# Zero Knowledge Range Proof:SmartProof

Amit Bhardia
*Dept.Computer Engineering*
*San Jose State University*
San Jose,California
amit.bharadia@sjsu.edu

Shubham Sawant
*Dept.Computer Engineering*
*San Jose State University*
San Jose,California
shubham.sawant@sjsu.edu

Rohan Kamat
*Dept.Computer Engineering*
*San Jose State University*
San Jose,California
rohansantosh.kamat@sjsu.edu

Nitish Joshi
*Dept.Computer Engineering)*
San Jose,California
San Jose,California
nitish.joshi@sjsu.edu

***Abstract*—Zero Knowledge Proof is an interactive methodology which allows the prover to prove the knowledge of a secret without revealing it. ZKP is just a methodology and can be implemented in a variety of ways. Majority of the applications are concerned with security domain. We provide a methodology which can verify the details of prover without the prover actually revealing them. The proposed methodology is suited for a variety of government and commercial applications.**

***Keywords*—Zero Knowledge Proof, ZKP ,Zero Knowledge Range Proof , Prover, Verifier, Three-Coloring Problem, Amazon S3**

## I. Introduction

Zero knowledge proof (ZKP) is an approach/method that we can use to solve this problem. In ZKP, prover proves his statement or in our case his identity to verifier without revealing it. So our solution goes like this, there will be a centralized trustworthy system. The prover (John/Karen), will submit their identity proof/supporting document to the system. It will have 2 kits for that document. One will be prover's kit and other will be verifier's kit. Kit is nothing but a program which runs on ZKP algorithm and creates encrypted proof for the document. This proof won't contain any kind of direct value of the original document. So there is no risk involved. Prover's kit will create an encrypted proof. (In first case it will be a valid passport ID and in second cases it will be confirmation of having age greater than 18 or so.) Verifier's kit will verify the encrypted proof with the help of public key. It will output the confirmation. This way John or Karen can submit their proof of identity without revealing any information. We will try to implement a generic system which will create verifier's and prover's kit dynamically. For example, if user wants to create a kit in order to prove his bank balance, then he will submit his balance and the system will create his kit (prover's kit) for him. On the same line, verifier's kit will be generated.

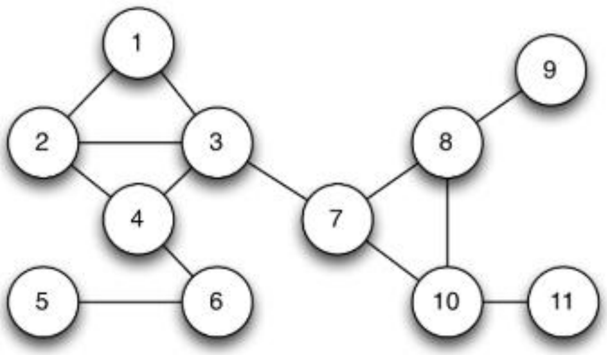
## II. Origins of zero knowledge proof

### A. First Introduction of Zero Knowledge Proof

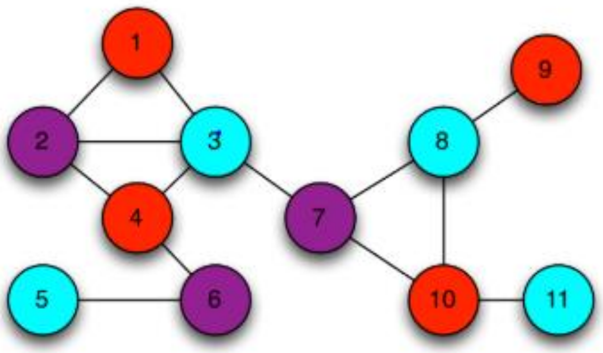
The notion of 'zero knowledge' was first proposed in the 1980s by MIT researchers Shafi Goldwasser, Silvio Micali and Charles Rackoff. The researchers were working on an interactive system where, a party(Prover) could exchange messages with another party(Verifier) to convince the verifier that some mathematical statement is true. Prior to this research all the study was completely focused on how to ensure that the prover is true, i.e. his identity is not compromised. What Goldwasser did was to turn the problem to the verifier itself. He questioned instead of worrying about the correctness of the identity of the prover, what if there is an instance of not trusting the verifier? This question was raised because it was necessary to understand, because during the process of verifying how much more information is verifier going to learn.

### B. A 'Real World' example

The discussion has been abstract as of now , consider a real-world example, a large telecom giant has established a cellular network. The network is represented below



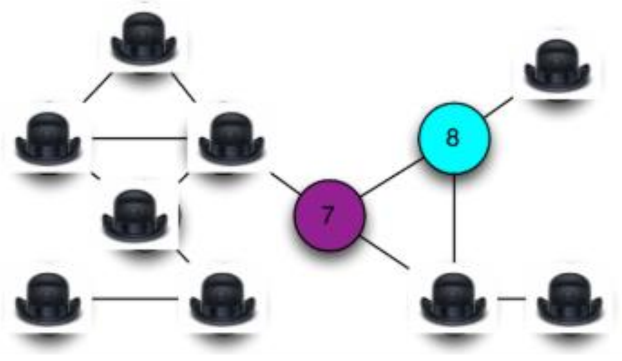
The vertices in the graph represent the cellular towers whereas the edges E represent the overlapping of signals because of being in the same location. The overlapping cause the signals to interfere with each other and are likely to scramble the reception. Fortunately, the infrastructure allows the towers to work on 3 different frequencies.



Thus the it is necessary to configure the network in such a manner that the adjacent nodes are working on a different frequency. This problem is also called as the graph three-coloring problem. Finding the solution is very difficult, and for some graph it is difficult even determining if solution exists It's easy to say solve the above toy problem by hand. The cellular giant has a very large network, which is so complicated that finding a solution needs very large computational power, something that is not at its disposal. So , they outsource it to some capable company like Google. So, Google uses its sophisticated infrastructure and find a coloring solution the graph, the problem here is Google won't reveal it's solution until payment is made by Cellular giant, and the latter needs to confirm that Google has a solution to its graph problem. Google cannot reveal its solutions or even a part of it, to the party. The only way is Google has to find some way to convince the cellular giant that it has the solution without actually revealing it. Here comes the role of ZKP.

## C. *The Solution*

A warehouse is chosen wherein, the network topology is laid on a sheet, then Google executives are asked to use their 3 color shuffle to layout their representation on the network, the vertices are colored and covered so as not to reveal the solution. The respective authorities from the Cellular giant can enter the warehouse for inspection. The are allowed to view any two adjacent nodes(vertices).



There are two outcomes of the experiments :

1. If the two revealed vertices are the same color (or aren't colored in at all!) then authorities definitely know that Google is lying . Clearly not going to pay Google a cent.
2. If the two revealed vertices are different colors, then Google might not be lying.

This requires much more consideration, Google might still be bluffing and a mere fluke that the outcome was favorable. Specifically, after one test they could succeed in cheating me with probability up to (E-1)/E (which for a 1,000 edge graph works out to 99.9% of the time). Fortunately Google has an answer to this. Lay down the architecture fresh again, Google again uses their 3 color shuffle to represent their solution, and similar procedure is followed. This procedure is iterated many times until the authorities from cellular company is convinced that Google has a solution to their problem.

## D. *What makes it "Zero Knowledge"*

Goldwasser, Micali and Rackoff proposed three following properties that every zero-knowledge protocol must satisfy. Stated informally, they are:

1. *Completeness.* If Google is telling the truth, then they will eventually convince me (at least with high probability).
2. *Soundness.* Google can *only* convince me *if* they're actually telling the truth.
3. *Zero-knowledgeness.* (Yes it's really called this.) I don't learn anything else about Google's solution.

## III. Outline of our work

We have implemented zero knowledge range proof, wherein the prover can simply provide his details in the form of some document , the verifier provides some range to check whether the prover details lies in that range. A Boolean value is generated which confirms that the information provided is verified. The above process ensures that no details of the prover are revealed, yet the verifier is convinced that the information provided is correct.

Lets understand the steps and the work flow of the application.

1. User signup as prover, verifier sign up as verifier.
2. The prover signs in.
   - 2.1 The prover selects either of the 2 options, age proof or bank balance.
   - 2.2 The prover then uploads the document.
   - 2.3 An ID is generated which is shared with the verifier.
3. The verifier then signs in.
   - 3.1 The verifier then gives a reference name to the request , provides the ID given by the prover.
   - 3.2 The required range proof is set by giving the upper and the lower limit.

## A. *How is it working*

1. When the prover upload the document , the document is uploaded to the Amazon S3 bucket . Here the document is scanned for the age/Balance field.
2. The verifier set the range in his portal, this range get notified to the solidity contract on the Hyperledger burrow node, the value extracted from the file is sent to the solidity contract where the verification process is done. The verification

process is basically checking whether the value provided by the prover lies within the range of the verifier.

## *B. Algorithms Used*

### *1. The Discrete Logarithm problemEquations*

In real numbers, taking the logarithm is a trivial operation. For example

$b^n = x$ then $\log_b x = n$
$2^3 = 8$ then $\log_2 = 8$

In any case, in the event that we take a cyclic gathering, figuring the discrete logarithm gives off an impression of being hard. A cyclic gathering is a gathering of numbers produced by a solitary component g, called the generator. The Discrete Logarithm issue can all the more formally be depicted as pursues: Given a gathering G, a generator g of the gathering and a component h of G, to locate the discrete logarithm to the base g of h in the gathering G. There are no polynomial time calculations that tackle the Discrete logarithm issue. Along these lines, in the event that we pick sufficiently substantial numbers settling, the discrete logarithm ends up unrealistic to comprehend. It can generally be fathomed given enough calculation time.

### *2. The Fiat-Shamir heuristic*

The purpose of this cryptographic technique is that a fact (e.g. knowledge of a certain number) can be proven without revealing the actual fact. For example, Alice wants to proof that she knows the solution to $b^n = x$. She picks a random integer v and calculates $s = b^v$ . Then, Alice computes $c = H(b, x, s)$, where H is a hash function. Also, she computes $r = v - (n * c)$, and publishes the proof (s,r). Then, anyone can verify that $s = b^r \ x^c = b^{v-nc} \, b^{nc} = b^v$ . Since the outcome of the hash function H was unpredictable, Alice must have known n to be able to compute r, because otherwise Alice was able to solve the Discrete Logarithm problem. Therefore, by only providing (s, r), Alice proofs the fact that she knows the number n.

### *3. The Integer Factorization problem*

Integer factorization is the decomposition of any composite number (i.e. a positive integer) into a product of smaller integers. An example is: $20 = 4 * 5$. If these smaller integers are restricted to prime numbers (e.g. $15 = 3 * 5$), and the composite number is sufficiently large (preferably with only large prime factors), then there currently exists no algorithm to solve this in polynomial time.

### *4. The Integer Factorization problem*

The secret order principle states that it is difficult to compute the order of an element in a group where the prime factors of the modulus are unknown This guideline depends on the Integer Factorization issue. The request of a component in a cyclic gathering is the type which decreases the component in the gathering to the personality component. As an example, in the integer modulo 7, 3 has the order 6 (7-1):
$3 * 3 * 3 * 3 * 3 * 3 \bmod 7 =$
$2 * 2 * 2 \bmod 7 =$
$8 \quad \bmod 7 = 1$

### *5. The Discrete Logarithm problemEquations*

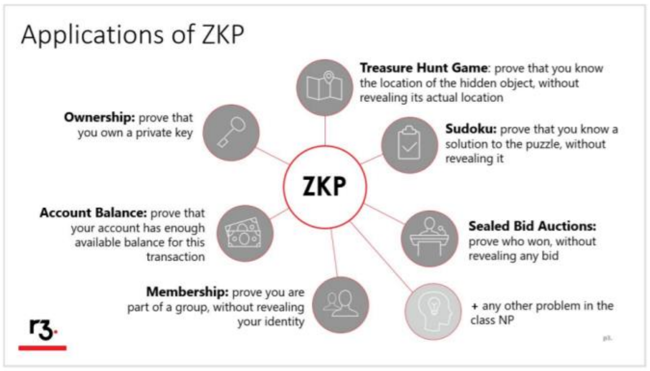
As stated by Wikipedia, if one knows the remainders of the Euclidean division of an integer n by several integers, then one can determine uniquely the remainder of the division of n by the product of these integers, under the condition that the divisors are pairwise coprime [15]. For example, what is the smallest number x that divided by 3 has a remainder of 2, dividing by 5 a remainder of 3, and divided by 7 a remainder of 2? The correct answer here can be provided by the theorem, and is 23.

### *6. Fujisaki-Okamoto Commitment*

Let N be a large composite number of which the prime factorization is unknown to Alice. Let g be a large element in the group $Z * N$ , and let h be an element of the group generated by g, such that both discrete logarithms are unknown. Let $E(x, r) = g^x \ h^r \bmod n$ be a commitment to x in base (g, h) where r is a random value. Then, it is computationally infeasible to compute x1, x2, r1, r2 where $x1 \neq x2$ such that $E(x1, r1) = E(x2, r2)$. Finally, $E(x, r)$ statistically reveals no information of x to Bob.

## *C. Future Work*

ZKP being an hot topic there are a variety of scope, the application was deployed in isolated individual microservices. This enables our application to work with any application and reduces the effort in integration. The ID which was generated can be substituted with a bar code. For now our application just validates range , which can be further enhanced to location, identity management in authentication mechanism.


Applications of ZKP
ZKP
Treasure Hunt Game: prove that you know the location of the hidden object, without revealing its actual location
Ownership: prove that you own a private key
Sudoku: prove that you know a solution to the puzzle, without revealing it
Account Balance: prove that your account has enough available balance for this transaction
Sealed Bid Auctions: prove who won, without revealing any bid
Membership: prove you are part of a group, without revealing your identity
+ any other problem in the class NP
r3.

## Acknowledgment

We are very thankful to Prof. Rakesh Ranjan San Jose State University for giving us the opportunity of working on ZKP. ZKP being a new concept very less material was available, we would like to express our gratitude towards ING Bank, as their ZKP implementation and repository was of great help.

## References

[1] Tommy Koens, Coen Ramaekers and Cees van Wijk, “Efficient Zero-Knowledge Range Proofs in Ethereum,”

[2] https://github.com/ing-bank/zkrangeproof

[3] https://www.linkedin.com/pulse/demonstrate-how-zero-knowledge-proofswork-without-using-chalkias

[4] https://isp.cs.ru.nl/2016/danezis.pdf

[5] https://blog.cryptographyengineering.com/2014/11/27/zero-knowledge-proofs-illustrated-primer/