Chapter 8
The kth-Ranked Element

In this chapter we describe the construction of Aggarwal et al. [1] for securely computing the kth-ranked element of the union of two distributed and sorted data sets in the presence of semi-honest and malicious adversaries. An important special case of this problem is that of securely computing the median. The construction of [1] is based on the iterative protocol of [74] for computing the kth-ranked element with low communication complexity. The secure implementation of this fundamental problem has been looked at by researchers in the last few years, mostly due to its potential applications. One particular setting is where the data sets contain sensitive data, yet the particular kth element is of mutual interest. For instance, two health insurance companies may wish to compute the median life expectancy of their insured smokers, or two companies may wish to compute the median salary of their employees. By running secure protocols for these tasks, sensitive information is not unnecessarily revealed. In this chapter we show how to securely compute the two-party kth element functionality  \( F_{kth} \), where each party enters a set of values from some predetermined domain and the output is the kth element of the sorted list comprised of the union of these sets.

8.1 Background

The functionality computing the kth-ranked element is defined as follows:

**Definition 8.1.1** Let X and Y be subsets of a domain \( \{0,1\}^{p(n)} \) for some known polynomial \( p(n) \). Then, the functionality  \( F_{kth} \) is as follows:

\[
((X,k),(Y,k)) \mapsto \begin{cases} ((X \cup Y)_k,(X \cup Y)_k) & \text{if } |X \cup Y| \geq k \\ (\perp, \perp) & \text{otherwise} \end{cases}
\]

where \((\Gamma)_i\) denotes the ith element within the sorted set \( \Gamma \).

We remark that the protocol of [1] presented here in Section 8.3 that is secure in the presence of malicious adversaries is a rare example of a protocol that achieves this level of security without having the parties commit to their inputs at the onset of the protocol (as in the general construction in Chapter 4). Rather, the simulator is able to construct the inputs used by the
adversary dynamically, by continually rewinding the adversary. This is feasible in this context because the communication complexity of the protocol is logarithmic in the input length, and so continued rewinding of the adversary (which can be exponential in the communication complexity) yields a polynomial-time strategy.

8.1.1 A Protocol for Finding the Median

We begin by providing a detailed description of a protocol for computing the median with only logarithmic communication; this protocol is due to [74]. This protocol is not supposed to be secure; however, it forms the basis of the secure protocols presented below. We focus our attention on the case in which the parties compute the median of two equally-sized disjoint sets, where the size of each set is a power of 2. (By definition, the median of a set $X$ with an even number of elements is often taken to be the mean of the two middle values. Yet, in this chapter we consider the element ranked $\lceil |X|/2 \rceil$.) We stress that the general case can be reduced to this simpler instance; see below for more details. The protocol is repeated in rounds where in each round the parties compare the medians of their updated sets, and then remove elements from their sets accordingly. Specifically, if, for instance, the median of party $P_1$ is larger than the median of $P_2$, then $P_1$ removes from its set the elements that are larger than or equal to its median, whereas $P_2$ removes the elements that are smaller than its median, and vice versa. The protocol is concluded when the data sets are of size 1, yielding that the number of iterations is logarithmic in the size of the sets. Let $\Gamma$ be a sorted list with $\ell$ items. We write $\Gamma^\top$ to denote elements $\gamma_{\ell/2+1}, \ldots, \gamma_\ell \subset \Gamma$ and by $\Gamma^\perp$ denote the elements $\gamma_1, \ldots, \gamma_{\ell/2} \subset \Gamma$. The formal details of the protocol are given in Protocol 8.1.2.

**PROTOCOL 8.1.2 (A Protocol for Computing the Median – FindMed)**

- **Input:** Disjoint data sets $X$ for party $P_1$ and $Y$ for party $P_2$, with $|X| = |Y| = 2^\ell$ for some $\ell$.
- **Output:** The median $m$ of $X \cup Y$.
- **The protocol:**
  1. If $|X| = |Y| = 1$, then output $\max(X, Y)$.
  2. Else (if $|X|, |Y| > 1$):
     a. $P_1$ computes the median $m_X$ of $X$ and sends it to $P_2$.
     b. $P_2$ computes the median $m_Y$ of $Y$ and sends it to $P_1$.
     c. If $m_X > m_Y$, then $P_1$ sets $X = X^\perp$ and $P_2$ sets $Y = Y^\top$.
     d. Else (if $m_X < m_Y$), then $P_1$ sets $X = X^\top$ and $P_2$ sets $Y = Y^\perp$.
     e. Return $\text{FindMed}(X,Y)$.

A protocol for computing the median with logarithmic communication