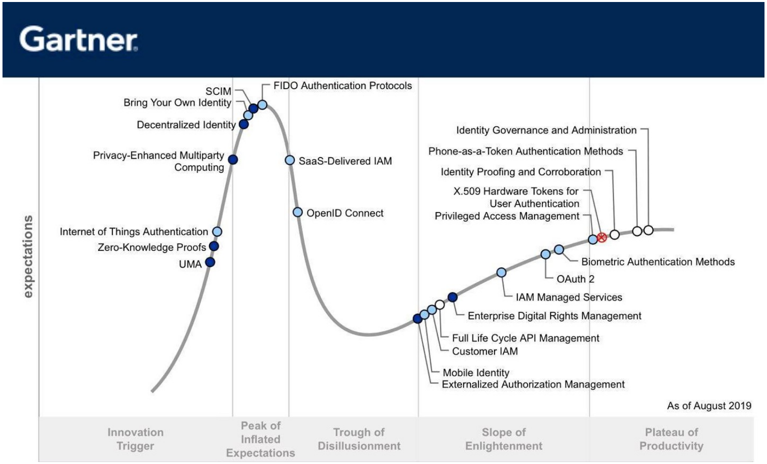
Zero Knowledge Proofs

Saravanan Vijayakumaran
sarva@ee.iitb.ac.in

Department of Electrical Engineering
Indian Institute of Technology Bombay

September 30, 2019

Gartner Hype Cycle for Identity



Source: `https://twitter.com/IdentityMonk/status/1158564314577612800`

Zero Knowledge Proofs

- Proofs that yield nothing beyond the validity of an assertion
- Examples of assertions
  - I know the discrete log of a group element wrt a generator
  - I know an isomorphism between two graphs G_1, G_2
- Proofs are a sequence of statements each of which is an axiom or follows from axioms via derivation rules
  - Traditional proofs do not have explicit provers and verifiers
- ZKPs involve explicit interaction between prover and verifier
- Prover and verifier will be modeled as algorithms or machines
  - Verifier is assumed to be probabilistic polynomial-time (PPT)
  - Prover may or may not be PPT

Examples of Interactive Proofs

- Proving that two chalks have different colours to a colour-blind verifier
- Proof of Quadratic Residuosity
  - For a positive integer N, x is called a quadratic residue modulo N if

$$x = w^2 \bmod N \text{ for some } w$$

  - Suppose $N = pq$ for distinct primes p and q with $|p| = |q| = n$.
  - Without knowing the factorization of N, the best algorithms for checking $x \in QR_N$ run in $\exp\left(\mathcal{O}(n^{\frac{1}{3}})\right)$ steps
  - Using the factorization of N, $x \in QR_N$ can be checked in time which is polynomial in n
- Proof of Quadratic Non-Residuosity
  - Exhaustive checking is not feasible
  - Use an idea similar to the chalks example
- More details on the last two examples
  `http://cyber.biu.ac.il/wp-content/uploads/2018/08/WS-19-1-ZK-intro.pdf`

Knowledge vs Information

- In information theory, entropy is used to quantify information
- Entropy of a discrete random variable X defined over an alphabet $\mathcal{X}$ is
$$H(X) = -\sum_{x\in\mathcal{X}} p(x)\log p(x)$$
- Knowledge is related to computational difficulty, whereas information is not
    - Suppose Alice and Bob know Alice's public key
    - Alice sends her private key to Bob
    - Bob has not gained new information (in the information-theoretic sense)
    - But Bob now knows a quantity he could not have calculated by himself
- Knowledge is related to publicly known objects, whereas information relates to private objects
    - Suppose Alice tosses a fair coin and sends the outcome to Bob
    - Bob gains one bit of information (in the information-theoretic sense)
    - We say Bob has not gained any knowledge as he could have tossed a coin himself

Modeling Assertions and Proofs

- The complexity class $\mathcal{NP}$ captures the asymmetry between proof generation and verification
- A language is a subset of $\{0,1\}^*$
- Each language $L \in \mathcal{NP}$ has a polynomial-time verification procedure for proofs of statements "$x \in L$"
  - Example: L is the encoding of pairs of finite isomorphic graphs
- Let $R \subset \{0,1\}^* \times \{0,1\}^*$ be a relation
- R is said to be polynomial-time-recognizable if the assertion "$(x,y) \in R$" can be checked in time poly$(|x|,|y|)$
- Each $L \in \mathcal{NP}$ is given by a PTR relation R_L such that

  $$L = \{x \mid \exists y \text{ such that } (x,y) \in R_L\}$$

  and $(x,y) \in R_L$ only if $|y| \leq \text{poly}(|x|)$
- Any y for which $(x,y) \in R_L$ is a proof of the assertion "$x \in L$"

Interactive Proof Systems

- Let $\langle A, B\rangle(x)$ denote the output of B when interacting with A on common input x
- Output 1 is interpreted as “accept” and 0 is interpreted as “reject”

Definition

A pair of interactive machines (P, V) is called an **interactive proof system for a language** L if machine V is polynomial-time and the following conditions hold:

- **Completeness**: For every $x \in L$,

$$\Pr[\langle P, V\rangle(x) = 1] \geq \frac{2}{3}$$

- **Soundness**: For every $x \notin L$ and every interactive machine B,

$$\Pr[\langle B, V\rangle(x) = 1] \leq \frac{1}{3}$$

- Remarks
  - Soundness condition refers to any possible prover while completeness condition refers only to the prescribed prover
  - Prescribed prover is allowed to fail with probability $\frac{1}{3}$
  - Arbitrary provers are allowed to succeed with probability $\frac{1}{3}$
  - These probabilities can be made arbitrarily small by repeating the interaction

Interactive Proof for Graph Non-Isomorphism

- Graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are isomorphic if there exists a bijection $\pi : V_1 \mapsto V_2$ such that $(u, v) \in E_1 \iff (\pi(u), \pi(v)) \in E_2$
- Graphs G_1 and G_2 are non-isomorphic if no such bijection exists
- Prover and verifier execute the following protocol
    - Verifier picks $\sigma \in \{1, 2\}$ randomly and a random permutation π from the set of all permutations over V_σ
    - Verifier calculates $F = \{(\pi(u), \pi(v) \mid (u, v) \in E\}$ and sends the graph $G' = (V_\sigma, F)$ to prover
    - Prover finds $\tau \in \{1, 2\}$ such that G' is isomorphic to G_τ and sends τ to verifier
    - If $\tau = \sigma$, verifier accepts claim. Otherwise, it rejects.
- Remarks
    - Verifier is a PPT machine but no known PPT implementation for prover
    - If G_1 and G_2 are not isomorphic, then verifier always accepts
    - If G_1 and G_2 are isomorphic, then verifier rejects with probability at least $\frac{1}{2}$
    - Repeated interactions can make false acceptance probability arbitrarily small

Zero Knowledge Interactive Proofs

- Consider an interactive proof system (P, V) for a language L
    - In an interactive proof, we need to guard against a malicious prover
    - To guarantee zero knowledge, we need to guard against a malicious verifier
- Recall that knowledge is related to computational difficulty
- Informal definition
    - An interactive proof system is **zero knowledge** if whatever can be efficiently computed **after interaction** with P on input x can also be efficiently computed from x (**without interaction**)
- Formal definition (ideal)
    - We say (P, V) is **perfect zero knowledge** if for every PPT interactive machine V^* there exists a PPT algorithm M^* such that for every $x \in L$ the random variables $\langle P, V^* \rangle(x)$ and $M^*(x)$ are **identically distributed**
    - M^* is called a **simulator** for the interaction of V^* with P
- Unfortunately, the above definition is too strict
- A relaxed definition is used instead

Perfect Zero Knowledge

Definition

Let (P, V) be an interactive proof system for a language L. We say that (P, V) is **perfect zero knowledge** if for every PPT interactive machine V^* there exists a PPT algorithm M^* such that for every $x \in L$ the following two conditions hold:

1. With probability at most $\frac{1}{2}$, machine M^* outputs a special symbol $\perp$
2. Let $m^*(x)$ be the random variable describing the distribution of $M^*(x)$ conditioned on $M^*(x) \neq \perp$. Then the random variables $\langle P, V^* \rangle(x)$ and $m^*(x)$ are **identically distributed**

- Remarks
  - M^* is called a **perfect simulator** for the interaction of V^* with P
  - By repeated interactions, the probability that the simulator fails to generate the identical distribution can be made negligible
- **Alternative formulation**: Replace $\langle P, V^* \rangle(x)$ with $\text{view}_{V^*}^{P}(x)$
  - A verifier's view consists of messages it receives and any randomness it generates
  - Simulator M^* has to change accordingly

ZK Proof for Graph Isomorphism

- An isomorphism ϕ between graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ exists
- Prover and verifier execute the following protocol
  - Prover picks a random permutation π from the set of permutations of V_2
  - Prover calculates $F = \{(\pi(u), \pi(v) \mid (u, v) \in E_2\}$ and sends the graph $G' = (V_2, F)$ to verifier
  - Verifier picks $\sigma \in \{1, 2\}$ randomly and sends it to prover
  - If $\sigma = 2$, then prover sends π to the verifier. Otherwise, it sends $\pi \circ \phi$ to the verifier where $(\pi \circ \phi)(v)$ is defined as $\pi(\phi(v))$
  - If the received mapping is an isomorphism between G_σ and G', the verifier accepts. Otherwise, it rejects
- Remarks
  - Verifier is a PPT machine. If ϕ is known to prover, it is a PPT machine
  - If G_1 and G_2 are isomorphic, then verifier always accepts
  - If G_1 and G_2 are not isomorphic, then verifier rejects with probability $\frac{1}{2}$
  - The prover is perfect zero knowledge (to be argued)

Simulator for Graph Isomorphism Transcript

- For an arbitrary PPT verifier V^*, $\text{view}_{V^*}^{P}(x) = \langle G', \sigma, \psi \rangle$ where ψ is an isomorphism between G_σ and G'
- The simulator M^* uses V^* as a subroutine
- On input (G_1, G_2), simulator randomly picks $\tau \in \{1, 2\}$ and generates a random isomorphic copy G'' of G_τ
    - Note that G'' is identically distributed to G'
- Simulator gives G'' to V^* and receives $\sigma \in \{1, 2\}$ from it
    - V^* is asking for an isomorphism from G_σ to G''
- If $\sigma = \tau$, then the simulator can provide the isomorphism $\pi : G_\tau \mapsto G''$
- If $\sigma \neq \tau$, then the simulator outputs $\perp$
- If the simulator does not output $\perp$, then $\langle G'', \tau, \pi \rangle$ is identically distributed to $\langle G', \sigma, \psi \rangle$

References

- Sections 4.1, 4.2, 4.3 of *Foundations of Cryptography, Volume I* by Oded Goldreich
- Alon Rosen's lecture in the 9th BIU Winter School on Cryptography
  - `https://cyber.biu.ac.il/event/the-9th-biu-winter-school-on-cryptography/`
  - `https://www.youtube.com/watch?v=6uGimDYZPMw`
  - `http://cyber.biu.ac.il/wp-content/uploads/2018/08/WS-19-1-ZK-intro.pdf`