Review of[5]
**Efficient Secure Two-Party Protocols:
Techniques and Constructions**
by Carmit Hazay and Yehuda Lindell
Springer, 2010

Reviewed by Jonathan Katz
Dept. of Computer Science, University of Maryland

1 Introduction

Textbooks and monographs (where a monograph is distinguished from a textbook by a monograph's focus on a relatively specialized topic) can play a significant role in defining a field. Books that are widely used in graduate courses can shape students' (and professors'!) perspectives on the subject; a text that becomes a 'bible' for graduate students can influence the direction a field moves. Textbooks and monographs can even have influence outside their own community, if they provide a treatment that is accessible to a broad audience from a wide range of backgrounds.

This potential impact of monographs (and, to a lesser extent, textbooks) appears under-appreciated in our community. More to the point, there seem to be fewer texts available than there "should" be — observe how often graduate computer science courses are taught without any assigned textbook, or how difficult it can be to generate a reading list for a beginning graduate student — and certainly there are fewer TCS-focused monographs than there are in disciplines such as chemistry or mathematics, to take two examples. While there may be legitimate reasons that partially account for this, the result is to the overall detriment of our field.

This situation has, thankfully, begun to change. Several excellent textbooks have become available in the past few years, and we have also witnessed the publication of many monographs that have become quite popular within their own niche areas. The book under review, constituting a detailed treatment of efficient secure two-party computation suitable for a graduate seminar or self-study, continues this positive trend. (Full disclosure: I have co-authored papers with both authors, and have written a textbook with one of them.)

2 Summary of the Book

The focus of this book is efficient protocols for secure two-party computation. This area of research has experienced a surge of interest lately, both within the cryptography and security research communities as well as in terms of government funding (at least in the US and EU) for work on the problem. Secure computation itself has a long history dating back to the early 1980s. Roughly, a protocol for secure computation of a function f allows two parties, holding inputs x and y respectively, to jointly compute $f(x, y)$ while ensuring several security properties, chiefly *privacy* (nothing is revealed to either party about the other party's input beyond what is revealed by $f(x, y)$) and *correctness* (neither party can cause the other party to output an incorrect result). Seminal results of Yao and Goldreich, Micali, and Wigderson show that *any* (polynomial-time) function f can be computed securely, based on standard cryptographic assumptions, in either the semi-honest

[5] ©2012 Jonathan Katz

or malicious settings and for any number of corrupted parties. (In the semi-honest setting, parties are assumed to follow the protocol but may then try to learn additional information about the other parties' inputs. In the malicious setting, parties can behave arbitrarily.) These results are among the most important in the field of cryptography, and contain beautiful ideas that every theoretical computer scientist should be aware of.

Initial results on secure computation focused primarily on feasibility. More recently researchers have developed and implemented highly efficient protocols for certain tasks. Roughly, research has progressed in three (overlapping) directions:

- Exploring (reasonable) weakenings of the security definitions in the hopes that these relaxed definitions will enable more efficient protocols.

- Optimizing protocols for "generic" secure computation that can be applied to arbitrary functions represented as boolean circuits.

- Developing efficient "special-purpose" protocols for particular functions of interest. These protocols exploit specific properties of the functions under consideration, and so do not (necessarily) rely on a boolean-circuit representation.

The present book describes results of the authors in each of the above areas. Following an overview of the book in Chapter 1, the book presents formal definitions of security for secure computation in Chapter 2. In addition to defining the "standard" notions of semi-honest and malicious security, the book also includes various intermediate notions including nonsimulation-based definitions of security and a newer definition proposed by Aumann and Lindell called *covert security.* Roughly, covert security does not *prevent* a malicious adversary from "cheating" and potentially violating various security guarantees; rather, it merely guarantees that any such cheating will be *detected* with high probability (at which point the second party can seek legal recourse). It has been argued that in many real-world settings, this threat of being caught will be sufficient to deter malicious behavior in the first place.

Chapters 3–5 describe "generic" protocols for secure computation of arbitrary functions f based on a boolean circuit computing f. Chapter 3 contains a full description of Yao's protocol for semi-honest adversaries, along with a detailed proof of security. (Amazingly, Yao's original paper includes neither the details of the protocol nor a security proof!) Chapters 4 and 5 discuss how to extend Yao's protocol so as to achieve malicious security and covert security, respectively.

The remainder of the book focuses on efficient protocols for specific functions. Thus includes several protocols (achieving different notions of security) for *oblivious transfer*, a fundamental primitive that is used as a building block of the generic protocols for secure computation mentioned above; protocols for oblivious pseudorandom function evaluation; for finding the median (or, more generally, the kth-ranked element) of two lists held by the parties; for keyword search; and for pattern matching.

If there is a fault with the book, it is that it focuses largely (though not exclusively) on results of the authors. These results do tell a coherent story; still, there were several results by other researchers that, in my mind, should have been included in a survey text of this sort. One prominent example is the work on *oblivious transfer extension*, which allows a large number of oblivious transfers to be carried out at essentially the cost of only k oblivious transfers (for k a statistical security parameter) and is the method of choice in practice when oblivious transfer is implemented. The authors also do not cover any results in the *random-oracle model.* Although there may be valid theoretical justification for omitting such results, protocols in the random-oracle model are generally more efficient and would again be preferred in practice. It would have been helpful if the authors had at least included pointers to this other literature at the end of every chapter.

3 Recommendation

Overall, the book is a pleasure to read, containing sufficient motivation, intuition, and informal discussion as well as detailed proofs of security. The book contains a superb treatment of both general secure two-party computation as well as several efficient protocols in this setting. The first three chapters of the book would serve as an accessible introduction to secure two-party computation for the interested graduate student; the rest of the book is an excellent starting point for the more specialized literature in the field. The book could also serve very nicely as a text for a graduate seminar in this area, or could even be used as a supplementary book at the end of a graduate "Introduction to Cryptography" class. (I am planning to use it this way later this semester.) It belongs on the shelf of every researcher interested in this area.