CS6222 Cryptography

Topic: Zero-Knowledge Proofs Date: Nov 21, 2024
Lecturer: Wei-Kai Lin (TA: Arup Sarker) Scriber: Alex Fetea and Nate Kimball

1 Introduction to Zero-Knowledge Proofs

Zero-Knowledge Proofs (ZKPs) allow one party (prover) to convince another party (verifier) that a certain statement is true without revealing anything about the underlying solution. This concept helps ensure both privacy and security and has a variety of applications such as secure authentication.

We focus on the Graph 3-Coloring problem as a concrete example.

2 Interactive Turing Machine for $L =$ Graph 3-Coloring

The Interactive Turing Machine (ITM) is used to define interactions between the prover (P) and the verifier (V) for a language L. L represents the Graph 3-Coloring problem in this case. This problem involves determining wheher graph nodes can be colored with three colors so that no two adjacent vertices share the same color. ITM (P, V) is an interactive (zero-knowledge) proof for language L if it satisfies the completeness and soundness (and zero-knowledge) properties:

2.1 Completeness

Completeness means that all true statements will be proven true with probability 1.

Formally: $\forall x \in L,\ \exists w \in \{0,1\}^*,\ \forall$ prior information z,

$$\Pr[\mathrm{out}_V(P(x,w) \leftrightarrow V(x,z)) = \mathrm{Accept}] = 1$$

2.2 Soundness

Soundness means that no adversarial prover can make the verifier accept a false statement with significant probability. For example, if a graph is not 3-colorable, the verifier should reject the proof with overwhelming probability.

Formally: $\forall x \notin L,\ \exists$ negligible $\epsilon(\cdot),\ \forall$ Adversary $P^*,\ \forall z$,

$$\Pr[\mathrm{out}_V(P^*(x) \leftrightarrow V(x,z)) = \mathrm{Accept}] \le \epsilon(n),$$

where $x \notin L$ means the graph is not 3-colorable.

2.3 Zero-Knowledge

Zero-knowledge guarantees that the verifier learns no extra information about the witness except for the validity of the statement being proven. This is done by showing the verifier's view can be simulated without any private information from the prover.

Formally: $\forall x \in L,\ w \in R_L(x),\ \forall z$,

$$\{\mathrm{view}_V[P(x,w) \leftrightarrow V(x,z)]\}_n \approx_c \{S(x,z)\}_n,$$

where $S(x, z)$ is a simulator that generates a view indistinguishable from the real one. This ensures privacy, since the verifier does not have any information about the witness w.

3 Commitment Schemes

A commitment scheme allows a party to commit to a value while keeping it hidden and ensuring it cannot be changed later.

3.1 Definition

$$\text{Com} : \{0,1\} \times \{0,1\}^n \to \{0,1\}^*,$$

satisfying the following properties:

- **Hiding:** The commitment hides the value being committed:

$$\forall v_0, v_1 \in \{0,1\}, \quad \{\text{Com}(v_0, r_0)\}_n \approx \{\text{Com}(v_1, r_1)\}_n.$$

- **Binding:** The commitment is binding, meaning the value cannot be changed or have multiple values:

$$\forall v_0 \neq v_1, \quad \text{Com}(v_0, r_0) \neq \text{Com}(v_1, r_1).$$

4 Graph 3-Coloring ZKP Construction

This examples demonstrates how ZKPs can be used for proving that a graph is 3-colorable without revealing the actual coloring.

4.1 Steps

1. The prover starts with a graph $G = (V, E)$ and a valid 3-coloring $w = (b_1, \ldots, b_n)$, where each vertex $i \in V$ has a color $b_i \in \{R, G, B\}$.
2. The prover selects a random permutation $\pi \leftarrow \text{perm}(3)$ of the colors to hide the true coloring.
3. The prover commits to the permuted colors for each vertex:

$$C_i = \text{Com}(\pi(b_i), r_i), \quad \forall i \in [n],$$

   where r_i is a random nonce.
4. The verifier selects a random edge $(u, v) \in E$ and asks the prover for the commitments for vertices u and v.
5. The prover reveals:

$$\pi(b_u), r_u, \quad \pi(b_v), r_v.$$

6. The verifier ensures:
   - The commitments match the values:

$$C_u = \text{Com}(\pi(b_u), r_u), \quad C_v = \text{Com}(\pi(b_v), r_v).$$

- The revealed colors are different:

$$\pi(b_u) \neq \pi(b_v).$$

If either fails, the verifier rejects. Otherwise, the protocol continues.

4.2 Proof of Completeness and Soundness

The proof of completeness is left to the reader.

We will give a sketch of the soundness proof:

$\exists (u^*, v^*) \in E$ s.t. $c_{u^*} = c_{v^*}$ because the statement is false
If the verifier selects $(u, v) = (u^*, v^*)$ then V rejects by binding of commitment
$\forall x \notin L,\ \exists$ negligible $\epsilon(\cdot),\ \forall$ Adversary $P^*,\ \forall z,$
$\Pr[\text{out}_v[P^*(x) \leftrightarrow V(x, z)] = \text{Accept}] \leq 1 - \frac{1}{|E|}$

If you repeat the protocol $n \cdot |E|$ times, then
$\forall x \notin L,\ \exists$ negligible $\epsilon(\cdot),\ \forall$ Adversary $P^*,\ \forall z,$
$\Pr[\text{out}_v[P^*(x) \leftrightarrow V(x, z)] = \text{Accept}] \leq (1 - \frac{1}{|E|})^{n \cdot |E|} \leq e^{-n}$

4.3 Proof of Zero-Knowledge (ZK)

We now show the protocol is zero-knowledge by constructing a simulator S that creates a view indistinguishable from the verifier's view.

4.3.1 Simulator Construction

The simulator S behaves as follows:

0. $(u, v) \leftarrow E$ (Verifier selects an edge).
1. $C_i \leftarrow \text{Com}(0, r_i), \quad \forall i \in \mathbb{N}$ (Random commitments).
2. $C_u = \text{Com}(d_0, r_u),\ C_v = \text{Com}(d_1, r_v), \quad d_0 \neq d_1$ (Simulator generates random colors).
3. $(u^*, v^*) \leftarrow V(x, z; \text{rand})$ (Verifier issues a random challenge).
4. Reveal d_0, r_u^* and d_1, r_v^* (Simulator opens the commitments for the selected vertices).

The simulator's output is indistinguishable because the commitments are computationally hiding, so the verifier cannot retrieve the original colors. As a result, the verifier cannot distinguish between the simulated and an actual interaction.

4.4 Malicious Verifier

The protocol is secure even if the verifier is malicious. A malicious verifier may not sample edges uniformly, and may strategically try to gain knowledge about the witness. This can be proved by constructing a simulator S that replicates the view of the malicious verifier without access to the prover's private witness.

Formally: $\exists$ PPT simulator S, $\forall$ NUPPT verifier V^*,

$$\{\text{view}_{V^*}[P \leftrightarrow V^*]\}_n \approx_c \{S\}_n.$$

This ensures that even a malicious verifier learns nothing about the prover's private information.

5 NP-Completeness and ZKPs

Zero-Knowledge Proofs can be constructed for any language $L \in NP$ by reducing it to an NP-complete problem, such as Graph 3-Coloring. There also exists a reduction for witnesses for every $L \in NP$ to witnesses for Graph 3-coloring.

Formally:

$\forall L \in NP, \exists$ poly-time algo R s.t. $(x', w') = R(x, w), \quad (x, w) \in L \Leftrightarrow (x', w') \in$ Graph 3-Coloring.

This means that for every $L \in NP$, there exists a ZKP protocol for L.

$$\forall L \in NP, \quad \exists \text{ a Zero-Knowledge Proof for } L.$$

This result demonstrates the universality of ZKPs and their applicability to a wide range of computational problems.

6 Application of ZKP

- **Setup:**
  - Generate a public-private key pair: (pk, sk).
  - The server stores pk, the public key, which is used for verifying signatures.
- **Protocol:**
  - The client signs a message m with their private key: $\text{Sign}_{sk}(m)$.
  - The signature includes a timestamp.
  - The client sends m and $\text{Sign}_{sk}(m)$ to the server.
  - The server verifies the signature using the public key pk: $\text{Verify}_{pk}(m, \text{Sign}_{sk}(m))$.
- **Problem:** While this protocol ensures authenticity, the server has the signed message $(m, \text{Sign}_{sk}(m))$, which it can show to anyone else to prove that the message originated from the signer, which breaks privacy.
- **ZKP Solution:**
  - Instead of sending the signature $\text{Sign}_{sk}(m)$, the client proves knowledge of the private key sk (the witness) through a ZKP.
  - This ensures the server knows the message's authenticity, but the server cannot show any evidence to convince a third party.