Yao's Garbled Circuits: Recent Directions and Implementations

Peter Snyder
University of Illinois at Chicago
Chicago, Illinois, USA
psndye2@uic.edu

ABSTRACT

Secure function evaluation, or how two parties can jointly compute a function without any other party learning about any other party's inputs, has been an active field in cryptography. In 1986 Andrew Yao presented a solution to the problem called *garbled circuits*, based on modeling the problem as a series of binary gates and encrypting the result tables. This approach was initial treated as theoretically interesting but too computationally expensive for practical use. However, in the decades since Yao's solution was initially published, much work has gone into both optimizing the protocol for practical use, and further securing the protocol to make it further secure.

This paper provides a thorough explanation of both Yao's original protocol and and its security characteristics. The paper then details additions to the protocol to make it both practical for computation and secure against untrusted parties. Implementations of Yao's protocol are also discussed, though the paper's emphasis is on the underlying enabling improvements to the protocol.

1. INTRODUCTION

Secure function evaluation (SFE) referrers to the problem of how can two parties collaborate to correctly compute the output of a function without any party needing to reveal their inputs to the function, either to each other or to a third party. A common example of this problem is the "millionaires problem", in which two millionaires wish to determine which of them has more money, without either party revealing how much money they have[5].

Many solutions have been developed for SFE. One category of solution is function specific, and depends on specific attributes of the function being executed to provide security[4]. These solutions, while interesting, are by definition of less general interest, since they apply to only a limited set of problems.

Another category of approach is more general, and seeks to provide a general solution for SFE by transforming arbitrary functions into secure functions. Approaches in this category include homomorphic encryption systems[2] which allow for arbitrary execution on encrypted data. Yao's *garbled circuits* protocol fits in this second category.

Yao's *garbled circuits* protocol transforms any function into a function that can be evaluated securely by modeling the function as a boolean circuit, and then encrypting the inputs and outputs of each gate so that the party executing the function cannot discern any information about the inputs or intermediate values of the function. The protocol is secure as long as both parties follow the protocol. A full description of the protocol and the related security definitions are provided later in this paper.

1.1 History of Protocol

Interestingly, Yao never published his *garbled circuits* protocol. Several of his publications discuss approaches to the SFE problem generally, specifically papers from 1982[5] and 1986[6]. These papers are much broader in scope and are much more abstract than providing a protocol that could be implemented. Yao first discussed the *garbled circuits* approach in a public talk on the latter paper, as a concrete example of how his broader strategies could be applied[1]. Only later and by other researchers would the protocol be documented formally[3], though still crediting Yao for the approach.

Yao having developed this foundational protocol, but never having published it, presents authors with the tricky question of what to cite when crediting to the *garbled circuits* approach. The common approach seems to be to cite Yao's two papers discussing his general approach the problem, even though those papers make no mention of garbled circuits or any similar concept.

1.2 Aims of the Paper

This paper aims to provide a full description of Yao's *garbled circuit*protocol and its security characteristics, namely what security the protocol does and does not provide. This paper also provides detailed explanations of related work done by other authors to improve the performance and security provided by the protocol.

This paper presumes no previous familiarity with Yao's protocol or cryptography in general in the explanation explanation of the protocol, beyond the general concepts of symmetric and asymmetric cryptography. Some background in cryptography is assumed in the sections on improvements and additions to the protocol. Formal proofs of the underlying concepts are not discussed and are left to their originating papers.

Some discussion is included of existing implementations of Yao's protocol. However, the focus here is on the promises, improvements and general techniques of the implementations, and not on implementation details like programming languages or hardware characteristics. Discussion of the implementations is mainly meant to inform how the protocol has developed and been improved, as opposed to a detailed comparison of how different implementations compare with each other.

1.3 Organization of the Paper

The remainder of the paper is structured as follows. Chapter 2 provides some security definitions used throughout the rest of the paper. Chapter 3 then provides a full explanation of the Yao's protocol and how to use *garbled circuits* to solve the *SFE* problem. Chapter 4 discusses oblivious transfer (OT), its role in the protocol, and a method for achieving OT in a manner that is compatible with the security guarantees of Yao's protocol. Chapter 5 discusses some of the security and performance limitations of Yao's protocol, and Chapters 6 and 7, respectively, discuss subsequent developments to Yao's protocol to address these issues. Chapter 8 provides a brief overview of some implementations of the protocol, and Chapter 9 concludes.

2. SECURITY DEFINITIONS

This section defines several security related terms that are used through out this paper. The terminology is not identical throughout the literature, but have mappings onto similar, equivalent terms.

2.1 Ideal Model

Attempting to abstractly but precisely defining the characteristics of a SFE protocol is difficult and can quickly devolve into a long enumeration of characteristics a SFE system should *not* do. Instead, Yao suggests[6] that a correct system should be compared to an ideal-oracle that fulfills three properties, and that a SFE system is correct if it performs identically to this imagined ideal-oracle.

This imagined ideal-oracle takes a function to execute (f), the first party (*P1*)'s input (i_{P1}) and the second party (*P2*)'s input (i_{P2}), executes the given function with the values provided, and then returns the function's output to both parties ($u \leftarrow f(i_{P1}, i_{P2})$).

2.1.1 Validity

A SFE system must be able to correctly calculate the given function and produce the same conclusion or output that a non-secure version of the function would produce. Note that this does not guarantee a correct result, since the function being computed in a secure manner could itself have a logic error in it, nor does it guarantee to produce any answer, if one of the parties submits an invalid input to the computation. This *validity* requirement merely requires that the function produce the same result as the insecure (or "pre-secured") version of the function being evaluated, given the same inputs.

2.1.2 Privacy

A SFE system must protect the privacy of both parties inputs by ensuring that if *P1* follows the protocol, *P2* is not able to learn anything about i_{P2} beyond what *P1* could learn by providing i_{P1} to the ideal oracle and examining u.

Note that that this definition of *privacy* does not guarantee that *P1* is not able to learn *P2*'s input by examining the function's result (if the function being executed allows for such reverse engineering). If, for example, the function being evaluated securely is multiplication, the fact that *P1* can learn i_{P2} through u/i_{P1} does not violate this *privacy* property, only that integer multiplication is not a function that makes sense in a two-party SFE computation.

2. Security Definitions - Ideal Model * Yao, How to Generate and Exchange Secrets [1986] - modeling trusted computing third party in three ways - validity (ie generate equally correct results in polynomial time) - privacy (ie if both parties follow the protocol, then neither party learns about the others inputs) - fairness (ie if one party cheats, they cannot get the right output while denying the other party the correct answer) - Semi-Honest (Honest but Curious) * Secure multi-party computation Goldreich, Oded. "Secure multi-party computation." Manuscript. Preliminary version (1998). - A protocol is secure under the Semi-Honest requirement then if it matches the Ideal model if all players follow the requirements of the protocol - Malicious Model * Secure multi-party computation Goldreich, Oded. "Secure multi-party computation." Manuscript. Preliminary version (1998).

References

[1] M. Bellare, V. T. Hoang, and P. Rogaway. Foundations of garbled circuits. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 784–796. ACM, 2012.

[2] C. Gentry. *A fully homomorphic encryption scheme.* PhD thesis, Stanford University, 2009.

[3] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 218–229. ACM, 1987.

[4] Y. Huang, D. Evans, J. Katz, and L. Malka. Faster secure two-party computation using garbled circuits. In *USENIX Security Symposium*, volume 201, 2011.

[5] A. C.-C. Yao. Protocols for secure computations. In *FOCS*, volume 82, pages 160–164, 1982.

[6] A. C.-C. Yao. How to generate and exchange secrets. In *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pages 162–167. IEEE, 1986.