CS 65500
Advanced Cryptography

Lecture 1: Indistinguishability

Instructor: Aarushi Goel

Spring 2025

Agenda

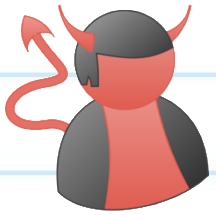
- → Independence / Perfect Secrecy
- → Statistical Indistinguishability
- → Computational Indistinguishability

Private Communication

$m \in \{0,1\}$

Alice

Bob

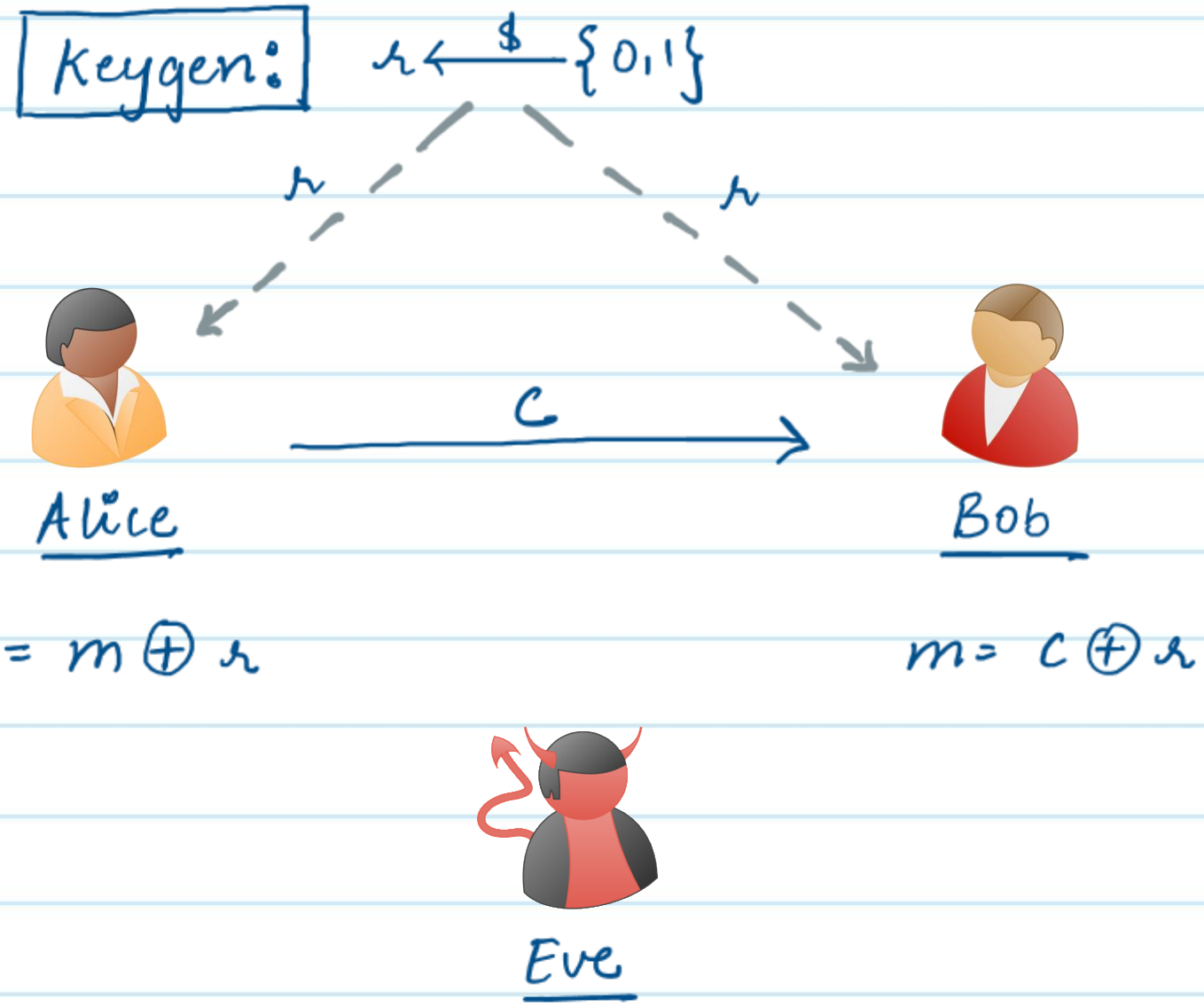


Eve

(Computationally Unbounded)

How can Alice send m to Bob, while keeping it hidden from an eavesdropper Eve?

One - Time Pad



One-Time Pad

Let $m = 0$. What are the possible values of c?

prob	r	$c = m \oplus r$
$1/2$	0	0
$1/2$	1	1

$\Rightarrow$ Whatever **Eve sees** (i.e., c) is independent of m

also called the "view" of the adversary

Secrecy

→ Is the message m really secret?

→ Eve could have easily guessed m with probability $1/2$.
In fact if they already knew something about m, they can do better.

NOTE: we did not claim that m is random

→ But Eve could have done this without looking at c.
c did not leak any additional information about m.

Secrecy

Typical goal in cryptography:

PRESERVE SECRECY!

Intuitively speaking, this is what we want:

> What Eve learns about m after seeing c, is the same as what they already knew about m.

Formalizing Secrecy

Event 1 : What did Eve already know about the message?

- Probability distribution over m
- i.e., $\forall m$, $Pr[msg = m]$

Event 2 : What does Eve learn after seeing c?

- New distribution $Pr[msg = m \mid view = c]$

What do we want for Secrecy?

Eve's Knowledge in Event 1

=

Eve's Knowledge in Event 2

Formalizing Secrecy

$$\Rightarrow \forall m, \forall c, \quad Pr[msg = m \mid view = c] = Pr[msg = m]$$

view is independent of msg

$$\Rightarrow \forall m, \forall c, \quad Pr[view = c \mid msg = m] = Pr[view = c]$$

for all possible values of msg, the view is distributed identically

$$\Rightarrow \forall c, \forall m_1, m_2, \quad Pr[view = c \mid msg = m_1] = Pr[view = c \mid msg = m_2]$$

Formalizing Secrecy (Summary)

These are equivalent formulations:

$$\forall m, \forall c, \quad Pr[msg = m \mid view = c] = Pr[msg = m]$$

$$\forall m, \forall c, \quad Pr[view = c \mid msg = m] = Pr[view = c]$$

$$\forall m, \forall c, \quad Pr[msg = m, \ view = c] = Pr[msg = m] \times Pr[view = c]$$

$$\forall c, \forall m_1, m_2, \quad Pr[view = c \mid msg = m_1] = Pr[view = c \mid msg = m_2]$$

Formalizing Secrecy

Is $Pr[msg = m_1 \mid view = v] = Pr[msg = m_2 \mid view = v]$?

Why / Why not ?

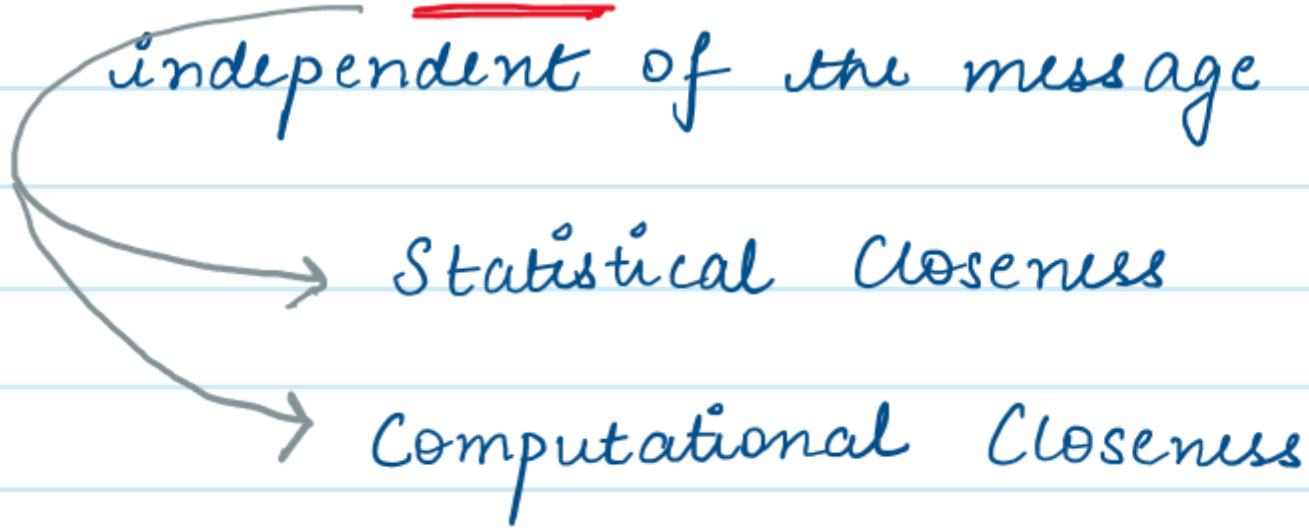
This is only true if msg is uniform.

Relaxing Secrecy Requirement

What if the view is not exactly independent of the message?

Next Best Thing:

View is close to a distribution that is independent of the message



Statistical Difference

Given two distributions A & B over some sample space, how well can a test T distinguish between them?

- T is given a sample drawn from A or B
- How differently does it behave in the two cases?

$$\Delta(A, B) = \max_T \left| \Pr_{x \leftarrow A}[T(x) = 0] - \Pr_{x \leftarrow B}[T(x) = 0] \right|$$

Statistical difference between A & B

max over all such possible tests

Statistical Indistinguishability

→ A and B are statistically indistinguishable from each other if the statistical difference between them is **negligible** Examples: 2^{-20}, 2^{-50}, 2^{-100}

we let the user decide which of these they want.

→ Decide how? Using a security parameter K

Security guarantees will be given asymptotically as a function of the security parameter.

→ Given $\{A_K\}$ & $\{B_K\}$, $\Delta(A_K, B_K)$ is a function of K.

we want this to be negligible function in K!

Negligible Functions

- The best distinguishing test T^* should have extremely small probability of success.
- If T^* has extremely small success probability given one sample, the best distinguishing test should also have extremely small success probability given polynomially many samples
- Functions that decay so quickly (i.e., approach to zero) that they cannot be rescued by any polynomial

Negligible Functions

Definition: A function $\nu(.)$ is negligible if for every polynomial $p(.)$, we have

$$\lim_{n \to \infty} p(k) \cdot \nu(k) = 0$$

$\Rightarrow$ A negligible function decays faster than all inverse polynomial functions.

Definition: A function $\nu(k)$ is negligible if $\forall c \geq 0$, $\exists N$ s.t., $\forall\ k > N$, $\nu(k) \leq \frac{1}{k^c}$

Statistical Indistinguishability

Definition: Distribution ensembles $\{A_K\}$, $\{B_K\}$ are statistically indistinguishable if

$\exists$ negligible $\nu(\cdot)$, s.t., $\forall K$, $\Delta(A_K, B_K) \leq \nu(K)$

$\Rightarrow$ $\exists$ negligible $\nu(\cdot)$, s.t $\forall$ tests T, $\forall K$

$$\left| \Pr_{x \leftarrow A_K} [T_K(x) = 0] - \Pr_{x \leftarrow B_K} [T_K(x) = 0] \right| \leq \nu(K)$$

Ques: is this equivalent to: $\forall$ tests T, $\exists$ negligible $\nu(\cdot)$ s.t. $\forall K$ $\left| \Pr_{x \leftarrow A_K} [T_K(x) = 0] - \Pr_{x \leftarrow B_K} [T_K(x) = 0] \right| \leq \nu(K)$?

0

Computational Indistinguishability

Definition: Distribution ensembles $\{A_K\}, \{B_K\}$ are computationally indistinguishable if

$\forall$ efficient tests T, $\exists$ negligible $\nu(\cdot)$ s.t. $\forall K$,

$$\left| \Pr_{x \leftarrow A_K} [T_K(x) = 0] - \Pr_{x \leftarrow B_K} [T_K(x) = 0] \right| \leq \nu(K)$$

What is efficient?

Cost of Computation

It can be helpful to think of cost of computation in terms of monetary value. Following costs are approximated using the pricing model of Amazon EC2

clock cycles	*approx cost*	*reference*
2^{50}	*$3.50*	*cup of coffee*
2^{55}	*$100*	*decent tickets to a Portland Trailblazers game*
2^{65}	*$130,000*	*median home price in Oshkosh, WI*
2^{75}	*$130 million*	*budget of one of the Harry Potter movies*
2^{85}	*$140 billion*	*GDP of Hungary*
2^{92}	*$20 trillion*	*GDP of the United States*
2^{99}	*$2 quadrillion*	*all of human economic activity since 300,000 BC*[4]
2^{128}	*really a lot*	*a billion human civilizations' worth of effort*

Computational Security

John Nash

"It doesn't really matter whether attacks are impossible, only whether attacks are computationally infeasible"

Modern cryptography is based on this principle.

Efficient = Probabilistic Polynomial Time. (PPT)

Computational Security

Non-uniform PPT T: A family of randomized programs $\{T_\kappa\}$ (one for each value of the security parameter κ), s.t. there is a polynomial $p()$ with each T_κ running for at most time $p(\kappa)$.

Uniform PPT T: where T is a single program that takes κ as an additional input.

By default: We will consider non-uniform PPT algorithms / adversaries / tests / distinguishers.

Choosing an Appropriate Security Parameter

Some example references for what extremely small probabilities are equivalent to.

probability	*equivalent*
2^{-10}	*full house in 5-card poker*
2^{-20}	*royal flush in 5-card poker*
2^{-28}	*you win this week's Powerball jackpot*
2^{-40}	*royal flush in 2 consecutive poker games*
2^{-60}	*the next meteorite that hits Earth lands in this square* →