Zero-Knowledge Proofs of Knowledge

Material taken from here, here, here
Redefining classical proofs

- Proof redefined as a game between a prover and a verifier
- Game can be **interactive**, where the verifier asks questions and the prover answers
- Further generalization to a **probabilistic** proof system
- **Zero-knowledge**: “Prove that I could prove it if I felt like it”
Proofs of knowledge

• In a regular ZK proof, the prover attempts to convince the verifier that some fact is true
  • “X is true”

• In a proof of knowledge, the prover attempts convince the verifier that it knows some secret information
  • “I know why X is true”

• Definition: An interactive proof system \((P, V)\) is a proof of knowledge for an NP relation \(R\) if there exists an efficient extractor \(E\) such that for any \(x\) and any prover \(P'\):
  \[
  Pr[w \leftarrow E(x) : (x, w) \in R] \geq Pr[\langle P', V \rangle(x) = 1] - \epsilon
  \]

  • \(w\) is the witness, \(\epsilon\) is the knowledge error; soundness error of at most \(\epsilon\)
Schnorr Protocol

Prover wants to prove that it knows the discrete logarithm $x$ of some group element $h = g^x \in \mathbb{G}$

$$r \leftarrow \mathbb{Z}_q, u = g^r$$

$$x, h = g^x$$

$$z \leftarrow r + cx$$

$$h = g^x$$

$$c \leftarrow \mathbb{Z}_q$$

$$g^z = u \cdot h^c$$
Schnorr protocol

- **Completeness:** if $z = r + cx$, then $g^z = g^{r+cx} = g^r \cdot (g^x)^c = u \cdot h^c$

- **Proof of knowledge:** Let $P'$ be a possibly malicious prover that convinces the honest verifier with probability $\delta = 1$. Construct the extractor $E$ as follows
  - Run $P'$ to obtain an initial message $u$
  - Send random challenge $c_1$ to $P'$ to get response $z_1$
  - Rewind the prover to its state after the first message
  - Send it another random challenge $c_2$ to get response $z_2$
  - Compute $x = \frac{z_1 - z_2}{c_1 - c_2}$
Schnorr protocol

- Proof of knowledge (cont’d):
  - Since the prover succeeds with probability 1, we know that
    \[ g^{z_1} = u \cdot h^{c_1}, \text{ and } g^{z_2} = u \cdot h^{c_2}. \]
  - Therefore, \( g^{z_1 - z_2} = h^{c_1 - c_2}, \) \( h = g^{\frac{z_1 - z_2}{c_1 - c_2}}, \) \( x = \frac{z_1 - z_2}{c_1 - c_2} \)
  - Extraction fails if \( c_1 = c_2, \) which happens with probability \( \frac{1}{q}, \) which is also equal to the knowledge error.
Schnorr protocol

• **Zero-knowledge:** let’s try to construct a simulator.
  
  • Simulator sends $u = g^r$, verifier responds with challenge $c$

  • Rewind and sample $s \in \mathbb{Z}_q$, and compute $u = g^s/h^c$

  • Restart the verifier and get challenge

• Problem: a malicious verifier could respond with a different challenge $c$ that depends on the $u$ that it receives!
Schnorr protocol

- Honest verifier zero-knowledge:
  - Simulator sends $u = g^r$, verifier responds with challenge $c$
  - Rewind and sample $s \in \mathbb{Z}_q$, and compute $u = g^s/h^c$
  - Restart the verifier and get challenge $c$ (verifier is honest, so it uses its random tape instead of adaptively choosing the challenge)
  - Simulator successfully answers with $s$, verifier checks that $g^s = g^s/h^c \cdot h^c$
Sigma protocols

- More general view of Schnorr’s protocol
- Protocols of the form
  - Prover sends a first message $u$ called a commitment
  - Verifier sends a uniformly random challenge $c$ from a finite challenge space
  - Prover generates and sends a response $z$
- HVZK can be turned into full ZK
- Fiat-Shamir heuristic to transform into NIZK in the random oracle model
Today’s reading: Zerocash
Next time

• Moral character of cryptographic work
  • No paper review, just one discussion question
• Project presentations
  • Will send out peer grading forms