Defining Encryption (ctd.)

Lecture 3

SIM & IND security

Beyond One-Time: **CPA** security

Onetime Encryption

Perfect Secrecy

A (2,2)-secret-sharing scheme: K and Enc(m,K) are shares of m

- **Perfect secrecy**: $\forall$ m, m' $\in \mathcal{M}$

  - $\{Enc(m,K)\}_{K \leftarrow KeyGen} = \{Enc(m',K)\}_{K \leftarrow KeyGen}$

  Distribution of the ciphertext is defined by the randomness in the key

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

- In addition, require **correctness**

  - $\forall$ m, K, Dec(Enc(m,K), K) = m

- E.g. One-time pad: $\mathcal{M} = \mathcal{K} = \mathcal{C} = \{0,1\}^n$ and
Enc(m,K) = m$\oplus$K, Dec(c,K) = c$\oplus$K

  - More generally $\mathcal{M} = \mathcal{K} = \mathcal{C} = \mathcal{G}$ (a finite group)
and Enc(m,K) = m+K, Dec(c,K) = c–K

Fix any m. For any c $\in \mathcal{G}$,
$Pr_{K \leftarrow \mathcal{G}}[Enc(m,K) = c]$
$= Pr_{K \leftarrow \mathcal{G}}[m + K = c]$
$= Pr_{K \leftarrow \mathcal{G}}[K = -m + c]$
$= 1/|\mathcal{G}|$

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 $Enc(m_b, K)$

  - Adversary returns a guess b'

  - Experiments outputs 1 iff $b'=b$

- IND-Onetime secure if for every adversary, $Pr[b'=b] = 1/2$

Key/Enc

$Enc(m_b, K)$

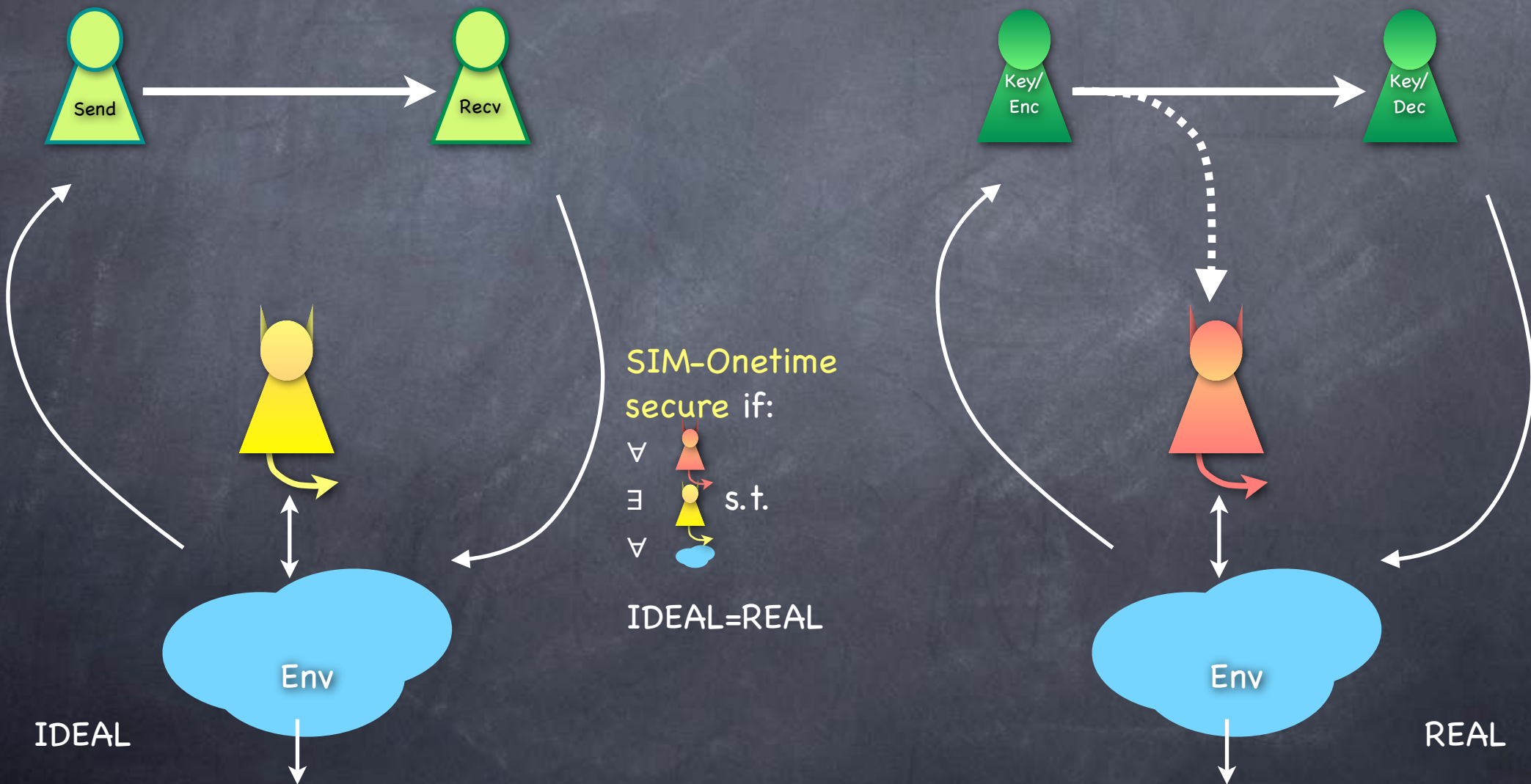m_b

m_0, m_1

b'

$b \leftarrow \{0,1\}$
$b'=b?$

Yes/No
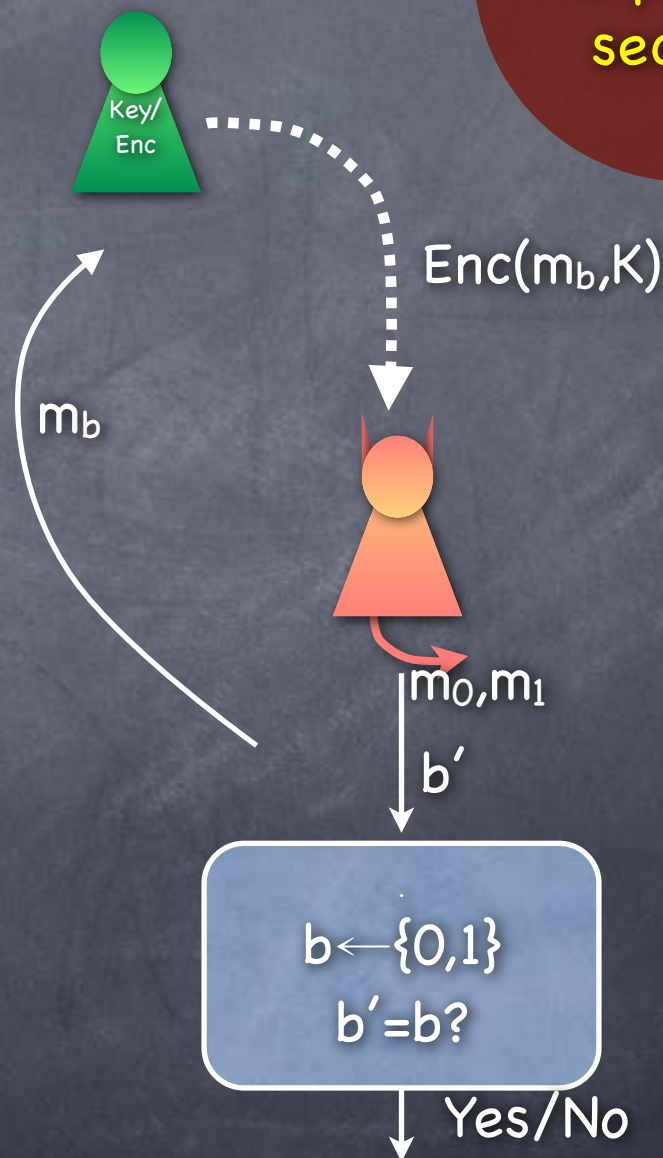
Onetime Encryption
SIM-Onetime Security

Equivalent to perfect secrecy + correctness

- Class of environments which send only one message

Send → Recv

Key/Enc → Key/Dec

SIM-Onetime secure if:

∀ ∃ s.t. ∀

IDEAL=REAL

Env

IDEAL

Env

REAL

Security of Encryption

- Perfect secrecy is too strong for multiple messages (though, as we shall see later, too weak in some other respects)

  - Requires keys as long as the messages

- Relax the requirement by restricting to computationally bounded adversaries (and environments)

- Coming up: Formalizing notions of "computational" security (as opposed to perfect/statistical security)

  - Then, security definitions used for encryption of multiple messages

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**: <u>Randomized</u>

    - 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

    - Dec: $\mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$

Symmetric-Key Encryption
Security Definitions

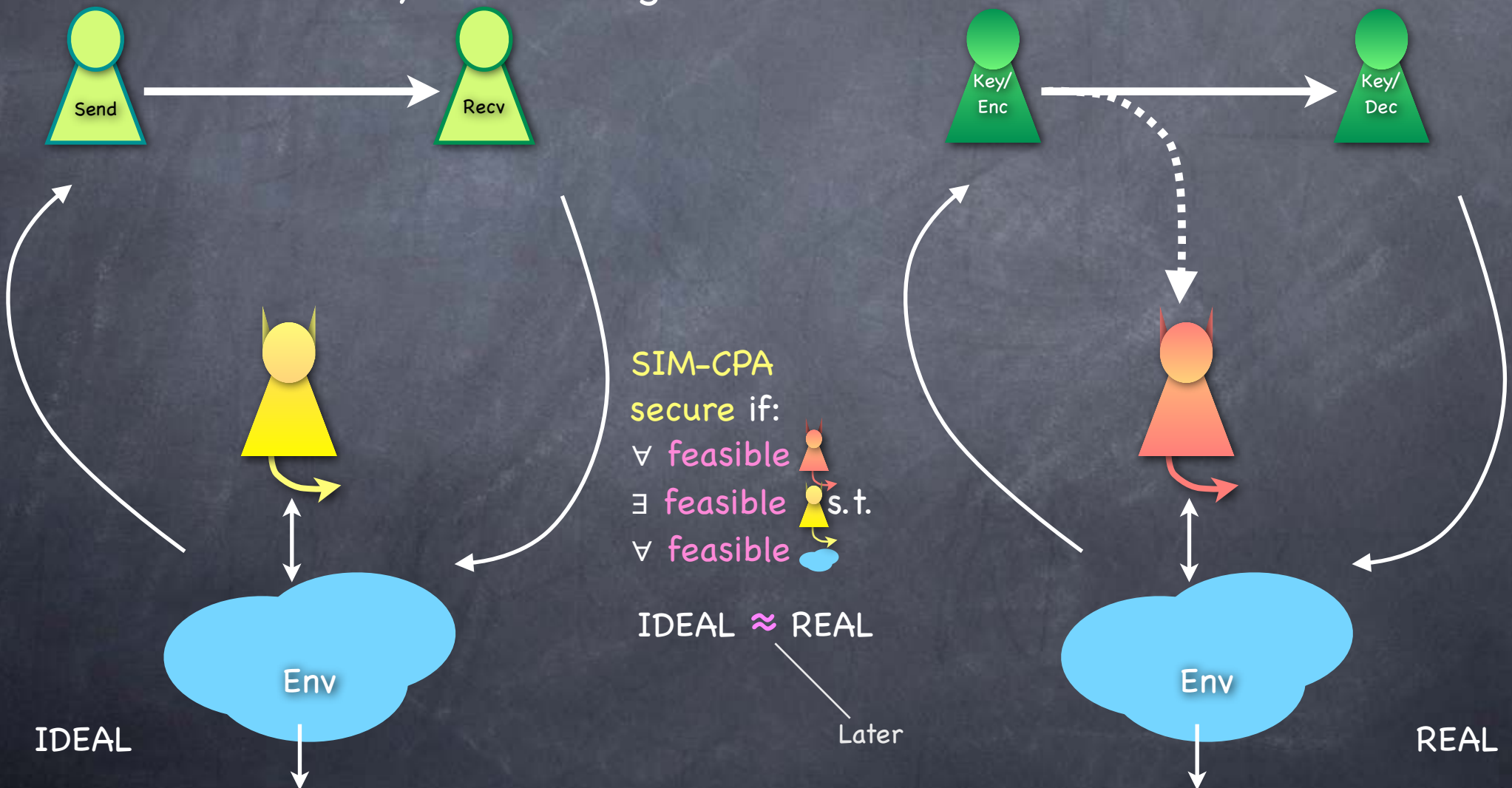| Security of Encryption	Information theoretic	Game-based	Simulation-based
One-time	Perfect secrecy & Perfect correctness $\equiv$	IND-Onetime & Perfect correctness $\equiv$	SIM-Onetime
Multi-msg		IND-CPA & correctness $\equiv$	SIM-CPA today
Active/multi-msg		IND-CCA & correctness $\equiv$	SIM-CCA

- CPA: Chosen Plaintext Attack

  - The adversary can influence/choose the messages being encrypted

  - Note: One-time security also allowed this, but for only one message

Symmetric-Key Encryption
SIM-CPA Security

Same as SIM-onetime security, but not restricted to environments which send only one message. Also, now all entities "feasible."

Send

Recv

Key/Enc

Key/Dec

SIM-CPA secure if:
∀ feasible
∃ feasible s.t.
∀ feasible

IDEAL ≈ REAL

Later

Env

Env

IDEAL

REAL

Symmetric-Key Encryption
IND-CPA Security

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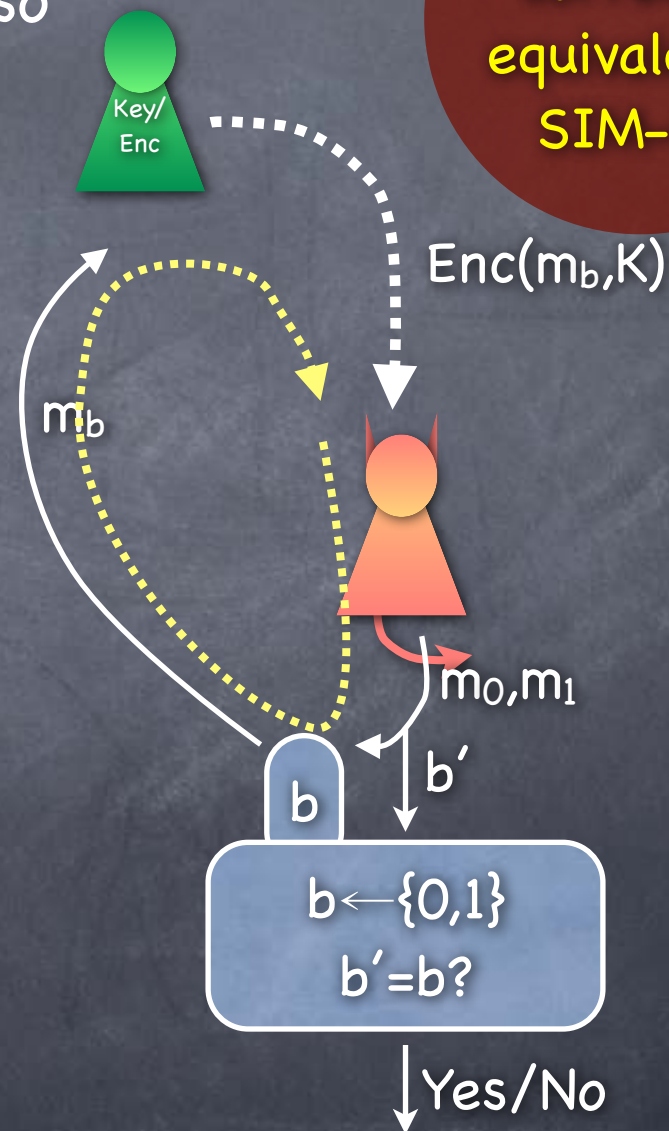
  - For as long as Adversary wants

    - Adv sends two messages m_0, m_1 to the experiment

    - Expt returns $Enc(m_b, K)$ 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

Key/Enc

$Enc(m_b, K)$

m_b

m_0, m_1

b'

b

$b \leftarrow \{0,1\}$
$b' = b?$

Yes/No

Feasible Computation
&
Negligible Advantage

Feasible Computation

- In analyzing complexity of algorithms: Rate at which computational complexity grows with input size

  - e.g. Can do sorting in $O(n \log n)$

- Only the rough rate considered

  - Exact time depends on the technology

  - Real question: Do we scale well? How much more computation will be needed as the instances of the problem get larger.

  - "Polynomial time" ($O(n)$, $O(n^2)$, $O(n^3)$, ...) considered feasible

■ Log ■ Poly ■ Exp
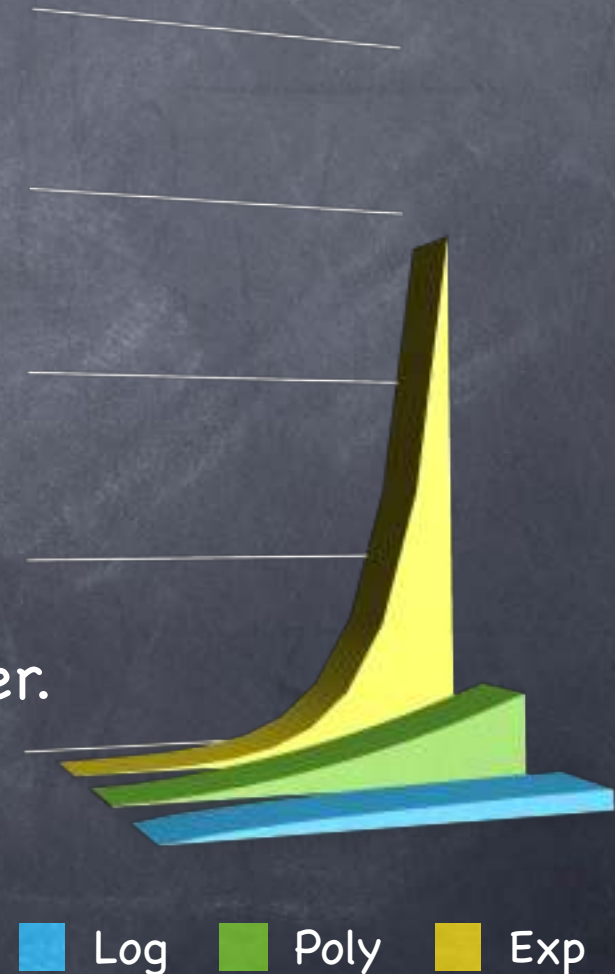
Infeasible Computation

- "Super-Polynomial time" considered infeasible

  - e.g. 2^n, $2^{\sqrt{n}}$, $n^{\log(n)}$

  - i.e., as n grows, quickly becomes "infeasibly large"

- Can we make breaking security infeasible for Eve?

  - What is n (that can grow)?

  - Message size?

    - We need security even if sending only one bit!

Security Parameter

- A parameter that is part of the encryption scheme

    - Not related to message size

    - A knob that can be used to set the security level

    - Will denote by k

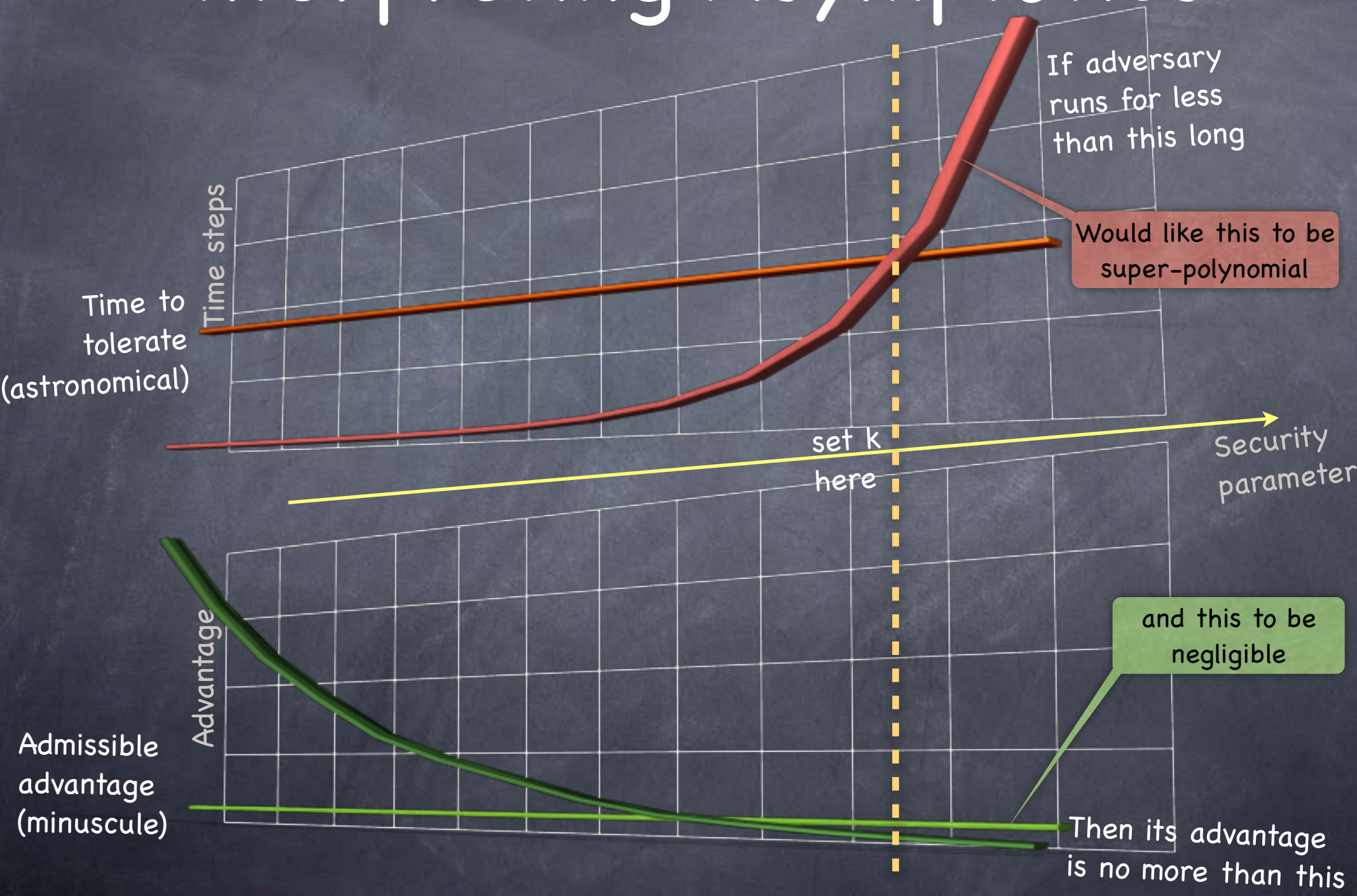- Security guarantees are given <u>asymptotically</u> as a function of the security parameter

Feasible and Negligible

- We want to tolerate Eves who have a running time bounded by some polynomial in k

    - Eve could toss coins: Probabilistic Polynomial-Time (PPT)

    - It is better that we allow Eve high polynomial times too (we'll typically tolerate some super-polynomial time for Eve)

        - But algorithms for Alice/Bob better be very efficient

    - Eve could be non-uniform: a different strategy for each k

- Such an Eve should have only a "negligible" advantage (or, should cause at most a "negligible" difference in the behaviour of the environment in the SIM definition)
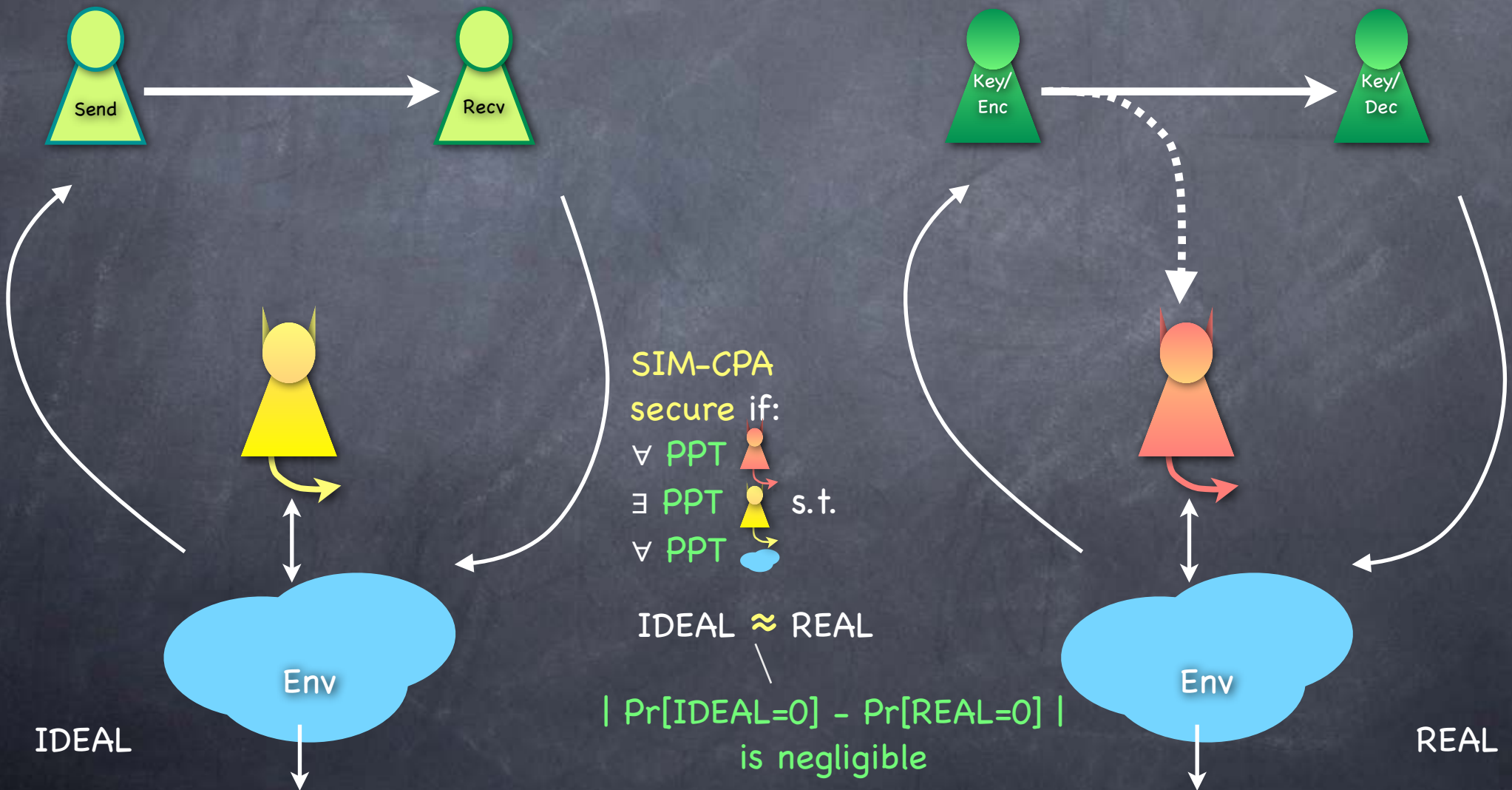
    - What is negligible?

Negligibly Small

- A negligible quantity: As we turn the knob the quantity should "decrease extremely fast"

  - Negligible: decreases as 1/superpoly(k)

    - i.e., faster than 1/poly(k) for every polynomial

    - e.g.: 2^{-k}, $2^{-\sqrt{k}}$, $k^{-(\log k)}$.

    - Formally: T negligible if $\forall c>0 \; \exists k_0 \; \forall k>k_0 \;\; T(k) < 1/k^c$

  - So that $negl(k) \times poly(k) = negl'(k)$

    - Needed, because Eve can often increase advantage polynomially by spending that much more time/by seeing that many more messages

Interpreting Asymptotics

Time steps

Time to tolerate (astronomical)

If adversary runs for less than this long

Would like this to be super-polynomial

set k here

Security parameter

Advantage

Admissible advantage (minuscule)

and this to be negligible

Then its advantage is no more than this

Symmetric-Key Encryption
SIM-CPA Security

Symmetric-Key Encryption
IND-CPA Security

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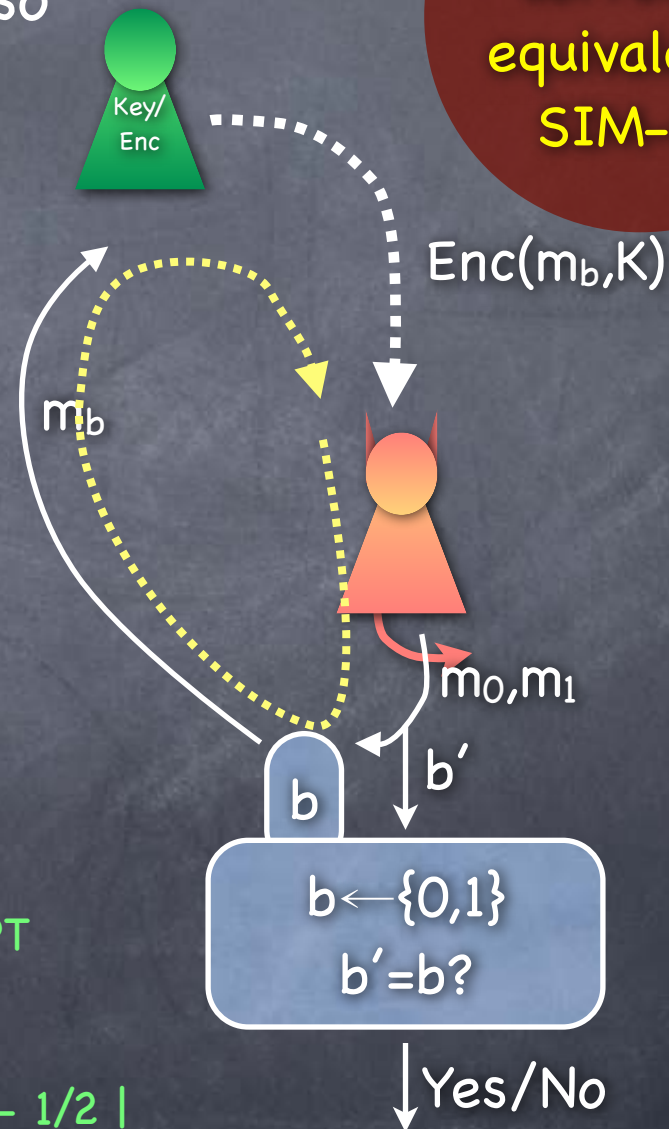
  - For as long as Adversary wants

    - Adv sends two messages m_0, m_1 to the experiment

    - Expt returns $Enc(m_b,K)$ to the adversary

  - Adversary returns a guess b'

  - Experiment outputs 1 iff b'=**b**   PPT

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

  | $Pr[b'=b] - 1/2$ | is negligible

IND-CPA + ~correctness equivalent to SIM-CPA

Key/ Enc

$Enc(m_b,K)$

m_b

m_0,m_1

b'

b

$b \leftarrow \{0,1\}$
b'=b?

Yes/No

Statistical Correctness

- We relaxed the "perfect" simulation requirement to IDEAL ≈ REAL

- In particular, this allows "almost perfect" correctness

  - Recall perfect correctness

    - $\forall$ m, $\Pr_{K \leftarrow KeyGen, Enc}[$ Dec(Enc(m,K), K) = m $] = 1$

  - Almost perfect correctness: a.k.a. **Statistical correctness**

    - $\forall$ m, $\Pr_{K \leftarrow KeyGen, Enc}[$ Dec(Enc(m,K), K) = m $] \approx 1$

      - i.e., $\Pr_{K \leftarrow KeyGen, Enc}[$ Dec(Enc(m,K), K) $\neq$ m $]$ is negligible

Next

- Constructing (CPA-secure) SKE schemes

    - Pseudorandomness Generator (PRG)

    - One-Way Functions (& OW Permutations)

    - OWP → PRG → (CPA-secure) SKE