Defining Encryption (ctd.)

Lecture 3

SIM & IND security

Beyond One-Time: **CPA** security

Computational Indistinguishability

Recall

Onetime Encryption

Perfect Secrecy

- **Perfect secrecy:** $\forall\ m, m' \in \mathcal{M}$
  - $\{\text{Enc}(m,K)\}_{K\leftarrow\text{KeyGen}} = \{\text{Enc}(m',K)\}_{K\leftarrow\text{KeyGen}}$

- Distribution of the 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

Assuming K uniformly drawn from $\mathcal{K}$

Pr[Enc(a,K)=x] = ¼,
Pr[Enc(a,K)=y] = ½,
Pr[Enc(a,K)=z] = ¼

Same for Enc(b,K).

- 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⊕K, Dec(c,K) = c⊕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

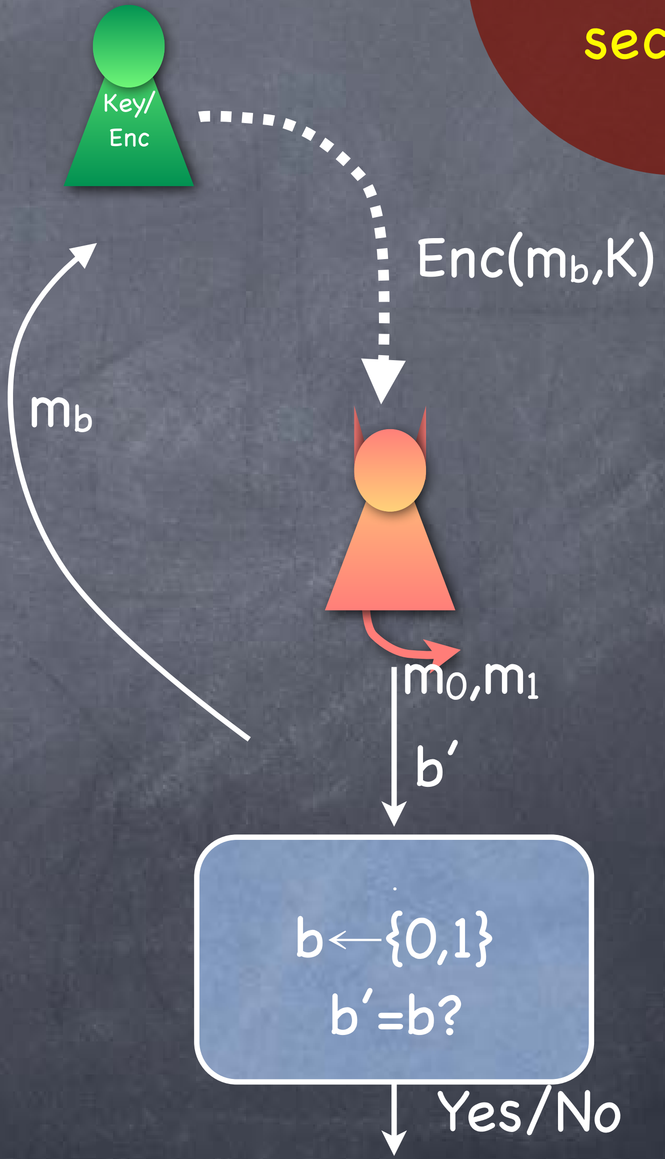
Recall

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



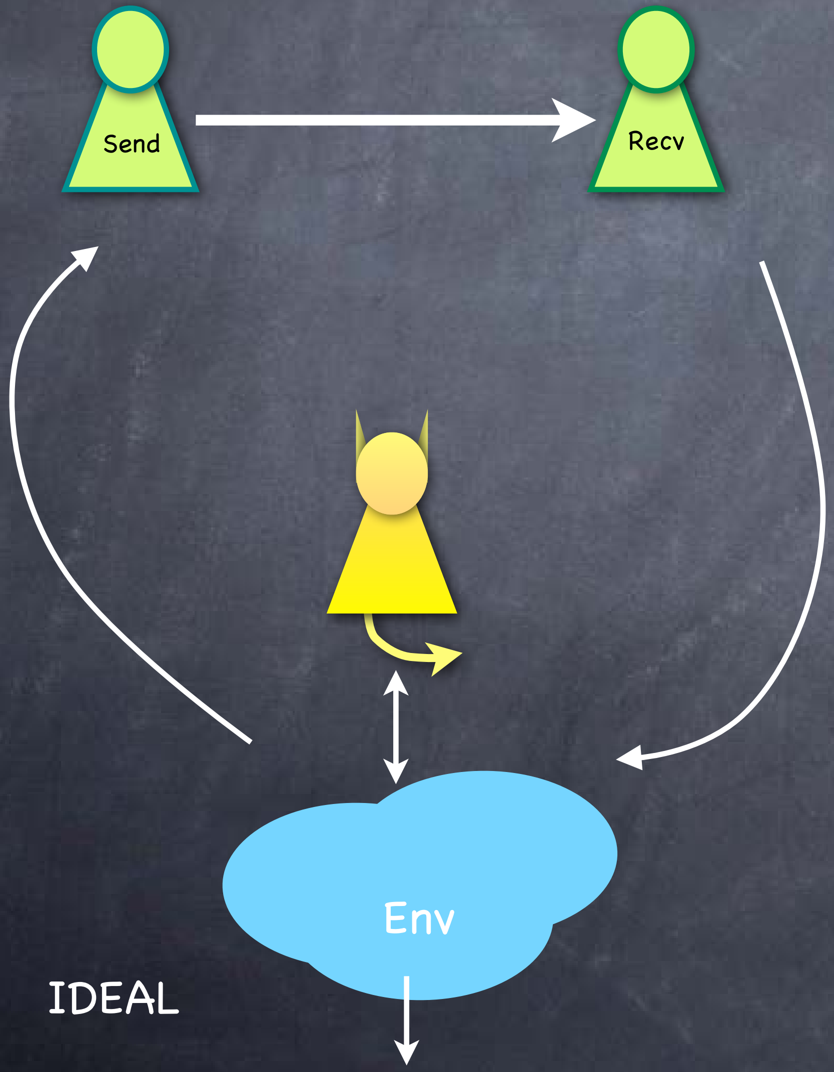
Recall

Onetime Encryption

SIM-Onetime Security

Equivalent to perfect secrecy + correctness

- Class of environments which send only one message



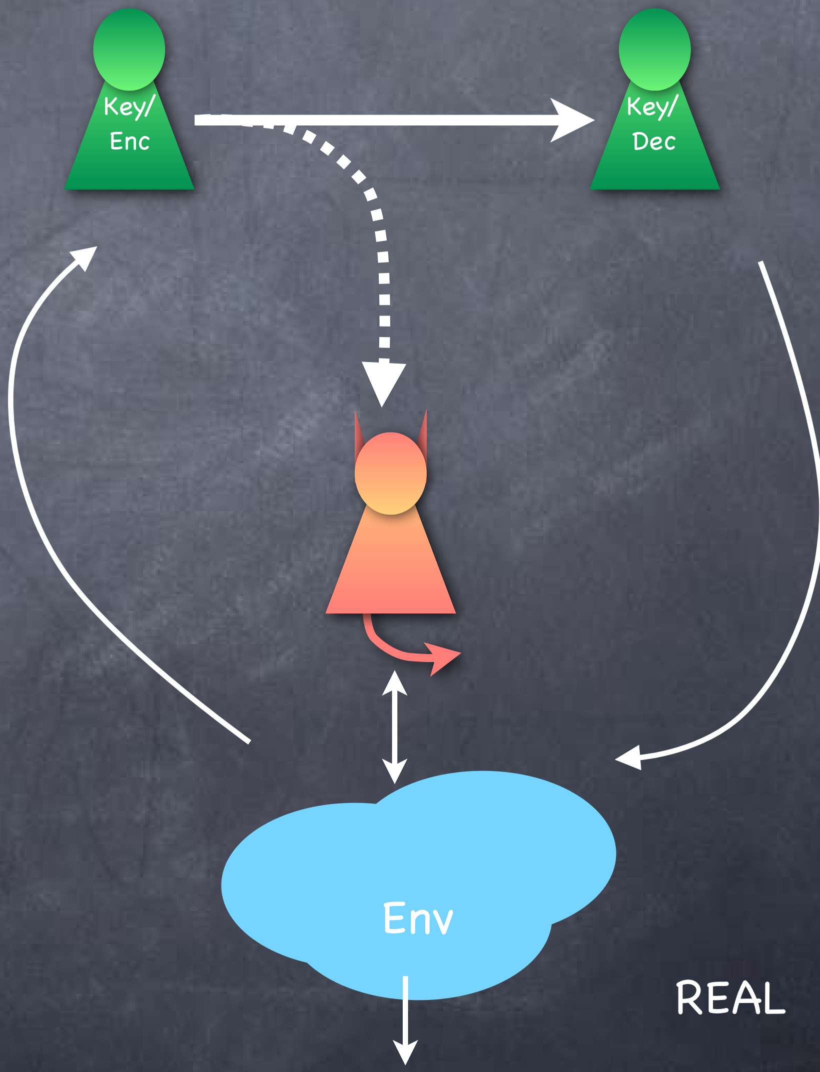
SIM-Onetime secure if:

∀

∃ s.t.

∀

IDEAL=REAL



Security of Encryption

- Perfect secrecy is too strong for multiple messages (though 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: Randomized
    - 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	≡	IND-Onetime & Perfect correctness	≡	SIM-Onetime
Multi-msg	🟥		IND-CPA & correctness	≡	SIM-CPA
Active/multi-msg	🟥		IND-CCA & correctness	≡	SIM-CCA

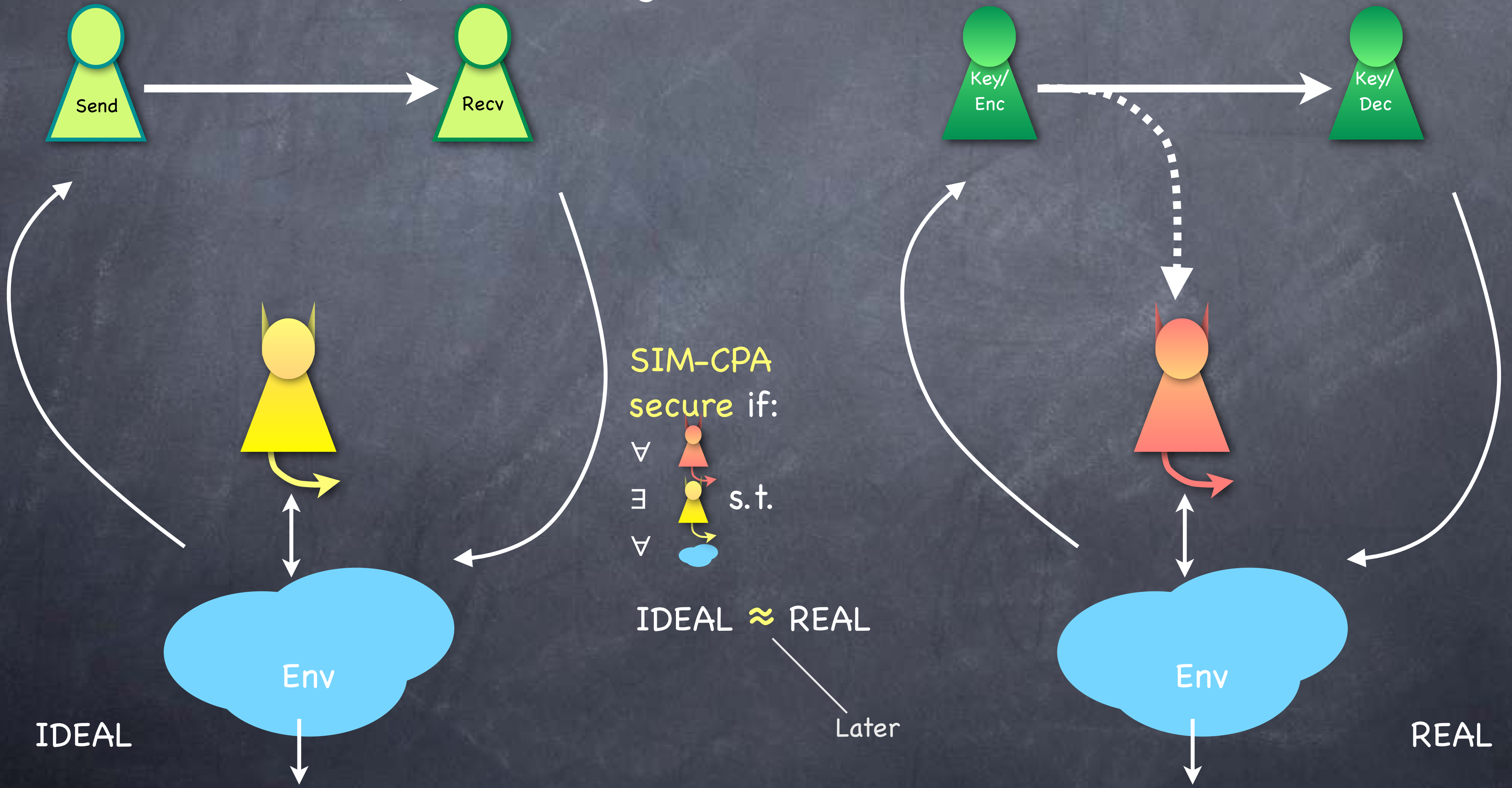
today

- 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 "efficient."



SIM-CPA secure if:

∀ ∃ s.t. ∀

IDEAL ≈ REAL

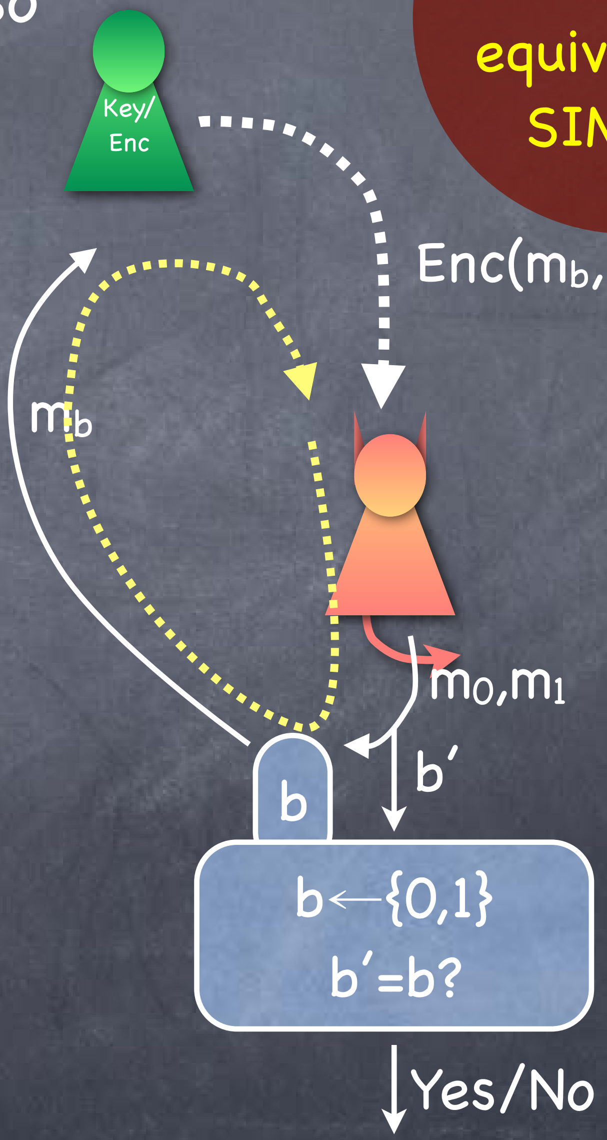
Later

Symmetric-Key Encryption

IND-CPA Security

IND-CPA + ~correctness equivalent to SIM-CPA

- 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 "efficient" adversaries Pr[b′=b] ≈ 1/2

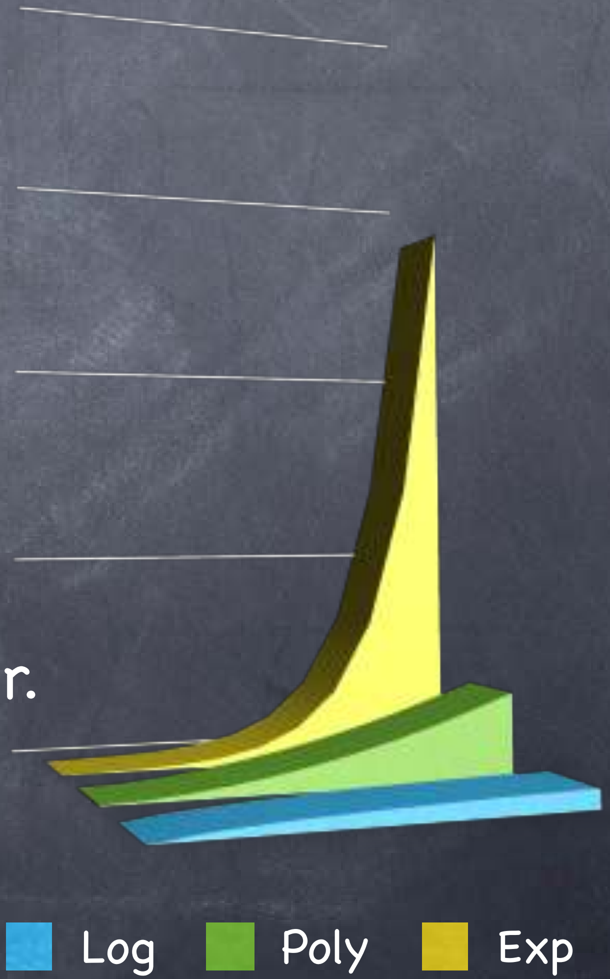


Almost Perfect

- For multi-message schemes we relaxed the "perfect" simulation requirement to IDEAL ≈ REAL
- In particular, we settle for "almost perfect" correctness
  - Recall perfect correctness
    - $\forall$ m, $\Pr_{K \leftarrow \text{KeyGen, Enc}}$ [Dec(Enc(m,K), K) = m] = 1
  - Almost perfect correctness: a.k.a. **Statistical correctness**
    - $\forall$ m, $\Pr_{K \leftarrow \text{KeyGen, Enc}}$ [Dec(Enc(m,K), K) = m] ≈ 1
  - But what is ≈ ?

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



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 asymptotically as a function of the security parameter

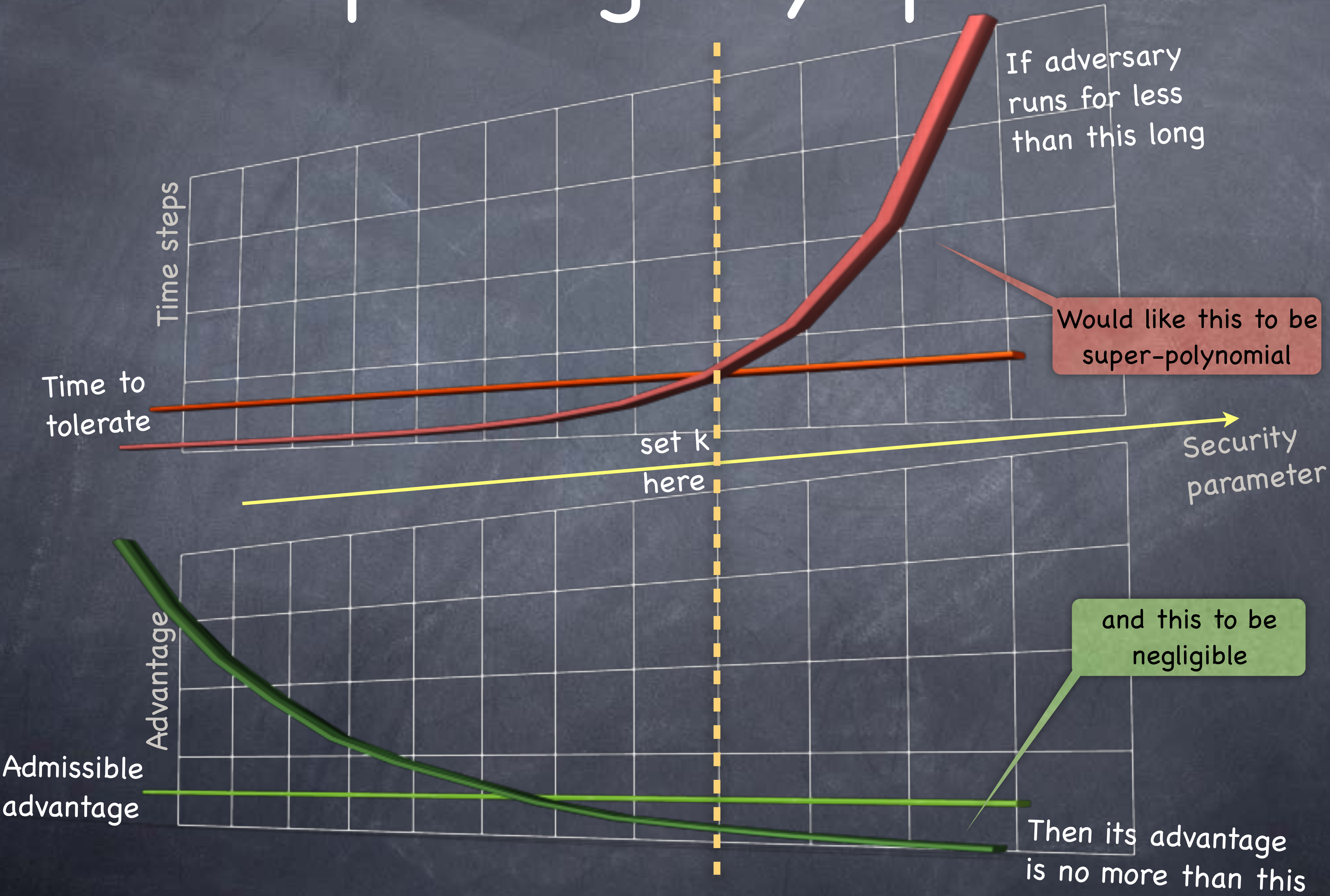
Feasible and Negligible

- 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 behavior 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}, $2^{-\surd k}$, $k^{-(\log k)}$.
    - Formally: T negligible if $\forall c{>}0\ \exists k_0\ \forall k{>}k_0\ \ T(k) < 1/k^c$
  - So that negl(k) × 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



If adversary runs for less than this long

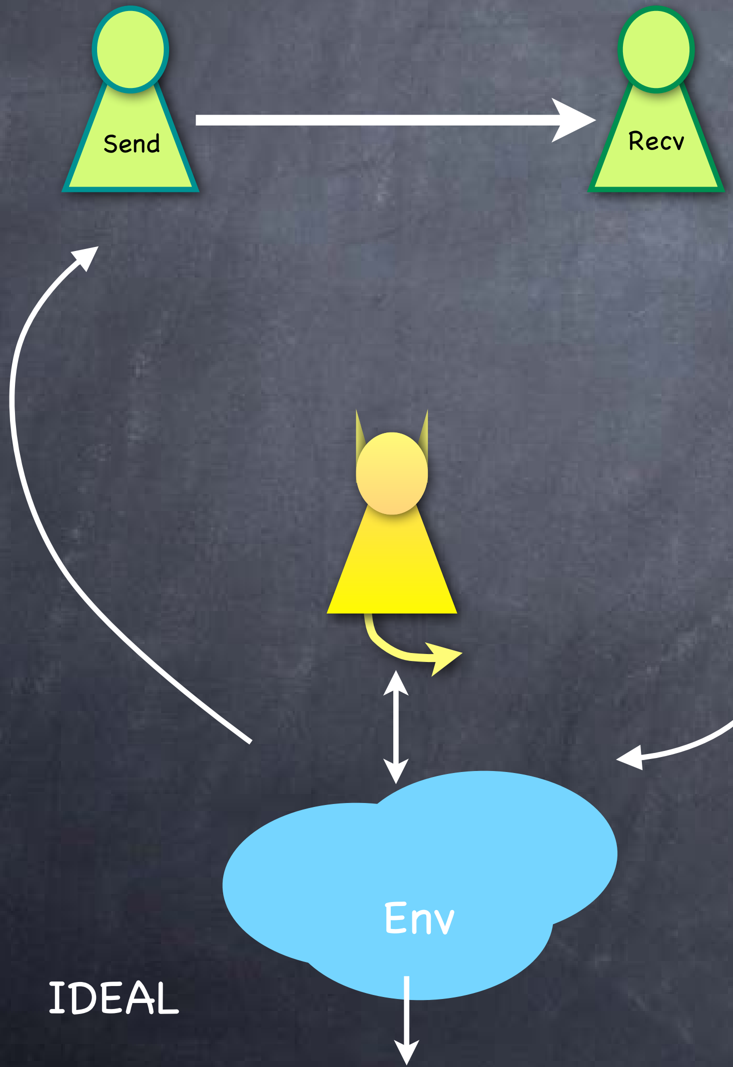
Would like this to be super-polynomial

and this to be negligible

Then its advantage is no more than this

Symmetric-Key Encryption

SIM-CPA Security



SIM-CPA secure if:

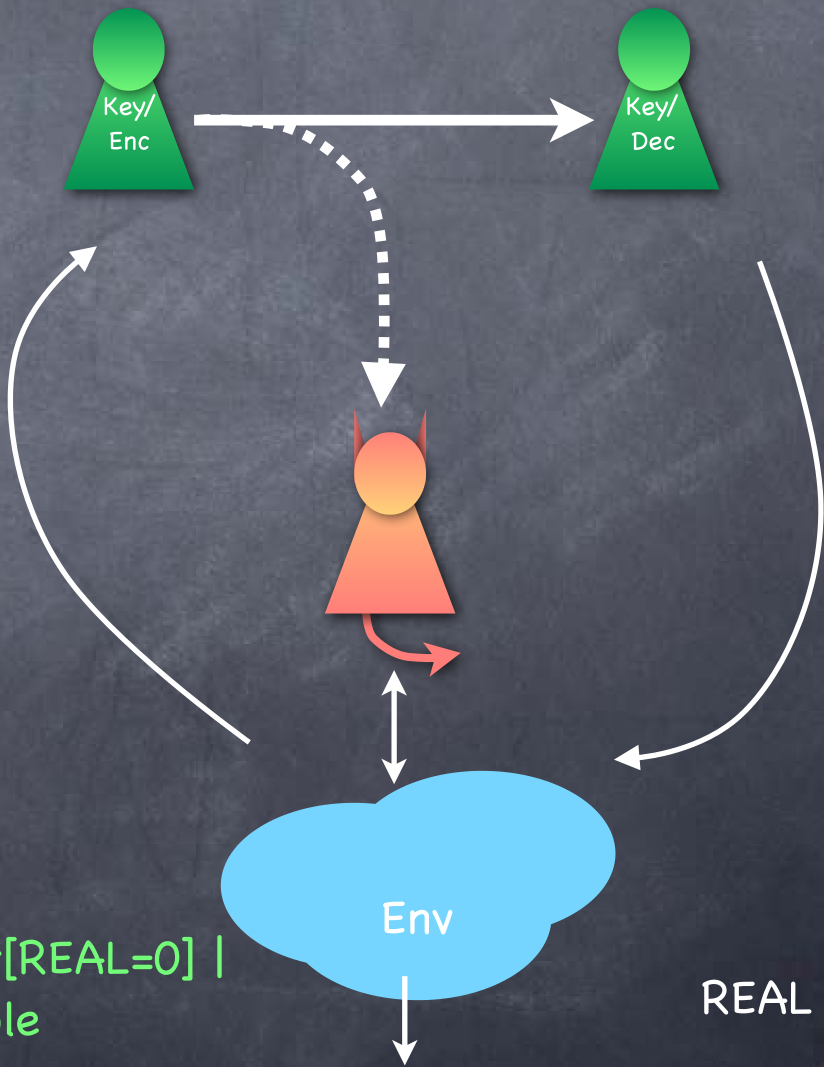
∀ PPT

∃ PPT s.t.

∀ PPT

IDEAL ≈ REAL

| Pr[IDEAL=0] - Pr[REAL=0] |
is negligible



Aside: Indistinguishability

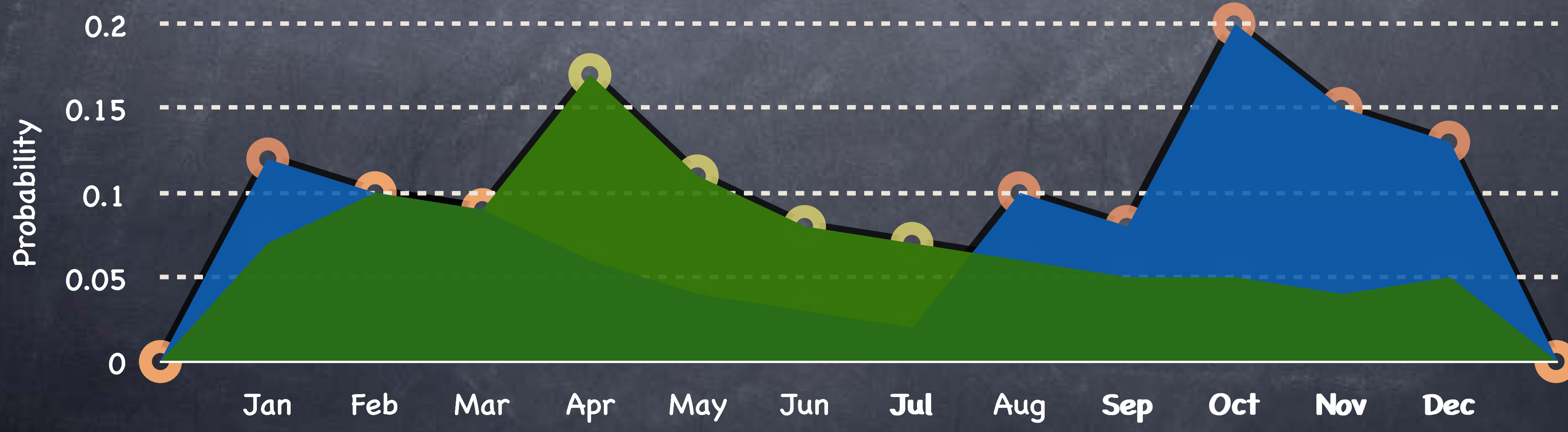
- Security definitions often refer to indistinguishability of two distributions: e.g., REAL vs. IDEAL, or Enc(m_0) vs. Enc(m_1)
- 3 levels of indistinguishability
    - Perfect: the two distributions are identical
    - Computational: for all PPT distinguishers, probability of the output bit being 1 is only negligibly different in the two cases
    - Statistical: the two distributions are "statistically close"
        - Hard to distinguish, irrespective of the computational power of the distinguisher

Statistical Indistinguishability

- Given two distributions A and B over the same sample space, how well can a (computationally unbounded) test T distinguish between them?
  - T is given a single sample drawn from A or B
  - How differently does it behave in the two cases?
- $\Delta(A,B) := \max_T | \Pr_{x \leftarrow A}[T(x)=1] - \Pr_{x \leftarrow B}[T(x)=1] |$

Statistical Difference (Distance) or Total Variation Distance

- Two distribution ensembles $\{A_k\}_k$, $\{B_k\}_k$ are **statistically indistinguishable** from each other if $\Delta(A_k,B_k)$ is negligible in k



Next

- Constructing (CPA-secure) SKE schemes
  - Pseudorandomness Generator (PRG)
  - One-Way Functions (& OW Permutations)
  - OWP → PRG → (CPA-secure) SKE