

lec 09: Non-interactive ZK.

- 1 Motivation: no NIZKs in standard model
- 2 Random Oracle Model (ROM)
- 3 ROM NIZKs via Fiat-Shamir
 - defining ZK-PoK in the ROM

Today: solving an impossible problem (NIZKs)

by "cheating" (ROM)

Some people view this as cheating, but that is misguided. Crypto as a field, is practice-oriented and model-flexible. Always be willing to change the model. (Alex's opinion)

↑ A very controversial topic in crypto, in the 2000s. Essential for most deployed crypto today. At GT, we ♥ it! Stanford too... not MIT...

logistics

- midterm: open paper notes

75 minutes, in 3wks

- final exam or project

- project proposals due 1wk after midterm
- instructions are up

① Last time: Sigma protocols ($\Sigma\Pi_s$)

$$\begin{array}{ccc} P & & V \\ & \xrightarrow{u} & \\ & \xleftarrow{c} & \\ & \xrightarrow{z} & \\ & & c \notin C \end{array}$$

- perfectly complete
- special sound $\exists \text{det, poly } \mathcal{E}(x, (u, c, z), (u, c', z')) \rightarrow \omega$
- special HVZK: $\exists \text{ PPT } \text{Sim}(x, c) \rightarrow (u, z)$
 - $\forall x, \text{Sim}(x, c)$ is accepting
 - $\forall x \in \mathcal{L}_\phi, \{ \text{Sim}(x, c) \} \approx_c \{ \text{view}_V[\langle P, V \rangle(x; \omega)] \}$
- $\Sigma\Pi_s$ have AND/OR combinators
- A $\Sigma\Pi$ for circuit via:
 - Pedersen commitments
 - Schnorr
 - AND
 - OR

[1] Motivation:

- Recall (lec04): If $\mathcal{L} \notin \text{BPP}$, then $\mathcal{L}$ does not have a NIZK "in the standard model".
- Today we learn about a non-standard model!
- It's called the Random Oracle Model

(ROM)

[2]

- We'll approach it gradually...
- Motivating Q:
 - Let H be a cryptographic hash function.
 - Is $H(m, r) \rightarrow c$ a hiding & binding commitment?
 - A: What does "cryptographic hash fn" mean?
 - Intuition: H should be deterministic, with "unstructured" outputs
 - That's $\sim$ "random" outputs
- The ROM captures this intuition.
 - We define $H: X \rightarrow Y$ as $H \stackrel{\$}{\leftarrow} \text{Fns}[X, Y]$
 - i.e. each output $H(x_i)$ is set to $y_i \stackrel{\$}{\leftarrow} Y$
 - all algs. query H (they don't have its code)
 - this makes H an "oracle"

- H 's code would be too big anyways!
- It's "code" is just an $X \rightarrow Y$ table. Huge for $X = \{0,1\}^\lambda$.
- 2^λ entries!

• Notation: for alg. A , A^H denotes A w/ oracle access to H

- Similar to E^{P^*} in knowledge soundness.
- There, E can also query P^*
- Specifically, its next-message functions

• So, is $H(m,r)$ hiding? ↙ input space $X = M \times R$

• Technically: $\text{Com}_{R_0}^H(m,r) \stackrel{\Delta}{=} H(m,r)$

• A: Yes! (For $R, C = \{0,1\}^\lambda$)

- Fix m, m'

- goal: $\{\text{Com}(m,r)\} \approx_c \{\text{Com}(m',r')\}$

- suffices to show $\{\text{Com}(m,r)\} \approx_c U(C)$

- Fix PPT A , gets c from D_b D_1 D_0

- What does A see?

- c (either $H(m,r)$ or $\perp^c$)

- Oracle responses c_i (to $H(m_i, r_i)$) $i \in [q]$

- w/LOH, assume (m_i, r_i) unique - A can cache duplicate queries internally...

- fact: if $b=0$, $c, c_1, \dots, c_q \sim U(C^{q+1})$

- fact: conditioned on $r \notin \{r_1, \dots, r_q\}$,

$c, c_1, \dots, c_q \sim U(C^{q+1})$ too!

(since $m, r \neq (m_i, r_i)$)

- so, if we can show $\Pr[r \in \{r_1, \dots, r_2\}] \leq \epsilon$,
then $|\Pr[A(D_0)=1] - \Pr[A(D_1)=1]| \leq \epsilon$ (data processing inequality)

- w LOG, assume (m_i, r_i) are unique
- (A can cache repeat queries internally)

- So, $\Pr[r \in \{r_1, \dots, r_q\}] \dots$
 $= \Pr[r=r_1] + \Pr[r \neq r_1 \wedge r=r_2] + \dots + \Pr[r \neq \{r_1, \dots, r_{q-1}\} \wedge r=r_q]$
 $= \sum_i \Pr[r \neq \{r_1, \dots, r_{i-1}\} \wedge r=r_i]$

this implies
r is 11 of A's
view

this
comes
from A

so, $\Pr[\dots] = 1/|R|$

$$= q/|R| \leq \text{negl}$$

□

- So, is $H(m, r)$ binding?

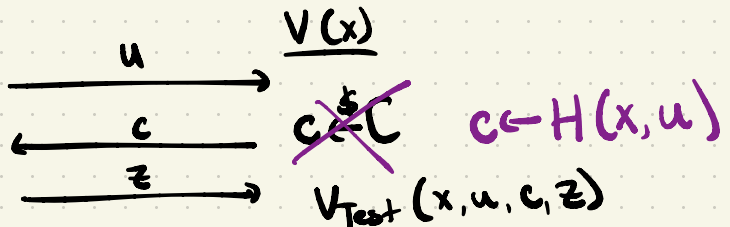
- Fix binding adversary $A^H(\cdot) \rightarrow (m, m', r, r')$
- Assume w LOG, A queries $H(m, r)$ and $H(m', r')$
 - (If not, defⁿ A' as A, but it makes ≤ 2 more queries)
 - Some m_i, r_i and m_j, r_j are the first collision...
 - this is actually a HW problem, so we'll stop here...

[3] Fiat Shamir

Let H be a random oracle.

Let $\Pi = (P, V)$ be a $\Sigma\Pi$ for ϕ .

idea:
replace
 V with H !



Define $\Pi_{\text{FS}} = (P_{\text{FS}}^H, V_{\text{FS}}^H)$ as follows:

$$P_{\text{FS}}^H(x, w)$$

$$V_{\text{FS}}^H(x)$$

$$u \leftarrow P(x, w)$$

$$c \leftarrow H(x, u)$$

$$z \leftarrow P(\dots, c)$$

$$(u, c, z)$$

$$c \stackrel{?}{=} H(x, u)$$

$$V_{\text{Test}}(x, u, c, z)$$

Clearly Π_{FS} is not a $\Sigma\Pi$

• wrong shape!

• We'll show Π_{FS} is a ZK-PoK.

• We'll have to define all properties in the ROM.

• Claim: Π_{FS} is perfectly complete

$$\forall x \in \mathcal{L}_\phi, \Pr_{P_{\text{FS}}, V_{\text{FS}}, H} [\langle P_{\text{FS}}^H, V_{\text{FS}}^H \rangle(x, w) = 1] = 1$$

- this is true, b/c Π accepted for all $c \in C$, which includes $H(x, u)$, for $H \in \text{Fns}[\dots]$

• Claim: Π_{FS} is ZK

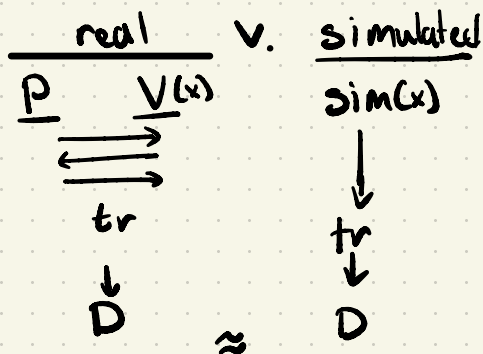
• notice: this is not HVZK!

• so, FS eliminates interaction and upgrades ZK.

• But, what does ZK mean in the ROM?

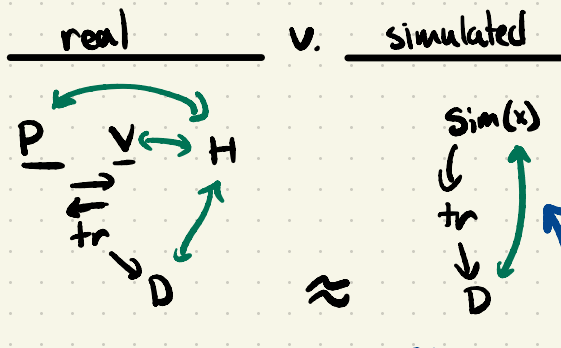
• First let's understand ZK as distinguishing a real v. simulated world in the standard model

Standard Model



"distinguisher"

ROM



Sim responds to $\cdot$

D 's queries, too responses

A: Sim produces tr and chooses RO

$\rightarrow$ Sim interacts w/ distinguisher

this is called programming the RO

So, assume Π has $Sim(x, c) \rightarrow u, z$

Let's define Sim_{FS} :

$c \in C$

$u, z \leftarrow Sim(x, c)$

set $H(x, u) \leftarrow c$ $\leftarrow$ if H is ever queried @ (x, u) , give c .

output (u, c, z)

respond to any new RO query (x_i, u_i) with $c_i \stackrel{r}{\leftarrow} \mathcal{C}$.

- caches responses to repeat queries

· Claim: $\forall D, x \in \mathcal{X} \setminus \emptyset$ note: since Π_{FS} is N.I., π is just $\pi - P_{FS}$'s message.

$$\Pr[D^H(\pi) = 1 : \pi \leftarrow P_{FS}^H(x, w)] \approx \Pr[D^{\text{Sim}_{FS}(x)}(\text{Sim}_{FS}(x)) = 1]$$

- we know $\pi \approx_c \text{Sim}_{FS}(x)$ by SHVZK for Π

- what about RO queries?

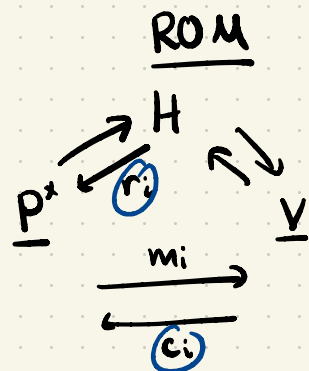
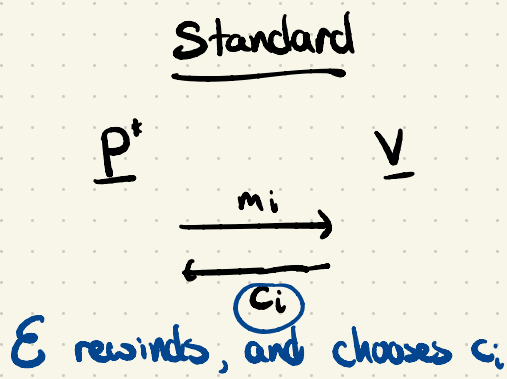
- in the real world: all UID, except:
 $c = H(x, w)$

- same as in the simulate world!

- note: Sim_{FS} sampled c before u , but D doesn't know that: it runs after

- this proof breaks if Sim_{FS} 's RO-responses are not uniform.

- Claim: Π_{FS} is knowledge-sound (KS)
- we need a ROM extractor $\mathcal{E}$



- ROM $\mathcal{E}$ can choose H 's responses.

(we need to explore all of P^* 's behavior)

- So, proving KS for Π_{FS} .

- Let Π have extractor $\mathcal{E}(x, (u, c, z), (u, c', z')) \Rightarrow \omega$

- Define $\mathcal{E}_{FS}$:

- run P_{FS}^H , responding to H -queries normally (w/ random, cached samples)
- get output (u, c, z) .
- rewind to query $c = H(x, u)$
 - if no such query, abort (1)

- set $H(x, u) \stackrel{P}{\leftarrow} C$ and continue to get (u', c', z')
- abort if $u \neq u'$ or $c = c'$ (2)
- output $E(x, (u, c, z), (u', c', z'))$

We must show: $\forall x \forall p \exists P_{FS}^*$

$$\Pr[P(x, \omega): \omega \leftarrow E_{P_{FS}^*}(x)] \geq \underbrace{P[\langle P_{FS}^*, V_{FS} \rangle(x) = 1]}_{\epsilon} - \text{negl.}$$

- see [Pointcheval + Stern '00] for the original sketch:

- assume P^* outputting ^{accepting} (u, c, z) always queries $c = H(x, u)$
 - this is WLOG: just add 1 query
- so (1) is impossible, and every output (u, c, z) defines a query index i s.t. $c = H(x, u_i)$.
- first run gives accepting (u, c, z) , $c = H(x, u_i)$ w/ $p \geq \epsilon$.
- 2nd run gives accepting (u', c', z') ^{$c' = H(x, u_j)$} w/ $p \geq \epsilon$.
 $w/ p \geq \epsilon/q, i=j \Rightarrow \underbrace{u = u_i = u_j = u'}_{\text{arrow}} \quad c = c'$
 $w/ p \geq 1/q, \quad c = c'$

so, E gets $(u, c, z), (u', c', z')$ w/ $u = u'$
 $c \neq c'$
 w/ prob. $\geq \epsilon(\epsilon/q - 1/q)$

- getting this takes some care, b/c the runs are not $\perp$. But dependence doesn't help P^* - it synchronizes the runs more.

in more detail:

let ϵ_j be the prob. a run succeeds with index j .
(so, $\sum_j \epsilon_j = \epsilon$).

note that after j , two runs are $\perp$.

let p be the share randomness up to j ,

let $\epsilon_{j,p}$ be the prob. of success w/ j for fixed p .

$$(\text{so } \epsilon_j = \mathbb{E}_p [\epsilon_{j,p}])$$

$$P_r [\text{both succeed at same } j]$$

$$= \sum_j \mathbb{E}_p [\epsilon_{j,p}^2]$$

$$\geq \sum_j (\mathbb{E}_p \epsilon_{j,p})^2$$

$$= \sum_j \epsilon_j^2$$

$$\geq (\sum_j \epsilon_j)^2 / 2$$

$$= \epsilon^2 / 2$$

(Jensen's: mean of squares)

(def. $\epsilon_{j,p}$)

(Cauchy Schwartz $x_i = \epsilon_{j,p} / \sqrt{2}$)

(def. ϵ_j)

$$y_i = 1$$

$$\langle x, x \rangle \cdot \langle y, y \rangle \geq \langle x, y \rangle^2$$

and, we fail if run 1 succeeds & $c' \neq c$.

$$\boxed{\epsilon^2 / 2 - \epsilon / |C|} \text{ knowledge error.}$$

• Q: How is a model different than an assumption?

• Models are higher-level.

• Assumptions restrict PPT algorithms within a model.

• Ex: $\forall$ PPT A , $\underbrace{\text{Adv}_{\text{dlog}}[A, G]}_{\text{restriction}} \leq \text{negl}$

• Models define how algorithms interact

• Ex: A w/ internal H fn v. A^H

• sec. defⁿ $\Rightarrow$ change...

- Standard model:
 - Algs have private memory
 - Algs interact w/ each other

• ROM

- All of the above
- Algs interact with $H \in \mathcal{F}_n[\dots]$ too!

• Other models too

- Generic Group Model: G-op oracle
- Whitebox cryptography: Memory is only ~ private
- Many more...

- Crucial fact: In practice, ROM is a heuristic
 - in our proofs, we analyze

$$A^H$$

$$H \xleftarrow{\$}$$

- in deployments, we set called instantiation

$$H \triangleq \text{SHA3} \quad \leftarrow \text{(or another)}$$

- does $\text{secure}(A^H) \Rightarrow \text{secure}(A^{\text{SHA3}})$?

- we can't prove it.
- but instantiation has never failed in practice
 - no secure computer system with a ROM proof has ever been broken.
- still, people worry
- later in the course, some ROM problems..