

Commitment Schemes

PETS'26

Example:

I know that OpenAI solved
another millenium problem,
but I won't tell you which one

Failed attempts

- Hashing: $c \leftarrow H(\text{"Riemann"})$
- Encryption: $c \leftarrow \text{Enc}(k, \text{"Riemann"})$

maybe I can find k'

s.t. $\text{Enc}(k', \text{"P vs NP"}) = c$

Ex: $m \in \{0, 1, \dots, 6\}$ $\text{Enc}_k(m) = k + m \bmod 7$
pick $k' = c - m' \bmod 7$

Properties we want: "hiding" } commitment
"binding" } decommitment

Indistinguishability

Informally: we often have some
random variable from distribution
 D_0 ("adversary's view")

To argue privacy, we want to show
that the adv. can't distinguish
 D_0 from another distribution D_1 ,
which "clearly preserves privacy"

Ex: $D_0 = m + k \bmod 7$

$$D_1 = c \leftarrow \{0, 1, \dots, 6\}$$

Claim: $D_0 = D_1$

Def indistinguishability

Let D_0, D_1 be two distributions

- D_0 and D_1 are statistically indist. $D_0 \equiv D_1$ if $D_0 = D_1$ or they are "negligibly close".

Here the adv. has unlimited comp. budget and can query D a poly. # of times

eg $D_0 = \{0\}$

$$D_1 = \begin{cases} 0 & \text{w.p. } 1 - \text{negl}(\lambda) \\ 1 & \text{w.p. } \text{negl}(\lambda) \end{cases}$$

$$D_0 \equiv D_1$$

- D_0 and D_1 are comp. indist. $D_0 \approx^c D_1$

if no poly time adv A can tell the difference between D_0 and D_1 with non negligible probability,

Define events W_0 and W_1 of the adversary outputting "1" in two possible "worlds"

$$W_0 = \{A(x)=1 : x \leftarrow D_0\}$$

$$W_1 = \{A(x)=1 : x \leftarrow D_1\}$$

Def: Advantage of A

$$\text{Adv}[A] = |\Pr[W_0] - \Pr[W_1]|$$

$$D_0 \approx^c D_1 \iff \text{Adv}[A] = \text{negl}(n)$$

Ex: Semantic security for Enc scheme

$$\{ \text{Enc}_k(m) : k \xleftarrow{\$} K \}$$

$$\{ \text{Enc}_k(0) : k \xleftarrow{\$} K \} \approx^c$$

Def Commitment Schemes

An algorithm $\text{Commit} : M \times R \rightarrow C$

$$\text{Commit}(m, r) = c$$

Two properties:

1. Statistical Hiding: the commitment leaks nothing about the message

$$\begin{aligned} \forall m_0, m_1 \quad & \{ \text{Commit}(m_0, r) : r \xleftarrow{\$} R \} \\ & \equiv \{ \text{Commit}(m_1, r) : r \xleftarrow{\$} R \} \end{aligned}$$

2. Computational Binding: the adv. can't change their mind. No PPT adv can find $m_0, m_1 \in M$ $r_0, r_1 \in R$

s.t. $\text{Commit}(m_0, r_0) = \text{Commit}(m, r)$
for $m_0 \neq m$,

Claim: statistical binding + hiding
is impossible.

Q: Why?

Terminology

Committer

$c = \text{Commit}(m, r)$
→

Receiver

"opening" r
 m →

check
that

$c = \text{Commit}(m, r)$

The Simplest Commitment in the world

$$\text{Commit}(m, r) = H(m, r)$$

that's it! ▽

How do we prove security?

Random Oracle Model

To prove security we "pretend" that
our hash function (e.g. SHA-3) is
a completely random function

$H: X \rightarrow Y$ where for each $x \in X$
the output $H(x)$ is sampled uniformly
at random from Y

Then: $\text{Commit}(m, r) = H(m, r)$

'is comp. hiding and binding

in the random oracle model

($|R|, |C|$ of size $2^{\text{poly}(\lambda)}$)

Proof:

Hiding: 'if H is a random function,
 $H(m, r)$ is uniformly random over C
to the Adv, as long as they don't
query H on (m, r)

$$\Pr[\text{Adv queries } H \text{ on } (m, r)] = \text{negl}(\lambda)$$

Binding: a random function is collision resistant

Controversy

ROM $\neq$

Hardness
Assumption
(factoring is
not pol-time)

in practice

SHA-3 is not
a random function

The hope: a proof in the ROM
"carries over" to the real world

Why do we do this? Schemes that
are secure if the ROM are simple
and efficient

Skepticism: it is possible to construct schemes (more or less contrived) that are secure in the ROM but insecure when you replace ROM with any concrete hash function

Pedersen Commitments

Setup: Let G is a prime order group $\leftarrow \mathbb{P}$
and $g, h \in G$ generators of G
such that the discrete log
between g and h is unknown
($g^x = h \pmod{q}$)

Commit: $\mathbb{Z}_q \times \mathbb{Z}_q \rightarrow G$

$$\text{Commit}(m, r) = g^m h^r = C$$

This scheme is linearly homomorphic

$$\text{Commit}(m_0, r_0) \cdot \text{Commit}(m_1, r_1)$$

$$= g^{m_1+m_2} \cdot h^{r_1+r_2} = \text{Commit}(m_1+m_2, r_1+r_2)$$

Statistically hiding: $\swarrow g^x$

$$\text{Commit}(m, r) = g^m h^r = g^{\underline{m+xr}}$$

Given any commitment $C = g^{\underline{\alpha}}$

and any message $m \in \mathbb{Z}_q$ there

exists a unique $r \in \mathbb{Z}_q$ such

that $g^m h^r = C$ (namely $r = \frac{\alpha - m}{x}$)

Computational binding:

breaking binding $\Leftrightarrow$ finding discrete logs

Def DLP. Given $g^x \in G$ with $x \in \mathbb{Z}_q$
find x

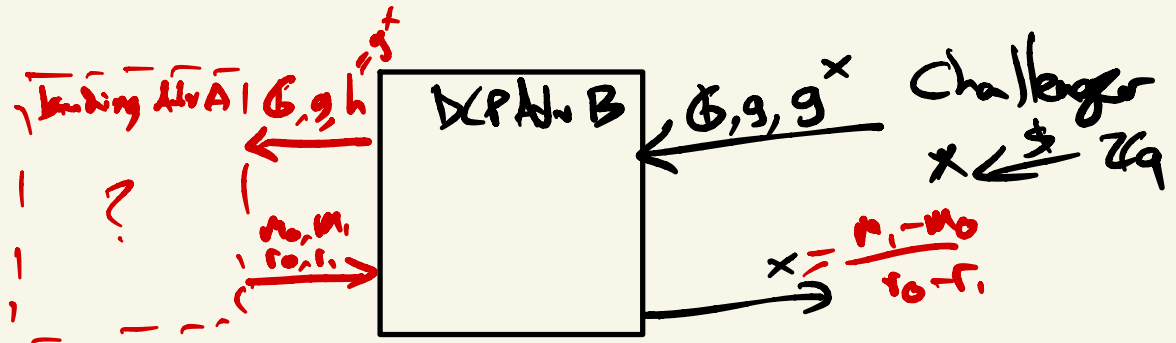
$$\text{AdvDLP}(A, G) = \Pr[A(g^x) = x : x \in \mathbb{Z}_q]$$

The DLP assumption: $\text{AdvDLP}(A, G) = \text{negl}(\lambda)$

To show: DLP hard $\Rightarrow$ Pedersen is binding

Contrapositive: assume Pedersen is not binding
 $\Rightarrow$ DLP is easy

We assume an Adv A that breaks binding, and use it to build an Adv B that breaks DLP



Life advice: if you want to break a dlog, find two different representations of the same group element

$$g^{m_0} h^{r_0} = c = g^{m_1} h^{r_1} \quad m_0 \neq m_1$$

$$g^{m_0 + x \cdot r_0} = g^{m_1 + x \cdot r_1}$$

$$m_0 + x \cdot r_0 = m_1 + x \cdot r_1 \quad \text{mod } q$$

$$x = \frac{m_1 - m_0}{r_0 - r_1} \quad \text{mod } q$$

Pseudo random Function $F: K \times X \rightarrow Y$

is a deterministic algorithm.

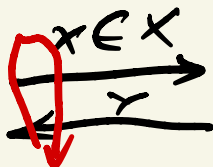
It is secure if it cannot be dist.
from a truly random function $X \rightarrow Y$
without the key.

$$\text{AdvPRF}(A, F) = \left| \Pr[A^{F(k, \cdot)} = 1 : k \xleftarrow{\$} K] - \Pr[A^{f(\cdot)} = 1 : f \xleftarrow{\$} \text{Funcs}[X, Y]] \right|$$

Game view:

PRF Adversary

guess b'



PRF challenger(b)

if $b = 0$

$k \xleftarrow{\$} K$

$f: F(k, \cdot)$

else $f \xleftarrow{\$} \text{Funcs}[X, Y]$

$y = f(x)$

$$\Delta_{b,b'} \text{PRF}[A, F] = |\Pr[b=b'] - 1/2|$$

A bad PRF

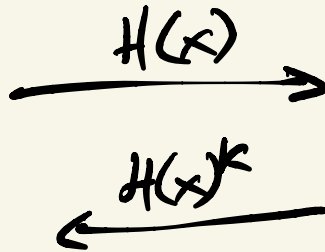
$$F(k, x) = H(x)^k$$

here $H: X \rightarrow \mathbb{G}$ of order q
 $\uparrow$ modeled as a random oracle

"oblivious PRF"

Client
x

Server
k



Def: DDH $g \in \mathbb{G}$ of order q

$$D_0 = \{ (g^x, g^y, g^{xy}) : x, y \in \mathbb{Z}_q \}$$

$$D_1 = \{ (g^x, g^y, g^z) : x, y, z \in \mathbb{Z}_q \}$$

$$\text{AdvDDH}[A, \mathbb{G}] = \left| \frac{\Pr[A(t)=1 : t \leftarrow D_0]}{\Pr[A(t)=1 : t \leftarrow D_1]} \right|$$

Thm: If DDH holds in $\mathbb{G}$

and $H: \mathbb{Z}_q \rightarrow \mathbb{G}$

is a random function then

The PRF $F(k, x) = H(x)^k$ is secure.

Proof

1. restrict to an Adv that makes a single PRF query

2. we prove

$$\text{Adv}_{\text{DDH}}(B, G) \geq \frac{\text{Adv}_{\text{PRF}}(A, F)}{Q_{\text{PRF}}}$$

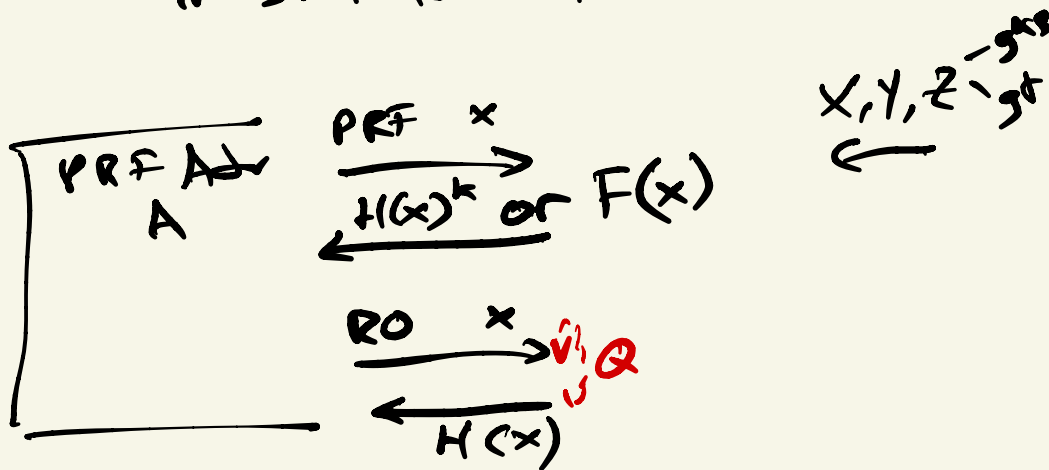
Assume a PRF Adv A



Goal: if $b=0$ (we get g^{AB}), we play the $b=0$ PRF game
if $b=1$ (we get g^A), we play the $b=1$ PRF game

Idea: We can "program" the RO

(we can choose the outputs $H(x)$ however we want, as long as it still looks like a random function)



Here's how we program the RO: distinct

- the Q queries to the RO are $x_1, \dots, x_Q$
- respond to $H(x_i) = X$, $H(x_i) \leftarrow \mathcal{B}$
- respond to "PRF on x_i " query with Z

$$Z = \begin{cases} g^{a \cdot b} = (g^a)^b = H(x_i) & \text{if } b=0 \\ g^0 = F(x_i) & \text{if } b=1 \end{cases} \quad \text{where } a = k$$