

Commitment Schemes

Yael Tauman Kalai

MIT 6.5620, Lecture 13

October 26, 2026

1 Coin Tossing over the Phone

Suppose Alice and Bob want to toss a coin, but they are not in the same room and do not trust each other. Alice can toss a coin and tell Bob the outcome, but why should Bob believe her? This is the coin-tossing problem introduced by Blum [Blum83].

A natural idea is to let each party contribute a random bit. Alice chooses $a \xleftarrow{\$} \{0, 1\}$, Bob chooses $b \xleftarrow{\$} \{0, 1\}$, and they output $a \oplus b$. If at least one of the bits is uniform and independent of the other, then $a \oplus b$ is uniform.

What goes wrong? Someone has to send the first message. If Alice sends a first, then Bob can choose $b = a$ to make the outcome 0, or $b = 1 \oplus a$ to make it 1. Reversing the order only changes which party can cheat. Asking them to send their bits “simultaneously” does not solve the problem in a model where a party can wait to receive the other party’s message before sending its own.

What we would like is for Alice to *fix* her bit before Bob chooses his, without *revealing* her bit until after Bob has sent his. Does this ring a bell?

A first attempt using encryption. How about Alice sends $c = \text{Enc}(\text{pk}, a; r)$, Bob sends b , and Alice then reveals (a, r) ? Bob checks that $c = \text{Enc}(\text{pk}, a; r)$. The question is who chooses the public key? If Bob chooses it then he can choose it together with a corresponding secret key, and thus decrypt a and choose his bit b depending on a . So the hiding property does not hold. If Alice chooses it, she can choose it maliciously, possibly not even in the support of the key generation algorithm, in a way that it cannot be decrypted, and binding may not hold.

One can prove that both the hiding and binding properties hold if pk is honestly generated by a trusted party, and the encryption scheme has perfect

correctness, namely, for all but negligible fraction of $(\mathbf{pk}, \mathbf{sk})$ in the support of key generation, every bit a , and every encryption random tape r ,

$$\text{Dec}(\mathbf{sk}, \text{Enc}(\mathbf{pk}, a; r)) = a.$$

As we show next, the primitive we are looking for, can be constructed from merely one-way functions.

Desired primitive: Digital locked box. Alice puts her bit in an opaque locked box and gives the box to Bob. Bob sends his bit, and Alice then gives Bob the key. We are assuming that Alice cannot change the contents after handing over the box, and Bob cannot see the contents before receiving the key. These are two different requirements:

- **Hiding:** Bob should not learn Alice's bit before choosing his own.
- **Binding:** Alice should not be able to change her bit after seeing Bob's.

A commitment scheme is a way to digitally implement this locked box.

The proposed protocol. For the moment, assume we have such a scheme.

1. Alice chooses $a \xleftarrow{\$} \{0, 1\}$ and sends a commitment to a .
2. Bob chooses $b \xleftarrow{\$} \{0, 1\}$ and sends b to Alice.
3. Alice opens the commitment, so that Bob obtains a .
4. The outcome is $a \oplus b$.

Remark 1. Commitments cannot force Alice to open, just as a locked box cannot force her to send the key. We therefore allow an abort outcome, denoted by $\perp$.

2 Commitment Schemes

We begin with computationally hiding, statistically binding commitments. We write $\approx_c$ for computational indistinguishability, and use security against nonuniform polynomial-size adversaries throughout.

Definition 1. A commitment scheme corresponding to a message space $\mathcal{M}$ consists of a pair of algorithms (Gen, Com) :

- **Gen** is a PPT algorithm that takes as input the security parameter 1^λ and outputs public parameters $\mathbf{pp} \in \{0, 1\}^{\text{poly}(\lambda)}$.
- **Com** is a polynomial-time computable function that takes as input public parameters $\mathbf{pp}$, a message $m \in \mathcal{M}$, and randomness $r \in \{0, 1\}^{\rho(\lambda)}$, for some polynomial ρ , and outputs a commitment $\text{Com}(\mathbf{pp}, m, r)$.

The following two properties are required to hold:

- **Computational Hiding:** For every polynomial-size algorithm $\mathcal{A}$ that, on input honestly generated $\mathbf{pp}$, outputs equal-length $m_0, m_1 \in \mathcal{M}$,

$$(\mathbf{pp}, \text{Com}(\mathbf{pp}, m_0, r_0)) \approx_c (\mathbf{pp}, \text{Com}(\mathbf{pp}, m_1, r_1)),$$

where $\mathbf{pp} \leftarrow \text{Gen}(1^\lambda)$, $(m_0, m_1) = \mathcal{A}(\mathbf{pp})$, and r_0, r_1 are independent uniform strings in $\{0, 1\}^{\rho(\lambda)}$.

- **Statistical Binding:**

$$\Pr_{\mathbf{pp} \leftarrow \text{Gen}(1^\lambda)} \left[\begin{array}{c} \exists m_0, m_1 \in \mathcal{M}, r_0, r_1 \in \{0, 1\}^{\rho(\lambda)} : \\ m_0 \neq m_1 \wedge \text{Com}(\mathbf{pp}, m_0, r_0) = \text{Com}(\mathbf{pp}, m_1, r_1) \end{array} \right] \leq \text{negl}(\lambda).$$

For bit commitments, the hiding requirement simply says that commitments to 0 and 1 are computationally indistinguishable.

Opening a commitment. To open c to a message m , the sender reveals (m, r) , and the receiver checks that $m \in \mathcal{M}$ and that $c = \text{Com}(\mathbf{pp}, m, r)$.

Remark 2. One can instead require *statistical hiding*: negligible statistical distance, even against an unbounded receiver. *Computational binding* only prevents efficient senders from finding two openings to different messages; such openings may exist. Perfect hiding means identical commitment distributions.

Can both guarantees be statistical? The answer is no. For fixed public parameters $\mathbf{pp}$, let

$$S_b(\mathbf{pp}) = \{\text{Com}(\mathbf{pp}, b, r) : r \in \{0, 1\}^{\rho(\lambda)}\}.$$

If the scheme is binding for $\mathbf{pp}$, then

$$S_0(\mathbf{pp}) \cap S_1(\mathbf{pp}) = \emptyset.$$

An all-powerful receiver can determine which one contains the commitment. Thus, for every binding choice of $\mathbf{pp}$, the committed bit is completely determined by the commitment. Since the public parameters are binding except with negligible probability, the distributions of commitments to 0 and 1, together with the public parameters, have statistical distance at least $1 - \text{negl}(\lambda)$. Thus, statistical hiding is impossible.

3 Constructing a Commitment Scheme from One-Way Functions

We will first construct a scheme from a pseudorandom generator. Since pseudorandom generators can be constructed from any one-way function [HILL99], this gives a construction from any one-way function. The commitment construction is due to Naor [Naor91].

The construction. Let $G : \{0, 1\}^\lambda \rightarrow \{0, 1\}^{3\lambda}$ be a pseudorandom generator,

- $\text{Gen}(1^\lambda)$ samples $u \xleftarrow{\$} \{0, 1\}^{3\lambda}$ and outputs $\mathbf{pp} = u$.
- To commit to $b \in \{0, 1\}$ using randomness $r \xleftarrow{\$} \{0, 1\}^\lambda$, define

$$\text{Com}(u, b, r) = G(r) \oplus (b \cdot u),$$

where $0 \cdot u = 0^{3\lambda}$ and $1 \cdot u = u$.

To open, reveal (b, r) . The receiver recomputes the commitment.

Hiding. Fix any $u \in \{0, 1\}^{3\lambda}$. By pseudorandomness,

$$G(r) \approx_c U_{3\lambda} \equiv U_{3\lambda} \oplus u \approx_c G(r) \oplus u,$$

where $U_{3\lambda}$ denotes a uniformly random string of length 3λ . This proves hiding. Note that the same argument works when a malicious receiver chooses u . Hiding holds even for maliciously chosen u , provided that u has the prescribed length and is chosen before the sender samples the fresh commitment randomness.

Binding. A commitment has openings to both 0 and 1 only if there exist $r_0, r_1 \in \{0, 1\}^\lambda$ such that

$$G(r_0) = G(r_1) \oplus u,$$

or equivalently, $u = G(r_0) \oplus G(r_1)$. There are at most $2^{2\lambda}$ pairs (r_0, r_1) and therefore at most $2^{2\lambda}$ bad choices of u . It follows that

$$\Pr_{u \xleftarrow{\$} \{0,1\}^{3\lambda}} [\exists r_0, r_1 : G(r_0) \oplus G(r_1) = u] \leq \frac{2^{2\lambda}}{2^{3\lambda}} = 2^{-\lambda}.$$

Notice that the binding proof does not use pseudorandomness! It only uses the fact that G has λ input bits and 3λ output bits. Pseudorandomness is used for hiding.

Theorem 1. *Assuming the existence of one-way functions, there exists a computationally hiding, statistically binding bit commitment scheme. Moreover, in the construction above, hiding holds even when the receiver chooses the public parameter u maliciously, and the binding error for uniform u is at most $2^{-\lambda}$.*

Who chooses the public parameters? For the applications in this lecture, the receiver can choose u and send it to the sender before the commitment phase. This does not require a trusted third party.

If the sender is dishonest, the receiver is honest and samples u uniformly. This gives binding. If the receiver is dishonest, we need hiding, and hiding holds for every choice of u . The two requirements are used in different corruption cases.

Why not let the sender choose u ? A dishonest sender could choose $u = 0^{3\lambda}$, in which case

$$\text{Com}(u, 0, r) = \text{Com}(u, 1, r)$$

for every r . Hiding is perfect for this u , but binding fails completely. More generally, the sender could choose $u = G(r_0) \oplus G(r_1)$ and retain the two openings.

Committing to longer messages. To commit to $m = (m_1, \dots, m_k) \in \{0, 1\}^k$, use the same u and independent randomness $r_1, \dots, r_k$, and send

$$(\text{Com}(u, m_1, r_1), \dots, \text{Com}(u, m_k, r_k)).$$

For polynomially bounded k , hiding follows by replacing the coordinates one at a time in a hybrid argument. Binding follows from the same good choice of u : two different message openings must disagree in at least one coordinate.

4 Returning to Coin Tossing

We can now implement our proposed protocol. Bob first samples $u \xleftarrow{\$} \{0, 1\}^{3\lambda}$ and sends it to Alice. Alice then commits to a uniform bit a , Bob sends a uniform bit b , and Alice opens her commitment. Thus, including the public-parameter message, the messages are

$$\begin{aligned} \text{Bob} &\longrightarrow \text{Alice} : u, \\ \text{Alice} &\longrightarrow \text{Bob} : c = G(r) \oplus (a \cdot u), \\ \text{Bob} &\longrightarrow \text{Alice} : b, \\ \text{Alice} &\longrightarrow \text{Bob} : (a, r). \end{aligned}$$

Alice checks that $u \in \{0, 1\}^{3\lambda}$ and $b \in \{0, 1\}$, and Bob verifies the opening. A missing or invalid message causes the honest party to output $\perp$.

When both parties are honest, they output the same uniform bit and never abort. However, a dishonest Alice can refuse to open if she does not like the outcome $a \oplus b$. Thus, in the presence of cheating, we cannot require the honest party to output each bit with probability $1/2 \pm \text{negl}(\lambda)$. We can prove the following upper bound.

Claim 1. *Let $Y \in \{0, 1, \perp\}$ be the honest party's output. For every polynomial-size cheating strategy of either party and every fixed $t \in \{0, 1\}$,*

$$\Pr[Y = t] \leq \frac{1}{2} + \text{negl}(\lambda).$$

Notice that Claim 1 is not a statement about the distribution conditioned on completion. Rather, it is an *unconditional* probability, including executions that abort.

Proof of Claim 1. Suppose first that Alice is dishonest and Bob is honest. Except with probability at most $2^{-\lambda}$ over Bob's choice of u , every commitment has at most one valid committed bit. Thus, when Bob receives Alice's commitment, her bit a is already fixed, if the commitment has a valid opening at all. Bob then chooses an independent uniform bit b . Alice can either open to this fixed bit or abort. Therefore, for every $t \in \{0, 1\}$,

$$\Pr[Y = t] \leq \frac{1}{2} + 2^{-\lambda}.$$

Now suppose that Bob is dishonest and Alice is honest. Hiding holds even when Bob chooses u maliciously. To make the outcome equal a fixed bit t , Bob must send $b = a \oplus t$. Thus, if Bob could make the honest Alice output t with probability noticeably greater than $1/2$, we could predict her uniform bit a from its commitment: run Bob and output $b \oplus t$. If Bob aborts or sends an invalid message, output an independent uniform guess. This would contradict hiding. Hence

$$\Pr[Y = t] \leq \frac{1}{2} + \text{negl}(\lambda).$$

□

Can Alice bias the coin among completed executions? Yes. Alice can generate an honest commitment to a random a , and after receiving b , open only if $a \oplus b = 0$. In this attack,

$$\Pr[Y = 0] = \frac{1}{2}, \quad \Pr[Y = 1] = 0, \quad \Pr[Y = \perp] = \frac{1}{2}.$$

Consequently,

$$\Pr[Y = 0 \mid Y \neq \perp] = 1.$$

This does not contradict Claim 1.

5 Commitments without Public Parameters

Naor's construction does not require a trusted third party, but it does require the receiver to send a public parameter before the sender commits. We now show how to remove this initial message, under the stronger assumption that injective one-way functions exist.

Fix an injective one-way function $f : \{0, 1\}^\lambda \rightarrow \{0, 1\}^{\text{poly}(\lambda)}$, and let

$$P(r, s) = \langle r, s \rangle \bmod 2 = \bigoplus_{i=1}^{\lambda} r_i s_i.$$

Recall the Goldreich–Levin theorem [GL89]: for independent uniform $r, s \in \{0, 1\}^\lambda$,

$$(f(r), s, P(r, s)) \approx_c (f(r), s, U_1).$$

The function P is deterministic; s is sampled uniformly as part of the commitment randomness. Formally, P is hardcore for $(r, s) \mapsto (f(r), s)$.

The idea. The value $f(r)$ fixes r , because f is injective. However, it does not allow an efficient receiver to predict $P(r, s)$. We can therefore use this bit as a mask for the committed bit.

The construction. To commit to $b \in \{0, 1\}$, choose independent $r, s \xleftarrow{\$} \{0, 1\}^\lambda$ and define

$$\text{Com}(b, (r, s)) = (f(r), s, P(r, s) \oplus b).$$

To open, reveal r . Given a commitment (y, s, c) and an opening $r \in \{0, 1\}^\lambda$, the receiver checks that $f(r) = y$. If so, it outputs $b = c \oplus P(r, s)$; otherwise, it rejects.

Hiding. For either $b \in \{0, 1\}$, the hardcore property implies

$$(f(r), s, P(r, s) \oplus b) \approx_c (f(r), s, U_1).$$

Thus, commitments to 0 and 1 are computationally indistinguishable.

Binding. Suppose

$$(f(r), s, P(r, s) \oplus b) = (f(r'), s', P(r', s') \oplus b').$$

Since f is injective, $f(r) = f(r')$ implies $r = r'$. Equality of the second components gives $s = s'$. Equality of the last components then gives $b = b'$. The scheme is therefore perfectly binding.

Where did we use injectivity? Only in the binding proof. If $f(r) = f(r')$ but $P(r, s) \neq P(r', s)$, the sender can use these two preimages to open the same commitment to either bit.

References

- [Blum83] M. Blum. Coin flipping by telephone: a protocol for solving impossible problems. *ACM SIGACT News*, 15(1):23–27, 1983. doi:10.1145/1008908.1008911. The coin-tossing problem was introduced at CRYPTO '81.
- [GL89] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. *STOC*, pp. 25–32, 1989. doi:10.1145/73007.73010.
- [HILL99] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999. doi:10.1137/S0097539793244708.
- [Naor91] M. Naor. Bit commitment using pseudorandomness. *Journal of Cryptology*, 4:151–158, 1991. doi:10.1007/BF00196774.