A Lightweight Auction Framework for Spectrum Allocation with Strong Security Guarantees

Ke Cheng*†‡, Liangmin Wang§, Yulong Shen*†‡, Yangyang Liu*†‡, Yongzhi Wang¶, and Lele Zheng*†‡
*School of Computer Science and Technology, Xidian University, Xi'an, China
†Shaanxi Key Laboratory of Network and System Security, Xidian University, Xi'an, China
‡State Key Laboratory of Integrated Service Networks (ISN), Xidian University, Xi'an, China
§Department of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang, China
¶Department of Computer Science and Information Systems, Park University, Parkville, MO, USA
Email: kecheng@stu.xidian.edu.cn; wanglm@ujs.edu.cn; ylshen@mail.xidian.edu.cn;
liuyangyang@stu.xidian.edu.cn; yongzhiwang@icloud.com; llzheng@stu.xidian.edu.cn

***Abstract*—Auction is an effective mechanism to distribute spectrum resources. Although many privacy-preserving auction schemes for spectrum allocation have been proposed, none of them is able to perform practical spectrum auctions while ensuring enough security for bidders' private information, such as geo-locations, bid values, and data access patterns. To address this problem, we propose SLISA, a lightweight auction framework which enables an efficient spectrum allocation without revealing anything but the auction outcome, i.e., the winning bidders and their clearing prices. We present contributions on two fronts. First, as a foundation of our design, we adopt a *Shuffle-then-Compute* strategy to build a series of secure sub-protocols based on lightweight cryptographic primitives (e.g., additive secret sharing and basic garbled circuits). Second, we improve an advanced spectrum auction mechanism to make it *data-oblivious*, such that data access patterns can be hidden. Meanwhile, the modified protocols adapt to our elaborate building blocks without affecting its validity and security. We formally prove the security of all protocols under a semi-honest adversary model, and demonstrate performance improvements compared with state-of-the-art works through extensive experiments.**

I. Introduction

Dynamic spectrum allocation via auction has become a promising approach to improve efficiency in spectrum utilization. There have been extensive works on designing truthful spectrum auctions, in which the auctioneer is assumed to be fully trusted, and all bidders would submit their true bids unreservedly [1]–[4]. In addition, due to the scarcity of spectrum resources, the bidders are usually required to provide the geo-location data for spectrum reuse. However, it is impractical to trust the auctioneer and bidders unconditionally, as they have probably disrupted a normal auction process to gain illicit benefits. For example, a dishonest auctioneer can simply adapt its pricing strategy to obtain extra profit by monitoring the bidders' bids. Through learning the historical bids of others, a bidder can choose a bid untruthfully to make the maximum profit. Furthermore, an external attacker is able to detect the service area of the bidders by obtaining geo-location information (possibly leaked from the auctioneer). Therefore, it is significant to design a secure auction framework for spectrum allocation, in which one's geo-location and bid should not be disclosed to the auctioneer and other rival bidders.

There exist a number of works on designing spectrum auctions in a privacy-preserving manner [5]–[12]. However, these advanced works fail to provide strong security guarantees for the all above-mentioned private information of bidders. In [5], [6], though bid values are submitted in an encrypted form, the auctioneer can deduce the order of these bids by observing the data access patterns during the auction. The works in [7]–[9] focus on how to protect bid values with no regard for bidders' location privacy. In [10]–[12], the data access patterns can be used to infer the position relationships between all bidders. But besides that, the performance of these works is barely satisfactory in most realistic cases. For example, the schemes in [5], [10]–[12] involve a large number of homomorphic encryption primitives, which are too costly to make these schemes scalable to support large-scale bidders. The systems in [7]–[9] heavily rely on great-depth garbled circuits [13], causing an impractical computation and communication cost. In short, conducting an efficient spectrum auction with strong security guarantees remains a challenge.

Motivated by the above arguments, in this paper we propose a Secure and LIghtweight Spectrum Auction framework, named SLISA. Our goal is to carry out a fully secure spectrum auction efficiently without revealing any information about bidders' geo-locations and bid values. To this end, we employ lightweight cryptographic primitives, such as additive secret sharing and basic garbled circuits, to design a set of sub-protocols that support secure arithmetic operations, secure maximum selecting, and secure sorting. Based on these elaborate building blocks, we achieve all phases of the spectrum auction in a privacy-preserving manner, while ensuring the privacy of bidders' geo-locations and bid values. In addition, to avoid the leakage of potential location relations and bid ranking caused by snooping data access patterns, our design is engaged in a two-pronged effort. First, we advocate a *Shuffle-then-Compute* strategy, which randomly permutes inputs before feeding it to original algorithms, for building sub-protocols to prevent leakage from data access patterns. Second, we design a data-oblivious auction algorithm whose execution path is independent of inputs, i.e., geo-locations and bid values.

In our work, the major challenges originate from the conflict-

ing requirements of high efficiency and strong security. Secure sorting is a dominant operation in the secure auction. Most of the existing works [14], [15] on the secure-sorting incur considerable overheads, since these methods involve plenty of time-consuming encryptions/decryptions or large-scale circuit evaluations. Moreover, the process of secure spectrum auction is generally complicated, meaning that extensive intermediate results exist in many phases of the auction. Unfortunately, such intermediate information has the potential to reveal indirect information about the results (e.g., whether a bid value is higher than another one) even if the exact value is protected. This makes it challenging to conduct a fully secure auction in which no adversary with arbitrary computation power can infer additional information except for auction outcomes. We summarize our contributions as follows:

1) In combination with additive secret sharing and garbled circuits, we present a set of sub-protocols to support secure arithmetic operations, secure maximum selecting and secure sorting, which can be used as building blocks for secure spectrum auction as well as other kinds of auctions and applications, e.g., virtual machine auction [16], [17], advertising auction [18], service recommendation [19] and service composition [20], [21].
2) Based on the building blocks, we propose a lightweight auction framework (SLISA) for spectrum allocation with strong security guarantees for bidders' private information including geo-locations and bid values. Our framework can remarkably improve auction efficiency, which benefits from the shuffle-then-compute design strategy and the use of lightweight cryptographic primitives.
3) By using proof-by-simulation techniques, we formally prove that all the protocols in SLISA are secure under a semi-honest adversary model. We conduct extensive experiments to show performance improvements compared with state-of-the-art works.

The rest of the paper is organized as follows. Section II presents the related work. Section III introduces the system model and design goals. The building blocks that used in SLISA are introduced in Section IV. The details of secure spectrum auction are presented in Section V. The security analysis and experimental evaluations are shown in Section VI and Section VII. Finally, Section VIII concludes the paper.

II. Related Work

There are a large body of works on spectrum auction design in the past decade. Zhou et al. propose VERITAS [22], a truthful single-sided spectrum auction system to enforce users to bid their true valuations of the spectrum. Then the same authors design TRUST [23], the truthful double spectrum auction mechanism, which eliminated the selfish behaviors of both sellers and buyers. Based on TRUST, Yao et al. put forward a more socially efficient double spectrum auction mechanism TDSA [24], which has a virtual group bidding mechanism to improve spectrum utilization. Continuously, a number of more advanced spectrum auction mechanisms have been proposed for many different purposes, such as reverse spectrum auction [25], heterogeneous spectrum auction [26], and combinatorial auction [27]. However, these spectrum auction mechanisms only focus on the achievement of truthfulness, while ignoring security guarantees for bidders.

TABLE I
Comparison of our work with previous secure schemes

Properties	[5] [6]	[7]–[9]	[10]–[12]	SLISA
Bid value privacy	√	√	√	√
Geo-location privacy	×	×	√	√
Hide data access pattern	×	×	×	√
Double auctions	×	√	√	√
Truthfulness	√	√	√	√

Furthermore, secure spectrum auctions have been extensively studied in recent years. The works in [5], [6] provide privacy-preserving solutions for single-sided spectrum auctions. To solve security suffering in double spectrum auctions, Chen et al. utilize garbled circuits to construct two secure auction frameworks PS-TRUST [7] and ITSEC [8], which can protect bids privacy in double spectrum auctions. In order to improve the efficiency of double spectrum auctions, Chen et al. [9] and Wang et al. [10], [11] propose a series of auction schemes by building secure mixed protocols based on multiple cryptographic primitives. In addition, ARMOR [12] leverages homomorphic encryption, order-preserving encryption and garbled circuits to devise a privacy-preserving protocol for combinatorial auction mechanisms. However, none of these works provides strong protections for geo-locations, bid values and data access patterns. For clarity, we summarize the differences between our work and previous secure schemes in Table I.

III. Problem Statement

A. System Model

Our auction framework is shown in Fig. 1, consisting of the following four entities: sellers, buyers, an auctioneer ($\mathcal{A}$) and an auction agent ($\mathcal{B}$). The auctioneer and auction agent, who are semi-honest and non-colluding, cooperate to run a privacy-preserving auction mechanism. We note that such a two-party model has been widely adopted in the literature under various secure applications (e.g., [9], [10], [28], [29], to just list a few) and our adoption also follows this popular trend.

We consider a double auction for spectrum allocation, where M sellers want to sublease their homogeneous channels to N buyers. We use $s_m(1 \leq m \leq M)$ to denote the set of sellers, each of whom provides one channel for sale. Let $b_n(1 \leq n \leq N)$ denote the set of buyers, each of whom requests one channel. In the auction, each seller s_m submit the sell-bid V_m to the auctioneer. After that, each buyer b_n submit a request (B_n, x_n, y_n, r_n), in which B_n is a buy-bid for one channel, (x_n, y_n) is a location, and r_n is a conflict radius, to the auctioneer. Note that, spectrum can be allocated to multiple conflict-free buyers, which improves the spectrum utilization. Based on this information, the auctioneer matches the winning sellers and buyers, and determines their respective clearing prices. We summarize the notations throughout this paper in Table II.

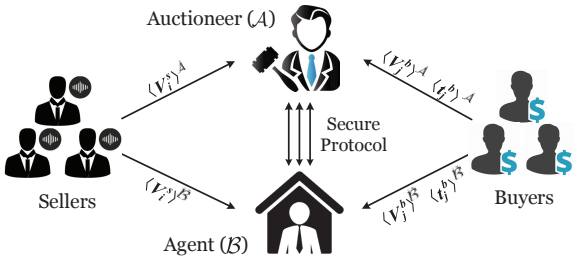


Fig. 1. System model.

B. Double Spectrum Auction Mechanism

We review a generic double spectrum auction mechanism TDSA [24]. Now we sketch its executable process as follows.

1) *Buyer Grouping*: The auctioneer first constructs a conflict graph of buyers based on the locations and interference ranges. Then, a conflict-free buyer group $G_t (t \in \{1, 2, \ldots, T\})$ subject to this conflict graph is formed by using a bid-independent grouping algorithm.

2) *Virtual Group Bidding*: Suppose each buyer group G_t has R_t members, then the virtual groups can be denoted as $VG_j, j \in \{1, 2, \ldots, R_t\}$ according to their size. Assume bids in group G_t are sorted as: $B_{t,1} \geq B_{t,2} \geq \ldots \geq B_{t,R_t}$. For virtual group VG_j, the virtual group bid is $B^v_{t,j} = B_{t,j} \times j$. Finally, the group bid GB_t for group G_t can be computed as $GB_t = \max_{j \in \{1,2,\ldots,R_t\}}(B^v_{t,j})$. Subsequently, each buyer group G_t will bid its group bid GB_t acting as a single "buyer".

3) *Preliminary Winner Determination*: The auctioneer first sorts the sell-bids $V_m (1 \leq m \leq M)$ in a non-descending order, and sort the group bids of buy group G_t in a non-ascending order as follows:

$$O' : V_1 \leq V_2 \leq \ldots \leq V_M$$
$$O'' : GB_1 \geq GB_2 \geq \ldots \geq GB_T$$

Then, the last profitable index can be calculated as follows:

$$K = \arg \max_{k \leq min(M,T)} (GB_k \geq V_k \text{ and } GB_k \neq GB_{k-1}). \quad (1)$$

The preliminary auction winners are first $k-1$ sellers in O' and first $k-1$ buyer groups in O''. The clearing price charged by the winning sellers is the K-th seller's bid, i.e., $P^s = V_k$. Similarly, the payment of the winning buyer group is the K-th buyer group's bid, i.e., $P^G = GB_K$.

4) *Washing Out*: After preliminary winners are selected, the auctioneer needs to remove buyers whose bids are too low to afford the payment. The remaining buyers are the final winners of the auction. Recall that the buyers' bids in group G_t is sorted in a non-ascending order, $B_{t,1} \geq B_{t,2} \geq \ldots \geq B_{t,R_t}$. The auctioneer begins to wash out buyers in G_t one by one from R_t to 1, until finding a virtual group $VG_{C_t} = B_{t,1}, B_{t,2}, \ldots, B_{t,C_t}$ whose virtual bid is larger than the price for winning buyer groups, i.e., $B^v_{t,C_t} = B_{t,C_t} \times C_t > P^G$.

5) *Final Pricing*: The auctioneer buys each winning seller's channel at the price of P^s, and sells a channel to each winning buyer group at the price of P^G. Moreover, in a wining group G_t, the clearing price for each buyer is $P^b_t = P^G / C_t$.

TABLE II
NOTATIONS AND DEFINITIONS

M, N, T	the numbers of sellers, buyers and buyer groups
V_m, B_n	the sell-bid and buy-bid
$(x_n, y_n), rd_n$	the location and conflict radius of buyer b_n
$\{G_t\}_{t=1}^T$	the set of non-conflict buyer groups
R_t	the number of the buyers in G_t
$B_{t,j}$	the buyer's bid in G_t, $1 \leq j \leq R_t$
$\{VG_j\}_{j=1}^{R_t}$	the virtual groups
$B^v_{t,j}$	the virtual group bid
GB_t	the group bid of group G_t
P^s, P^G, P^b_t	the clearing price for winning sellers, buyer groups and buyers
$\langle x \rangle$	the secret-shared form of the value x, $\langle x \rangle = (\langle x \rangle^{\mathcal{A}}, \langle x \rangle^{\mathcal{B}})$
$\langle x \rangle^{\mathcal{A}} / \langle x \rangle^{\mathcal{B}}$	the share of x stored in $\mathcal{A}$ or $\mathcal{B}$

C. Threat Model and Design Goals

In our content, we consider the security threats that flow largely from the auctioneer and the auction agent. That is, these two parties are considered as adversaries. Following most popular security works [8]–[12], [28], [29], we assume that our auction scheme is in the semi-honest and non-colluding model, such that the auctioneer and the agent would faithfully carry out the auction protocol while they attempt to infer private information besides the auction outcome.

Our aim is to design a secure and efficient auction scheme for spectrum allocation based on TDSA. To provide strong protection for both location privacy and bid privacy, $\mathcal{A}$ and $\mathcal{B}$ are required to know nothing about the buyers' location information (x_n, y_n, rd_n), buy-bids B_n, and sell-bids V_m. This would imply that all protocols involved in the auction scheme should achieve *information-theoretic security* [30]. In other words, the security of these protocols can be proven formally under a standard security model. In addition, facing plenty of auction users and real-time requirements, our auction scheme should be efficient with low computation and communication overheads. For that, we do not adopt some computation-intensive encryption mechanisms (e.g., homomorphic encryptions) in the online stage.

IV. Building Blocks: Secure Basic Operations on Secret-shared Data

Before working with secure auction specifics, we describe in detail the building blocks: first, secure arithmetic operations on secret-shared data, and second, secure maximum/sort protocol based on the alternating use of secret sharing and garbled circuit. These operations are implemented via secure interactions of two parties $\mathcal{A}$ and $\mathcal{B}$ described earlier in Section III-A.

A. Secure Arithmetic Operations on Secret-shared Data

For practical cost efficiency, we do not adopt homomorphic-encryption-based methods, while we resort to the lightweight cryptographic primitive – additive secret sharing [31], to achieve the encryption of bid information. Given an ℓ-bit value x, a random number $r \in \mathbb{Z}_{2^\ell}$ is generated to additively secret-share x as follows: $\langle x \rangle^{\mathcal{A}} = (x - r) \mod 2^\ell$ and $\langle x \rangle^{\mathcal{B}} = r \mod 2^\ell$, where $\langle x \rangle^{\mathcal{A}} / \langle x \rangle^{\mathcal{B}}$ is only held by party $\mathcal{A}/\mathcal{B}$. A shared value x is denoted as $\langle x \rangle$, i.e., $\langle x \rangle = (\langle x \rangle^{\mathcal{A}}, \langle x \rangle^{\mathcal{B}})$, meaning that each party obtains a secret share of x. To recover

(Rec$(\cdot,\cdot)$) the value x, party $\mathcal{A}$ ($\mathcal{B}$) sends $\langle x\rangle^{\mathcal{A}}$($\langle x\rangle^{\mathcal{B}}$) to party $\mathcal{B}$ ($\mathcal{A}$) who computes $x = \langle x\rangle^{\mathcal{A}} + \langle x\rangle^{\mathcal{B}}$.

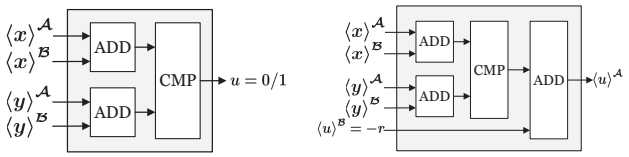
Secure addition and multiplication. To compute the sum of two shared values $\langle x\rangle$ and $\langle y\rangle$ without disclosing x and y to party $\mathcal{A}$ and $\mathcal{B}$, we let party $\alpha(\alpha \in \{\mathcal{A},\mathcal{B}\})$ locally computes $\langle x+y\rangle^{\alpha} = \langle x\rangle^{\alpha} + \langle y\rangle^{\alpha}$. To multiply a shared value $\langle x\rangle$ with a public constant c, $\langle c\cdot x\rangle = c\cdot\langle x\rangle$ can be computed as follows: party α locally computes $\langle c\cdot x\rangle^{\alpha} = c\cdot\langle x\rangle^{\alpha}$. In order to perform secure multiplication $\langle x\cdot y\rangle = \langle x\rangle\cdot\langle y\rangle$, we leverage precomputed multiplication triplets [32] of the form $\langle c\rangle = \langle a\rangle\cdot\langle b\rangle$, which are secret-shared among the two parties. Party α locally computes $\langle e\rangle^{\alpha} = \langle x\rangle^{\alpha} - \langle a\rangle^{\alpha}$ and $\langle f\rangle^{\alpha} = \langle y\rangle^{\alpha} - \langle b\rangle^{\alpha}$. Both parties run e=Rec($\langle e\rangle^{\mathcal{A}}$, $\langle e\rangle^{\mathcal{B}}$) and f=Rec($\langle f\rangle^{\mathcal{A}}$,$\langle f\rangle^{\mathcal{B}}$), then party $\mathcal{A}$ sets $\langle x\cdot y\rangle^{\mathcal{A}} = f\cdot\langle a\rangle^{\mathcal{A}} + e\cdot\langle b\rangle^{\mathcal{A}} + \langle c\rangle^{\mathcal{A}}$ and $\mathcal{B}$ sets $\langle x\cdot y\rangle^{\mathcal{B}} = e\cdot f + f\cdot\langle a\rangle^{\mathcal{B}} + e\cdot\langle b\rangle^{\mathcal{B}} + \langle c\rangle^{\mathcal{B}}$. Note that the secret shares of multiplication triplets can be distributed to two parties offline due to data-independent characteristics.

Vectorization in the secret-shared setting. Operating on the secret-shared vectors and matrices is critical to elevate the efficiency. In particular, given two shared matrices $\langle\mathbf{X}\rangle$ and $\langle\mathbf{Y}\rangle$, we can achieve matrix addition non-interactively as $\langle\mathbf{X}+\mathbf{Y}\rangle^{\alpha}$=$\langle\mathbf{X}\rangle^{\alpha} + \langle\mathbf{Y}\rangle^{\alpha}$ for $\alpha \in \{\mathcal{A},\mathcal{B}\}$. To multiply two shared matrices $\langle\mathbf{X}\rangle$ and $\langle\mathbf{Y}\rangle$, we leverage shared matrices $\langle\mathbf{U}\rangle, \langle\mathbf{V}\rangle, \langle\mathbf{Z}\rangle$, in which $\mathbf{U}$ has the same size as $\mathbf{X}$, $\mathbf{V}$ has the same size as $\mathbf{Y}$, each element in $\mathbf{U}$ and $\mathbf{V}$ is uniformly random in $\mathbb{Z}_{2^\ell}$, and $\mathbf{Z} = \mathbf{U}\cdot\mathbf{V} \mod 2^\ell$. Firstly, party α computes $\langle\mathbf{P}\rangle^{\alpha} = \langle\mathbf{X}\rangle^{\alpha} - \langle\mathbf{U}\rangle^{\alpha}$ and $\langle\mathbf{Q}\rangle^{\alpha} = \langle\mathbf{Y}\rangle^{\alpha} - \langle\mathbf{V}\rangle^{\alpha}$. Then, two parties jointly reconstruct $\mathbf{P}$ and $\mathbf{Q}$. Finally, $\mathcal{A}$ sets $\langle\mathbf{X}\cdot\mathbf{Y}\rangle^{\mathcal{A}} = \langle\mathbf{U}\rangle^{\mathcal{A}}\cdot\mathbf{Q} + \mathbf{P}\cdot\langle\mathbf{V}\rangle^{\mathcal{A}} + \langle\mathbf{Z}\rangle^{\mathcal{A}}$ and $\mathcal{B}$ sets $\langle\mathbf{X}\cdot\mathbf{Y}\rangle^{\mathcal{B}} = \mathbf{P}\cdot\mathbf{Q} + \langle\mathbf{U}\rangle^{\mathcal{B}}\cdot\mathbf{Q} + \mathbf{P}\cdot\langle\mathbf{V}\rangle^{\mathcal{B}} + \langle\mathbf{Z}\rangle^{\mathcal{B}}$. We have to note that, this protocol also supports to multiply a shared vector with a shared matrix, and works on real-field data. We refer the readers to [29] for the generation of the desired shared multiplication triplets and the proof of security of the vectorized extension.

B. Secure Comparison Protocol

To support secure comparison on secret-shared values, we leverage Yao's garbled circuits (GC) [13], which enables two parties to compute a function $f(x,y)$ on their respective inputs x and y without disclosing the inputs beyond what can be deduced from the function output. We list four garbled circuits used in this work. On the premise of not disclosing two input values x and y, 1) an ADD circuit outputs $z = x + y$, 2) a CMP circuit outputs 1 if $x \geq y$ and 0 otherwise, 3) a NEQ circuit outputs 1 if $x \neq y$ and 0 otherwise, and 4) a DIV circuit outputs an approximate result of x/y with a given precision.

On the basis of ADD and CMP circuits, we first customize a COMP circuit for securely computing a flag u that indicates whether x is larger than y ($u = 1$) or not ($u = 0$). Fig. 2(a) shows the structure of COMP circuit, where one ADD circuit takes $\langle x\rangle^{\mathcal{A}}$ and $\langle x\rangle^{\mathcal{B}}$ as inputs while the other takes $\langle y\rangle^{\mathcal{A}}$ and $\langle y\rangle^{\mathcal{B}}$ as inputs, then the two outputs serve as the inputs of the CMP circuit. Specifically, party $\mathcal{A}$ can serve as the circuit constructor while party $\mathcal{B}$ can serve as the circuit evaluator. $\mathcal{B}$ runs oblivious transfer (OT) protocol [33] with $\mathcal{A}$ to obliviously obtain the garbled input corresponding to its private input, then evaluates the circuit to get the result. If the result is public, $\mathcal{A}$ directly sends it to $\mathcal{B}$. If the required output u is in a secret-shared form, we can insert an ADD circuit into COMP circuit for dividing the result. As depicted in Fig. 2(b), the output of a newly-built circuit COMP-ADD is a secret share $\langle u\rangle^{\mathcal{A}}$ while $\langle u\rangle^{\mathcal{B}}$ is set as a random number $-r$ ($r \in \mathbb{Z}_{2^\ell}$). In a subsequent design, we will calculate inequality relation and division by NEQ-ADD and DIV-ADD circuits. The structure of them is similar to that of COMP-ADD circuit, and the only change is to use NEQ/DIV circuit to replace CMP circuit. Furthermore, we can easily compute the maximum of two secret-shared data by leveraging COMP-ADD circuit. Secure maximum (SecMax) protocol is based on the following equations: $\max(x,y) = y + u\cdot(x-y), u := x \geq y$. Specifically, $\mathcal{A}$ and $\mathcal{B}$ first obtain the flag $\langle u\rangle$ = COMP-ADD($\langle x\rangle, \langle y\rangle$), then compute $\langle\max(x,y)\rangle = \langle y\rangle + \langle u\rangle\cdot(\langle x\rangle - \langle y\rangle)$.



(a) COMP circuit (b) COMP-ADD circuit

Fig. 2. The structure of garbled circuits

C. Secure Sort (SST) Protocol

From the underlying auction in Section III-B, we find that the bid sorting is a dominant operation. In this subsection, we present a two-party protocol for a secure sort that runs between party $\mathcal{A}$ and $\mathcal{B}$. Given secret shares of an original sequence $\mathbf{x} = [x_1, x_2, \ldots, x_n]$, the goal of secure sort (SST) protocol is sorting $\mathbf{x}$ into a monotonically increasing sequence $\mathbf{x}^{\phi} = [x_{\phi(1)}, x_{\phi(2)}, \ldots, x_{\phi(n)}]$ under the secret-shared form, where ϕ is a permutation of indices 1 to n that corresponds to the indices of $\mathbf{x}$'s elements sorted by its values x_i. During the execution of SST protocol, no information regarding the plaintext of $\mathbf{x}$ is revealed to $\mathcal{A}$ and $\mathcal{B}$.

For ensuring both efficiency and information-theoretic security, we adopt a *Shuffle-then-Compute* strategy to devise secure sort protocol in which we add a step that securely shuffles the data before feeding it to a conventional sorting process. Let us consider a quick sort algorithm in which original input is first randomly permuted. During the execution, the adversary is able to infer sensitive information, such as the relationship of size between the elements, on the shuffled input by observing access patterns. Yet this information cannot be linked back to that of the original input, such that our sorting method is sufficient to prevent leakage from access patterns.

Specifically, we obfuscate the elements' orders by *permutation matrices* that can be generated by an arbitrary number of row elementary operation on the identity matrix. Each permutation matrix $\mathbf{E}_j$ represents a permutation function π_j. Multiplying the original sequence $\mathbf{x}$ with the permutation matrix $\mathbf{E}_j$ yields the new shuffled array $\mathbf{x}^{\pi_j}$. We give an illustration for the case $n = 3$ in Fig. 3. In the design of our secure two-party protocol, we first permute the original

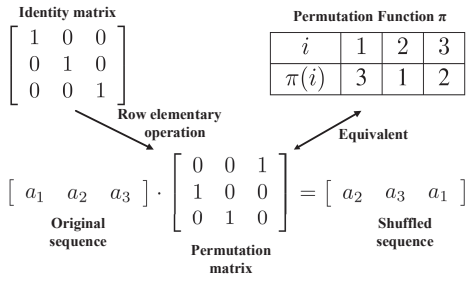


Fig. 3. The illustration of permutation matrix for $n = 3$, i denotes the position in the original sequence and $\pi(i)$ denotes that in the shuffled sequence.

sequence $\mathbf{x} = [x_1, x_2, \ldots, x_n]$ by two $n \times n$ random permutation matrices $\mathbf{E}_1$ (only known to $\mathcal{A}$) and $\mathbf{E}_2$ (only known to $\mathcal{B}$) as follows:

$$\mathbf{x} \cdot \mathbf{E}_1 \cdot \mathbf{E}_2 = [x_{\pi_2^{-1}(\pi_1^{-1}(1))}, x_{\pi_2^{-1}(\pi_1^{-1}(2))}, \ldots, x_{\pi_2^{-1}(\pi_1^{-1}(n))}],$$

where π_j^{-1} is the inverse function of π_j. So neither of $\mathcal{A}$ and $\mathcal{B}$ can recover the original index of $x_{\pi_2^{-1}(\pi_1^{-1}(i))}(1 \leq i \leq n)$ as long as the private permutation matrices are not leaked out. After that, we use the shuffled sequence as an input for a subsequent sorting process. In order to greatly reduce the computation complexity, our construction of sorting is built on quick sort algorithm.

Algorithm 1 summarizes the steps of SST protocol. Both the input and the output of SST protocol are a sequence in a secret-shared form. To start with, $\mathcal{A}$ generates a random permutation matrix $\mathbf{E}_1$ and a random number matrix $\mathbf{R}_1$, then computes $\langle \mathbf{E}_1 \rangle^{\mathcal{A}} = \mathbf{E}_1 - \mathbf{R}_1$ and sends $\mathbf{R}_1$ to $\mathcal{B}$. After receiving $\mathbf{R}_1$, $\mathcal{B}$ sets $\langle \mathbf{E}_1 \rangle^{\mathcal{B}} = \mathbf{R}_1$. Through these steps, the permutation matrix $\mathbf{E}_1$ is secret-shared between two parties while being kept private to $\mathcal{A}$. Then, $\mathcal{A}$ and $\mathcal{B}$ take similar actions to generate the secret shares of permutation matrix $\mathbf{E}_2$ which is only known to $\mathcal{B}$. Subsequently, following the operations during the vectorizing procedure in Section IV-A, $\mathcal{A}$ and $\mathcal{B}$ can easily compute $\langle \mathbf{x}' \rangle = \langle \mathbf{x} \cdot \mathbf{E}_1 \cdot \mathbf{E}_2 \rangle = \langle \mathbf{x} \rangle \cdot \langle \mathbf{E}_1 \rangle \cdot \langle \mathbf{E}_2 \rangle$. At this point, $\mathcal{A}$ and $\mathcal{B}$ jointly complete the *shuffle* process, which cuts off the link between the original sequence and the new one. Next, $\langle \mathbf{x}' \rangle$ will be fed to a quick sort process. In contrast to the traditional quick-sort algorithm, we use the customized COMP circuit to achieve comparison operations on the secret-shared data. And base on this, the underlying quick-sort method with recursion is easily extended to support the shared sequence.

Key-Value Sort. Benefited from the above realization method of sorting a set of shared values, our sorting protocol is easily expandable to work on arrays where every element is a pair of shared values representing a key-value pair (k_i, x_i) and the value x_i is used to sort the array. Firstly, the input is updated with a set of pairs of shared values $(\langle k_1 \rangle, \langle x_1 \rangle), (\langle k_2 \rangle, \langle x_2 \rangle), \ldots, (\langle k_n \rangle, \langle x_n \rangle)$, and similarly for the output. Secondly, we need to use the same random permutation matrices $\mathbf{E}_1, \mathbf{E}_2$ to shuffle of k_i and x_i simultaneously, i.e., compute $\langle \mathbf{x}' \rangle = \langle \mathbf{x} \cdot \mathbf{E}_1 \cdot \mathbf{E}_2 \rangle$ and $\langle \mathbf{k}' \rangle = \langle \mathbf{k} \cdot \mathbf{E}_1 \cdot \mathbf{E}_2 \rangle$. Thirdly, the swap operations (as shown on Line 16 and Line 20) are used not only on the values but also on the keys, such that the keys would be arranged to keep pace with their values.

Algorithm 1 Secure Sorting (SST) Protocol

Input: $\mathcal{A}$ inputs $\langle \mathbf{x} \rangle^{\mathcal{A}} = [\langle x_1 \rangle^{\mathcal{A}}, \ldots, \langle x_n \rangle^{\mathcal{A}}]$
$\mathcal{B}$ inputs $\langle \mathbf{x} \rangle^{\mathcal{B}} = [\langle x_1 \rangle^{\mathcal{B}}, \ldots, \langle x_n \rangle^{\mathcal{B}}]$
Output: $\mathcal{A}$ outputs $\langle \mathbf{x}^{\phi} \rangle^{\mathcal{A}} = [\langle x_{\phi(1)} \rangle^{\mathcal{A}}, \ldots, \langle x_{\phi(n)} \rangle^{\mathcal{A}}]$
$\mathcal{B}$ outputs $\langle \mathbf{x}^{\phi} \rangle^{\mathcal{B}} = [\langle x_{\phi(1)} \rangle^{\mathcal{B}}, \ldots, \langle x_{\phi(n)} \rangle^{\mathcal{B}}]$

1: $\underline{\mathcal{A}}$:
2: Generate a random permutation matrix $\mathbf{E}_1$ and a random number matrix $\mathbf{R}_1$.
3: Set $\langle \mathbf{E}_1 \rangle^{\mathcal{A}} = \mathbf{E}_1 - \mathbf{R}_1$ and send $\mathbf{R}_1$ to $\mathcal{B}$.
4: $\underline{\mathcal{B}}$:
5: Receive $\mathbf{R}_1$ from $\mathcal{A}$ and set $\langle \mathbf{E}_1 \rangle^{\mathcal{B}} = \mathbf{R}_1$.
6: $\underline{\mathcal{A}}$ and $\underline{\mathcal{B}}$:
7: Similar to above, generate the secret shares of random permutation matrices matrix $\mathbf{E}_2$ (only known to $\mathcal{B}$), making $\mathcal{A}$ get $\langle \mathbf{E}_2 \rangle^{\mathcal{A}}$ and $\mathcal{B}$ get $\langle \mathbf{E}_2 \rangle^{\mathcal{B}}$.
8: Compute $\langle \mathbf{E}_1 \cdot \mathbf{E}_2 \rangle = \langle \mathbf{E}_1 \rangle \cdot \langle \mathbf{E}_2 \rangle$.
9: Compute $\langle \mathbf{x}' \rangle = \langle \mathbf{x} \rangle \cdot \langle \mathbf{E}_1 \cdot \mathbf{E}_2 \rangle = \langle \mathbf{x} \cdot \mathbf{E}_1 \cdot \mathbf{E}_2 \rangle$.
10: Set $low = 1$ and $up = n$.
11: **if** $low < up$ **then**
12: $\underline{\mathcal{A}}$: $\langle p \rangle^{\mathcal{A}} = \langle \mathbf{x}'_{up} \rangle^{\mathcal{A}}$, $\underline{\mathcal{B}}$: $\langle p \rangle^{\mathcal{B}} = \langle \mathbf{x}'_{up} \rangle^{\mathcal{B}}$.
13: Set $i = low$.
14: **for** $j = low$ to $(up - 1)$ **do**
15: **if** COMP$(\langle p \rangle, \langle \mathbf{x}'_j \rangle)$ **then**
16: $\underline{\mathcal{A}}$: Swap$(\langle \mathbf{x}'_i \rangle^{\mathcal{A}}, \langle \mathbf{x}'_j \rangle^{\mathcal{A}})$, $\underline{\mathcal{B}}$: Swap$(\langle \mathbf{x}'_i \rangle^{\mathcal{B}}, \langle \mathbf{x}'_j \rangle^{\mathcal{B}})$.
17: $i = i + 1$.
18: **end if**
19: **end for**
20: $\underline{\mathcal{A}}$: Swap$(\langle \mathbf{x}'_i \rangle^{\mathcal{A}}, \langle \mathbf{x}'_{up} \rangle^{\mathcal{A}})$, $\underline{\mathcal{B}}$: Swap$(\langle \mathbf{x}'_i \rangle^{\mathcal{B}}, \langle \mathbf{x}'_{up} \rangle^{\mathcal{B}})$.
21: Set $up = i - 1$ and repeat the steps from line 11 to 23.
22: Set $low = i + 1$ and repeat the steps from line 11 to 23.
23: **end if**
24: $\underline{\mathcal{A}}$: Set $\langle \mathbf{x}^{\phi} \rangle^{\mathcal{A}} = \langle \mathbf{x}' \rangle^{\mathcal{A}}$, $\underline{\mathcal{B}}$: Set $\langle \mathbf{x}^{\phi} \rangle^{\mathcal{B}} = \langle \mathbf{x}' \rangle^{\mathcal{B}}$.

Accordingly, our protocol is able to sort *multidimensional array* in which each element is a group of shared values through similar updates. We provide experimental evaluations for this extension in Section VII-B. The detailed procedure is not described here due to a page limit.

V. Secure Spectrum Auction

In this section, we propose a lightweight auction framework (SLISA) for spectrum allocation with strong security guarantees. As mentioned above, our design rationale is based on the TDSA scheme. However, the technical difficulty is to execute the auction on the secret-shared data in a data-oblivious manner. To achieve this, we improve the TDSA auction mechanism to make it data-oblivious, and refactor it by using secure sub-protocols presented in Section IV. The main auction scheme consists of the following phases.

A. Bid Sharing and Submitting

At first, all sellers $s_1, s_2, \ldots, s_M$ divide their IDs ID_m^s and sell-bids $V_m (1 \leq m \leq M)$ locally by the additive secret sharing, and then submit the shared IDs and bids to the auctioneer $\mathcal{A}$ and the auction agent $\mathcal{B}$. All buyers' IDs ID_n^b

Algorithm 2 Privacy-preserving Buyer Grouping Protocol

Input: Secret-shared IDs and requests of buyers.
Output: Secret-shared data of non-conflict buyer groups.

1: $\mathcal{A}$ and $\mathcal{B}$:
2: Generate two secret-shared random permutation matrices $\langle \mathbf{E}_1 \rangle$ and $\langle \mathbf{E}_2 \rangle$, where $\mathbf{E}_1$ and $\mathbf{E}_2$ are only known to $\mathcal{A}$ and $\mathcal{B}$, respectively.
3: Use $\langle \mathbf{E}_1 \rangle$ and $\langle \mathbf{E}_2 \rangle$ to shuffle the secret-shared tuples $\{(\langle ID_n^b \rangle, \langle B_n \rangle, \langle x_n \rangle, \langle y_n \rangle, \langle rd_n \rangle)\}_{n=1}^N$ to get a new tuple array $\{(\langle ID_i' \rangle, \langle B_i' \rangle, \langle x_i' \rangle, \langle y_i' \rangle, \langle rd_i' \rangle)\}_{i=1}^N$.
4: Initialize edge set $\mathbb{E} = \varnothing$.
5: **for** $i = 1$ to N **do**
6: **for** $j = i + 1$ to N **do**
7: Compute $\langle rsum \rangle = (\langle rd_i' \rangle + \langle rd_j' \rangle)^2$, $\langle dis \rangle = (\langle x_i' \rangle - \langle x_j' \rangle)^2 + (\langle y_i' \rangle - \langle y_j' \rangle)^2$.
8: **if** COMP($\langle rsum \rangle, \langle dis \rangle$) **then**
9: $\mathbb{E} = \mathbb{E} \cup (b_i, b_j)$.
10: **end if**
11: **end for**
12: **end for**
13: Construct the conflict graph $CG(\mathbb{B}, \mathbb{E})$.
14: Get the set of non-conflict buyer groups $G = \{G_1, G_2, \ldots, G_T\}$ by GMIN algorithm.
15: Reorganize buyers' IDs and buy-bids for each group G_t, to get $(\langle ID_{t,j} \rangle, \langle B_{t,j} \rangle), 1 \le j \le R, 1 \le t \le T$.

and requests $(B_n, x_n, y_n, rd_n)(1 \le n \le N)$ are divided and submitted in the same way by buyers. The tuples submitted by the sellers and buyers are as follows:
Sellers: $(\langle ID_m^s \rangle, \langle V_m \rangle)$, $1 \le m \le M$
Buyers: $(\langle ID_n^b \rangle, \langle B_n \rangle, \langle x_n \rangle, \langle y_n \rangle, \langle rd_n \rangle)$, $1 \le n \le N$

B. Privacy-preserving Buyer Grouping

In this phase, the auctioneer and the agent perform a privacy-preserving buyer grouping (PPBG) protocol for high spectrum utilization. We first construct a conflict graph based on the shared locations of buyers, then group buyers subject to the conflict graph. To prevent potential privacy leakage of the location relations, we adopt once again the shuffle-then-compute strategy to design PPBG protocol.

As described in Algorithm 2, the auctioneer $\mathcal{A}$ and the agent $\mathcal{B}$ jointly generate two secret-shared random permutation matrices $\langle \mathbf{E}_1 \rangle$ and $\langle \mathbf{E}_2 \rangle$ ($\mathbf{E}_1$ and $\mathbf{E}_2$ are only known to $\mathcal{A}$ and $\mathcal{B}$ respectively), and use them to shuffle the tuple array submitted by buyers. Then, $\mathcal{A}$ and $\mathcal{B}$ construct the conflict graph according to the shuffled tuple array $\{(\langle ID_i' \rangle, \langle B_i' \rangle, \langle x_i' \rangle, \langle y_i' \rangle, \langle rd_i' \rangle)\}_{i=1}^N$. The interference area of buyer b_i is represented by a circle centered on location (x_i, y_i) with radius rd_i. Two buyers b_i and b_j are non-conflict if the following equation holds: $(x_i - x_j)^2 + (y_i - y_j)^2 \ge (rd_i + rd_j)^2$. By using the arithmetic operations and COMP circuit in Section IV-A, we can recognize whether b_i is conflicted with b_j. Let $\mathbb{B}$ denote all buyers and $\mathbb{E}$ denote conflict-edge set. If b_i and b_j are conflict, we add an edge (b_i, b_j) into the edge set $\mathbb{E}$. Furthermore, we can construct the conflict graph $CG(\mathbb{B}, \mathbb{E})$ by checking the conflict relationships between all buyers (Line 4-13). Next, we use a classic greedy algorithm GMIN [34] (We omit its details due to a page limit), which takes $CG(\mathbb{B}, \mathbb{E})$ as input, to yield the set of non-conflict buyer groups $G = \{G_1, G_2, \ldots, G_T\}$. Assume that the group G_t has R_t buyers and $R = \max_{t \in \{1,2,\ldots,T\}}(R_t)$. Finally, we reorganize buyers' IDs and buy-bids for each group G_t to get $(\langle ID_{t,j}^b \rangle, \langle B_{t,j} \rangle), 1 \le j \le R, 1 \le t \le T$. That is, the original buyers' IDs $\langle ID_n^b \rangle$ and bids $\langle B_n \rangle$ are rearranged according to different groups. Note that we pad G_t with $(\langle 0 \rangle, \langle 0 \rangle)$ to R buyer tuples for executing the subsequent processing in a data-oblivious manner.

C. Privacy-preserving Spectrum Allocation

At the beginning, we construct a set of secret-shared multidimensional arrays $\mathbb{S}$, $\mathbb{G}$ and $\mathbb{G}_t (t \in \{1, 2, \ldots, T\})$ to represent all sellers, all buyer groups, and all buyers belonged to group G_t. In these arrays, each seller, each group, and each buyer exist in the form of tuples as follows:
Sellers: $(\langle ID_m^s \rangle, \langle V_m \rangle, \langle f_m^s \rangle)$, $1 \le m \le M$
Groups: $(\langle GB_t \rangle, \langle C_t \rangle, \langle P_t^b \rangle, \langle f_t^g \rangle)$, $1 \le t \le T$
Buyers: $(\langle ID_{t,j}^b \rangle, \langle B_{t,j} \rangle, \langle B_{t,j}^v \rangle, \langle f_{t,j}^b \rangle), 1 \le j \le R, 1 \le t \le T$
where f_m^s, f_t^g and $f_{t,j}^b$ are binary flags to indicate whether the seller, group, and buyer are a winner (1) or not (0), respectively. Then, we initialize these arrays as follows:

$$\mathbb{S} = \begin{vmatrix} m: & 1 & \cdots & M \\ \langle ID_m^s \rangle: & \langle ID_1^s \rangle & \cdots & \langle ID_M^s \rangle \\ \langle V_m \rangle: & \langle V_1 \rangle & \cdots & \langle V_M \rangle \\ \langle f_m^s \rangle: & \langle 0 \rangle & \cdots & \langle 0 \rangle \end{vmatrix} \quad \mathbb{G} = \begin{vmatrix} t: & 1 & \cdots & T \\ \langle GB_t \rangle: & \langle 0 \rangle & \cdots & \langle 0 \rangle \\ \langle C_t \rangle: & \langle 0 \rangle & \cdots & \langle 0 \rangle \\ \langle P_t^b \rangle: & \langle 0 \rangle & \cdots & \langle 0 \rangle \\ \langle f_t^g \rangle: & \langle 0 \rangle & \cdots & \langle 0 \rangle \end{vmatrix}$$

$$\mathbb{G}_t = \begin{vmatrix} j: & 1 & \cdots & R \\ \langle ID_{t,j}^b \rangle: & \langle ID_{t,1}^b \rangle & \cdots & \langle ID_{t,R}^b \rangle \\ \langle B_{t,j} \rangle: & \langle B_{t,1} \rangle & \cdots & \langle B_{t,R} \rangle \\ \langle B_{t,j}^v \rangle: & \langle 0 \rangle & \cdots & \langle 0 \rangle \\ \langle f_{t,j}^b \rangle: & \langle 0 \rangle & \cdots & \langle 0 \rangle \end{vmatrix}$$

After that, we elaborate a data-oblivious spectrum allocation, and present a privacy-preserving spectrum allocation (PPSA) protocol as shown in Algorithm 3.

1) Secure Virtual Group Bidding (Line 1-8): This step uses two looping statements to obtain virtual group bids $B_{t,j}^v$ and group bids GB_t. In the outer for-loop, $\mathcal{A}$ and $\mathcal{B}$ sort all buyers of $\mathbb{G}_t$ in a non-ascending order of buy-bid $B_{t,j}$ by SST protocol. Recall that the extended SST protocol is able to sort a multidimensional array. For simplicity, we use the same indices $j \in [1, R]$ after the secure sorting without affecting correctness. In the nested for-loop, $\mathcal{A}$ and $\mathcal{B}$ first compute all virtual group bids $B_{t,j}^v$ of group $\mathbb{G}_T$ based on the secure arithmetic operations on secret-shared data, and then select the maximum of all virtual group bids as the group bid GB_t by SecMax protocol.

2) Secure Preliminary Winner Determination (Line 9-22): In this step, we will obtain winning sellers' IDs and clearing prices for winning sellers and winning groups. At first, $\mathcal{A}$ and $\mathcal{B}$ sort all sellers $\mathbb{S}$ in a non-descending order of sell-bid V_m, and sort all group-bids GB_t in a non-ascending order. From the underlying TDSA scheme described in Section III-B, it is

vital to determine the critical index K according to Equation (1). Next, inspired by the previous work [9], we leverage two arrays of binary flags λ_k and η_k to find out K in a data-independent manner:

- λ_k: indicates whether k is less than or equal to the critical index j^* ($\lambda_k = 1$) or not ($\lambda_k = 0$) (Line 11 & 13).
- η_k: indicates whether k is equal to the critical index K ($\eta_k = 1$) or not ($\eta_k = 0$) (Line 18 & 20)

To facilitate understanding, we state the value-taking pattern of these flags as follows:

$$\left|\begin{array}{cccccccc} k: & 1 & \cdots & K-1 & K & K+1 & \cdots & \theta \\ \lambda_k: & 1 & \cdots & 1 & 1 & 0 & \cdots & 0 \\ \eta_k: & 0 & 0 & 0 & 1 & 0 & \cdots & 0 \\ f_k^s: & 1 & 1 & 1 & 0 & 0 & \cdots & 0 \end{array}\right|$$

With inputting the secret shared group bids and buy-bids, $\mathcal{A}$ and $\mathcal{B}$ compute the secret shares of λ by COMP-ADD and NEQ-ADD circuits, where $\lambda = (GB_k \geq V_k \wedge GB_k \neq GB_{k-1})$. Then, it is straightforward to compute $\eta_k = \lambda_k - \lambda_{k+1} (1 \leq k \leq \theta - 1)$ in term of the above pattern. After that, the flags f_k^s for sellers can be set to 1 when $1 \leq k \leq K-1$ by $f_{k-1}^s = \lambda_k (2 \leq k \leq \theta)$ (Line 13). Based on these flags, the clearing prices and IDs of winning sellers can be computed in a data-oblivious manner as follows: $P^s = P^s + V_k \cdot \eta_k$ (Line 21), $ID_m^s = ID_m^s \cdot f_m^b$ such that the failures' ID is set to 0 (Line 15-17). Similarly, the clearing prices of winning groups can be obtained by $P^G = P^G + GB_k \cdot \eta_k$ (Line 21).

3) Secure Washing Out (Line 23-36): This step removes the buyers in the winning group, whose bids are too low to afford the clearing price P_G. We first compute the flags $f_t^g = (GB_t > P^G)$ for groups by COMP-ADD circuit (Line 24). Next, to compute the number C_t of winning buyer in group G_t in a data-independent manner, we introduce an array of binary flags δ_j to indicate whether j is less than or equal to C_t ($\delta_j = 1$) or not ($\delta_j = 0$). In a winning group, the value-taking pattern of flags δ_j and $f_{t,j}^b$ as follows:

$$\left|\begin{array}{cccccccc} j: & 1 & \cdots & C_t-1 & C_t & C_t+1 & \cdots & R \\ \delta_j: & 1 & \cdots & 1 & 1 & 0 & \cdots & 0 \\ f_{t,j}^b: & 0 & 0 & 0 & 1 & 0 & \cdots & 0 \end{array}\right|$$

We compute these flags and C_t by $\delta_j = f_t^g \wedge (B_{t,j}^v > P^G)$, $f_{t,j}^b = \delta_j - \delta_{j+1}$, and $C_t = C_t + j \cdot f_{t,j}^b$ (Line 25-31). Finally, the clearing prices and IDs of winning buyers can be obtained by $ID_{t,j}^b = ID_{t,j}^b \cdot f_{t,j}^b$ and $P_t^b = f_t^g \cdot (P^G / C_t)$ (Line 32-35).

4) Result Return (Line 37): In this step, $\mathcal{A}$ and $\mathcal{B}$ publish the IDs and clearing prices of all winning sellers and winning buyers as final auction results.

VI. Security Analysis

We first employ composition theory [30] to prove the security of our protocols in the semi-honest adversary model [30] (as shown in Definition 1), then demonstrate that our secure spectrum auction can meet the desired security requirements.

Definition 1. *Suppose that a two-party protocol Π asks $\mathcal{A}$ (resp. $\mathcal{B}$) to compute the function $f^{\mathcal{A}}(x,y)$ (resp. $f^{\mathcal{B}}(x,y)$), where x, y are the inputs of $\mathcal{A}$ and $\mathcal{B}$, respectively. Let $view^{\mathcal{A}}(x,y) = (x, r_{\mathcal{A}}, m_1, \cdots, m_t)$ (resp. $view^{\mathcal{B}}(x,y) = (y, r_{\mathcal{B}}, m_1, \cdots, m_t)$) is the view of $\mathcal{A}$ (resp. $\mathcal{B}$) during an execution of Π on (x,y), where $r_{\mathcal{A}}$ (resp. $r_{\mathcal{B}}$) represents randomness of $\mathcal{A}$ (resp. $\mathcal{B}$) and m_i represents the i-th message passed between the parties. Let $O^{\mathcal{A}}(x,y)$ (resp. $O^{\mathcal{B}}(x,y)$) denotes $\mathcal{A}$'s (resp. $\mathcal{B}$'s) output. We say that protocol Π is secure against semi-honest adversaries if there exist probabilistic polynomial time (PPT) simulators S_1 and S_2 such that:*

Algorithm 3 Privacy-preserving Spectrum Allocation Protocol

Input: Secret-shared arrays $\mathbb{S}$, $\mathbb{B}$ and $\mathbb{G}_t (t \in \{1, 2, \ldots, T\})$.
Output: Secret-shared IDs and clearing prices of the winning sellers and buyers.

1: $\mathcal{A}$ and $\mathcal{B}$:
2: **for** $1 \leq t \leq T$ **do**
3: Sort $\mathbb{G}_t$ in a non-ascending order of buy-bid $B_{t,j}$ by SST protocol (use same indices after sorting).
4: **for** $1 \leq j \leq R$ **do**
5: $\langle B_{t,j}^v \rangle = j \cdot \langle B_{t,j} \rangle$
6: $\langle GB_t \rangle = \text{SecMax}(\langle GB_t \rangle, \langle B_{t,j}^v \rangle)$.
7: **end for**
8: **end for**
9: Sort $\mathbb{S}$ in a non-descending order of sell-bid V_m by SST protocol.
10: Sort the group-bids $\{GB_t\}_{t=1}^T$ in a non-ascending order by SST protocol.
11: $\theta = \min(M, T)$, $\langle \lambda_1 \rangle = \text{COMP-ADD}(\langle GB_1 \rangle, \langle V_1 \rangle)$.
12: **for** $2 \leq k \leq \theta$ **do**
13: $\langle \alpha \rangle = \text{COMP-ADD}(\langle GB_k \rangle, \langle V_k \rangle)$, $\langle \beta \rangle = \text{NEQ-ADD}(\langle GB_k \rangle, \langle GB_{k-1} \rangle)$, $\langle \lambda_k \rangle = \langle \alpha \rangle \cdot \langle \beta \rangle, \langle f_{k-1}^s \rangle = \langle \lambda_k \rangle$.
14: **end for**
15: **for** $1 \leq m \leq M$ **do**
16: $\langle ID_m^s \rangle = \langle ID_m^s \rangle \cdot \langle f_m^b \rangle$.
17: **end for**
18: $\langle \eta_\theta \rangle = \langle \lambda_\theta \rangle$
19: **for** $1 \leq k \leq \theta - 1$ **do**
20: $\langle \eta_k \rangle = \langle \lambda_k \rangle - \langle \lambda_{k+1} \rangle$.
21: $\langle P^s \rangle = \langle P^s \rangle + \langle V_k \rangle \cdot \langle \eta_k \rangle, \langle P^G \rangle = \langle P^G \rangle + \langle GB_k \rangle \cdot \langle \eta_k \rangle$.
22: **end for**
23: **for** $1 \leq t \leq T$ **do**
24: $\langle f_t^g \rangle = \text{COMP-ADD}(\langle GB_t \rangle, \langle P^G \rangle)$.
25: **for** $1 \leq j \leq R$
26: $\langle \gamma \rangle = \text{COMP-ADD}(\langle B_{t,j}^v \rangle, \langle P^G \rangle), \langle \delta_j \rangle = \langle f_t^g \rangle \cdot \langle \gamma \rangle$.
27: **end for**
28: **for** $1 \leq j \leq R - 1$ **do**
29: $\langle f_{t,j}^b \rangle = \langle \delta_j \rangle - \langle \delta_{j+1} \rangle$.
30: $\langle C_t \rangle = \langle C_t \rangle + j \cdot \langle f_{t,j}^b \rangle$.
31: **end for**
32: $\langle P_t^b \rangle = \langle f_t^g \rangle \cdot \text{DIV-ADD}(\langle P^G \rangle, \langle C_t \rangle)$.
33: **for** $1 \leq j \leq R$ **do**
34: $\langle ID_{t,j}^b \rangle = \langle ID_{t,j}^b \rangle \cdot \langle f_{t,j}^b \rangle$.
35: **end for**
36: **end for**
37: **Return** $\langle ID_{t,j}^b \rangle, \langle P_t^b \rangle (1 \leq t \leq T, 1 \leq j \leq R)$, $\langle ID_m^s \rangle (1 \leq m \leq M), \langle P^s \rangle$.

$$(S_1(x, f^{\mathcal{A}}(x,y)), f(x,y)) \stackrel{c}{\equiv} (view^{\mathcal{A}}(x,y), O(x,y)) \quad (2)$$

$$(S_2(y, f^{\mathcal{B}}(x,y)), f(x,y)) \stackrel{c}{\equiv} (view^{\mathcal{B}}(x,y), O(x,y)) \quad (3)$$

where $\stackrel{c}{\equiv}$ denotes computational indistinguishability.

We state that the arithmetic operations and garbled circuits in Section IV are secure under the semi-honest adversaries model. The formal security proofs of them can be found in [29], [31]. COMP, COMP-ADD, NEQ-ADD, and DIV-ADD circuits and SecMax protocol are direct applications of Yao's garbled circuits, whose security proof can be found in [35]. Then, we prove the security of all protocols as follows:

Theorem 1. *As long as the arithmetic operations and COMP circuit are secure against semi-honest adversaries, SST protocol is secure under the semi-honest adversaries model.*

Proof. To prove the security of SST protocol, we construct simulators in two distinct cases depending on which party is corrupted. We show that for all PPT adversaries, the corrupted party's view based on the interaction between $\mathcal{A}$ and $\mathcal{B}$ is indistinguishable to its view when it interacts with a simulator instead. The interactive messages of SST protocol include the data transferred in executing secure arithmetic operations and COMP circuit, and the output of COMP circuit. Since the arithmetic operations and COMP circuit are secure against semi-honest adversaries, any PPT adversary cannot distinguish the simulator's views from the data transferred in these operations. As for the output of COMP circuit, we construct a simulator $\mathcal{S}_1$ to simulate $\mathcal{A}$'s view. $\mathcal{S}_1$ chooses an integer σ from $\{0,1\}$ uniformly at random to simulate the output of COMP circuit. Recall that the COMP circuit's input values are shuffled by $\mathbf{E}_1 \cdot \mathbf{E}_2$ and these matrices are generated randomly. So the case in which the output of COMP is assigned to 0 or 1 is randomly given, such that any PPT adversary cannot distinguish σ from COMP$(\langle p \rangle, \langle \mathbf{x}'_j \rangle)$. Clearly, a simulator $\mathcal{S}_2$ can do the same works as above to simulate $\mathcal{B}$'s view. According to the composition theory [35], we can claim that SST protocol is secure under the semi-honest adversaries model. □

Theorem 2. *As long as the arithmetic operations, garbled circuits, SecMax protocol, and SST protocol are secure against semi-honest adversaries, PPBG protocol and PPSA protocol are secure under the semi-honest adversaries model.*

The proof of Theorem 2 is similar to that of Theorem 1, and we omit this due to a page limit. Next, we demonstrate that our auction scheme can achieve the desired security properties, which is trivially guaranteed by the security of the above protocols. Recall that each buyer divides the location information (x_n, y_n, r_n) into two shares by additive secret sharing locally before submitting them to the auctioneer and the agent. Since PPBG protocol is secure against semi-honest adversaries, no information about (x_n, y_n, r_n) is disclosed to $\mathcal{A}$ and $\mathcal{B}$, i.e., the location privacy can be preserved. As the same as the location information, buy-bids B_n and sell-bids V_m are secret-shared before uploading to $\mathcal{A}$ and $\mathcal{B}$. Since PPBG protocol and PPSA protocol are secure against semi-honest adversaries, no information about B_n and V_m is disclosed to $\mathcal{A}$ and $\mathcal{B}$.

VII. Performance Evaluation

A. Experiment Setup

We implement all protocols in C++ and run experiments on the two Amazon EC2 t2.large machines (as $\mathcal{A}$ and $\mathcal{B}$) with Linux 16.04 and 8 GB of RAM each. The communication bandwidth between two machines for LAN setting is set to 1 GB/s. Specifically, we use the unsigned long integer type in C++ for secure arithmetic operations and use a state-of-the-art computation framework EMP-toolkit [36] for garbled circuits. In our experiment, buyers are randomly distributed in $1000 \times 1000\text{m}^2$ area, and the interference range is randomly from 50m to 100m. The sell-bids and buy-bids are uniformly from (0,10]. To ensure security and accuracy, we set the bit length of each values $\ell = 32$ with $\ell_F = 16$ bits in the fractional part, and the symmetric security parameter $\kappa = 128$ for garbled circuits.

B. Benchmarking of secure operations on secret-shared data

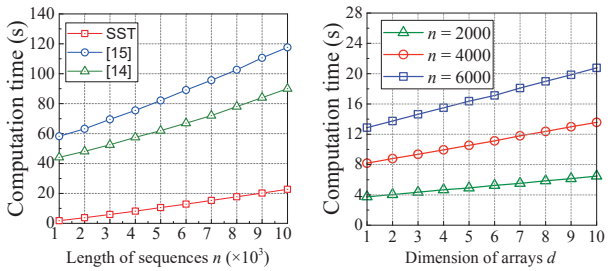
We evaluate the performance of secure basic operations on secret-shared data. We report results for both offline and online time separately. The offline stage includes the computation and communication that are independent of the input including the precomputation of multiplication triplets and garbled circuits, and the generation of permutation matrices in SST protocol, while the online phase consists of all input-dependent steps.

We first list the computation time and communication cost of secure arithmetic operations, our custom-built garbled circuits, and secure maximum (SecMax) protocol in Table III. The running time includes the computation time and data transmission time of both sides. We find that these basic operations enjoy a high performance during the online stage, which lays a good foundation for more complex protocols. Next, as shown in Fig. 4(a), we compare online computation time (for varying lengths of the sequences) of secure sort (SST) protocol with two state-of-the-art sort protocols in [14] and [15], which are based on additive homomorphic encryption and pure garbled circuits, respectively. We find that the time cost in SST protocol is almost increased linearly with the length of sequences, which is consistent with the computational complexity of underlying quick sort $(O(n \log n))$. Compared with previous works, the online computation time in SST protocol has remarkable advantages, as it does not involve

TABLE III
Running time (in μs) and communication cost (in $bytes$) for one secure operation on secret-shared data

Operations	Online Stage		Offline Stage	
	Time	Comm	Time	Comm
Secure addition	1	0	0	0
Secure multiplication	145	16	276911	1050
COMP	516	1596	6643	0
COMP-ADD	568	1642	8447	0
NEQ-ADD	538	1536	8256	0
DIV-ADD	594	1824	8543	0
SecMax	816	1702	280456	1050

any time-consuming cryptographic operation (e.g., homomorphic encryptions in [14] and large-scale circuit operation in [15]). Then, we also measure the online time for varying dimensions of arrays when performing secure sorting on a multidimensional array. Since the previous protocols in [14] and [15] cannot be directly applied to multidimensional-array sorting, we only present the impact of different dimensions on SST protocol in Fig. 4(b). The time cost in SST protocol with different n increases at a linear speed as the dimension of arrays (d) increases, since the number of the added operations is linear with the extended dimensions. But for all this, the online time remains relatively small, as the added operations only include secure multiplications and element swaps.



(a) Impact of length of sequences (b) Impact of dimension of arrays

Fig. 4. Online computation time of SST protocol.

C. Performance of secure spectrum auction

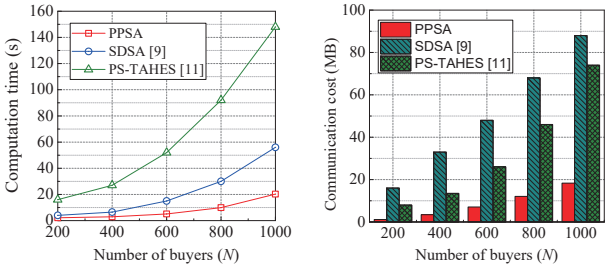
We evaluate the performance of our secure spectrum auction that includes three phases, i.e., bid submitting, buyer grouping, and spectrum allocation. To make a comprehensive performance evaluation, we compare our protocols with two advanced scheme SDSA [9] and PS-TAHES [11]. For a fair comparison, we assume that a bidder has the same bid for all channels in the evaluation. It is worth noting that the security level of these schemes is lower than ours, e.g., SDSA cannot support privacy-preserving buyer grouping, and PS-TAHES discloses the order of locations.

During the bid submitting, sellers and buyers use additive secret sharing to divide their data before submission, which is very suitable for lightweight computing devices. According to our evaluation, single execution time for data dividing can be at microsecond level. Next, we illustrate the performance of privacy-preserving buyer grouping (PPBG) protocol and privacy-preserving spectrum allocation (PPSA) protocol in Table IV. Note that the number of sellers M is set to 50 and the number of buyers N varies from 200 to 1000. We find that buyer grouping takes up most of the auction time, as this stage involves abundant COMP-circuit computations. For instance, when $M = 50$ and $N = 200$, the online computation time and communication cost of PPBG protocol are 25.87s and 66.85MB. This result still owns an important advantage compared with PS-TAHES, which takes 287.21s and 275.92MB to group buyers when $M = 20$ and $N = 100$. Furthermore, we compare the performance of spectrum allocation between PPSA protocol and previous works. As shown in Fig. 5(a), PPSA protocol is up to 2.8× faster than SDSA and 7.3× faster than PS-TAHES. Fig. 5(b) shows that PPSA protocol also has

TABLE IV
COMPUTATION TIME AND COMMUNICATION COST OF BUYER GROUPING AND SPECTRUM ALLOCATION WHEN $M = 50$

Stage		Buyer grouping	Spectrum allocation
$N = 200$	Time (s)	25.87	2.17
	Comm (MB)	66.85	1.15
$N = 400$	Time (s)	84.43	2.91
	Comm (MB)	256.39	3.44
$N = 600$	Time (s)	172.54	5.08
	Comm (MB)	578.61	7.09
$N = 800$	Time (s)	30.16	9.94
	Comm (MB)	1024.54	12.06
$N = 1000$	Time (s)	496.54	20.26
	Comm (MB)	1662.38	18.32

remarkable advantages in communication cost over others. The underlying reason is that our protocol reduces time-consuming cryptographic operations as much as possible by employing the lightweight additive-secret-sharing technique and basic garbled circuits to execute the majority of secure computations. In contrast to our scheme, SDSA needs to perform large-scale garbled-circuit computations, while PS-TAHES involves a mass of homomorphic encryptions/decryptions and operations on homomorphic ciphertexts.



(a) Computation time (b) Communication cost

Fig. 5. Performance comparison in the stage of spectrum allocation.

VIII. CONCLUSION

In this paper, we propose a lightweight spectrum auction framework with strong security guarantees for both users' locations and bids. Technically, we design a set of secure basic operations by leveraging lightweight cryptographic primitives (i.e., additive sharing and garbled circuits) on secret-shared data, to achieve security, efficiency, and scalability simultaneously. Formal security analysis is presented showing all protocols involved in our auction are secure under the semi-honest adversary model. We demonstrate the high-efficiency of our design by comparing with state-of-the-art works through extensive experiments. In the future, we plan to extend our framework to support a variety of secure auction schemes.

ACKNOWLEDGMENT

The corresponding authors are Liangmin Wang and Yulong Shen. This research was supported in part by the National Key Research and Development Program of China (Grant No. 2018YFE0207600), the National Natural Science Foundation of China (Grant No. U1736216, 61571352 and 61602364), Key R&D Program of Shaanxi Province (Grant No. 2019ZDLGY12-03, 2019ZDLGY13-06), the Fundamental Research Funds for the Central Universities and the Innovation Fund of Xidian University.

REFERENCES

[1] X. Dong, Y. Gong, J. Ma, and Y. Guo, "Protecting operation-time privacy of primary users in downlink cognitive two-tier networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 7, pp. 6561–6572, 2018.

[2] X. Dong, X. Yang, Y. Wang, A. Salem, Y. Shen, and J. Ma, "Salute: A strategy-proof auction mechanism for flexible multichannel allocation," in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops*. IEEE, 2018, pp. 1–2.

[3] X. Dong, Q. Kang, Y. Xu, Z. Ma, and T. Li, "A practical sybil-proof incentive mechanism for multichannel allocation," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops*. IEEE, 2019, pp. 1047–1048.

[4] X. Dong, T. Zhang, D. Lu, G. Li, Y. Shen, and J. Ma, "Preserving geo-indistinguishability of the primary user in dynamic spectrum sharing," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 9, pp. 8881–8892, 2019.

[5] M. Pan, J. Sun, and Y. Fang, "Purging the back-room dealing: Secure spectrum auction leveraging paillier cryptosystem," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 4, pp. 866–876, 2011.

[6] Q. Huang, Y. Tao, and F. Wu, "Spring: A strategy-proof and privacy preserving spectrum auction mechanism," in *2013 Proceedings IEEE INFOCOM*. IEEE, 2013, pp. 827–835.

[7] Z. Chen, L. Huang, L. Li, W. Yang, H. Miao, M. Tian, and F. Wang, "Ps-trust: Provably secure solution for truthful double spectrum auctions," in *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*. IEEE, 2014, pp. 1249–1257.

[8] Z. Chen, L. Huang, and L. Chen, "Itsec: An information-theoretically secure framework for truthful spectrum auctions," in *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2015, pp. 2065–2073.

[9] Z. Chen, X. Wei, H. Zhong, J. Cui, Y. Xu, and S. Zhang, "Secure, efficient and practical double spectrum auction," in *2017 IEEE/ACM 25th International Symposium on Quality of Service (IWQoS)*. IEEE, 2017, pp. 1–6.

[10] Q. Wang, J. Huang, Y. Chen, C. Wang, F. Xiao, and X. Luo, "*prost*: Privacy-preserving and truthful online double auction for spectrum allocation," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 2, pp. 374–386, 2018.

[11] Q. Wang, J. Huang, Y. Chen, X. Tian, and Q. Zhang, "Privacy-preserving and truthful double auction for heterogeneous spectrum," *IEEE/ACM Transactions on Networking (TON)*, vol. 27, no. 2, pp. 848–861, 2019.

[12] Y. Chen, X. Tian, Q. Wang, M. Li, M. Du, and Q. Li, "Armor: A secure combinatorial auction for heterogeneous spectrum," *IEEE Transactions on Mobile Computing*, 2018.

[13] A. C.-C. Yao, "Protocols for secure computations," in *FOCS*, vol. 82, 1982, pp. 160–164.

[14] F. Baldimtsi and O. Ohrimenko, "Sorting and searching behind the curtain," in *International Conference on Financial Cryptography and Data Security*. Springer, 2015, pp. 127–146.

[15] Z. Chen, L. Chen, L. Huang, and H. Zhong, "On privacy-preserving cloud auction," in *2016 IEEE 35th Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 2016, pp. 279–288.

[16] J. Li, Y. Zhu, J. Yu, C. Long, G. Xue, and S. Qian, "Online auction for iaas clouds: Towards elastic user demands and weighted heterogeneous vms," in *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*. IEEE, 2017, pp. 1–9.

[17] K. Cheng, Y. Shen, Y. Zhang, X. Zhu, L. Wang, and H. Zhong, "Towards efficient privacy-preserving auction mechanism for two-sided cloud markets," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, May 2019, pp. 1–6.

[18] J. Hartline, N. Immorlica, M. R. Khani, B. Lucier, and R. Niazadeh, "Fast core pricing for rich advertising auctions," in *Proceedings of the 2018 ACM Conference on Economics and Computation*. ACM, 2018, pp. 111–112.

[19] Y. Zhang, C. Yin, Q. Wu, Q. He, and H. Zhu, "Location-aware deep collaborative filtering for service recommendation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–12, 2019.

[20] Y. Zhang, K. Wang, Q. He, F. Chen, S. Deng, Z. Zheng, and Y. Yang, "Covering-based web service quality prediction via neighborhood-aware matrix factorization," *IEEE Transactions on Services Computing*, 2019.

[21] Y. Zhang, G. Cui, S. Deng, F. Chen, Y. Wang, and Q. He, "Efficient query of quality correlation for service composition," *IEEE Transactions on Services Computing*, 2018.

[22] X. Zhou, S. Gandhi, S. Suri, and H. Zheng, "ebay in the sky: Strategy-proof wireless spectrum auctions," in *Proceedings of the 14th ACM international conference on Mobile computing and networking*. ACM, 2008, pp. 2–13.

[23] X. Zhou and H. Zheng, "Trust: A general framework for truthful double spectrum auctions," in *IEEE INFOCOM 2009*. IEEE, 2009, pp. 999–1007.

[24] E. Yao, L. Lu, and W. Jiang, "An efficient truthful double spectrum auction design for dynamic spectrum access," in *2011 6th International ICST Conference on Cognitive Radio Oriented Wireless Networks and Communications (CROWNCOM)*. IEEE, 2011, pp. 181–185.

[25] Y.-J. Chen, X.-Y. Yin, and J. Zhang, "A reverse auction framework for hybrid access in femtocell network," *Journal of Computer Science and Technology*, vol. 32, no. 6, pp. 1250–1264, 2017.

[26] X. Feng, Y. Chen, J. Zhang, Q. Zhang, and B. Li, "Tahes: Truthful double auction for heterogeneous spectrums," in *2012 Proceedings IEEE INFOCOM*. IEEE, 2012, pp. 3076–3080.

[27] Z. Zheng, F. Wu, and G. Chen, "A strategy-proof combinatorial heterogeneous channel auction framework in noncooperative wireless networks," *IEEE Transactions on Mobile Computing*, vol. 14, no. 6, pp. 1123–1137, 2014.

[28] Y. Zheng, H. Duan, X. Tang, C. Wang, and J. Zhou, "Denoising in the dark: Privacy-preserving deep neural network based image denoising," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[29] M. Payman and Z. Yupeng, "Secureml: A system for scalable privacy-preserving machine learning," in *Security and Privacy (SP), 2017 IEEE Symposium on*. IEEE, 2017.

[30] O. Goldreich, "The foundations of cryptography, vol. 2, chapter general cryptographic protocols," *Cambridge Univ. Press*, 2004.

[31] D. Demmler, T. Schneider, and M. Zohner, "Aby-a framework for efficient mixed-protocol secure two-party computation." in *NDSS*, 2015.

[32] M. Atallah, M. Bykova, J. Li, K. Frikken, and M. Topkara, "Private collaborative forecasting and benchmarking," in *Proceedings of the 2004 ACM workshop on Privacy in the electronic society*. ACM, 2004, pp. 103–114.

[33] Y. Ishai, J. Kilian, K. Nissim, and E. Petrank, "Extending oblivious transfers efficiently," in *Annual International Cryptology Conference*. Springer, 2003, pp. 145–161.

[34] S. Sakai, M. Togasaki, and K. Yamazaki, "A note on greedy algorithms for the maximum weighted independent set problem," *Discrete Applied Mathematics*, vol. 126, no. 2-3, pp. 313–322, 2003.

[35] Y. Lindell and B. Pinkas, "A proof of security of yaos protocol for two-party computation," *Journal of cryptology*, vol. 22, no. 2, pp. 161–188, 2009.

[36] X. Wang, A. J. Malozemoff, and J. Katz, "Emp-toolkit: Efficient multiparty computation toolkit," 2016.