1 Application of ZKP

[Jan: Applications of ZKPs]

In order to give the reader a motivation to explore further on Zero-Knowledge Proofs, we present in this section some interesting applications:

1. **Applications of Zero-Knowledge Range proof** [?] [Jan: "proofs"] let [Jan: "Let", please take a look at typos like that.] us mention a special kind of ZKP, Zero-Knowledge Range Proofs (ZKRP). ZKRP allows us to verify that a number lies within a certain range. Note that ZKRP holds the same three properties as ZKP, which are completeness, soundness, and zero-knowledge.
   - Over 18 ZKRP (zero-knowledge range proofs): We can use this application in services that need to validate their user's age before he gets to access their server. The interesting feature in ZKRP is that a user can validate his age without revealing it.
   - Know Your Customer (KYC): Since ZKRP allows us to validate that a certain amount of private information is in a specific range, we can use this property to ensure compliance while maintaining the user's privacy. [Jan: Can you give a more specific example?]
   - Mortgage risk assessment: This application is more important for financial institutions, with it we can prove that the salary of someone is more than some amount to get a mortgage approved.
   - E-voting: By using ZKP we can construct a secure, and trusted e-voting system to ensure mutual authentication between the election authority server and the voters.
   - Electronic auctions and procurement: Using the protocol introduced in [?] alongside ZKP we can present a protocol that allows participants to ensure that an auction has run correctly without revealing the bid values of other participants while increasing the robustness of the auction protocol.
2. **Authentication systems**: ZKP help the user who wants to verify its identity (password for example) via some secret to a second party, but the second party should not learn anything about this secret.[?] [Jan: Correct English in that sentence.]
3. **Ethical behavior**: ZKP uses to oblige a user to prove that his behavior is correct according to the protocol, because ZKP saves the privacy of the user's secret during the process of providing the proof.[?] [Jan: Same as with KYC, is there a more specific example?]
4. **Nuclear disarmament**: ZKP assists inspectors to confirm whether or not an object is indeed a nuclear weapon without recording, sharing, or revealing the internal workings which might be secret.[?]
5. **DLT (Distributed ledger technology) and blockchain**: by using ZKP we can perform a valid transaction with [Jan: "while"] keeping the sender, the recipient, and all other transaction details remain [Jan: delete "remain"] hidden:
   - Confidential Transactions:

In a usual transaction in DLT, the amount of money spent is public for anyone in the network, by using CT [Jan: What is CT?], each party can use a commitment scheme to hide the amount they send or receive. Hence, no adversary can see this amount.

- Provisions:
  Bitcoin exchanges work like banks, by holding their customer's bitcoin securely on their behalf, but these exchanges should have a proof of solvency, to avoid big problems like customer losing their money permanently. A proof of solvency shows that the exchanges have sufficient reserves to settle each customer's account. Provisions (a privacy-preserving proof of solvency) is when an exchange could have a proof of solvency without revealing its Bitcoin addresses; total funds or liabilities; or any information about its customers. [Jan: This is nice, but there should be a reference somewhere.]

2 ZKP for NP problems

[?] shows that all language in NP have Zero-Knowlege proof system, in this section we explore ZKP for the graph three-colorability problem (G3C) and ZKP for Fiat-Shamir Identification Protocol, with short explanation, the protocol, and the idea of the proof for each one of them.

2.1 Graph Three-Colorability Problem (G3C) [?]

G3C is NP-complete, therefore it can be used for any language in NP, the protocol rests on a "computational assumption". [Jan: These comments should come after you explain the problem. Also maybe you want to explain why ZK protocol for an NP-complete problem implies that you have ZK protocol for every problem in NP. For this you want to define what is a reduction (look it up in Chapter 2 of Arora-Barak) and then say that for any other language prover and verifier can first reduce their inputs to 3-coloring and then run ZK proof for 3-coloring.]

first, let's give the idea of G3C problem : The idea of this problem is, we want to color a graph G with 3 different colors such that no two adjacent vertices have the same color. Suppose $G = (E, V)$ with, $\mid E \mid = m$ [Jan: Don't use \mid for this, just $|E|$.]

and $\mid V \mid = n$ The following protocol should apply [Jan: apply $->$ be repeated] m^2 times using independent coin tosses: [Jan: Before you start, explain that you assume that the prover has a 3-coloring of the graph and that you call it ϕ.]

1. The prover chooses random permutation Π [Jan: Rephrase to make it clear that it is permutation of colors of ϕ.] for the proper coloring (ϕ) to get new coloring and put these colors into n locked boxes and send them without their keys to the verifier.

2. The verifier chooses randomly an edge $e \in E$ and sends it to the prover.

3. Let e=(u,v), if $e \in E$ then the prover sends the corresponding [Jan: Which keys are corresponding?] keys to the verifier, else prover do nothing.

4. The verifier opens the boxes and sends accept iff they are different. [Jan: If what is different?] !

[Jan: I suggest you don't write as theorem/proof, just write a couple of paragraphs where you sketch why this protocol has three properties.] !

Theorem 1 *G3C protocol satisfies completeness, soundness and zero-knowledge.*

Proof 1 [Jan: Divide it into paragraphs.] ! *Completeness. Since $G \in G3C$, then for any edge $e \in E$, therefore the verifier will see any corresponding boxes they don't have the same color. So the verifier returns accept for all m^2 iteration. Soundness. If $G \notin G3C$ then at least there is one edge has the same color on its vertices, so in each round the verifier will reject with probability at least $1/m$. The protocol satisfies soundness $(1/m)^{m^2}$* [Jan: It's $(1-1/m)^{m^2} \leq \exp(-1/m \cdot m^2) = \exp(-m)$.] ! *Zero-knowledge. We have to construct a simulator S such that it produces a transcript which is computationally indistinguishable from that produced from the protocol between the prover and the verifier. Since we are dealing with any verifier define a black box V^* to get the edge e from it to avoid any attempt from the verifier to get more information from the verifier. Since π also selected randomly and independently at each round, so the transcript that produced from S is indistinguishable from the one between the prover and the verifier.* [Jan: The reason this protocol is zero-knowledge is because: 1) Verifier cannot learn anything from unopened commitments. 2) Information from open commitments are indistinguishable from random different colors (a, b), which can be generated by the simulator.] !