Zero-Knowledge Against Quantum Attacks

John Watrous

Department of Computer Science
University of Calgary

January 16, 2006

Zero-Knowledge Proof Systems [Goldwasser, Micali & Rackoff, 1985]

Assume that a **promise problem** $A = (A_{\text{yes}}, A_{\text{no}})$ has been fixed. A **zero-knowledge proof system** for the problem A is a pair (V, P) of interacting parties; a (computationally bounded) **verifier** and a **prover**.

Interaction:

Both parties receive an input string $x \in A_{\text{yes}} \cup A_{\text{no}}$, exchange messages with one another, and finally the verifier V produces an output string denoted $(V, P)(x)$.

Conditions:

Completeness: If $x \in A_{\text{yes}}$, then it must be the case that $(V, P)(x) = 1$ (accept) with high probability.

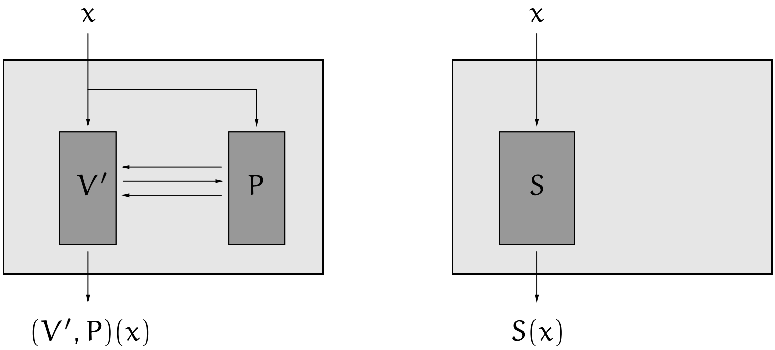
Soundness: If $x \in A_{\text{no}}$, then it must be the case that $(V, P')(x) = 0$ (reject) with high probability for every possible cheating prover P'.

Zero-knowledge: If $x \in A_{\text{yes}}$, then no cheating verifier V' can extract knowledge from an interaction with P.

What does it mean to “extract knowledge”?

The notion of **knowledge** is a complexity-theoretic notion, and is different from **information**; it is formalized by means of the **simulator paradigm**.

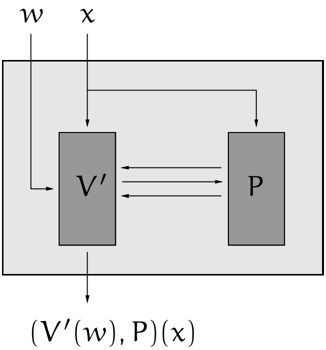
Informally: a verifier V' **learns nothing** (i.e., fails to extract knowledge) from P if there exists a **polynomial-time simulator** S that produces an output that is **indistinguishable** from the output V' would produce when interacting with P on any $x \in A_{\text{yes}}$:

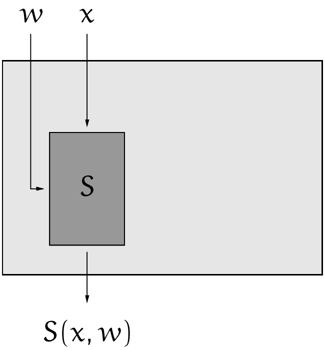


Auxiliary inputs

The previous informal definition is not quite strict enough to capture the notion of zero-knowledge, and gives rise to a class of protocols lacking certain desirable properties...

We need to allow the cheating verifier V' (as well as the simulator S) to take an **auxiliary input** string w. The outputs of these two processes should be indistinguishable provided $x \in A_{\text{yes}}$:





Auxiliary inputs

This **auxiliary input** definition captures the idea that zero-knowledge proofs should not **increase** knowledge, and is closed under sequential composition.

Definition of Zero-Knowledge (classical)

An interactive proof system (P, V) for a given problem $A = (A_{\text{yes}}, A_{\text{no}})$ is **zero-knowledge** if, for every polynomial-time verifier V' there exists a **polynomial-time simulator** S such that, for every w and $x \in A_{\text{yes}}$,

$$(V'(w), P)(x) \quad \text{and} \quad S(x, w)$$

are indistinguishable.* [Goldwasser, Micali & Rackoff, 1989].

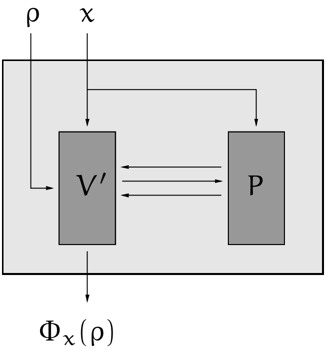
---

* Different notions of indistinguishability give rise to different variants of zero-knowledge, such as **statistical** and **computational** zero-knowledge.

Quantum version of the definition

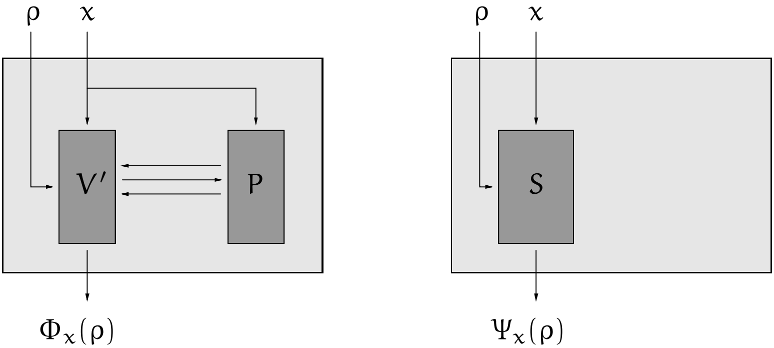
Suppose that some verifier V' tries to use **quantum information** to extract knowledge from P. (Note that the prover P is still classical, so the input x and any information exchanged between V' and P must be classical.)

The interaction between V' and P on input x induces some **admissible mapping** on the auxiliary input:



Quantum version of the definition

If P is zero-knowledge even against a verifier V' that uses quantum information, then there should exist a simulator S that performs an admissible mapping Ψ_x on the auxiliary input that is **indistinguishable** from Φ_x (when $x \in A_{\text{yes}}$):



Problem with the quantum definition?

These definitions are fairly straightforward. . . but have been considered problematic for several years. (The problem was apparently first identified by Jeroen van de Graaf in his 1997 PhD thesis.)

The problem: No nontrivial protocols were previously shown to be zero-knowledge with respect to these definitions, even protocols already proved zero-knowledge in the classical setting.

In order to describe the problem, it will be helpful to consider a simple and well-known zero-knowledge proof system for the **Graph Isomorphism** problem:

Input: Two graphs G_0 and G_1 (given by adjacency matrices).

Yes: G_0 and G_1 are isomorphic ($G_0 \cong G_1$).

No: G_0 and G_1 are not isomorphic ($G_0 \ncong G_1$).

A zero-knowledge proof system for Graph Isomorphism

The following protocol (described for honest parties) is a zero-knowledge protocol for Graph Isomorphism [Goldreich, Micali & Widgerson, 1991].

The GMW Graph Isomorphism Protocol

Assume the input is a pair (G_0, G_1) of n-vertex graphs. Let $\sigma \in S_n$ be a permutation satisfying $\sigma(G_1) = G_0$ if $G_0 \cong G_1$, and let σ be arbitrary otherwise.

Prover's step 1: Choose $\pi \in S_n$ uniformly at random and send $H = \pi(G_0)$ to the verifier.

Verifier's step 1: Choose $a \in \{0, 1\}$ randomly and send a to the prover. (Implicit: challenge prover to show $H \cong G_a$.)

Prover's step 2: Let $\tau = \pi\sigma^a$ and send τ to the verifier.

Verifier's step 2: Accept if $\tau(G_a) = H$, reject otherwise.

Sequential repetition reduces soundness error...

Zero-knowledge property for the GMW protocol

The **completeness** and **soundness** properties are straightforward. Let us consider the **zero-knowledge** property…

Consider a **classical** cheating verifier V':

Verifier's step 1: Perform some **arbitrary** polynomial-time computation on (G_0, G_1), auxiliary input w, and H to obtain $a \in \{0, 1\}$. Send a to P.

Verifier's step 2: Perform some **arbitrary** polynomial-time computation on (G_0, G_1), auxiliary input w, H, and τ to produce output.

Simulator for V':

1. Choose $b \in \{0, 1\}$ and $\tau \in S_n$ uniformly, and let $H = \tau(G_b)$.
2. Simulate whatever V' does given prover message H. Let a denote the resulting message back to the prover.
3. If $a \neq b$ then rewind: go back to step 1 and try again.
4. Output whatever V' would after receiving τ.

Simulator for a cheating quantum verifier?

Suppose that we have a cheating **quantum** verifier V' that starts the protocol with an auxiliary quantum register W.

Verifier's step 1: Perform some arbitrary polynomial-time quantum computation on (G_0, G_1), auxiliary input register W, and H to obtain $a \in \{0, 1\}$. Send a to P.

> **For example:** let a be the outcome of some binary-valued projective measurement $\{\Pi_0^H, \Pi_1^H\}$ of W that depends on H.

Verifier's step 2: Perform some arbitrary polynomial-time quantum computation to produce an output.

How can we simulate such a verifier?

The "no quantum rewinding" issue

Two principles are working against us:

- The **no cloning theorem** prevents making a copy of the auxiliary input register's state.
- Measurements are **irreversible**.

Suppose that we randomly choose b and τ, and let $H = \tau(G_b)$ as for our simulator before. If the simulator guesses incorrectly (meaning $a \neq b$), then the original state of W may not be recoverable.

> *"Rewinding by reversing the unitary transformation induced by [the verifier], or taking snapshots is impossible.*
>
> *But... showing that rewinding by reversing or by taking snapshots is impossible does not show that no other ways to rewind in polynomial time exist."*
>
> [van de Graaf, 1997]

New results

In the remainder of this talk I will argue that the GMW Graph Isomorphism protocol is indeed zero-knowledge against quantum verifiers:

- For any quantum verifier V', there exists a simulator S that induces precisely the same admissible mapping as the interaction between V' and P (on a "yes" input to the problem).
- The method gives a way to "rewind" the simulator, but it requires more than just reversing the verifier's actions. (The entire simulation will be quantum, even though the prover is classical.)
- The method generalizes to several other protocols (but I will only discuss the Graph Isomorphism example in this talk for simplicity).

Assumptions on V'

Assume V' uses three registers:

W: stores the auxiliary input.
V: represents workspace of arbitrary size.
A: single qubit representing the message sent by V'.

Register W starts in the auxiliary state, and registers V and A are initialized to all zeroes.

Assume V' operates as follows:

- For each graph H on n vertices, V' has a corresponding unitary transformation V_H that acts on $(\mathsf{W}, \mathsf{V}, \mathsf{A})$.
- Upon receiving H from P, the V' applies V_H to $(\mathsf{W}, \mathsf{V}, \mathsf{A})$, measures A in the standard basis, and sends the result a to P.
- After P responds with some permutation τ, V' simply outputs $(\mathsf{W}, \mathsf{V}, \mathsf{A})$ along with the prover messages H and τ.

Simulator construction

The simulator will use registers W, V, and A along with:

P_1: stores the prover's first message.
B: stores the simulator's guess b for a.
P_2: stores the prover's second message.
R: stores "randomness" used to generate transcripts.

Define a unitary operator V on (W, V, A, P_1) that represents a unitary realization of V':

$$V = \sum_{H} V_H \otimes |H\rangle \langle H| .$$

Define T to be a unitary operation on registers (P_1, B, P_2, R) for which

$$T : |00 \cdots 0\rangle \mapsto \frac{1}{\sqrt{2n!}} \sum_{b,\tau} |\tau(G_b)\rangle |b\rangle |\tau\rangle |b, \tau\rangle .$$

The operation T produces a superposition over *transcripts*.

Simulator construction

Now define the simulator as follows:

Simulator

1. Perform T, followed by V.
2. Perform a measurement $\{\Pi_0, \Pi_1\}$ whose outcome corresponds to the XOR of A and B (in the computational basis).
3. If the measurement outcome is 1, we need to **rewind and try again:**
   - Perform V^* followed by T^*.
   - Perform a **phase flip** in case any of the qubits in any of the registers (V, A, P_1, B, P_2, R) is set to 1 (i.e., perform $2\Delta - I$, where $\Delta = I_W \otimes |00\cdots 0\rangle\langle 00\cdots 0|$.)
   - Perform T followed by V.
4. Output registers (W, V, A, P_1, P_2). (Registers B and R are traced out.)

Analysis of simulator

Assume that the auxiliary input is $|\psi\rangle$, and $x = (G_0, G_1)$ for $G_0 \cong G_1$. Let

$$|\varphi\rangle = |\psi\rangle\,|00\cdots 0\rangle$$

be the state of all registers given this input.

The simulator performs T, then V, then measures w.r.t. $\{\Pi_0, \Pi_1\}$. Assuming $G_0 \cong G_1$, the outcome will **always be uniformly distributed.**

First, suppose that the measurement $\{\Pi_0, \Pi_1\}$ gives **outcome 0**. The resulting state of all registers is

$$|\sigma_0\rangle = \sqrt{2}\Pi_0 VT\,|\varphi\rangle\,.$$

This is the **target state**: it represents a successful simulation because

$$\mathrm{tr}_{B,R}\,|\sigma_0\rangle\,\langle\sigma_0| = \Phi(|\psi\rangle\,\langle\psi|).$$

(Nothing is surprising here...the simulator has been lucky and didn't need to rewind.)

Analysis of simulator

Suppose on the other hand that the **measurement outcome was 1**. The resulting state is

$$|\sigma_1\rangle = \sqrt{2}\Pi_1 V T |\varphi\rangle .$$

Time to rewind and try again...

Performing the "rewind and try again" procedure results in the state

$$VT(2\Delta - I)T^* V^* |\sigma_1\rangle .$$

Claim

$$VT(2\Delta - I)T^* V^* |\sigma_1\rangle = |\sigma_0\rangle \quad \text{(the target state).}$$

Note: this would not happen for **arbitrary** choices of $|\varphi\rangle$, V, T, Π_0, Π_1, etc... the claim relies on the fact that the measurement $\{\Pi_0, \Pi_1\}$ gives outcome 0 and 1 with equal probability for **all** choices of $|\psi\rangle$.

Proof of claim

The fact that the measurement $\{\Pi_0, \Pi_1\}$ gives outcomes 0 and 1 with equal probability for **all** choice of $|\psi\rangle$ implies

$$\Delta T^* V^* \Pi_0 V T \Delta = \Delta T^* V^* \Pi_1 V T \Delta = \frac{1}{2}\Delta.$$

Therefore

$$\begin{aligned}
&\langle \sigma_0 | V T (2\Delta - I) T^* V^* | \sigma_1 \rangle \\
&\quad = 2 \langle \varphi | T^* V^* \Pi_0 V T (2\Delta - I) T^* V^* \Pi_1 V T | \varphi \rangle \\
&\quad = 4 \langle \varphi | T^* V^* \Pi_0 V T \Delta T^* V^* \Pi_1 V T | \varphi \rangle \\
&\qquad - 2 \langle \varphi | T^* V^* \Pi_0 V T T^* V^* \Pi_1 V T | \varphi \rangle \\
&\quad = 4 \langle \varphi | \Delta T^* V^* \Pi_0 V T \Delta T^* V^* \Pi_1 V T \Delta | \varphi \rangle \\
&\quad = \langle \varphi | \Delta | \varphi \rangle \\
&\quad = 1,
\end{aligned}$$

so $VT(2\Delta - I)T^* V^* |\sigma_1\rangle = |\sigma_0\rangle$. □

Analysis of simulator

This establishes that the admissible map Ψ agrees with the map Φ corresponding to the actual interaction on all pure state auxiliary inputs:

$$\Psi(|\psi\rangle\langle\psi|) = \Phi(|\psi\rangle\langle\psi|)$$

for all $|\psi\rangle$.

Admissible maps are **completely determined** by their actions on pure state inputs, however, so

$$\Psi = \Phi;$$

the simulator **agrees precisely** with the actual interaction on **every possible state** of the auxiliary input register (including the possibility it is entangled with another register).

Other protocols

The simulation method just described can be adapted to prove several other protocols are zero-knowledge against quantum attacks, including:

- Quantum protocols for any problem having an **honest verifier** quantum statistical zero-knowledge proof system:

$$\mathrm{QSZK} = \mathrm{QSZK}_{\mathrm{HV}}.$$

- The Goldreich-Micali-Wigderson **Graph 3-Coloring** protocol assuming unconditionally binding and quantum computationally concealing bit commitments. (See [ADCOCK & CLEVE, 2002].)
- Presumably several other proof systems. . .

Adapting the simulator to other protocols may require iterating the “rewind and try again” process.

Future work/open questions

1. Find further applications and generalizations of the method.
2. Identify limitations of the method.
3. Identify good candidates for quantum one-way permutations.