivalent to perfect secrecy

- IND-Onetime Experiment

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$
- Adversary sends two messages  $m_0, m_1$  to the experiment
- Experiment replies with  $\text{Enc}(m_b, K)$
- Adversary returns a guess  $b'$
- Experiments outputs 1 iff  $b' = b$
- IND-Onetime secure if for every adversary,  $\Pr[b = b'] = 1/2$



# Symmetric-Key Encryption

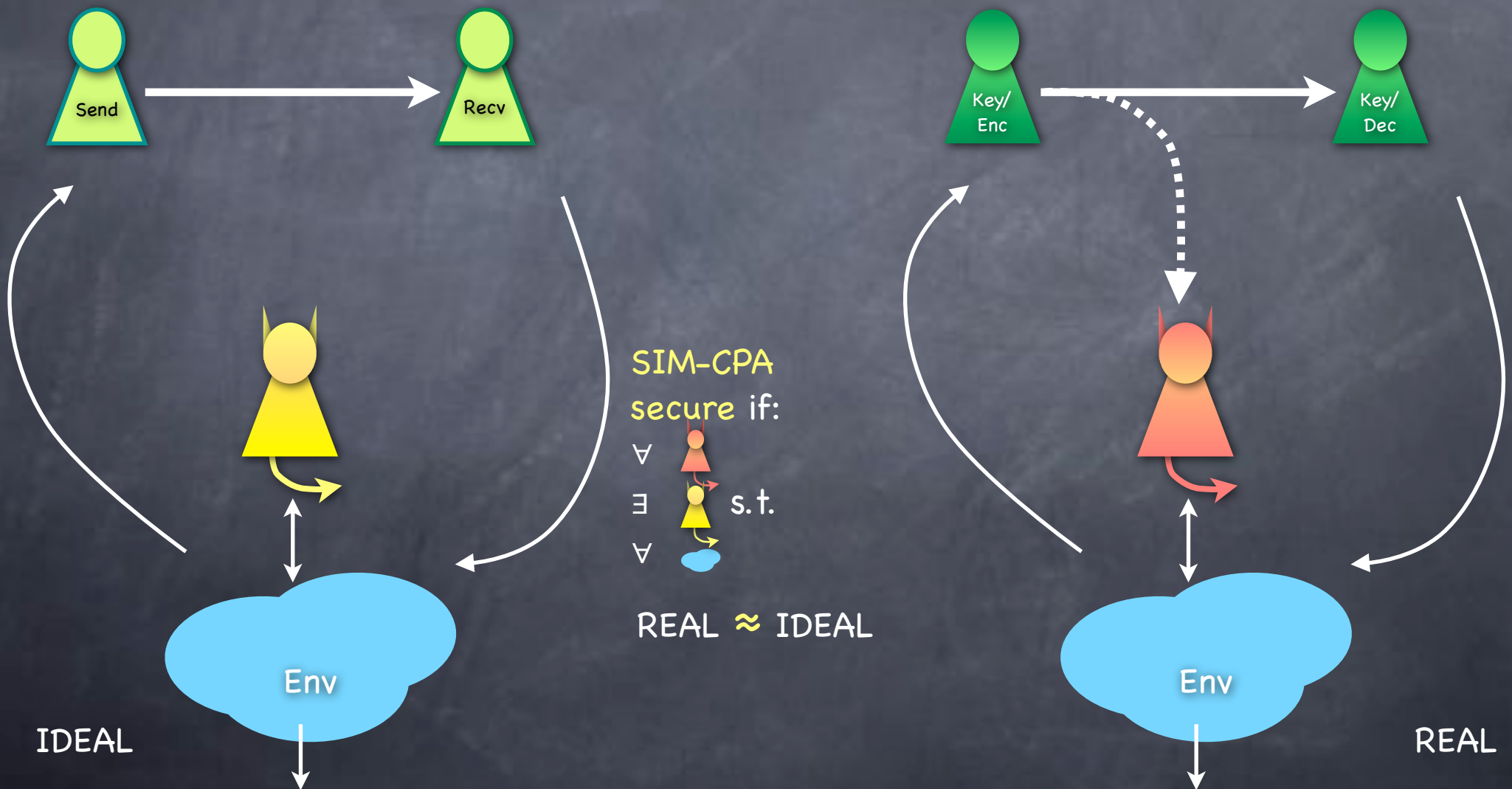
## The Syntax

- Shared-key (Private-key) Encryption
  - **Key Generation:** Randomized
    - $K \leftarrow \mathcal{K}$ , uniformly randomly drawn from the key-space (or according to a key-distribution)
  - **Encryption:** Randomized
    - $\text{Enc}: \mathcal{M} \times \mathcal{K} \times \mathcal{R} \rightarrow \mathcal{C}$ . During encryption a fresh random string will be chosen uniformly at random from  $\mathcal{R}$
  - **Decryption:** Deterministic
    - $\text{Dec}: \mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$



# Symmetric-Key Encryption

## SIM-CPA Security

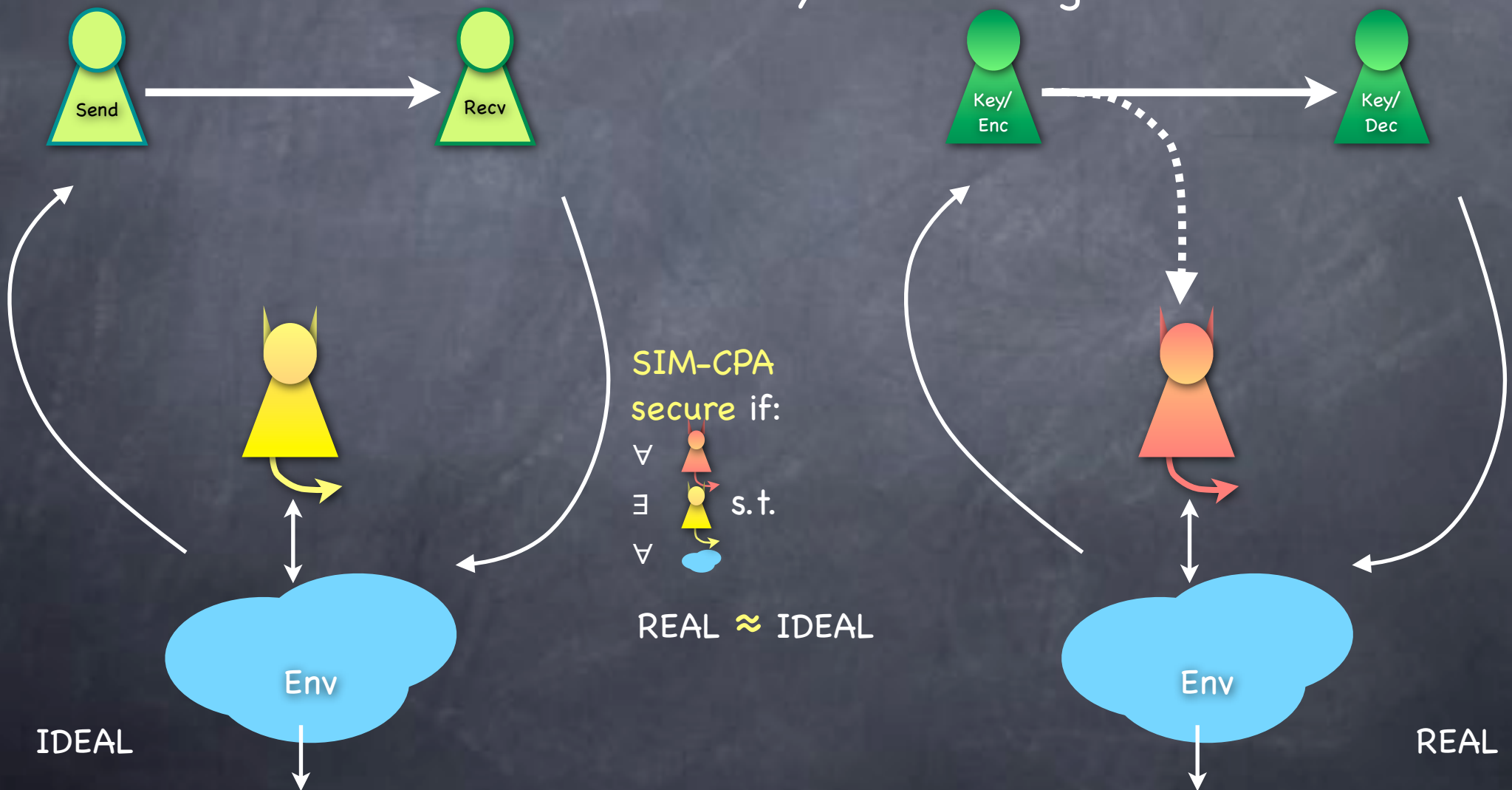




# Symmetric-Key Encryption

## SIM-CPA Security

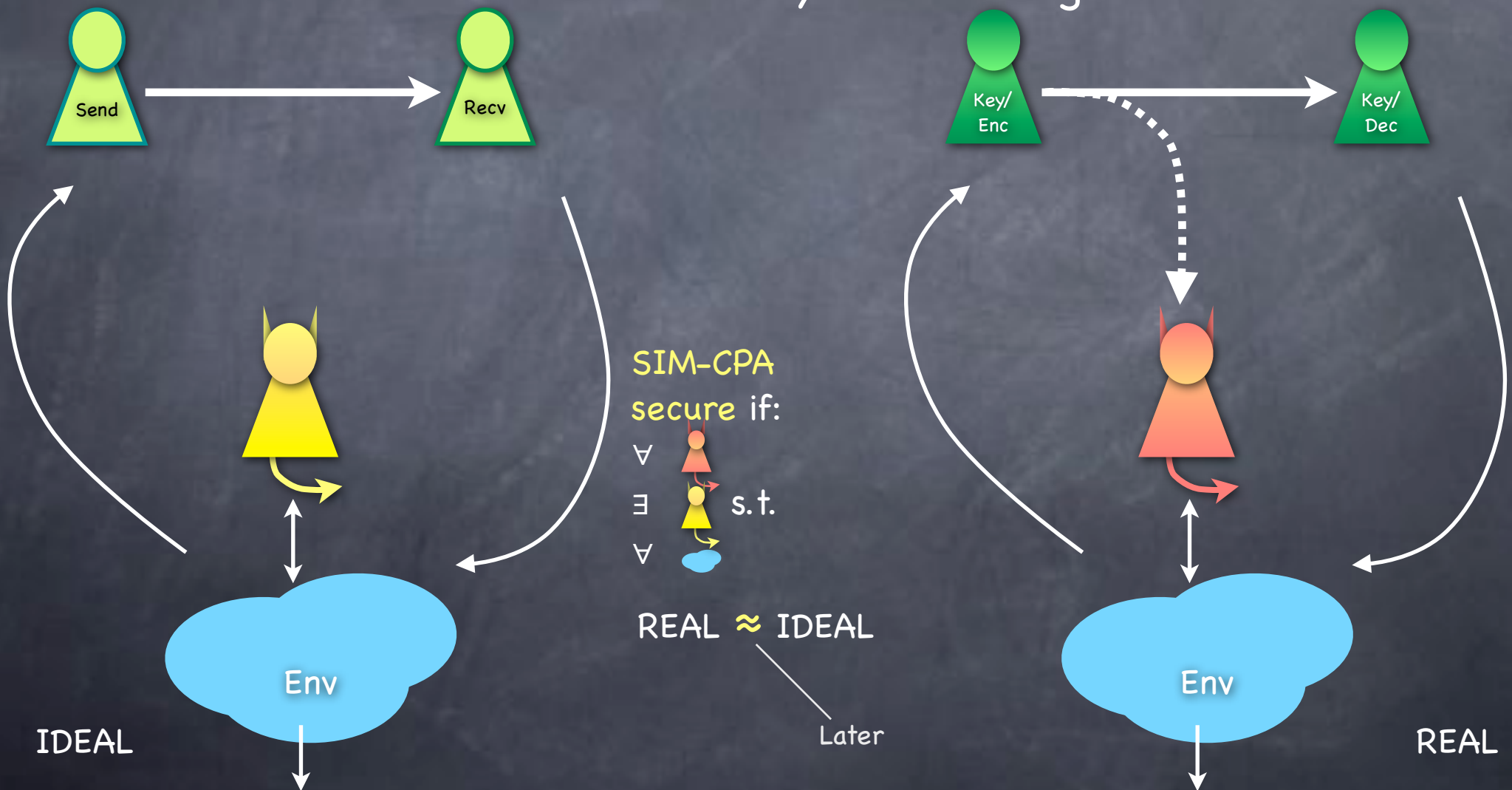
- Same as SIM-onetime security, but not restricted to environments which send only one message



# Symmetric-Key Encryption

## SIM-CPA Security

- Same as SIM-onetime security, but not restricted to environments which send only one message



# Symmetric-Key Encryption

## IND-CPA Security



# Symmetric-Key Encryption

## IND-CPA Security

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$



$b \leftarrow \{0,1\}$

# Symmetric-Key Encryption

## IND-CPA Security

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$
- For as long as Adversary wants



$b \leftarrow \{0,1\}$



# Symmetric-Key Encryption

## IND-CPA Security

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$



- For as long as Adversary wants

- Adv sends two messages  $m_0, m_1$  to the experiment



$b \leftarrow \{0,1\}$

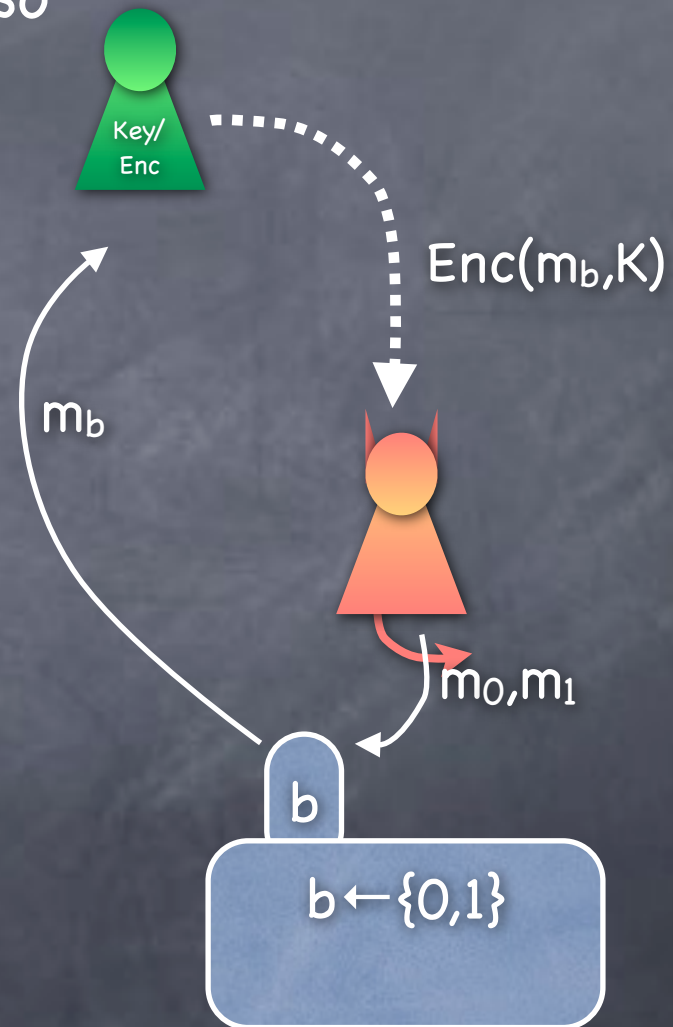
# Symmetric-Key Encryption

## IND-CPA Security

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$

- For as long as Adversary wants

- Adv sends two messages  $m_0, m_1$  to the experiment
- Expt returns  $\text{Enc}(m_b, K)$  to the adversary



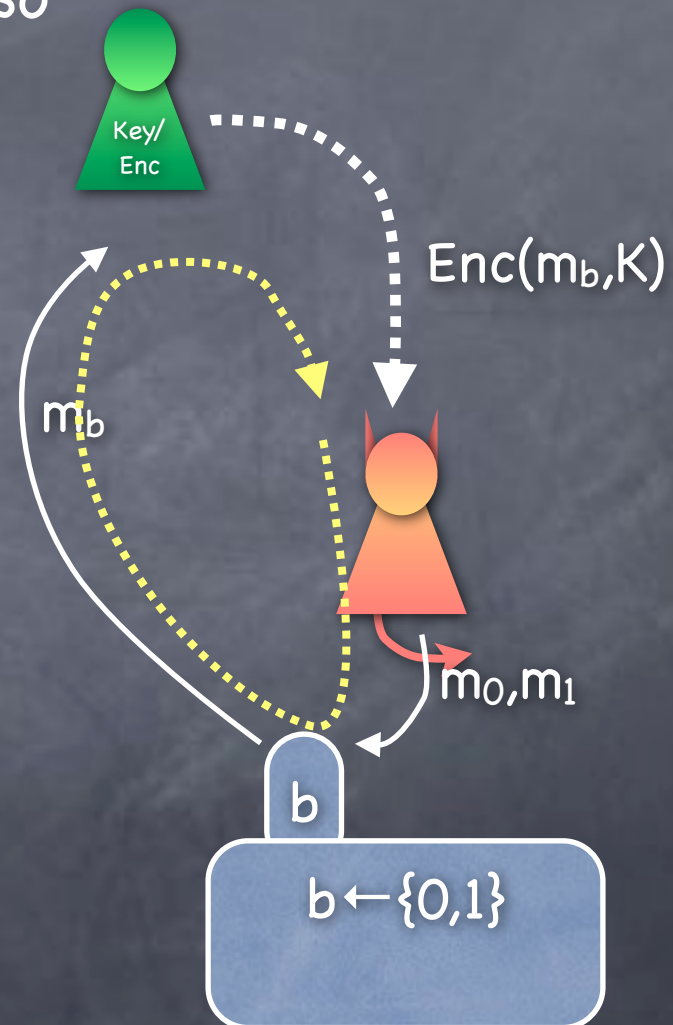
# Symmetric-Key Encryption

## IND-CPA Security

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$

- For as long as Adversary wants

- Adv sends two messages  $m_0, m_1$  to the experiment
- Expt returns  $\text{Enc}(m_b, K)$  to the adversary



# Symmetric-Key Encryption

## IND-CPA Security

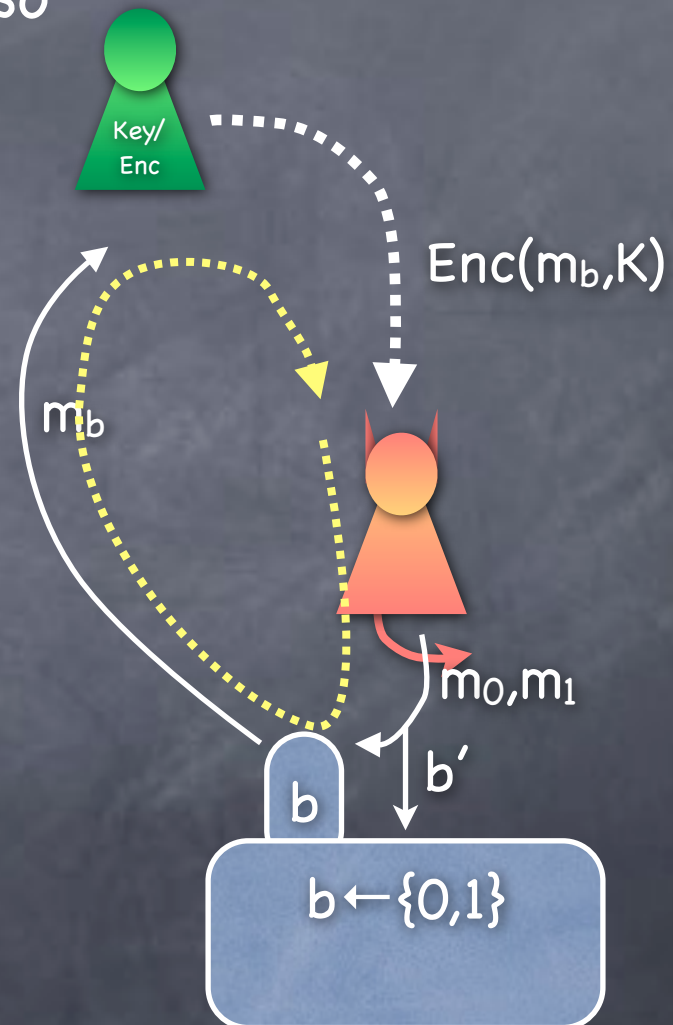
- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$

- For as long as Adversary wants**

- Adv sends two messages  $m_0, m_1$  to the experiment

- Expt returns  $\text{Enc}(m_b, K)$  to the adversary

- Adversary returns a guess  $b'$





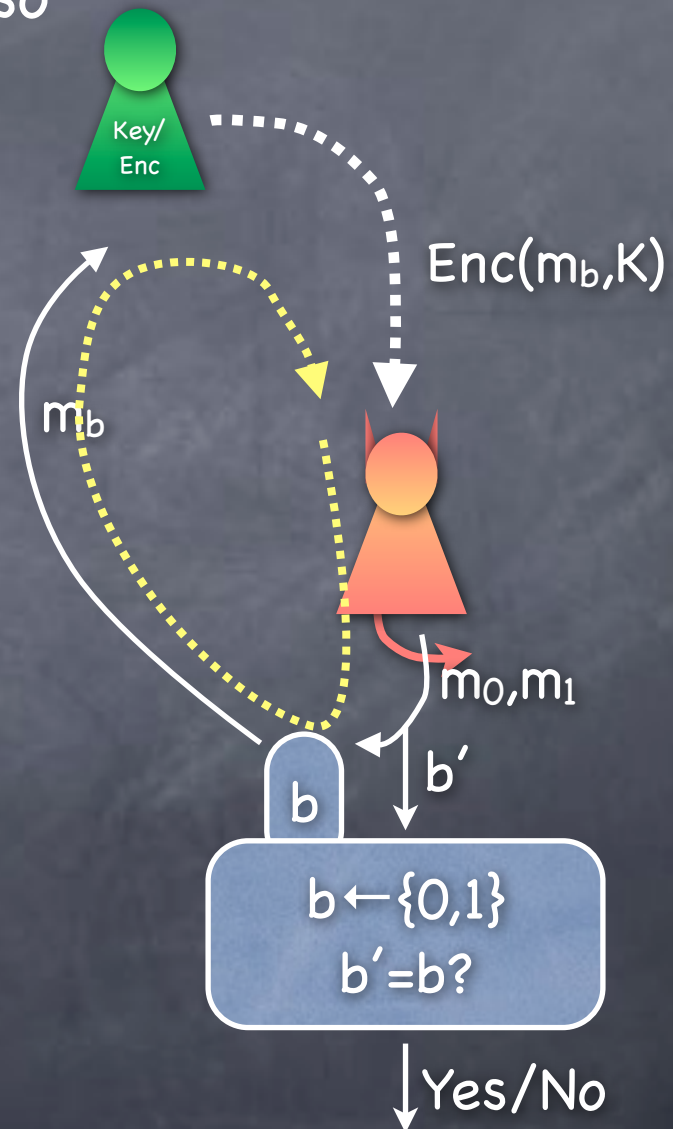
# Symmetric-Key Encryption

## IND-CPA Security

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$

- For as long as Adversary wants

- Adv sends two messages  $m_0, m_1$  to the experiment
- Expt returns  $\text{Enc}(m_b, K)$  to the adversary
- Adversary returns a guess  $b'$
- Experiment outputs 1 iff  $b' = b$





# Symmetric-Key Encryption

## IND-CPA Security

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$

- For as long as Adversary wants

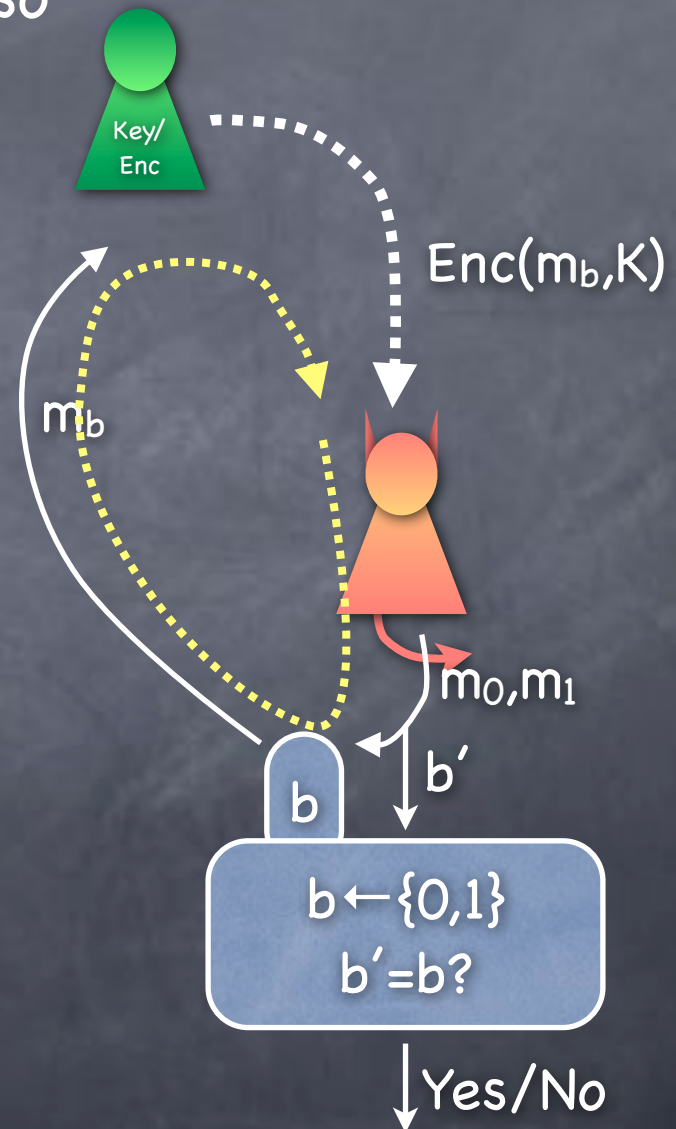
- Adv sends two messages  $m_0, m_1$  to the experiment

- Expt returns  $\text{Enc}(m_b, K)$  to the adversary

- Adversary returns a guess  $b'$

- Experiment outputs 1 iff  $b' = b$

- IND-CPA secure if for all “feasible” adversaries  $\Pr[b' = b] \approx 1/2$



# Symmetric-Key Encryption

## IND-CPA Security

- Experiment picks a random bit  $b$ . It also runs KeyGen to get a key  $K$

- For as long as Adversary wants

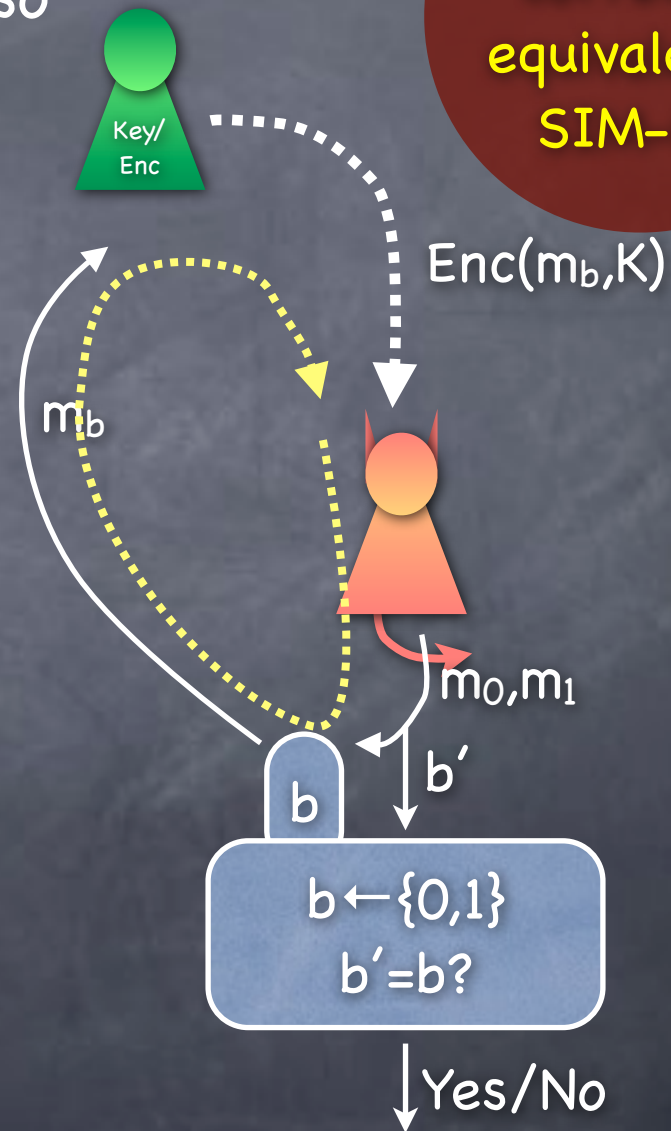
- Adv sends two messages  $m_0, m_1$  to the experiment

- Expt returns  $\text{Enc}(m_b, K)$  to the adversary

- Adversary returns a guess  $b'$

- Experiment outputs 1 iff  $b' = b$

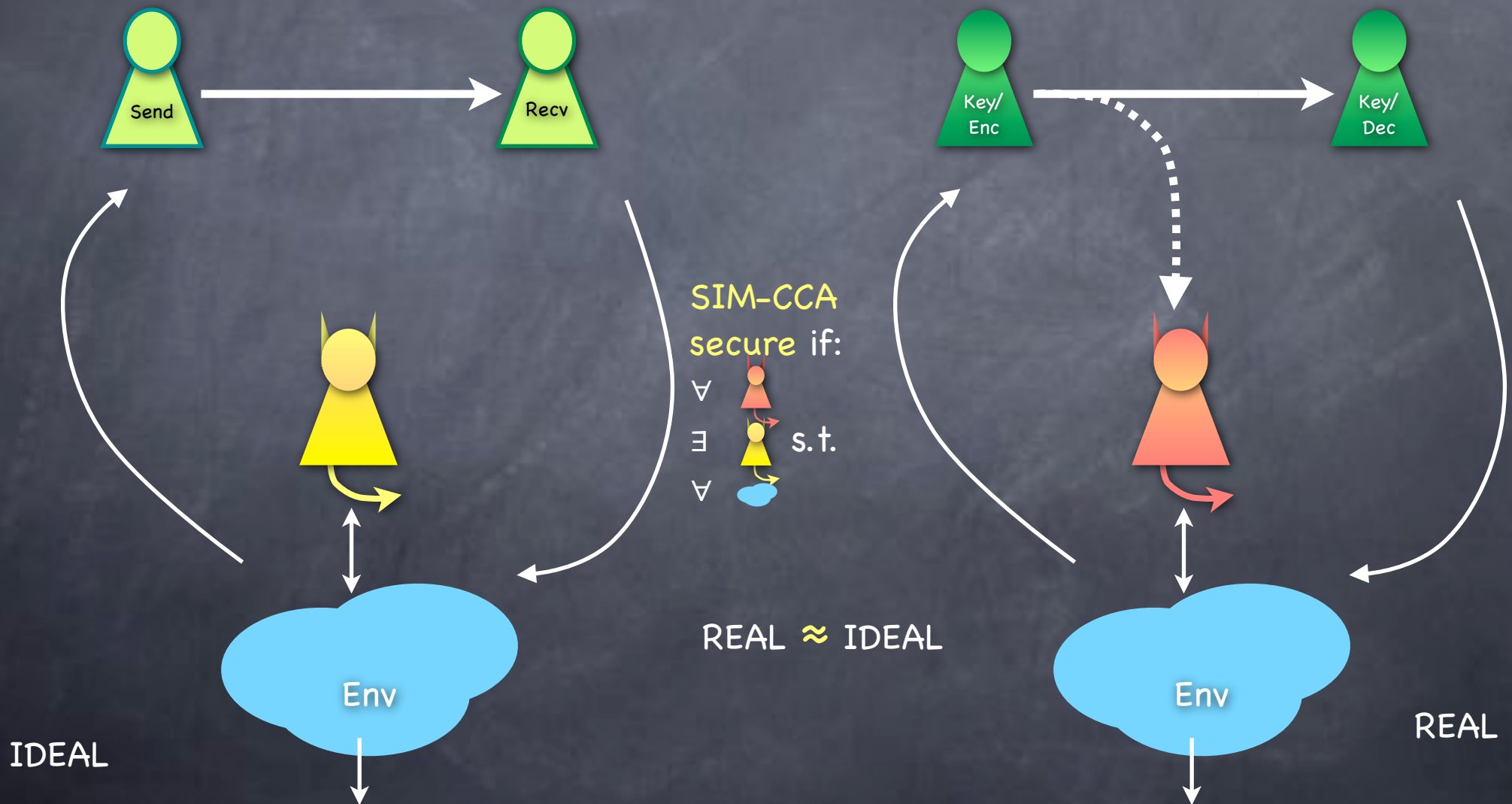
- IND-CPA secure if for all “feasible” adversaries  $\Pr[b' = b] \approx 1/2$



IND-CPA +  
~correctness  
equivalent to  
SIM-CPA

# Symmetric-Key Encryption

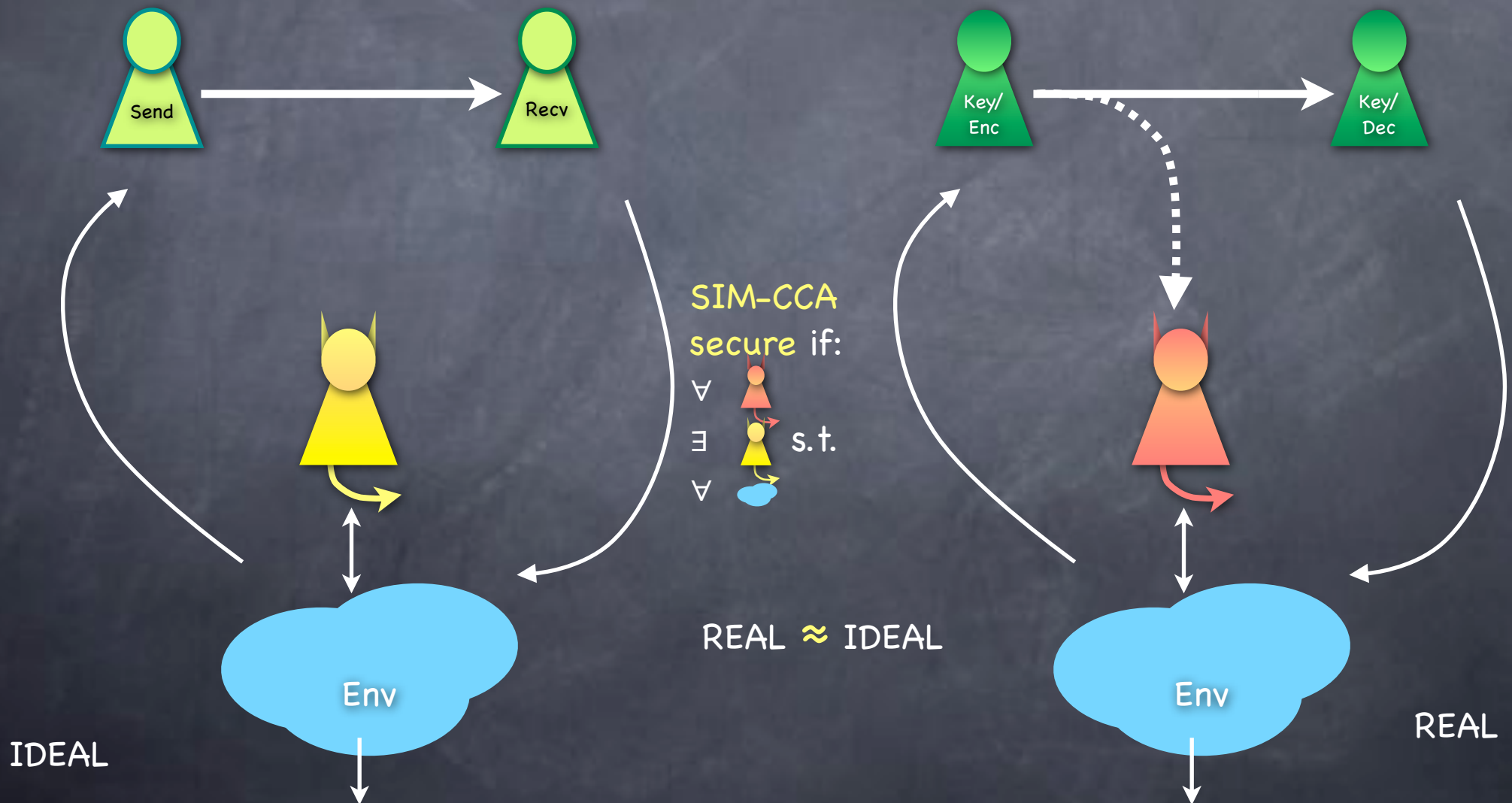
## SIM-CCA Security



# Symmetric-Key Encryption

## SIM-CCA Security

- An active adversary can inject messages into the channel

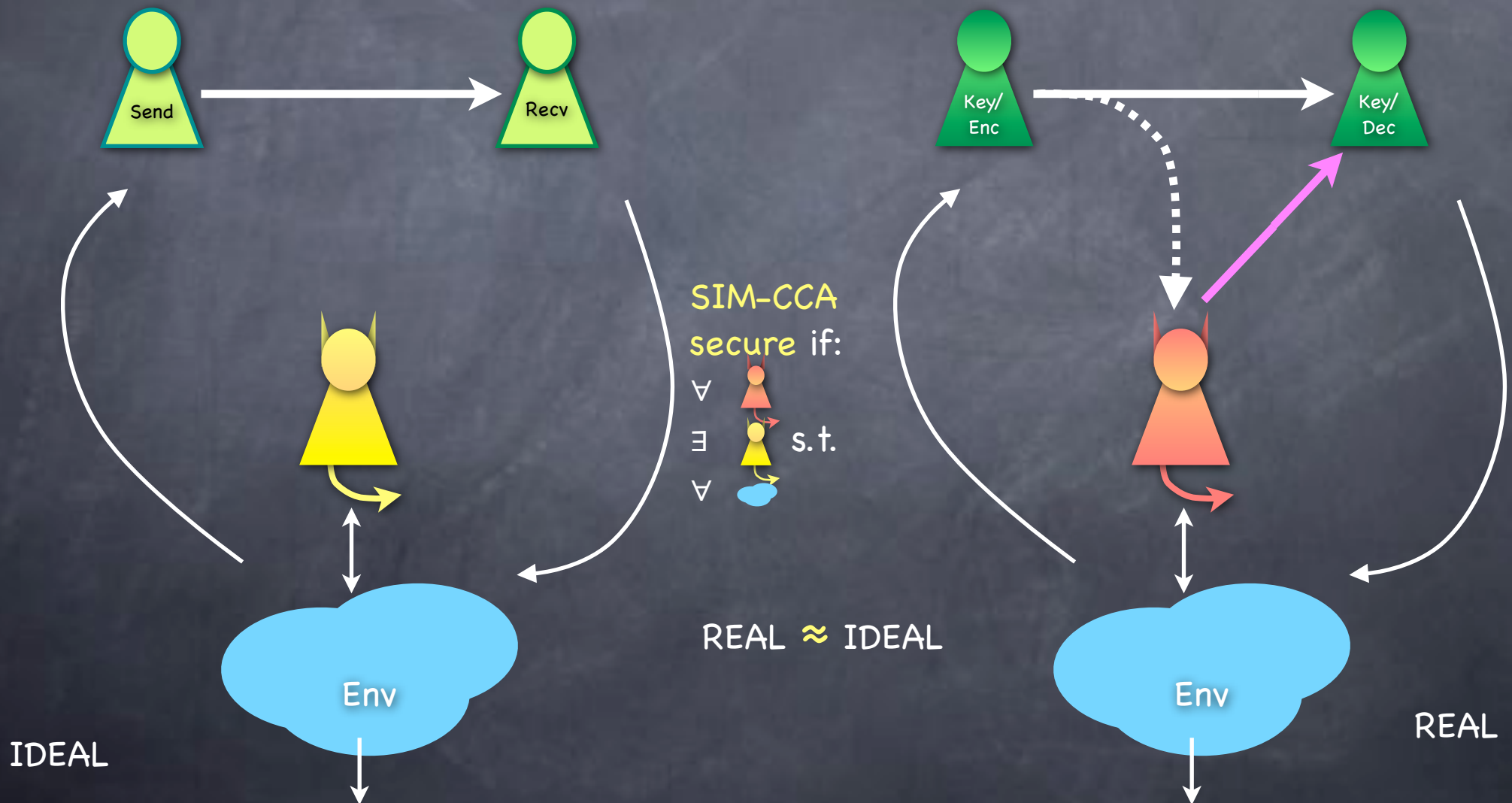




# Symmetric-Key Encryption

## SIM-CCA Security

- An active adversary can inject messages into the channel

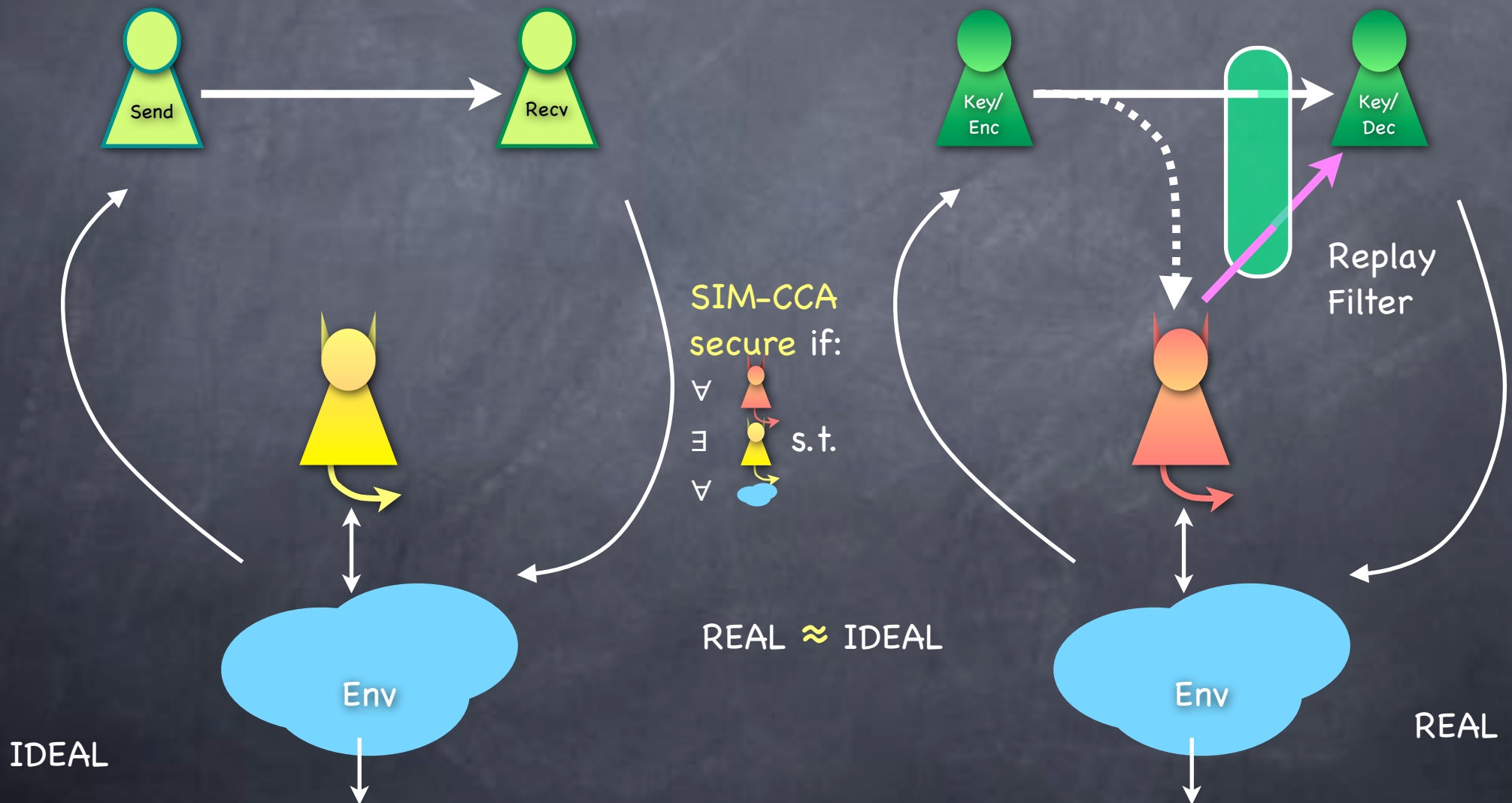




# Symmetric-Key Encryption

## SIM-CCA Security

- An active adversary can inject messages into the channel



# Symmetric-Key Encryption

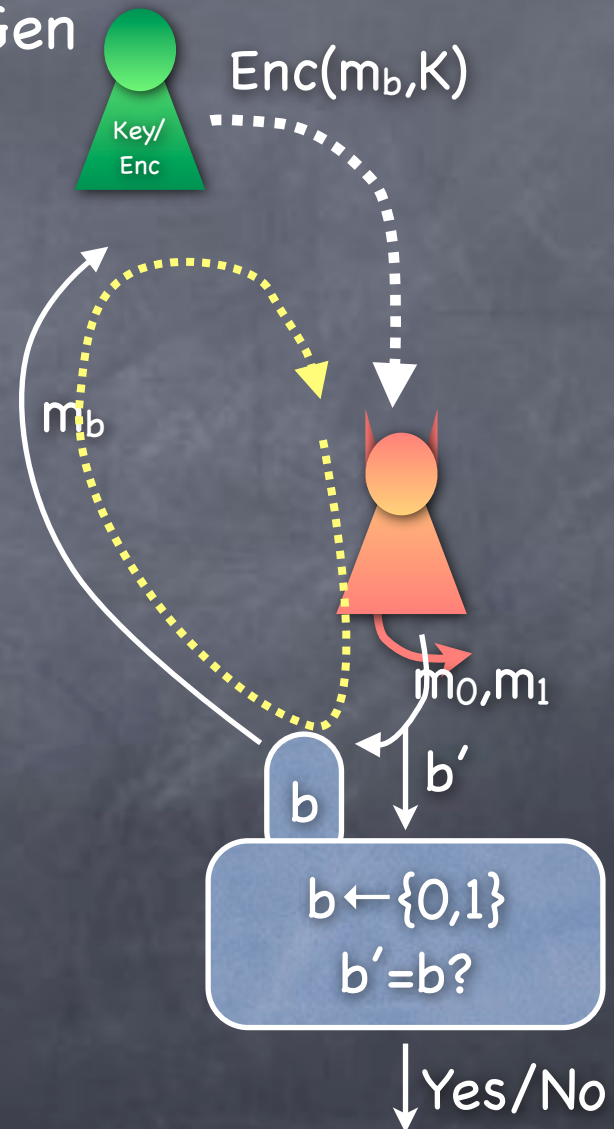
## IND-CCA Security

- Experiment picks  $b \leftarrow \{0,1\}$  and  $K \leftarrow \text{KeyGen}$

- For as long as Adversary wants**

- Adv sends two messages  $m_0, m_1$  to the experiment
- Expt returns  $\text{Enc}(m_b, K)$  to the adversary

- Adversary returns a guess  $b'$
- Experiment outputs 1 iff  $b' = b$
- IND-CCA secure** if for all feasible adversaries  $\Pr[b' = b] \approx 1/2$



# Symmetric-Key Encryption

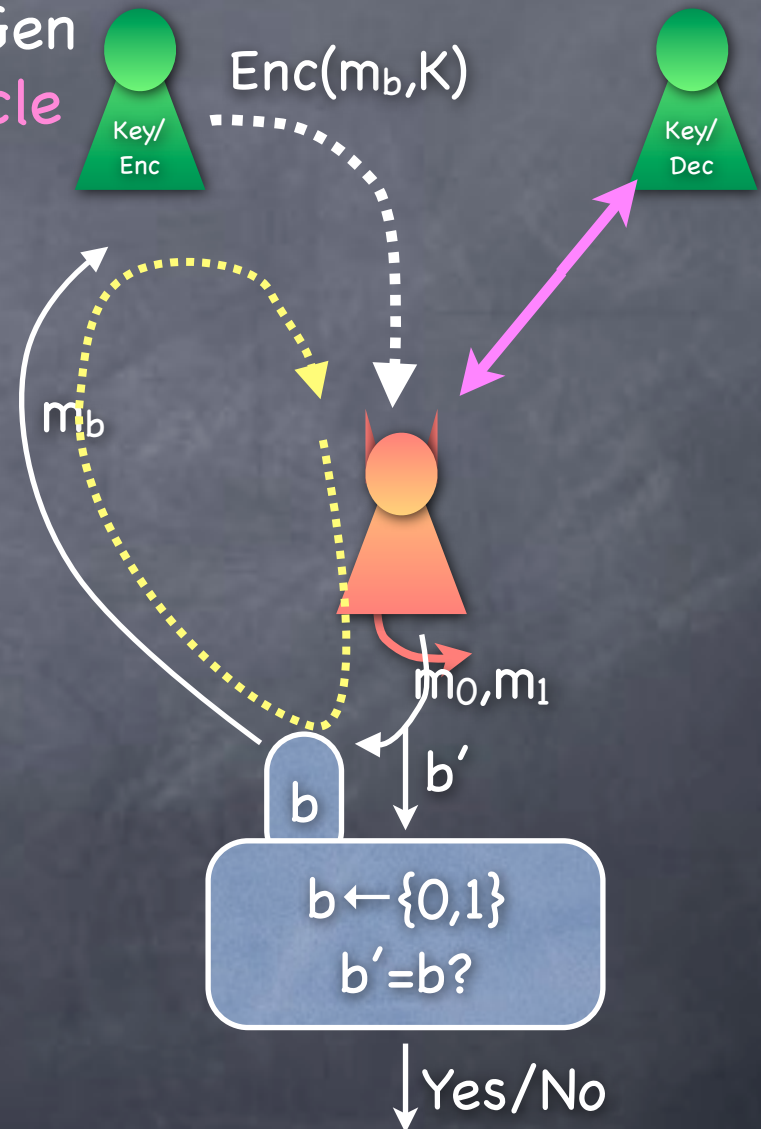
## IND-CCA Security

- Experiment picks  $b \leftarrow \{0,1\}$  and  $K \leftarrow \text{KeyGen}$   
Adv gets (guarded) access to  $\text{Dec}_K$  oracle

- For as long as Adversary wants

- Adv sends two messages  $m_0, m_1$  to the experiment
- Expt returns  $\text{Enc}(m_b, K)$  to the adversary

- Adversary returns a guess  $b'$
- Experiment outputs 1 iff  $b' = b$
- IND-CCA secure if for all feasible adversaries  $\Pr[b' = b] \approx 1/2$





# Symmetric-Key Encryption

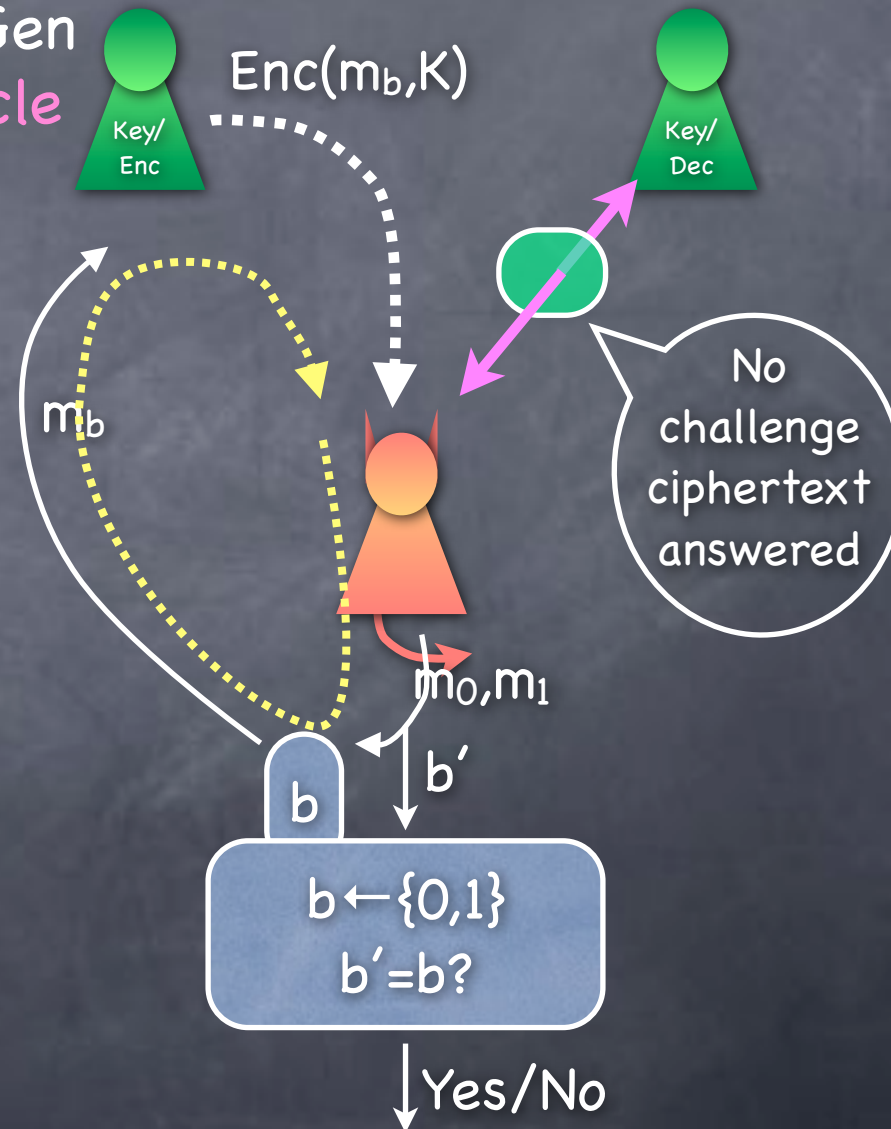
## IND-CCA Security

- Experiment picks  $b \leftarrow \{0,1\}$  and  $K \leftarrow \text{KeyGen}$   
Adv gets (guarded) access to  $\text{Dec}_K$  oracle

- For as long as Adversary wants

- Adv sends two messages  $m_0, m_1$  to the experiment
- Expt returns  $\text{Enc}(m_b, K)$  to the adversary

- Adversary returns a guess  $b'$
- Experiment outputs 1 iff  $b' = b$
- IND-CCA secure if for all feasible adversaries  $\Pr[b' = b] \approx 1/2$



# Symmetric-Key Encryption

## IND-CCA Security

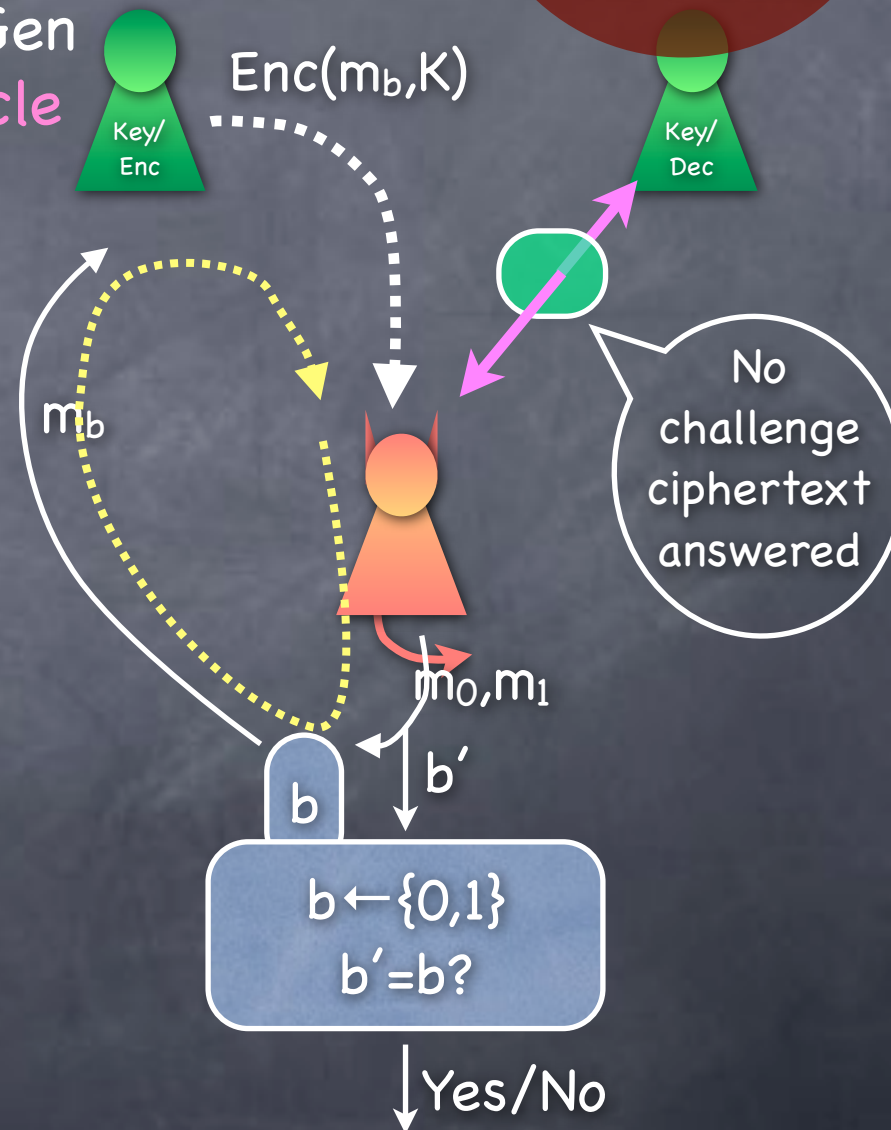
IND-CCA +  
~correctness  
equivalent to  
SIM-CCA

- Experiment picks  $b \leftarrow \{0,1\}$  and  $K \leftarrow \text{KeyGen}$   
Adv gets (guarded) access to  $\text{Dec}_K$  oracle

- For as long as Adversary wants

- Adv sends two messages  $m_0, m_1$  to the experiment
- Expt returns  $\text{Enc}(m_b, K)$  to the adversary

- Adversary returns a guess  $b'$
- Experiment outputs 1 iff  $b' = b$
- IND-CCA secure if for all feasible adversaries  $\Pr[b' = b] \approx 1/2$





# Perspective on Definitions

# Perspective on Definitions

- “Technical” vs. “Convincing”

# Perspective on Definitions

- “Technical” vs. “Convincing”
- For simple scenarios technical definitions could be convincing

# Perspective on Definitions

- “Technical” vs. “Convincing”
- For simple scenarios technical definitions could be convincing
  - e.g. Perfect Secrecy

# Perspective on Definitions

- “Technical” vs. “Convincing”
- For simple scenarios technical definitions could be convincing
  - e.g. Perfect Secrecy
- IND- definitions tend to be technical: more low-level details, but may not make the big picture clear. Could have “weaknesses”



# Perspective on Definitions

- “Technical” vs. “Convincing”
- For simple scenarios technical definitions could be convincing
  - e.g. Perfect Secrecy
- IND- definitions tend to be technical: more low-level details, but may not make the big picture clear. Could have “weaknesses”
- SIM- definitions give the big picture, but may not give details of what is involved in satisfying it. Could be “too strong”

# Perspective on Definitions

- “Technical” vs. “Convincing”
- For simple scenarios technical definitions could be convincing
  - e.g. Perfect Secrecy
- IND- definitions tend to be technical: more low-level details, but may not make the big picture clear. Could have “weaknesses”
- SIM- definitions give the big picture, but may not give details of what is involved in satisfying it. Could be “too strong”
- Best of both worlds when they are equivalent:
  - use IND- definition while say proving security of a construction;
  - use SIM- definition when low-level details are not important

# Summary

# Summary

- Security definitions:

# Summary

- Security definitions:
  - SIM-Onetime, SIM-CPA, SIM-CCA



# Summary

- Security definitions:
  - SIM-Onetime, SIM-CPA, SIM-CCA
  - IND-Onetime/Perfect Secrecy, IND-CPA, IND-CCA

# Summary

- Security definitions:
  - SIM-Onetime, SIM-CPA, SIM-CCA
  - IND-Onetime/Perfect Secrecy, IND-CPA, IND-CCA
- Next Lecture

# Summary

- Security definitions:
  - SIM-Onetime, SIM-CPA, SIM-CCA
  - IND-Onetime/Perfect Secrecy, IND-CPA, IND-CCA
- Next Lecture
  - For multi-message schemes we relaxed the “perfect” simulation requirement

# Summary

- Security definitions:
  - SIM-Onetime, SIM-CPA, SIM-CCA
  - IND-Onetime/Perfect Secrecy, IND-CPA, IND-CCA
- Next Lecture
  - For multi-message schemes we relaxed the “perfect” simulation requirement
  - But what is  $\approx$  ?

# Summary

- Security definitions:
  - SIM-Onetime, SIM-CPA, SIM-CCA
  - IND-Onetime/Perfect Secrecy, IND-CPA, IND-CCA
- Next Lecture
  - For multi-message schemes we relaxed the “perfect” simulation requirement
  - But what is  $\approx$  ?
  - And, how to build symmetric-key encryption schemes