

lec 07: Proofs of Knowledge

0 Recall: ZKPs for NP

1 Proofs of Knowledge.

2 Schnorr's Protocol

This is our 4th lecture on ZK

So far

1. IP

2. ZK (for GI)

3. ZK for NP

4. ZK Proofs of Knowledge
(Today!)

⊠ Notation + Recap

Let $\phi(x, w)$ be a relation.

The language $\mathcal{L}_\phi$ is the set $\{x \mid \exists w, \phi(x, w) = 1\}$

• Recap: security properties for an IP (P, V)

- Completeness : $\forall x \in \mathcal{L}_\phi, \Pr[(P, V)(x; w) = 0] \leq \text{negl}$.
- Soundness : $\forall x \notin \mathcal{L}_\phi, \forall P^* \Pr[(P^*, V)(x) = 1] \leq \text{negl}$
- HVZK : $\exists \text{Sim}, \forall x \in \mathcal{L}_\phi \{ \text{Sim}(x) \} \approx_c \{ \text{view}_V[(P, V)(x, w)] \}$

• Observation:

• Soundness is existential. It means that if P^* succeeds, then a w exists s.t. $\phi(x, w) = 1$, (with high probability).

• What if we also want to ensure that P^* knows such a w ?

• We want a PoK (proof of knowledge).

• Does existence imply knowledge?

• No! many examples:

- Primality testing $\in P$, but factoring is hard
- for all $h \in \mathbb{G}, \exists x, g^x = h$, but cd-log is hard
- Search problems can be harder than decision...

⊠ Defining PoK

• How can we define PoK?

- We want to ensure P^* knows w .
 - but, we still want to hide w from V : ZK
- Attempt 1: P^* must send w .
 - e.g. (P, V) is a Pok if P 's 1st message is w .

$$\frac{P(x, w)}{\quad} \xrightarrow{w} \frac{V(x)}{\quad} \quad \phi(x, w) \stackrel{?}{=} 1$$

- issue: incompatible with ZK!
 - if d_p is NP-hard and $BPP \neq NP$, this isn't ZK.

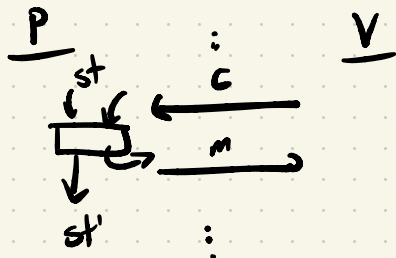
- Attempt 2: Compute w from m_i

$$\frac{P(x, w)}{\quad} \xrightleftharpoons[c_i]{m_i} \mathcal{G} \quad \frac{V(x)}{\quad}$$

- there must be a PPT $A(m_i) \rightarrow w$
- same issue: ZK implies m_i are simulatable
 - so w is computable,
 - so $L \in BPP \dots$

- Attempt 3: Compute w from P^* 's state st .

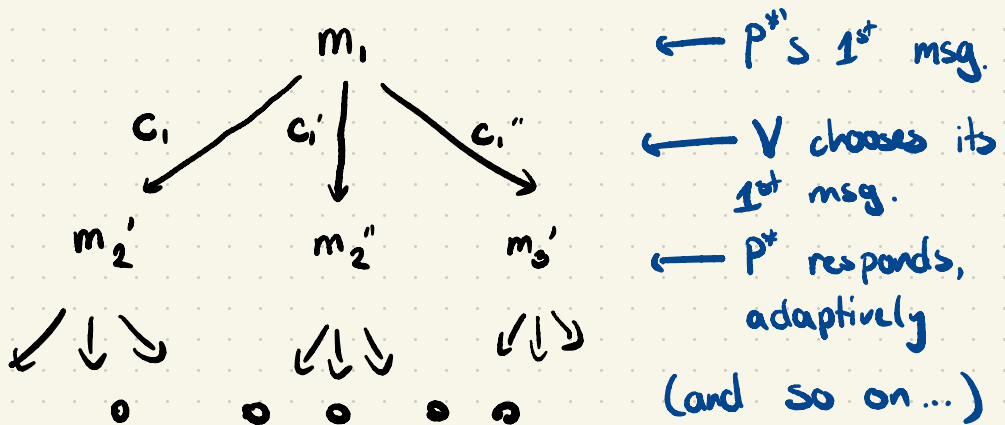
- model P^* as an initial state st_0
 - and a state transformer: $P^*(st_i, c_i) \rightarrow st_{i+1}, m_i$



- require PPT $A(st_0) \rightarrow \omega$
- this is compatible with $\mathbb{ZK}$.
 - Simulation is for m_i , not st.
- issue: hard to formalize
 - P^* is an arbitrary algorithm!!
 - What form does st have?!
- issue: P^* could be unbounded. How can PPT A read its state? $|st|$ could be $\Omega(2^n)$.
- it would be nice to base of def. on the m_i
 - but we just saw that the m_i aren't enough.
 - at least, not from 1 transcript.

• Observation :

- P^* defines a whole tree of transcripts based on what the c_i are.



- we'll compute ω from this tree.
 - compatible w/ $\mathbb{ZK}$: Sim produces just 1 path.
 - the tree can be super-poly, so we'll

represent it carefully.

• Observation:

• An interactive alg. is defined by next-message fns

$\underbrace{P^*}_{r \in \{0,1\}^*}$ private randomness

$\xleftarrow{c_1}$
 $\xrightarrow{m_1}$
 $\vdots$
 $\xleftarrow{c_n}$
 $\xrightarrow{m_n}$

$\underline{V}$

$P_1^*(r, c_1) \rightarrow m_1$

$P_2^*(r, c_1, c_2) \rightarrow m_2$

$\vdots$

$P_n^*(r, c_1, \dots, c_n) \rightarrow m_n$

• Notation: for a PPT $\mathcal{E}$,

$\mathcal{E}^{P^*}$

denotes that $\mathcal{E}$ has oracle access to $\{P_i^*(r, \cdot)\}_i$ for common $r \in \{0,1\}^*$.

• $\mathcal{E}$ can call the P_i^* to explore P^* 's tree.

• called "rewinding"

• backing up the tree, without changing r .

Defⁿ (P, V) is a PoK for ϕ with knowledge error κ if $\exists$ PPT extractor $\mathcal{E}$ s.t. $\forall x, P^*$

$$\Pr[\phi(x, \omega) = 1 : \omega \leftarrow \mathcal{E}^{P^*}(x)] \geq \Pr[(P^*, V)(x) = 1] - \kappa$$

(there are many variations of this definition,

but in all, E 's success probability is lower-bounded by some function of p^* 's.)

- We say that (P, V) is a PoK for ϕ if its knowledge error is negligible.
 - equivalently, we call (P, V) "knowledge sound"
- Let's do a warmup.
 - Thm. If (P, V) is a PoK for ϕ , then it is sound for $\mathcal{L}_\phi$.

• Pf.

• Fix $x \notin \mathcal{L}_\phi$

• We have:

$$\begin{aligned} \Pr[\phi(x, \omega) = 1 : \omega \leftarrow E^{P^*}(x)] &\geq \Pr[(P^*, V)(x) = 1] - K \\ &\quad \text{(no } \omega \text{ exists)} \qquad \qquad \qquad 0 \geq \Pr[(P^*, V)(x) = 1] - K \\ &\quad \text{(} K \leq \text{negl)} \qquad \qquad \qquad \text{negl} \geq \Pr[(P^*, V)(x) = 1] \end{aligned}$$

• But, the converse is not true. ▣

• Example:

$$\phi_{d\text{-log}}(x \in G, \omega \in \mathbb{Z}_p) \stackrel{\Delta}{=} g^{\omega} = x$$

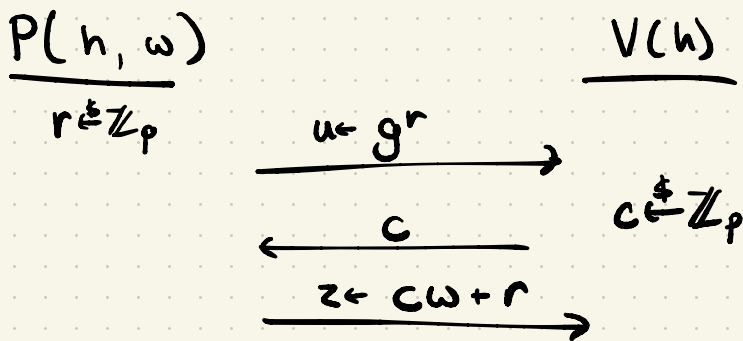
$$\mathcal{L}_{\phi_{d\text{-log}}} = \text{all of } G.$$

- so, the empty IP is trivially sound.
- but, it is not a PoK if $d\text{-log}$ is hard...

② Schnorr's Protocol.

- Schnorr is a PoK for a discrete logarithm.
 - Let G be a group of order p , generated by g .
 - Fix $\phi(h, \omega)$ to hold when
$$h = g^\omega$$
- As we just noted, $\Delta_\phi = G$ (trivial).
- So Schnorr is exciting because it is a PoK for ϕ .

• The protocol:



V can't ask for $\log h$, $g^z \stackrel{?}{=} \underline{uh^c}$
so it asks for $\log(uh^c)$

$u = g^r$ protects P 's privacy, c protects soundness

- This protocol is hugely impactful, gives the best signatures:
 - '91: Schnorr patents them
 - DSA: Schnorr, modified to dodge patent [NIST '94]
 - Recommended by most browsers + certificate authorities
 - EdDSA: Schnorr [RFC 8032 '11] (patent expired in '10)

Claim: Schnorr is complete, HVZK, and a PoK.

- Complete

- there is 0 completeness error: fix $h = g^w$:

$$\Pr[\langle P, v \rangle(h; w) = 1]$$

$$= \Pr[g^z = uh^c]$$

(def. P, v)

$$= \Pr[g^{r+cw} = g^r \cdot (g^w)^c]$$

(def. z, u, h)

$$= 1$$

- HVZK

- fix $h = g^w$

- define $\text{Sim}(h) \rightarrow u, c, z$:

- $c \xleftarrow{\$} \mathbb{Z}_p$

- $z \xleftarrow{\$} \mathbb{Z}_p$

- $u \leftarrow g^z / h^c$

observe:

$g^z = uh^c$ by construction.

- Claim:

$$\{\text{Sim}(h)\} = \{\text{view}_v[\langle P, v \rangle(h; w)]\}$$

$$\{u_s, c_s, z_s\} = \{u_\pi, c_\pi, z_\pi\}$$

- $c_s, c_\pi \sim U(\mathbb{Z}_p)$

- $z_s \xleftarrow{\$} \mathbb{Z}_p$ & $z_\pi \sim U(\mathbb{Z}_p)$, since $z_\pi = r_\pi + c_\pi w$

- and that determines u_s, u_π

- Side Q: Is Schnorr ZK?

- Rejection sampling is too slow, since $c \xleftarrow{\$} \mathbb{Z}_p$,
and $p = -\Omega(2^\lambda)$.

- GK'96: Schnorr isn't black-box ZK (in std. model)
(no 3-move Π can be...)

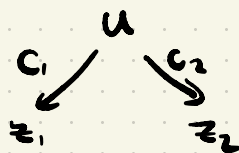
- How to get z_K : V commits to c , before u .
- Now, PoK.
 - Fix P^* .
 - Fix $h \in G$
 - Let $\epsilon = \Pr[\langle P^*, V \rangle(h) = 1]$
 - We must show

$$\Pr[g^w = h : w \leftarrow \mathcal{E}^{P^*}(h)] \geq \epsilon - \text{negl}.$$

for some PPT $\mathcal{E}$.

- For simplicity, assume $\epsilon = 1$ (revisit later)

- Define $\mathcal{E}^{P^*}(h) \rightarrow w$:



- get u from P^*
- send $c_1 \xleftarrow{\$} \mathbb{Z}_p$, get z_1
- send $c_2 \xleftarrow{\$} \mathbb{Z}_p$, get z_2 (same u)
- since $\epsilon = 1$, (u, c_1, z_1) and (u, c_2, z_2) are both accepting transcripts.

$$g^{z_1} = u h^{c_1}$$

$$g^{z_2} = u h^{c_2}$$

$$g^{\frac{z_1}{h^{c_1}}} = u$$

$$= g^{\frac{z_2}{h^{c_2}}} \left| \begin{array}{l} \text{One } G \text{ element } (u), \text{ with} \\ \text{two representations.} \\ \text{attack d-log!} \end{array} \right.$$

$$g^{z_1 - z_2} = h^{c_1 - c_2}$$

$$g^{\frac{z_1 - z_2}{c_1 - c_2}} = h$$

- output $w \leftarrow \frac{z_1 - z_2}{c_1 - c_2}$

- Claim: $\Pr[\mathcal{E} \text{ wins}] \geq 1 - 1/p$

$$\Pr[E \text{ wins}] \geq \Pr[E \text{ wins} \wedge c_1 \neq c_2]$$

$$= \Pr[c_1 \neq c_2] \quad (\text{if } c_1 \neq c_2,$$

$$= 1 - \Pr[c_1 = c_2] \quad g^{\frac{z_1 - z_2}{c_1 - c_2}} \text{ is defined and } = h)$$

$$= 1 - 1/p$$

$$= \epsilon - 1/p$$

$$\geq \epsilon - \text{negl}$$



• Q: Pok for $\epsilon < 1$?

• Same E

• See Boneh & Shoup Lemma 19.2

$$\Pr[E \text{ wins}] \geq \epsilon^2 - \epsilon/p$$

both transcripts $\uparrow$ good $\quad \uparrow$ not (1st good & $c_1 = c_2$)

└──────────┘
union bound

• hard part: r (in P^*) is shared

• redemption: r, c_1, c_2 are $\perp$.

• Recap:

• knowledge soundness / Pok

• > (existential) soundness

• Schnorr: a Pok for D-log

• Next time:

• A ZK-Pok for circuit SAT