Secure Auctions using Multi Party Computation

Abhishek Khandelwal (220040)
Indian Institute of Technology Kanpur
Kanpur, Uttar Pradesh, India
abhishekkh22@iitk.ac.in

Pallav Goyal (220747)
Indian Institute of Technology Kanpur
Kanpur, Uttar Pradesh, India
pallavg22@iitk.ac.in

Abstract

This report explores secure auction protocols that utilize multi-party computation (MPC) to enable privacy-preserving bidding without relying on a trusted auctioneer. By leveraging cryptographic techniques, such as secret sharing, commitment schemes, and homomorphic encryption, these protocols ensure bid confidentiality, integrity, and fair winner determination. We examine two primary research papers—SEAL and an array of "bidder resolved" protocols that extend MPC techniques for decentralized auction design. We examine two primary research papers—SEAL and an array of protocols that extend MPC techniques for decentralized auction design. Through an in-depth analysis and implementation of one selected protocol, we illustrate the conceptual feasibility of MPC-based auctions, emphasizing their potential to enhance privacy and trust in competitive bidding systems.

1 Introduction

In the modern world, auctions play a vital role in facilitating the exchange of goods and services. They are widely used across diverse fields, including government procurement, financial markets, and e-commerce. Among various auction types, *sealed-bid auctions* are particularly valued for their efficiency and ability to maintain privacy. In a sealed-bid auction, each participant submits a private bid, and the highest bidder wins without revealing losing bids. However, the presence of a trusted auctioneer in traditional sealed-bid auctions introduces vulnerabilities. A dishonest auctioneer could manipulate results, disclose losing bids, or otherwise compromise the auction's integrity and fairness.

A popular variation of the sealed-bid format is the *Vickrey auction*, where the highest bidder wins but pays the amount of the second-highest bid. Vickrey auctions encourage truthful bidding by ensuring that each participant has an incentive to bid their true valuation. Yet, despite their desirable properties, both traditional sealed-bid and Vickrey auctions are susceptible to the same risks associated with relying on a central, trusted auctioneer.

To address these concerns, research in secure auctions aims to eliminate the need for a central auctioneer and replace it with decentralized mechanisms that ensure privacy, fairness, and transparency. This shift from conventional to secure and private auction protocols mitigates trust issues and improves the reliability of auction systems.

In this report, we focus on two research papers that tackle the problem of secure auctions without a centralized auctioneer. The first paper, *SEAL: Sealed-Bid Auction Without Auctioneers* by S. Bag, F. Hao, S. F. Shahandashti, and I. G. Ray, introduces a decentralized protocol that guarantees public verifiability and removes the need for secret communication channels among bidders. The second paper, *Secure and Private Auctions Without Auctioneers* by F. Brandt, explores a similar goal but introduces a bidder-resolved auction protocol, leveraging cryptographic techniques to uphold privacy and fairness. We examined the protocols presented in both papers and implemented one of the protocols from the second paper. Specifically, we implemented and analyzed the B-share protocol, focusing on both implementation details and experimental findings.

2 Background

In secure auctions, cryptographic primitives play a crucial role in maintaining bid confidentiality, ensuring integrity, and preventing collusion among participants. The protocols discussed in this report utilize a combination of cryptographic methods to achieve privacy and decentralized auction functionality. The following cryptographic primitives form the foundation of these protocols:

2.1 Secret Sharing

Secret sharing allows a secret (in this case, the bid) to be divided into multiple shares and distributed across several participants. The key property of secret sharing is that no single participant can reconstruct the secret without the cooperation of others. In the threshold scheme, for example, a secret S is split into n shares, and any t shares can reconstruct the secret S. Mathematically, this can be expressed as:

$$S = \text{Share}_1 + \text{Share}_2 + \cdots + \text{Share}_t \mod p$$

where p is a large prime, and the shares are generated in such a way that no individual share reveals any information about the secret.

This technique ensures that no bidder can reconstruct the value of their bid independently, preventing information leakage. In the implemented protocol, each bidder generates additive shares of their bid b and distributes them among the participants, enabling secure collaborative computation without exposing individual bid values.

2.2 Commitment Schemes

Commitment schemes ensure that participants cannot alter their bids after submission. A common scheme involves two phases: the commitment phase, where a cryptographic hash $H(b)$ of the bid is computed, and the reveal phase, where the original bid b is disclosed and verified. The commitment is computed as follows:

$$C = H(b, r)$$

where r is a random value (the *blinding factor*) used to ensure that the commitment C cannot be predicted before the bid is revealed. This ensures that the bid cannot be changed once committed, thus preventing manipulations.

In a secure auction, the integrity of the commitment is critical, especially in public verifiability settings where it allows for consistency checks across all bids.

2.3 Homomorphic Encryption

Homomorphic encryption allows operations to be performed on encrypted data, yielding encrypted results that, when decrypted, are equivalent to the result of operations performed on the plaintext. Specifically, for a bid b, a homomorphic encryption scheme allows the encryption $E(b)$ to support operations like addition and multiplication:

$$E(b_1) \oplus E(b_2) = E(b_1 + b_2)$$
$$E(b_1) \otimes E(b_2) = E(b_1 \times b_2)$$

In secure auctions, while the protocols do not use full homomorphic encryption, they utilize partial techniques to compute outcomes such as the highest bid while preserving privacy. This ensures that the auction outcome can be determined without revealing the individual bids.

2.4 Bitwise Comparison Protocols

Bitwise comparison protocols compare two numbers bit by bit, starting from the most significant bit (MSB), to determine the largest value without revealing the actual numbers. For two bids b_1 and b_2, the bitwise comparison protocol proceeds as follows:

$$b_1 > b_2 \quad \text{if} \quad b_{1,i} > b_{2,i} \quad \text{for the first bit where} \quad b_{1,i} \neq b_{2,i}$$

This technique enables the determination of the highest bid while keeping the non-winning bids confidential. It is particularly useful in sealed-bid auction setups, where all bids are hidden, and only the winning bid needs to be revealed.

2.5 One-Way Functions

One-way functions are cryptographic primitives that are easy to compute in one direction but computationally hard to invert. In the context of commitment schemes, one-way functions are used to ensure that once a bid is committed, it cannot be revealed or tampered with before the reveal phase. Mathematically, if f is a one-way function:

$$f(x) \to y \text{ is easy to compute, but } f^{-1}(y) \text{ is computationally hard}$$

These functions are fundamental in maintaining the integrity of bid commitments, making it infeasible for a participant to alter their bid once it is committed.

2.6 Ring Transfer Protocol

The ring transfer protocol is a cryptographic technique that allows secure message passing among participants in a cyclic or sequential manner. This is often used in combination with one-way functions and random multipliers to facilitate secure computations in decentralized systems. In the auction context, a message m can be passed along a ring of participants such that only the intended recipient can decrypt the message. The transfer process can be formalized as:

$$m_i = f(m_{i-1}, \text{key}_i)$$

where m_{i-1} is the message from the previous participant, and key_i is a cryptographic key for participant i. This technique ensures the privacy of bids and the correctness of auction outcomes, as no participant can view the bids of others during the transfer.

2.7 Discrete Logarithm Problem

The discrete logarithm problem plays a key role in the security assumptions of some auction protocols. It asserts that given a group G and an element $g \in G$, it is computationally infeasible to determine the logarithm of an element y with respect to g, i.e., to solve for x in:

$$y = g^x \mod p$$

This problem provides the foundation for the security of various cryptographic operations, such as generating secure multipliers or bid shares. The hardness of the discrete logarithm problem ensures that even if an adversary has access to encrypted data, they cannot easily decipher the shared values, thus maintaining bid privacy.

2.8 Summary of Cryptographic Tools

Each of these cryptographic tools plays a pivotal role in supporting the privacy, security, and efficiency goals of secure auction protocols. By leveraging secret sharing, commitment schemes, homomorphic encryption, bitwise comparison protocols, one-way functions, ring transfer protocols, and the discrete logarithm problem, decentralized auction systems can be designed to protect against bid tampering, cheating, and information leakage while ensuring privacy and fairness.

3 Literature Survey

In advancing secure auction mechanisms, the elimination of a trusted auctioneer while maintaining bid privacy and integrity has become a pivotal research focus. This section reviews two prominent contributions in the field: the SEAL protocol by Bag et al., which proposes a decentralized approach for sealed-bid auctions, and Brandt's fully private protocol for secure auctions, which further enhances privacy by preventing bid information leakage even under potential collusion among participants.

3.1 SEAL Protocol: Decentralized Sealed-Bid Auction Without an Auctioneer

The SEAL protocol, developed by Bag et al. [1], introduces an auctioneer-free approach to sealed-bid auctions, enabling participants to securely determine the winning bid without a centralized authority. This protocol ensures public verifiability, privacy of losing bids, and security against collusion, while maintaining linear computation and communication complexity, $O(c)$, in terms of the bit-length c of the bids.

3.1.1 ***Commitment Phase :*** In the initial phase, each bidder V_i commits to their bid b_i by encoding it bitwise as $b_i = b_{i1}||b_{i2}||\ldots||b_{ic}$. For each bit b_{ij}, a cryptographic commitment is created as:

$$\epsilon_{ij} = \langle g^{\alpha_{ij}\beta_{ij}} g^{b_{ij}}, g^{\alpha_{ij}}, g^{\beta_{ij}} \rangle$$

where α_{ij} and β_{ij} are random values selected by V_i. This commitment scheme, based on the Decisional Diffie-Hellman (DDH) assumption, conceals individual bit values while ensuring verifiability.

3.1.2 ***Comparison Phase :***
The protocol determines the maximum bid bit-by-bit, starting from the most significant bit, by utilizing a modified Anonymous Veto protocol (AV-net) by Hao and Zieliński. In this phase, each bidder V_i privately sets a bit d_{ij} corresponding to each bit position j of their bid. Zero-knowledge proofs (ZKPs) are used to verify that $d_{ij} = b_{ij}$, without revealing actual bit values. A secure logical OR for each bit position j is computed as:

$$T_j = d_{1j} \vee d_{2j} \vee \cdots \vee d_{nj}$$

If $T_j = 1$ at any position, it signifies the *deciding position* (junction), indicating where bids diverge. After that all those bidders whose defeat is confirmed commits 0 bit value while other commits as usual.

3.1.3 ***Modified Veto Protocol:***
The SEAL protocol's AV-net modification is crucial for the secure, bitwise comparison of bids. Each bidder V_i selects random elements x_i and r_i and computes:

$$X_i = g^{x_i}, \quad R_i = g^{r_i}$$

along with ZKPs demonstrating knowledge of x_i and r_i.

Each voter V_i for $i \in [1, n]$ computes:

$$Y_i = \frac{\prod_{j=1}^{i-1} X_j}{\prod_{j=i+1}^{n} X_j}$$

Then, bidders publish encrypted ballots b_i for each bit of their bid, where:

$$b_i = \begin{cases} Y_i^{x_i} & \text{if } d_{ij} = 0 \\ R_i^{x_i} & \text{if } d_{ij} = 1 \end{cases}$$

This approach ensures privacy while keeping computations verifiable; $b_i \neq 1$ if at least one $d_{ij} = 1$.

3.1.4 ***Privacy and Verification:***
SEAL achieves public verifiability through ZKPs, allowing any participant or third party to validate the correctness of the auction. Privacy of losing bids is upheld by only revealing the highest bid, while ZKPs confirm each bidder's compliance without exposing individual bid values. The protocol ensures exclusive privacy, where colluding parties gain no knowledge about the bids other than the winning bid, unless they collude with the winner.

3.1.5 ***Efficiency:***
The protocol has a complexity of $O(c)$ exponentiations per bidder per bit, ensuring efficient computational and communication costs. Experimental results demonstrate scalability with larger bit lengths and more bidders, affirming its feasibility for extensive applications.

3.1.6 ***Conclusion:***
In summary, SEAL represents a robust, secure, and efficient solution for decentralized auctions by balancing verifiability, privacy, and efficiency without a trusted auctioneer, making it suitable for diverse real-world auction scenarios.

3.2 Brandt's Protocol: Secure and Private Auctions Without Auctioneers

Felix Brandt's work [2] introduces a framework for secure and private auctions without a centralized auctioneer, where bids are processed directly by the bidders in a fully decentralized manner. The protocols developed by Brandt focus on full privacy, aiming to prevent any information leakage about bids, even if most participants collude. Three primary protocols are outlined: B-share, MB-share, and YMB-share, each designed for different auction types and levels of security.

3.2.1 ***B-share Protocol: First-Price Auction :***
The B-share protocol supports first-price auctions, where the highest bidder pays their bid amount. In this protocol, each bidder i divides their bid b_i into additive shares $b_{i1}, b_{i2}, \ldots, b_{in}$, such that:

$$b_i = \sum_{j=1}^{n} b_{ij} \pmod{p}$$

where p is a large prime modulus. Each share is distributed among the participants, who then collaboratively compute the maximum bid without disclosing individual values. The sum function

$$f(X_1, X_2, \ldots, X_n) = \sum_{i=1}^{n} X_i$$

is used to determine the winner. Although simple, B-share can detect and penalize participants who fail to comply, ensuring auction integrity.

3.2.2 ***MB-share Protocol: Enhanced Privacy for First-Price Auctions :***
The MB-share protocol improves upon B-share by adding a shared random multiplier M to mask the computation of the maximum bid, thus enhancing privacy. In this protocol, each participant computes:

$$f(X_1, X_2, \ldots, X_n) = g^{\sum_{i=1}^{n} X_i M} \pmod{p}$$

where g is a generator in a finite field. This ensures that individual bids remain private due to the discrete logarithm problem, which makes it computationally infeasible to deduce X_i values from the aggregated output. The MB-share protocol is robust against collusion but incurs a higher communication cost than B-share, making it more suitable for high-security auctions with fewer bidders.

Note: The authors' formula in the MB-share protocol involves taking the modulus within the sum in the exponent and then applying exponentiation, which can lead to ambiguity. Specifically, $(a^{(x+y) \bmod p}) \bmod p$ is not the same as $(a^x \cdot a^y) \bmod p$. The expression $(a^{(x+y) \bmod p}) \bmod p$ implies that the sum $x + y$ is first reduced modulo p before applying the exponentiation, whereas $(a^x \cdot a^y) \bmod p$ does not assume this intermediate reduction, which can result in different outcomes in modular arithmetic.

3.2.3 ***YMB-share Protocol: Fully Private Vickrey Auction :***
The YMB-share protocol is designed for second-price or Vickrey auctions, where the winner pays the amount of the second-highest bid. Each bidder generates two values Y and N for each bid, indicating their willingness to pay at each price level. A unique key

K_{ij} for each bidder i and price level j is jointly computed, allowing only the winning bidder to privately learn their obligation:

$$K_{ij} = f(N_1, \ldots, Y_i, \ldots, N_n)$$

The winning bidder is determined by comparing K_{ij} values without revealing other bids. This protocol achieves high privacy by masking each bid level, but it requires substantial computational resources due to the need for multiple computations and secure key generation.

3.2.4 ***Summary of Protocols :*** Brandt's three protocols provide flexible and progressively secure solutions for auction scenarios. B-share provides a foundational method for bid privacy in first-price auctions, while MB-share enhances security against collusion with random multipliers. YMB-share further extends this to Vickrey auctions, offering full privacy at the cost of increased computation. Together, these protocols form a comprehensive toolkit for implementing secure, decentralized auctions without a trusted auctioneer.

3.3 Comparison and Implications

Both SEAL and Brandt's protocols represent foundational contributions to secure auction frameworks. SEAL's efficiency and ease of implementation make it suitable for larger auctions, particularly where the threat model does not include collusion among bidders. In contrast, Brandt's protocol offers enhanced privacy and resistance to collusion, ideal for high-stakes environments where the auctioneer's presence could jeopardize bid confidentiality. However, its higher computational demands restrict its use to smaller auctions.

The methods employed by SEAL and Brandt underline the inherent trade-offs between computational complexity and privacy in auction design. SEAL prioritizes transparency and simplicity, while Brandt's work pushes the boundaries of privacy by ensuring that no bid information is leaked unless all bidders collude. Both protocols lay essential groundwork for further exploration in secure multiparty computation and privacy-preserving auction mechanisms.

4 Implementation Description

In this project, we implemented a secure auction system where multiple bidders can securely exchange their bids using socket communication. The solution leverages the Boost ASIO library for network communication, enabling secure data transfer between multiple bidder instances running on separate processes. The primary objective is to ensure that the auction process is efficient, concurrent, and secure. The key components of the implementation are as follows:

4.1 System Overview

- The auction system consists of multiple bidders, each represented as a separate process. The communication between these bidders is managed using TCP sockets via Boost ASIO.
- The goal is to securely distribute shares of random values among bidders, ensuring that no single bidder knows the complete value except through collaboration.
- After the initial share distribution, each bidder publishes some part of their data corresponding to their bid for a specific price.
  - This published data is carefully designed to reveal enough information to identify a winning bid, while keeping the actual bid values and bidder identities hidden from others.
  - Only the winning bidder can recognize that they have won based on the published data. Other bidders are unable to identify the winner.
  - This mechanism ensures the privacy of the bidding process while still enabling a fair determination of the winning bid.

4.2 Core Components

- **Bid Selection**: Each bidder selects a random bid index from the range $[0, \texttt{PRICES} - 1]$.
- **Share Distribution**:
  - Each bidder generates random shares for all other bidders such that the sum of all shares for a given price index matches a pre-determined value.
  - The bidder's own share is adjusted based on the sum of shares sent to other bidders to ensure consistency.
- **Inter-Bidder Communication**:
  - Each bidder starts a server on a unique port (`12345 + bidder_index`) to receive shares from other bidders.
  - Concurrently, it connects to other bidders' servers to send its shares. This is achieved using multiple threads to handle connections in parallel.
- **Data Aggregation**:
  - Each bidder receives shares from others and combines them to reconstruct the values, which are then compared to determine the auction result.
- **Winner Announcement**:
  - Once the shares are aggregated, each bidder makes their summations public and then through the public values each bidder computes whether he has won or not.

4.3 Implementation Details

- **Boost ASIO for Networking**:
  - The `boost::asio::ip::tcp` library is used for setting up TCP connections between bidders. Both synchronous and multithreaded approaches are employed to optimize communication.
- **Multi-threading**:
  - A separate thread is used for running the server (`start_server`) to accept incoming connections from other bidders.
  - Multiple threads are used for outbound connections (`connect_to_bidder`) to reduce latency when sending data to other bidders.
- **Random Number Generation**:
  - C++ standard library components like `std::mt19937` and `std::uniform_int_distribution` are used for generating random bids and shares.
- **Mutex Locking**:

  - To handle shared resources and avoid race conditions when updating global timing variables, mutex locks are used.
- **Time Measurement**:
  - Execution time is measured at various stages to analyze performance using high-resolution clocks, specifically the time taken for:
    1. **Random Share Selection** (`time_select`)
    2. **Inter-Bidder Communication**
- **File Storage**:
  - Experimental results such as time taken for selecting bids, reading data, and writing data are stored in separate files for analysis.
- **Data Visualization**:
  - Python scripts are used to parse the stored data files and generate graphs, providing visual insights into performance metrics such as computation time and communication overhead.

4.4 Experimental Results

The experiments were conducted on a multi-core system, varying two key parameters: k, the total number of discrete prices, and n, the number of bidders. Each bidder operates as a separate process. The experiment was divided into two parts:

1. Varying the total number of bid prices (k) while keeping the number of bidders (n) constant at 8.
2. Varying the number of bidders (n) while keeping the number of prices (k) constant at 1024.

- **Observations:**
  - When the number of bidders (n) is kept constant, there is a linear relationship between the computational cost and the number of discrete prices (k). This indicates that the computation scales linearly with the total number of prices.
  - Similarly, when the number of prices (k) is kept constant, the computational cost increases linearly with the number of bidders (n). Thus, the computation scales linearly with the number of participants.
  - The overall computational complexity for each bidder is $O(nk)$, which aligns with the observed experimental data.
  - The communication times remain largely unaffected when the number of bidders (n) is kept constant, regardless of changes in the number of prices (k).
  - However, when the number of prices (k) is fixed, the communication time exhibits a linear relationship with the number of bidders (n). This suggests that communication overhead scales linearly with the number of participants.
  - The communication complexity for each bidder is $O(n)$, as reflected in the recorded data points.
- **Analysis of Computational Cost**:
  - The computational complexity is $O(nk)$ because:
    * Each bidder is required to generate a random share for every other bidder.
    * For each bidder, this share generation needs to be repeated for every price, resulting in $n \times k$ computations.
  - Consequently, increasing either the number of bidders (n) or the number of prices (k) leads to a linear increase in computational cost, which aligns with the observed data.
- **Analysis of Communication Time**:
  - The communication time remains constant with respect to k because:
    * Each bidder sends a single vector of size k to every other bidder.
    * Increasing k only changes the size of the vector but does not affect the number of communication operations.
  - However, when varying n, the communication time increases linearly due to:
    * The current implementation reads and writes data sequentially, where each bidder interacts one at a time.
    * This results in a communication complexity of $O(n)$, as each bidder waits for its turn to communicate.
- **Future Optimizations for Parallel Communication**:
  - The sequential nature of the current implementation can be optimized by:
    * Performing all read operations from different bidders in parallel.
    * Similarly, conducting all write operations simultaneously.
  - This parallel approach would reduce communication complexity to $O(1)$ per bidder, making the communication time independent of the number of bidders (n) and significantly enhancing performance.

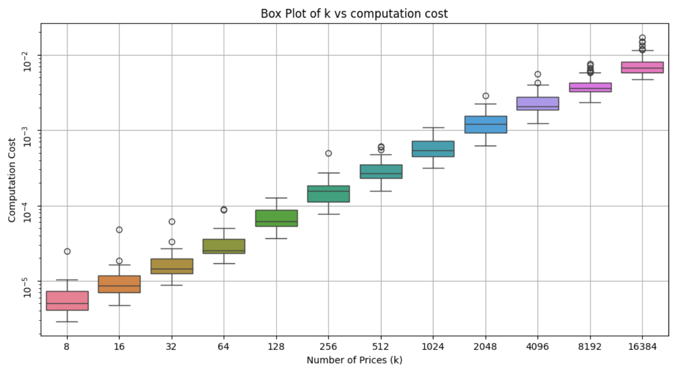

Figure 1: Computation Cost vs number of Prices

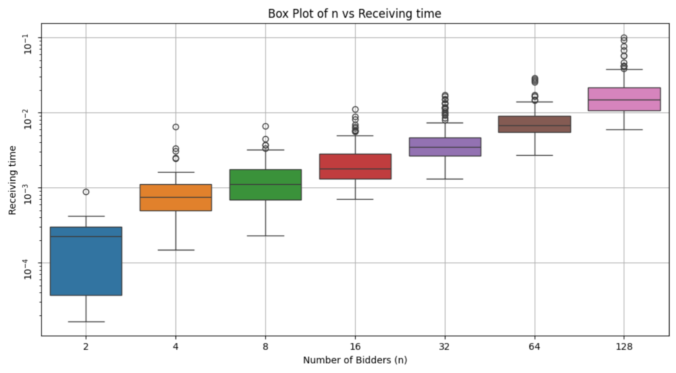

Figure 4: Receiving time vs number of Bidders

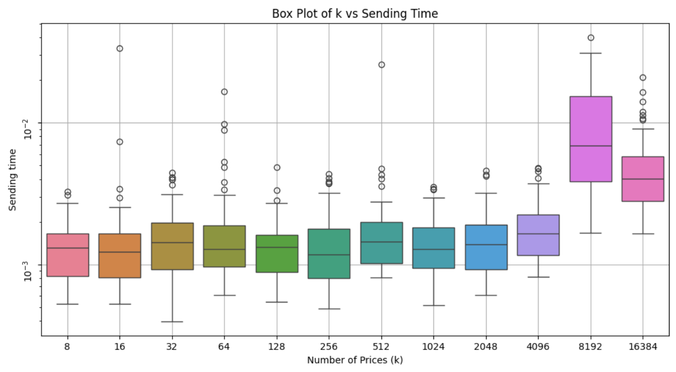

Figure 5: Sending time vs number of Prices

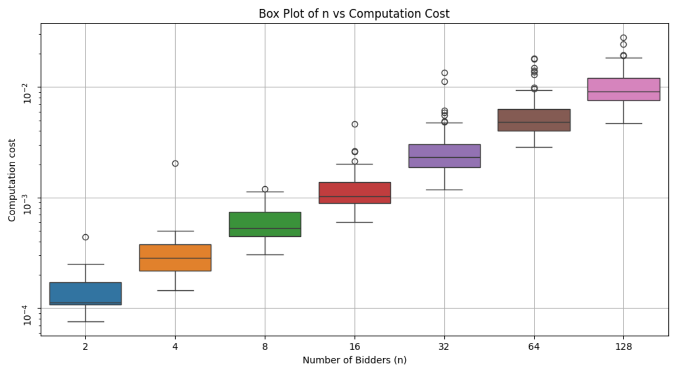

Figure 2: Computation Cost vs number of Bidders

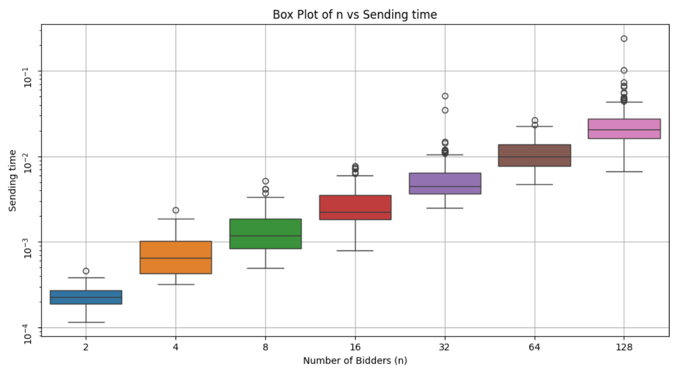

Figure 6: Sending time vs Number of Bidders

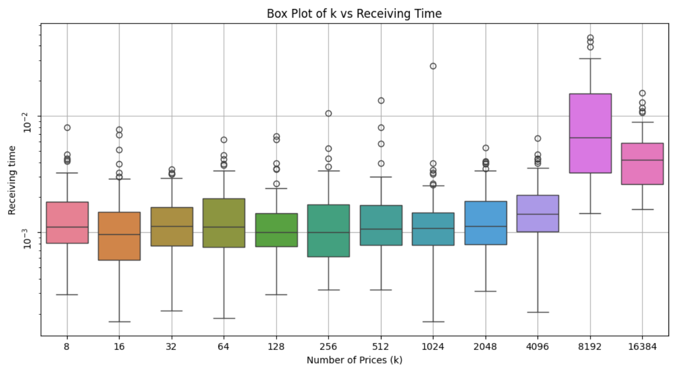

Figure 3: Receiving time vs number of Prices

Acknowledgments

We would like to express our sincere gratitude to our professor, Dr. Adithya Vadapalli, for his constant guidance and support throughout this project. His insightful suggestions and encouragement have been invaluable in helping us navigate the complexities of cryptographic protocols and secure auctions.

We also appreciate the resources and assistance provided by our department, which facilitated our research and the preparation of this report.

References

[1] Samiran Bag, Feng Hao, Siamak Shahandashti, and Indranil Ghosh Ray. Seal: Sealed-bid auction without auctioneers. *Information Forensics and Security, IEEE Transactions on*, PP, 11 2019.

[2] F. Brandt. Secure and private auctions without auctioneers, 2002.