1. Zero Knowledge Proofs

A zero knowledge (ZK) proof system is a way of convincing someone of a fact without giving them any additional knowledge. But what does ‘not giving them any additional knowledge’ mean?

Let us consider an example of a ZK proof.

Suppose that a prover \( P \) wants to prove to a verifier \( V \) that two graphs \( G_1 \) and \( G_2 \) are isomorphic.

\[ P \rightarrow V \text{ \{two graphs are isomorphic\}} \]


2. Why does this work?

- If \( P \) knows an isomorphism between \( G_1 \) and \( G_2 \), then \( P \) can prove on request that either \( (G_1, G_2) \) and \( (H_1, H_2) \) are isomorphic (if not, he has a 50 percent chance of failure).
- \( P \) repeats this, say 100 times. If it gets it right every time, \( V \) is quite convinced that \( G_1 \) and \( G_2 \) are isomorphic (or that \( P \) is incredibly lucky).
- Moreover, \( P \) does not learn anything, because he could have generated the conversation (including \( P \)'s responses) on his own, using a simulator that selects \( b \) and then computes a random isomorphic copy of \( G_2 \).

3. Intuitive Definition

A pair of protocols \((P, V)\) for a prover \( P \) and a verifier \( V \) is a perfect zero knowledge proof system for \( L \) if it is

- Sound: if \( x \notin L \), then \( P(x) \text{ accepts} \) with probability \( 1/2 \).
- Complete: if \( x \in L \), then \( P(x) \text{ accepts} \) with probability \( 1/2 \).
- Simulatable: no matter what protocol \( V \)'s verifier uses, there is a probabilistic polynomial time "simulator" \( S_L \), that he could use to simulate possible conversations with the prover.

- Formally, for every \( x \in L \), \((P, V, S_L)\) is the set of possible runs of the protocol \((P, V)\) on input \( x \) and \( S_L(x) \) are independently distributed.

- So there is no verifier the prover can do (as protocol he can follow) to learn anything he shouldn’t.

There is an analogous definition of computational ZK.

- This requires only \((P, V)\)'s verifier input \( x \) and \( S_L(x) \) be indistinguishable by a polynomial-time verifier.

4. What is “Knowledge”?

CRYPTOGRAPHY

- Defined with respect to computational ability
- Bob gains knowledge after interacting with Alice if, after the interaction, Bob can easily compute something that was hard for him earlier

EPISTEMIC LOGIC

- Defined with respect to what the agent considers possible
- Bob gains knowledge of fact \( \varphi \) after interacting with Alice if, after the interaction, \( \varphi \) is true in every world Bob considers possible (whereas it was false in some worlds he considered possible before the interaction)

How are these notions related?

5. Previous Work

- Halpern, Moses and Tuttle [HMT 1988] proposed a logical definition of "generating a \( \psi \) satisfying \( R(x, \psi) \) for a relation \( R \)."

They showed that, if \( R \) is variable polynomial time and the verifier can generate a \( \psi \) satisfying \( R(x, \psi) \) at the end of a ZK proof, he can do so at the start.

- They called this property provability security.
- They left open the question of finding an epistemic statement that is sufficient for ZK.

- We provide such a statement.

6. The Runs and Systems Framework

- [Tinag, Halpern, Moses and Vardi, 1995]

- Each agent starts in some initial local state, its local state then changes over time.
- A global state is a tuple of local states.
- A run is an infinite sequence of global states—a possible execution of a protocol. Given a run \( r \) and a time \( t \), we refer to \( r(t) \) as a point.
- A system is a set of runs.
- Often the set of all possible runs of a protocol.
- We start with a collection of primitive facts:
  - \( \text{\{Fact 1\}} \)
  - \( \text{\{Fact 2\}} \)

7. Knowledge as Ability to Generate a Witness

- Intuitively, in a ZK proof, the verifier learns nothing about the initial state of the system.
- Of course, the verifier may learn facts like "the prover sent \( 337 \) in the second round of the interaction."
- Let \( \text{\{Fact 3\}} \)
- The system \( R \) is relation holding for \( L \), if for all relations \( R \), \( \text{Algorithm M} \), and times \( m \), there exists an algorithm \( M' \) and a negligible function \( \text{\{Fact 4\}} \)

8. Formalizing Generating a Witness for \( R \)

- We want to capture the ability of the verifier to generate witnesses for \( R \) using just its local state.
- Formally, the verifier has an algorithm \( M \) that, given as input the local state \( s \), of the verifier, generates a witness \( \psi \) such that \( R(s, \psi) \) holds.
- The input \( s \) (for which we want to check membership in \( L \)) is in the verifier's local state.
- \( M \) does not get the prover's state \( s \) as input.

9. Relation Hiding

- We consider interactive proofs of languages \( L \) that have a "witness relation" \( R \), that is computable in polynomial time in \( |s| \).
- \( \text{\{Fact 5\}} \)
- Let \( R(s, \psi) \) such that \( \psi \notin R \).

- The system \( R \) is relation hiding for \( L \) if, for all relations \( R \), \( \text{Algorithm M} \), and times \( m \), there exists an algorithm \( M' \) and a negligible function \( \text{\{Fact 6\}} \)

10. Characterizing ZK

- We can model a concurrent ZK system with a single verifier and an infinite number of provers.
- All the provers have the same initial state and use the same protocol \( P \).
- \( P \) is such that provers talk only to the verifier (they do not talk to each other).
- Given a protocol \( P \), let \( P \times \langle \psi \rangle \) denote the system with runs of this form, where all provers run \( P \) and the verifier runs some probabilistic polynomial time protocol.

Theorem 2: The interactive proof system \((P, V)\) for \( L \) is computational concurrent zero knowledge iff the system \( P \times \langle \psi \rangle \) is relation hiding for \( L \).

11. Proofs of Knowledge

In a proof of knowledge, the prover not only convinces the verifier of \( \varphi \) but also that it possesses, or can "knowledgeably compute," a witness for \( \varphi \) from its initial secret information.

Witness Convincing

- Define a relation \( R \), \( R_x \) such that \( x \in R \) iff \( R(x, y) \) holds.
- The system \( L \) is witness convincing for \( x \) if, for all algorithms \( M \), there exists an algorithm \( M' \) and a negligible function \( \text{\{Fact 7\}} \)

If \( t \) is in \( \text{\{Fact 8\}} \) (accepts) with probability \( \frac{1}{2} \).

Halpern, Moses and Tuttle [HMT 1988] proposed a logical definition of "generating a \( \psi \) satisfying \( R(x, \psi) \) for a relation \( R \)."

- They showed that, if \( R \) is variable polynomial time and the verifier can generate a \( \psi \) satisfying \( R(x, \psi) \) at the end of a ZK proof, he can do so at the start.

- They called this property provability security.

12. Characterizing Concurrent ZK

- We can model a concurrent ZK system with a single verifier and an infinite number of provers.
- All the provers have the same initial state and use the same protocol \( P \).
- \( P \) is such that provers talk only to the verifier (they do not talk to each other).
- Given a protocol \( P \), let \( P \times \langle \psi \rangle \) denote the system with runs of this form, where all provers run \( P \) and the verifier runs some probabilistic polynomial time protocol.

Theorem 2: The interactive proof system \((P, V)\) for \( L \) is computational concurrent zero knowledge iff the system \( P \times \langle \psi \rangle \) is relation hiding for \( L \).

13. Proofs of Knowledge

In a proof of knowledge, the prover not only convinces the verifier of \( \varphi \) but also that it possesses, or can "knowledgeably compute," a witness for \( \varphi \) from its initial secret information.

Witness Convincing

- Define a relation \( R \), \( R_x \) such that \( x \in R \) iff \( R(x, y) \) holds.
- The system \( L \) is witness convincing for \( x \) if, for all algorithms \( M \), there exists an algorithm \( M' \) and a negligible function \( \text{\{Fact 7\}} \)

If \( t \) is in \( \text{\{Fact 8\}} \) (accepts) with probability \( \frac{1}{2} \).

Intuitively, this says that if the prover convinces the verifier that \( x \in L \), then the prover knows how to generate a witness \( y \in R(x) \) at the beginning of the protocol.

Theorem 3: The interactive proof system \((P, V)\) for \( L \) is a proof of knowledge iff the system \( P \times \langle \psi \rangle \) is witness convincing for \( L \).

- The runs of system \((P, V)\) for \( L \) are all possible interactions of a verifier running \( V \) with a prover running some probabilistic polynomial time protocol.

14. Future Work: The Evolution of Belief

- Relation hiding restricts the verifier's knowledge at the beginning of the interaction (at time 0) about what he can do at some future time \( m \).
- Intuitively, we would expect that the verifier does not learn something new at any point of zero-knowledge proof.
- This does not hold if we consider only objective probabilities on the verifier’s possible worlds.
- At the end of a run, neither the verifier can generate a witness or not.
- Nevertheless, the verifier may have subjective uncertainty about whether he can generate a witness.
- However, subjective beliefs can be arbitrary.
- What are appropriate constraints/assumptions for how the verifier’s subjective beliefs change during a ZK proof?