ZERO KNOWLEDGE PROOFS APPLIED TO AUCTIONS

Luiz Thomaz do Nascimento
lthomaz@mit.edu

Sapna Kumari
skumari@mit.edu

Vedavinayagam Ganesan
vedag@mit.edu

May 16, 2019

Contents

1 Introduction **3**

2 Auctions **3**

2.1 Definition . . . 3

2.2 Types of auctions . . . 3

2.3 Governmental Auctions Requirements . . . 4

2.4 Zero-knowledge Proofs Applied to Auctions . . . 4

3 Related work **4**

4 Zero-knowledge proofs **5**

4.1 Intuition . . . 5

4.2 Definition . . . 5

4.3 Interactive Proofs . . . 6

4.4 Non-Interactive Proof . . . 6

4.5 Implementations of Non-Interactive Proofs . . . 7

5 Bulletproofs **8**

5.1 Inner Product Argument . . . 8

5.2 Range Proofs . . . 8

5.3 Arithmetic Circuits . . . 9

6 Code Implementation **9**

6.1 Hyrax . . . 9

6.2 BulletproofLib . . . 10

7 Future work **10**

8 Conclusion **10**

1 Introduction

Modern technology is transforming manual work flows into digital automated processes with more efficiency and less human dependency. Digital transformation and hyper connectivity of entities, which include businesses, households, governments, and other public and private organizations, have led to widespread use of digital transactions. The massive digitization of transactions has increased the need of security, privacy and confidentiality. Since the same technological advances are also accessible to dishonest parties, the security and transparency of such processes are of paramount importance.

Online auctions is one example of a digital transaction that recently has gained a great deal of popularity as a replacement for traditional buy-sell market place settings since online auctions are much more convenient for auctioneers and bidders who do not have to physically attend an auction. Given that online auctions can reach wider range of bidders, the auctioneer can get more competitive prices, compared to traditional auctions. From security and privacy standpoint, online auctions require same level of security, confidentiality and privacy as it is required in any other electronic transaction settings. Research on cryptography field provides technologies that can be applied in auction applications. Zero-knowledge proof, ZKP, is one of those. This project is about the application of ZKP on transactions in online auctions and its goal is to apply this technology to provide transparency and privacy in governmental auctions settings. First, we introduce the auction process and different types of auctions. Next, we briefly review literature on security issues in auctions and application of various cryptographic schemes for electronic auctions. Then, we discuss ZKP concepts and the implementation of such constructions, in particular Bulletproofs which was the ZKP used in this project. After this, we discuss our code implementation and finally we conclude with the findings of our experience in applying Bulletproof cryptography protocol in the auction use case.

2 Auctions

2.1 Definition

Auction is a form of price negotiation. In traditional e-commerce setting, the seller or producer publishes an asking price for a product or service. This mechanism allow auctioneers to get the highest price for a product, service or prize. In a typical setting, bidders make their bids and those represent a commitment to pay that amount offered by the auctioned item. Bids can be opened or closed, meaning that bidders can or cannot see others bid values. Additionally, auctions can be forward or reverse. On the first type buyers are the bidders and on the second one the auctioneer is the buyer who is interested in paying the lowest price offered by several suppliers, or bidders, for one product of service.

Online auctions are the digital representation of this work-flow on the internet. Buyers and sellers located around the world can negotiate and exchange goods and services. The auction process is run by software agents with website interfaces which facilitate bidding and selection of the winning.

2.2 Types of auctions

There are several types of auctions. At the primary level there are four types of auctions.

Open Cry Auction: This is the traditional type of auction in which the bidder publicly announces their bid. In English-Auction setting, the auctioneer invites bids. Once a bid has been made, the subsequent bids are higher than the previous bid. The latest bidder is the winner. This type of auction is also called ascending price auction since each subsequent bidder bids higher price than the previous one. On the other hand, in Dutch-Auction setting, the auctioneer offers goods and service at a high price. If no one buyer accepts the price, the price is lowered and so on. The first bidder wins the auction

Sealed Bid Auction:This auction setting may involve several phases. In the first phase, many bidders bid but the bidding value is not known by the public. In the second phase, the bids are opened and the auctioneer may select the winner with lowest value. This sequence may be iterated several times until the auctioneer gets a price and quality required.

Multi-items auctions: In this auction, multiple identical items are for sale. Each winner pays the price he bid (Discriminative Auction) or each winner pays the price of the lowest winning bid (Non-discriminative auction)

Vickrey Auction: This is also called as Secondary price auction. The winner pays the second highest price bid.

In addition to the above major four types of auctions there are several other types auctions based on who bids, who sells, number of bidders or number of sellers. Reverse auction is an auction where the role of buyer and sellers are reversed.

There is only one buyer receiving bids from multiple sellers who are making their bids to sell their offering. The reverse auction methods is commonly used in procurement processes by governments. Governmental organizations engage in public auctions by issuing public tenders. In this setting, the suppliers compete to sell their goods and services. The government wants to involve public without any discrimination. Fair chance is given to all participants of the society. In this fair process, the government selects a competitive price among the sellers who meets government's minimal requirements predefined criteria. In this setting, typically bidders do not want to share their price information during auction process. This process is similar to the sealed-bid auction, described before.

2.3 Governmental Auctions Requirements

Governmental auctions have several implications since they deal with public goods. Below is list of typical requirements in such public auctions.

(i) Fairness:
This is most important requirement for this auction setting. All participants must be treated equally. During the auction, all bids should remain confidential. After the bidding phase is completed, no bidder should be able to modify the committed bid (Unforgeability). Finally, during the opening phase, all the bids must be opened and lowest bid must win.

(ii) Confidentiality:
Except the winning bid all the other bids must remain confidential

(iii) Anonymity:
Information about the identity of the bidders must be confidential. However the identity of the winning bid may not be confidential.

2.4 Zero-knowledge Proofs Applied to Auctions

Online public auctions could benefit from the properties of zero-knowledge proofs. This technology provides proofs keeping the privacy which is the paradox faced by such auctions. It is necessary to provide transparency by showing that the lowest price won while keeping price privacy of all other bidders. The central idea of the proposed solution in this project is to design a proof system that uses zero-knowledge proofs to demonstrate the selection of the winning bid followed the rules without leaking any information that needs to remain confidential. Below, is a list of main steps of the solution work-flow.

(a) Auction Initiation: The auctioneer initiates the auction by starting an auction repository. This auction repository will stores all committed bids, not their openings. This repository is public to all public auction participants.

(b) Bid commitments: The bidders commit their bids by submitting the cryptographic commitments of their bid value

(c) Opening Bids: After the commitment phase is finished, the auctioneer opens all commitments using the instructions private to him.

(d) Proof generation: It is generated one proof for each losing bid. This proof demonstrate that the difference between the losing value and winning value is positive. All proofs use the pre-committed values and no plan text values should be disclosed.

(e) Proof Verification: Each proof can be publicly verified by any interested party.

3 Related work

Recent literature covering online reverse auctions is quite limited. However, we found some research covering online forward auctions. In [1], there are recommendations for electronic auctions where researchers spotted security and privacy issues while simulating the open-cry auctions. Slow completion rate and lack of anonymity are inherent in open-cry auctions. Internet security challenges make implementation of online open cry auctions challenging [2, 3].

There are some previous work on sealed bid auctions. Part of them address issues from bidder perspective such as refusal of opening bid by the bidder and communication failure [3, 4]. Those proposals suggested usage of undeniable signature schemes, electronic cash technology with anonymity and non interactive proofs. Other related research address issues from auctioneer perspective such as trust and fairness issues [2, 5, 6, 7]. Those proposals recommend usage of verifiable signature sharing to distribute electronic version of auction escrow among distributed auctioneers,

which are expected to force bidders to open their bids. Another recommendation is based on a complex polynomial secret sharing scheme in combination with bid splitting, essentially to enforce trust. These protocol proposals have their own issues or they did not address all security and privacy issues of auctions. They also incur high communication and computational costs.

There are some recent auctions related work using non-interactive proofs and other cryptography protocols. This mentioned work discuss auctions in Smart contract settings in Blockchain. For example the article in [8], it is provided an application design short zero-knowledge proofs to confirm consistency of auctions.

4 Zero-knowledge proofs

Zero-knowledge proof [9] is a method by which one party, the prover, can convince another party, the verifier, that they possess knowledge of some information without revealing the knowledge itself. A prover can for example convince a verifier that a confidential transaction is valid without revealing why that is the case, i.e. without leaking the transacted values. While the prover can prove his possession of some knowledge by simply revealing that knowledge to the verifier, the challenge of ZKP is to prove such knowledge without revealing the information itself or any additional information at all.

4.1 Intuition

There is a cave with a single entrance but with two interconnected tunnels that form a ring. In the ring, there is a locked door on the opposite side of the entrance which opens when some secret code is input on its puzzle-lock.

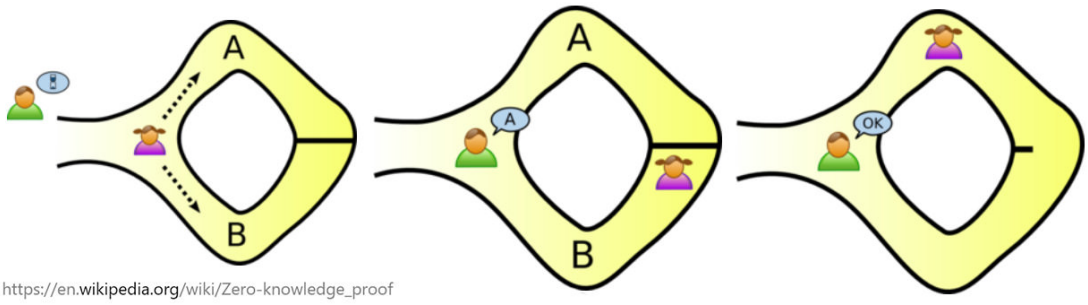

Figure 1: Ali Baba cave story

The figure 1 shows the layout of this cave. Peggy, wearing a pink shirt in the figure 1, knows the puzzle-lock secret and she, as prover, wants to demonstrate to the verifier Victor, who is wearing the green shirt, that she knows the code that unlocks the door but without revealing the secret code itself. To proceed with this proof, they engage in the following protocol:

- Peggy enters the cave from one of the two tunnels to the ring, A or B, while Victor waits outside.
- Then, Victor goes to the cave entrance and shouts to Peggy, asking her to leave the cave from either side A or B. Victor must choose his exit request randomly.
- If Peggy is on side B and Victor requested exit through side A, she can open the door and leave the cave from the right side. Otherwise, she can just leave the cave from the same side B. The point is that if Peggy knows the door secret, she should always be able to satisfy Victor's exit request regardless of what Victor chooses.

Because there is a 50% probability that Victor will request Peggy to exit from the same side that she entered from, Peggy may get lucky and satisfy Victor's requests without actually knowing the secret code. However, if request for exit side is truly random, and the protocol is repeated enough times, the probability of Peggy satisfying all of Victor's requests without knowing the secret becomes vanishingly small.

4.2 Definition

In [10], zero-knowledge proof is defined by a protocol involving two parties, prover and verifier, in which prover convinces the verifier that a statement of the form $u \in \mathcal{L}$ holds true where $\mathcal{L}$ is a language in NP. A witness w is a piece

of information that allows one to efficiently verify that that the statement $u \in \mathcal{L}$ is true. The protocol must satisfy three properties:

- **Completeness**: A prover holding a witness w to $u \in \mathcal{L}$ can convince the verifier.
- **Soundness**: A cheating prover P^* cannot convince the verifier when $u \notin \mathcal{L}$ except with some small probability.
- **Zero-knowledge**: The interaction only shows that statement $u \in \mathcal{L}$ is true. It reveals nothing else, in particular it does not disclose anything of the witness w.

4.3 Interactive Proofs

Figure 2 shows the typical structure of interactive zero-knowledge proofs. This structure, also known as Sigma protocol, comprises of three communication steps between prover and verifier:

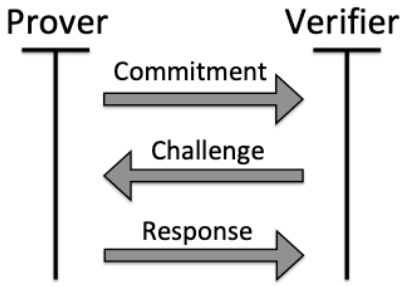

Figure 2: ZKPs - Sigma protocol

1. **Commitment**: Prover commits to a particular value and transfers the commitment to the verifier. In the cave story, this step is equivalent to Peggy choosing one of the two sides to enter the cave, and letting verifier know once she has entered the cave.
2. **Challenge** Verifier sends a random challenge to the prover. This step is equivalent to Victor challenging Peggy to exit from one of two sides chosen at random.
3. **Response** Prover computes a response based on the challenge and witness, and sends it to verifier for the task of verification. This step corresponds to Peggy leaving the cave from the side requested by Victor given her secret knowledge. Victor, waiting at the mouth of the cave, can check whether Peggy returns from the requested side.

The interactive ZKP protocols are rarely used in practice due to the restrictions they impose on the proof system. First, the interactive protocol often assumes that prover is very powerful with unbounded computational capacity since prover may have to conduct exponentially large number of rounds to convince an honest verifier that the proof is valid. Second, it is a synchronous protocol meaning that prover and verifier need to interact with each other in real time to execute the protocol. It goes without saying that such restrictions limit the suitability of these proofs for most applications which require concurrency and non-interactivity.

4.4 Non-Interactive Proof

In several settings, it is necessary to have an offline verification process. For example, in cryptocurrency, zero-knowledge proofs can be useful in validating integrity of a transaction stored in the blockchain while protecting the participant's information. This validation needs to be performed by several validators in the blockchain. It is not practical to make transaction participants interact with all the validators. Moreover, they will most likely not be available to interact with each other at the same time. Therefore, a mechanism to allow a verification process that does not depend on interaction between prover and verifier is very useful in this example as well as many other applications of ZKPs.

Fortunately, there is a very simple and powerful construction to transform a interactive proof into a non-interactive proof under some assumptions. Fiat-Shamir heuristic [11] takes an interactive proof and creates a non-interactive version provided that the original proof is public coin. This assumption basically states that the challenges made by the verifier on the interactive protocol are public. The central idea of this protocol is to use a hash function on the commitment aiming to generate a random and unpredictable challenge from the perspective of the prover. This way, even though the prover generates an entire transcript of the protocol without interacting with verifier, the prover cannot cheat as the

output of the hash function is beyond his control. Figure 3 presents the structure of non-interactive protocol where challenge is replaced by hash of the previous commitments in the protocol. Pointcheval and Stern [12] proved that Fiat-Shamir protocol is secure against chosen message attack as long as the hash function used behaves like a random oracle. The resulting Fiat-Shamir transcript is a digital signature of the proof and it can be verified multiple times and at any time the verifier wants to check the proof.

Figure 3: Zero-knowledge proof - Fiat Shamir heuristic

4.5 Implementations of Non-Interactive Proofs

There are several implementations of non-interactive zero-knowledge (NIZK) proofs; each implementation targets a specific goal and it has their pros and cons. One distinguishing feature among various implementations is related to whether they provide proof protocol for a specific type of problem (specialized NIZK) or a more general class of problems (generic NIZK). Specialized constructions use a specific mathematical formulation to address the task in hand that leads to efficiency either in size or execution performance. An interesting example can be found in [13]. In this paper, the authors construct a zero-knowledge proof to prove if a number is inside a specific range. This construction is optimized to produce a small and fast proof to be executed as part of smart contracts in the Ethereum platform since blockchain ecosystem charges contracts based on their execution time and storage cost.

Generic statement proof systems allow us to prove seemingly different problem statements using the same framework. For example, proving of knowledge of secret input of a known function with a known output is generic in the sense that any known function with known output can be used. In [14], it is described how proofs for such statements can be constructed using low degree polynomials such that verifier can succinctly check the computational integrity statements. This construction, called Fast Reed-Solomon Interactive Oracle Proofs of Proximity, is beyond the scope of this report and it can be found in [15]. Nonetheless, the technique requires provers to construct the polynomials according to the formulation preventing them from cheating i.e it requires prover to be honest. This problem is addressed in two main approaches. The first approach uses cryptographic constructions to hide the challenge in which the constructed polynomial will be tested. This is associated with the need of trusted setup, meaning that there must be a previously trusted source of randomness that needs to be shared between the prover and verifier. The second approach requires additional algebra and extra committed proofs that enable the verifier to check the consistency of the proof system. The trade-off between these two solutions is between the size of the proof and the need for the trusted setup. In some settings, such as cryptocurrency systems, the requirement of a previous trusted setup generated by a third-party can be a relevant source of vulnerability.

The three perhaps most popular NIZK systems use the polynomial constructions discussed in [15] and aim to solve the problem mentioned above, using one of the two approaches. These systems are ZK-SNARK (Zero-Knowledge Succinct Non-interactive ARgument of Knowledge), ZK-STARK (Zero-Knowledge Scalable Transparent ARguments of Knowledge) and Bulletproofs. The table 1, extracted from [16], shows the characteristics of these three systems. Analyzing the data from this table, it is possible to see that ZK-SNARK and ZK-STARK are in two different poles when it comes to proof size. Basically, we need to trade off small proof size with the need of trusted setup. STARKS proof on the other hand does not require this inconvenient previous trusted setup and also is quantum resistant. However, this system generates the biggest proof size. ZK-STARK and ZK-SNARK proof systems have relatively similar prover and verification time in contrast with Bulletproofs which has the poorest performance. The biggest benefit of the Bulletproofs is the relatively small proof size as well as the fact that this proof system does not require trusted setup.

Table 1: Most popular zero-knowledge proof systems

	Trusted Setup	Proof Size	Prover Time	Verification Time
SNARK	Required	288 bytes	2.3s	10ms
STARK	Notrequired	45-200kB	1.6s	16ms
Bulletproofs	Not required	1.3kB	30s	1100ms

5 Bulletproofs

Bulletproof system [10, 8] was designed to tackle two major problems of previously existent NIZK systems. ZK-SNARKs require a trusted setup and ZK-STARKs have long size proofs and demand a high computation during the proof construction. None of these characteristics are desirable and in the cryptocurrency systems, they can be prohibitively expensive. Bulletproof solves the problem of trusted setup requirement with reasonable trade off on size. In terms of computational requirements, this proof system is outperformed by ZK-SNARKs and ZK-STARKs, but still practical to be used in cryptocurrency systems. For these characteristics as well the fact that Bulletproof is well-suited for proving statements on committed values, we decided to use Bulletproof protocols in our project.

This system is zero-knowledge argument of knowledge, meaning that a prover is able to convince a verifier that a statement holds without revealing any information about the inputs of the statement. It is assumed that the prover is computationally bounded which limits his ability to forge proofs. In addition, the argument of knowledge needs to be public coin, meaning that the challenges sent by verifier are independent of prover's messages and correspond to the randomness generated independently by the verifier. This last assumption allows the interactive proof design to be transformed into a non-interactive proof system, NIZK, by using Fiat-Shamir heuristic. Finally, Bulletproofs rely on Pedersen commitments to hide the secret inputs and provide computational integrity check.

5.1 Inner Product Argument

The most important building block of Bulletproof is its efficient algorithm to calculate an inner-product argument for two independent binding vector Pedersen Commitments. The argument is an argument of knowledge that the prover knows openings $\mathbf{a}, \mathbf{b} \in \mathbb{Z}_p^n$ of binding Pedersen vector commitments that satisfy an inner product relation $c = \langle \mathbf{a,b} \rangle$ as shown below.

$$(\mathbf{g,h} \in \mathbb{G}^n, P \in \mathbb{G}, c \in \mathbb{Z}_p; \mathbf{a,b} \in \mathbb{Z}_p^n) : P = \mathbf{g}^{\mathbf{a}}.\mathbf{h}^{\mathbf{b}} \quad and \quad c = \langle \mathbf{a,b} \rangle$$

where $\mathbb{G}$ denotes cyclic group of prime order p, $\mathbb{Z}_p$ denotes a ring of integers modulo p and $\mathbb{G}^n$, $\mathbb{Z}_p^n$ denote vector spaces of dimension n over $\mathbb{G}$ and $\mathbb{Z}_p$ respectively.

Here, we give the simple overview of how this proof is constructed. Given the initial relation, the algorithm uses the homomorphic properties of Pedersen commitments to derive an interactive proof system to prove the original inner product statement. In the commitment step, commitment vectors $\mathbf{g}^{\mathbf{a}}\mathbf{h}^{\mathbf{b}}$ are split in two other vectors L and R which are multiplied by the random values received from the verifier. After this computation, the modified vectors L and R are sent to the verifier, who is able to compute again the inner product using these changed vectors that are consistent with the original statement. Given homomorphic properties of the commitments, the verifier is able to conclude if the proof is complete and sound. Finally, this proof system can be converted into a non-interactive one by using Fiat-Shamir heuristic. The details of this proof system are beyond the scope of this report and can be found in [10, 8].

5.2 Range Proofs

Bulletproofs have a construction to handle range proof, which is a proof system that allows prover to convince verifier that a secret committed value is inside a particular range. In fact, the formulation presented in [8], allows proofs for the range $[0, 2^n - 1]$. The relation presented on the previous sub-section is modified to the equation 1.

$$\{(g, h \in \mathbb{G}, V, n; v, \gamma \in \mathbb{Z}_p) : V = g^v.h^\gamma \wedge v \in [0, 2^n - 1]\} \tag{1}$$

This construction uses a bit representation of the value being proved to be in the range. Let $a_L = (a_1, a_2, ..., a_n) \in \{0, 1\}^n$ to be the binary representation of v, the value we want to prove to be in the range $[0, 2^n - 1]$. The equations 2, ensures that a_L is the bit representation of v. In the first equation each non-zero bit is multiplied by its correspondent 2 power to generate the value v. The second equation generates the vector a_R which contains -1's on the correspondent 0 bits of a_L and 0's on the correspondents 1's, assuming that a_L has only 0's and 1's. Finally, the third equation just check that a_L is composed by 0's and 1's and a_R is composed by -1's and 0's. These three set of equations constraint a_L to be the bit representation of v using only multiplication and additions. This is necessary to fit in the inner product formulation.

$$\langle \mathbf{a}_L, \mathbf{2}^n \rangle = v \quad and \quad \mathbf{a}_R = \mathbf{a}_L - \mathbf{1}^n \quad and \quad \mathbf{a}_L \circ \mathbf{a}_R = \mathbf{0}^n \tag{2}$$

Using this algebraic construction, it is possible to derive another set of equations that uses a similar construction of the inner product proof presented before. The details of this formulation goes beyond the scope of this paper and can be found in [8]. With this formulation is possible to prove that a number v is inside a range $[0, 2^n - 1]$.

5.3 Arithmetic Circuits

Bulletproofs also offer a formulation to prove the satisfiability of arithmetic circuits. Those are expressed by set of addition and multiplication operations organized like a electronic digital circuit. Even though only these two operations are allowed, this representation is powerful and expressive enabling the formulation of quite general statements. The reason why only addition and multiplication are allowed is due to the fact that addition can be easily computed using the homomorphic properties of Pedersen commitments and multiplication can be manipulated to generate a inner product. Using these facts, arithmetic circuit satisfiability can be proved using the constructions presented in the subsections before and discussed in details in [8].

6 Code Implementation

To build the prototype of this project, we experimented with two Bulletproofs implementations. The first one is called Hyrax [17] which is actually a doubly-efficient zk-SNARK implementation that contains code for Bulletproofs as well. This code base was developed and is maintained by Riad S. Wahby from Stanford university. The other implementation is called BulletproofLib [18] and was developed by Benedikt Bünz who is one of the main authors of Bulletproofs paper [8].

For the sake of transparency, buying auctioneer aims to prove to participating bidders that he chose the winning bid fairly i.e the winning bid is the lowest bid among all bids and the winning bid is chosen from one of the committed bids in the bidding phase. For convenience, we restate below the workflow in reverse auction setting.

- **Setup phase**:Auctioneer announces tender request for a service in a smart contract
- **Bidding phase**: Every bidder sends commitment for the bid offer and the commitment is publicly available
- **Revealing phase**: Auctioneer opens all bid commitments and picks the winning bid as the bid with lowest offer price. Auctioneer then announces the winning bid commitment and does not reveal the winning bid offer publicly.
- **Proof phase**: Let l denote a particular losing bid value, and w the winning bid value, then $d = l - w$. If $l > w$, then $d > 0$. If n denotes the bit size of a maximum possible bid value, then the maximum possible difference between any bid offers is also given by n bits. Auctioneer sets up a zero-knowledge range proof to show that commitment $C_d = C_l - C_w$ is opened by d which lies in the range $[0, 2^n - 1]$ without revealing d. The difference between any two arbitrary bids lies in the interval $[-(2^n - 1), 2^n - 1]$. If the group size p is large enough, at least twice the length of the range interval $[0, 2^n - 1]$, then negative values of d would lie outside the range interval since small negative values modulus large p would lie close to p. Auctioneer sends a relevant range proof transcript to each verifier where the transcript aims to convince the verifier of the following relation:

  $$(g, h \in \mathbb{G}, C, n; d, \gamma \in \mathbb{Z}_p) : C = d.g + \gamma.h \wedge d \in [0, 2^n - 1]$$

  assuming the commitment scheme is Pedersen under Elliptic curve field.
- **Verification phase** Each losing bidder, as an independent verifier, first computes C_d based on the publicly available commitment of the winner and their own commitment to ensure that they received the correct transcript, and then verifies the transcript received from the prover.

6.1 Hyrax

We started with the proof of concept development using Hyrax. This implementation does not have an implementation for range proofs but it does have an implementation for arithmetic circuits. Therefore, we coded an arithmetic circuit that output the most significant bit of a subtraction between two numbers. Assuming a signed bit representation for the values of a losing and winner bit, we were able to determine that one value is greater than other, by checking the signed bit of the difference. Hence, we input two values, the losing bid value and the winner bid value, into this arithmetic circuit to generate the transcript to be sent to the correspondent losing bidder. This would allow the bidder to verify the correctness of the workflow performed by the auctioneer. However, we realize that HyraxZK implementation generates the commitments automatically at the time that prover generates the proof. Conversely, in our workflow,

we need touse and provide the pre-committed values, the ones used during the bid phase, as part of the proof in order to check correctness of the auctioneer procedure. To address this issue, it would be necessary to change the Hyrax code so that we could use pre-committed values on the proof generation. However, this code modification was beyond our implementation capacity since there was not much documentation available. As result, we decided to use another Bulletproof implementation.

6.2 BulletproofLib

BulletproofLib is a Java implementation of all the protocols covered in the Bulletproofs paper [8] written by the author of the paper named Buenz. This implementation makes use of other Java cryptography APIs such as BouncyCastle for use of elliptic curve and group element primitives. We focused on the range proof implementation and tried our auction proof design with this code. Since this code was not production ready, there were some minor bugs that we had to fix, and because there was no executable main file either, we first created a main file in which we called the Prover and Verifier interface methods to setup the proof. To briefly summarize, we first create public parameters using a parameter generation method where we pass in the interval bit size n, as well as the specific type of elliptic curve (p is a 256 bit prime, so it is quite large). Next, we generate two random blinding factors, one for each of the two bid values, and create the two commitments, one for losing bid and one for winning bid. Next, we call the prover who takes in the parameters, difference committed value, and the witness (stores the the two bid values, and their corresponding randomness) and generates a proof transcript in a non-interactive manner. Next, the verifier takes the parameters, difference committed value, and the proof transcript. We checked that verifier accepts whenever the difference is positive, and rejects whenever the difference is negative. Thus, it seems our zero-knowledge design for auction scenario is feasible. The code for my Java main file is attached in the appendix at the end.

7 Future work

One important step of the reverse auction workflow is the bid commitment phase. Our proposed setup assumes that every bidder knows all bid commitments from all other bidders. This previous knowledge is necessary to make the proof consistent. The losing bidders receive proofs that the difference of their bid values and the winning bid value is positive. Hence, they need to confirm that this winning value is part of the original commitments, otherwise they would not be confident that the auctioneer did not create a fake commitment to benefit somebody else. As a result, all bidders must know all other commitments. However, this leads to another problem, which is related to the malleability of Pedersen commitments, which are used in the Bulletproofs. One bidder could generate a lower value commitment based on previous commitments already sent. Therefore a dishonest bidder could exploit opportunity to take advantage in the process. However, this bidder will not be able to open this commitment since he does not know the original value neither the hiding factor of the commitment, assuming that the openings are not public. Hence, it seems reasonable to leave all commitments public. It is still necessary though to have a setup that allows all participant bidders to track which commitments are part of the auction. One possible solution could be the usage of smart contract and blockchain to guarantee transparency and immutability of this information. For example, a smart contract could collect all bid commitments and store them all in the blockchain. This way, everybody could check how this smart contract was implemented and trace all bid commitments into the blockchain. This should provide trust to all bidders that nobody could create a fake commitment to take advantage in the auction. Although this approach seems to be reasonable to solve this problem, it should be implemented and tested to check its robustness and consistency. For sure, this topic should be an area of future work.

8 Conclusion

This project allowed us to gain an insight into an exciting albeit complex field of zero-knowledge cryptography. We delved into the implementations of zero-knowledge proofs and designed a proof system to generate transparency alongside privacy in online auctions. This cryptographic construction is very fascinating as it enables us to put together the two contrasting objectives of privacy and transparency. As mentioned in the earlier sections, transparency in public reverse auctions is a big concern and addressing it properly can bring several benefits to the society. We have seen that there are several zero-knowledge proofs cryptographic constructions that can be used in this problem. We decided to use Bulletproof construction, which represents a good trade-off between the security assumption and performance of the proof system. The experiments that we ran gave us several insights about how this system should work in practice but as we discussed in the previous section, there are other aspects that need to be considered from the design standpoint.

The prototype construction was very important as a thought experiment. Our solution design changed significantly as we worked on the proof of concept because we had substantial practical feedback from the available implementations

and testing environment. Some concepts such as usage of public commitments for the bids were envisioned after we put together the prototype. Therefore, the development of such proof concepts is crucial to achieve a solid solution design.

Finally, zero-knowledge proof cryptography has become a hot research area in the wake of advances in cryptocurrency and blockchain technology. However other application domains, including auctions, could also benefit from this very useful technology. The need for transparency is more relevant than ever before due to massive digitization of organizational workflows, especially in the public sector and zero-knowledge proofs could be a cryptographic tool to generate confidence among the players and stake holders involved. However, the current implementations for zero-knowledge proofs are still experimental and not very user or developer friendly. The lack of production ready tools is an obstacle for broad adoption in the near future but it can be overcome as people continue to work on it.

Appendix

```
package edu.stanford.cs.crypto.efficientct.rangeproof;

import java.math.BigInteger;
import edu.stanford.cs.crypto.efficientct.GeneratorParams;
import edu.stanford.cs.crypto.efficientct.VerificationFailedException;
import edu.stanford.cs.crypto.efficientct.algebra.BouncyCastleECPoint;
import edu.stanford.cs.crypto.efficientct.algebra.COCOGroup;
import edu.stanford.cs.crypto.efficientct.algebra.GroupElement;
import edu.stanford.cs.crypto.efficientct.commitments.PeddersenCommitment;
import edu.stanford.cs.crypto.efficientct.util.ProofUtils;

public class Main {

   public static void main(String[] args) throws VerificationFailedException {
      COCOGroup curve = new COCOGroup();

      int n = 16;
      GeneratorParams parameters = GeneratorParams.generateParams(n,curve);

        BigInteger x1 = BigInteger.valueOf(10);
      BigInteger x2 = BigInteger.valueOf(20);
      BigInteger x_diff = x2.subtract(x1);

      BigInteger r1 = ProofUtils.randomNumber();
      BigInteger r2 = ProofUtils.randomNumber();
      BigInteger r_diff = r2.subtract(r1);


      GroupElement v1 = parameters.getBase().commit(x1, r1);
      GroupElement v2 = parameters.getBase().commit(x2, r2);


      PeddersenCommitment<?> witness1 = new PeddersenCommitment<>(parameters.getBase(),x1, r1);
      PeddersenCommitment<?> witness2 = new PeddersenCommitment<>(parameters.getBase(),x2, r2);

      PeddersenCommitment<?> witness = new PeddersenCommitment<>(parameters.getBase(), x_diff,
          r_diff);

      GroupElement diff = parameters.getBase().commit(x_diff, r_diff);
      GroupElement other = v2.subtract(v1);
      System.out.println( diff.equals(other) );

      RangeProofProver prover = new RangeProofProver();
      RangeProof proof = prover.generateProof(parameters, diff, witness);

      RangeProofVerifier verifier = new RangeProofVerifier();
      verifier.verify(parameters, other, proof);

   }
}
```

References

[1] Colin Boyd and Wenbo Mao. *Security issues for electronic auctions*. Hewlett-Packard Laboratories Bristol (UK), 2000.

[2] Michael Harkavy, J Doug Tygar, and Hiroaki Kikuchi. Electronic auctions with private bids. In *USENIX Workshop on Electronic Commerce*, 1998.

[3] Michael P Wellman and Peter R Wurman. Real time issues for internet auctions. In *IEEE Workshop on Dependable and Real-Time E-Commerce Systems*. Citeseer, 1998.

[4] Kouichi Sakurai and S Miyazaki. A bulletinboard based digital auction scheme with bidding down strategy. In *Proc. 1999 International Workshop on Cryptographic Techniques and E-Commurce*, pages 180–187, 1999.

[5] Kapali Viswanathan, Colin Boyd, and Ed Dawson. A three phased schema for sealed bid auction system design. In *Australasian Conference on Information Security and Privacy*, pages 412–426. Springer, 2000.

[6] Matthew K Franklin and Michael K Reiter. The design and implementation of a secure auction service. *IEEE Transactions on Software Engineering*, 22(5):302–312, 1996.

[7] Kazue Sako. An auction protocol which hides bids of losers. In *International Workshop on Public Key Cryptography*, pages 422–432. Springer, 2000.

[8] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Gregory Maxwell. Bulletproofs: Short proofs for confidential transactions and more. *2018 IEEE Symposium on Security and Privacy (SP)*, pages 315–334, 2017.

[9] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, February 1989.

[10] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In *EUROCRYPT*, 2016.

[11] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, 1986.

[12] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In Ueli Maurer, editor, *Advances in Cryptology — EUROCRYPT '96*, pages 387–398, Berlin, Heidelberg, 1996. Springer Berlin Heidelberg.

[13] Tommy Koens, Coen Ramaekers, and Cees van Wijk. Efficient zero-knowledge range proofs in ethereum. ING, blockchain@ing.com.

[14] Eli Ben Sasson. Starks i - arithmetization eli ben sasson technion cyber computer security. `https://www.youtube.com/watch?v=9VuZvdxFZQo`, Nov 2017.

[15] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Fast reed-solomon interactive oracle proofs of proximity. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:134, 2017.

[16] Elena Nadolinski. Demystifying zero-knowledge proofs. `https://docs.google.com/presentation/d/1gfB6WZMvM9mmDKofFibIgsyYShdf0RV_Y8TLz3k1Ls0/edit#slide=id.p`.

[17] Riad S. Wahby. Hyrax reference implementation. `https://github.com/hyraxZK/hyraxZK`.

[18] Benedikt Bünz. Bulletprooflib. `https://github.com/bbuenz/BulletProofLib`.