International Journal of Advanced Research in

Computer Science Engineering and Information Technology

Volume: 3 Issue: 1 26-Jun-2014,ISSN_NO: 2321-3337

Secure MultiParty Computation

[1.] Dr.K.Ravikumar, [2.] B. Gnanam Jeyanthi

Asst. Professor Dept. of computer Science, Research scholar

Department of Computer Science

Tamil University, Thanjavur

ABSTRACT--*In Secure Multiparty computation model, each processes contain a tamper-proof security module. Security modules can be trusted by other processes and can establish secure channels between each hosts. The availability of security modules is restricted by their hosts because a corrupted party can stop the execution of its own security module and also drop message sent by or to its security module. We show that SMC can be implemented to reduce faults based on the security modules of its own.*

1, INTRODUCTION

Secure Multiparty computation is a subfield of cryptography. The goal of this field is to create methods that enables parties jointly compute a function over their inputs, while at the same time keeping these inputs private. An important primitive in MPC is *oblivious transfer*. Unconditionally or information theoretically secure MPC is closely related to the problem of *secret sharing*. Secure Multiparty computation is one of the most fundamental security problem. Consider, a set of *n* parties jointly want to compute the result of an *n*-ary function F. Every party provide its own(*private*) input to this function but the inputs should remain secret to the other parties, except what can be derived from the result of F. The problem is easy to solve if you assume the existence of a trusted third party (TTP) which collects the inputs, computes F and distributes the result to everyone. However, the problem is very challenging if you assume that there is no TTP available and parties can misbehave arbitrarily, that, they can send wrong messages or fail to send messages at all. Still, the protocol must correctly and securely compute F as if a TTP were available.

SMC can be used to solve various real life problems such as distributed voting, private bidding and online auctions, sharing of signature or decryption functions and so on. Unfortunately, solving SMC is without extra assumptions very expensive in terms of communication (number of messages), resilience (amount of redundancy), and time(number of synchronous rounds).

2, PROCESSES AND CHANNELS

The system consists of a set of processes interconnected by a synchronous communication network with reliable secure bidirectional channels. Two Processes connected by a channel are said to be adjacent.

A reliable secure channel connecting processes X and Y satisfies the following properties:

- *No loss* (No messages are lost during the transmission over the channel.
- *No Duplication*(All messages are delivered at most once)
- *Authenticity*(If a message is delivered at Q, then it was previously sent by P)
- *Integrity* (Message contents are not tampered with during transmission. i.e., any changer during the transmission will be detected and the message will be discarded.)
- *Confidentiality* (Message Contents remain secret from unauthorized entities).

In a synchronous network, communication proceeds in rounds. In each round, a party first receives inputs from the user and all messages sent to it in the previous round(if any), processes them and may finally send some messages to other parties or give outputs to the user.

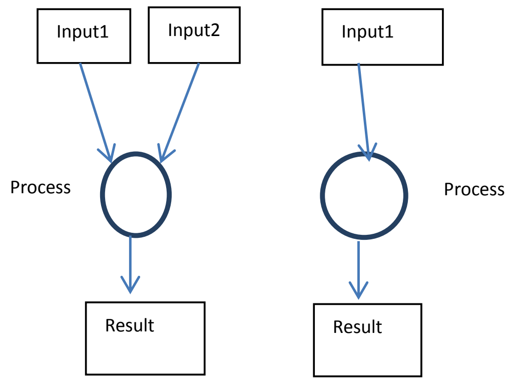
3, FRAMEWORK

We introduce a transformation framework that systematically transforms normal computations to secure multi-party computations. We start from describing two different models of computation (without the privacy requirements), and then we show how to transform them to models enhanced with privacy requirements, thus generating new SMC problems. The model after the transformation is the Secure Multi-party Computation (SMC) model. According to the number of distinguished inputs, we classify computations into two different models:

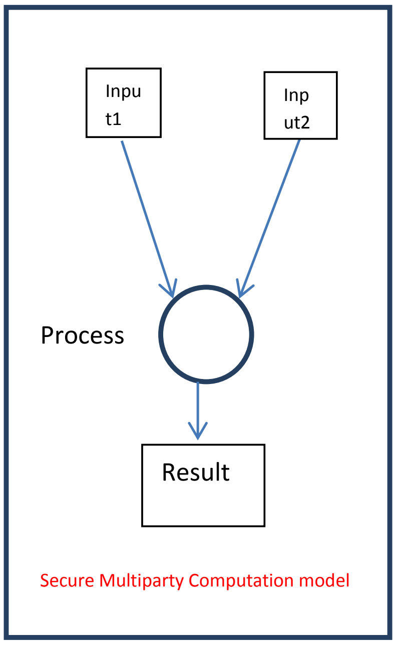
- The multi-input computation model and
- The single-input computation model.

The multi-input computation model usually has two distinguishable inputs. For instance, client/server computation is a multi-input computation model. The single-input computation model usually has one input or one set of inputs.

Next we want to transform both models to the Secure Multiparty Computation model, in which, the input from each participating party is considered as private, and nobody is willing to disclose its own inputs to the other parties. In certain specific cases, the computation results could also be private, namely some party should not learn the results. For the multi-input computation model, its transformation to the corresponding SMC model is straightforward because the model naturally has at least two inputs.

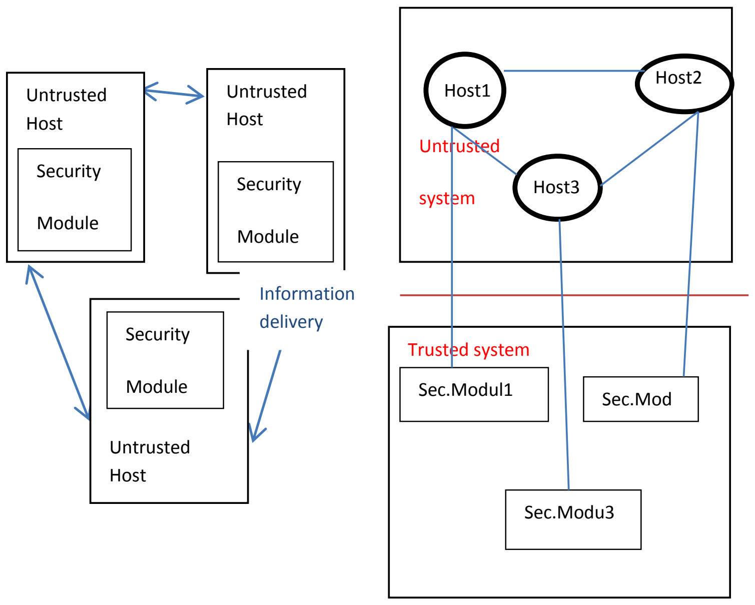


Multi Input Computation Model Single Input Computation Model



Secure Multiparty Computation model

Assumes Synchronous settings..



[4]Untrusted hosts and Security Modules:

The set of processes is divided into two disjoint classes: Untrusted hosts and Security modules. We assume that there exists a fully connected communication topology between the hosts, i.e., any two hosts are adjacent. We denote by n the number of hosts in the system. Since the security modules trust each other we called it as the trusted system.

Formally, the connection between the untrusted and trusted system is achieved by associating each process in the untrusted system (i.e., each host) with exactly one process in the trusted system (i.e., security module) and vice versa.

5, SMC (Secure Multiparty Computation)

In secure multi-party computation(SMC), a set of processes p1,…,p*n*, each starting with an input value xi, wants to compute the result of a deterministic function F, i.e., *r*=F(x1,…..,x*n*) while making sure that the following conditions are satisfied:

Correctness: the correct value of *r* is computed and

Privacy : *r* is the only new information that is released.

Computing r such that privacy and correctness are achieved is referred to as computing *r* securely. Result *r* should be computed reliably and securely, i.e., as if they were using a trusted third party (TTP). This means that the individual inputs remain secret to other processes and that malicious processes can neither prevent the computation from taking place nor influence *r* in favorable ways. We assume that F is a well-known deterministic function with input domain X and output domain Y upon which all processes have agreed upon beforehand and that all correct processes jointly begin the protocol. We say that r is an F-result if r was computed using F. Since faulty process cannot be forced to submit their input value, F may be computed using special values instead of the input value of a faulty process.

The protocol is a set of instructions or rules for the parties to follow for sending and receiving messages. A protocol solves secure multiparty computation (SMC) if it satisfies the following properties:

- SMC validity – If a process receives an F-result, then F was computed with at least the inputs of all correct processes.
- SMC Agreement – If some process pi receives F-result r_i and some process p_j receives f-result r_j then r_i=r_j.
- SMC Termination – Every correct process eventually receives an F-result.
- SMC Privacy – Faulty processes learn nothing about the input values of correct processes.

The objectives of SMC are,

- ➢ To define and compare single and multi-third party computing model
- ➢ To analyse and store the behavior of third parties in several rounds of computation.
- ➢ To identify and reduce malicious conduct of trusted third parties in order to obtain correct results of computation.
- ➢ To make computational secured.

6, Maintaining secrecy in Trusted systems:

We have to carefully analyze the security properties of the protocols which run within the trusted system in order to maintain confidentiality and be able to implement SMC-privacy.

7, Trusted System - Security Properties of Protocols

A protocol running in the trusted system satisfies the following two properties to be of use as a building block in SMC:

- Content Secrecy – Hosts cannot learn any useful information about other hosts input from observing the messages in transit.
- Control Flow Secrecy – Hosts cannot learn any useful information about other hosts input from observing the message pattern.

8, Conclusion

The need of using multiparty computation model is that of privacy concern as parties providing inputs for computation may not be able to know the third party computation as the TTPs are selected at run time from the pool of TTPs. While using multiparty environment for computation, different cases were studied for identification of malicious conduct by TTPs. The behavior of TTPs are analysed in several rounds.

References

1)Roberto Cortin˜ as, Felix C. Freiling, Marjan Ghajar-Azadanlou,Alberto Lafuente, Mikel Larrea, Lucia Draque Penso, and Iratxe Soraluze, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, Secure Failure Detection and Consensus in TrustedPals, VOL. 9, NO. 4, JULY/AUGUST 2012.

*2) Zinaida Benenson1, Milan Fort2, Felix Freiling3,
Dogan Kesdogan2, and Lucia Draque Penso3 TrustedPals: Secure Multiparty Computation Implemented with Smart Cards.*

3) INTERNATIONAL JOURNAL OF INNOVATIVE TECHNOLOGIES, VOL. 01, ISSUE 05, OCT 2013 ISSN 2321 –8665,Secure Failure Detection and Aggrements In Trusted Pals.

4)Ivan damgad,Yuval Ishai, and Mikkel korigaard,Perfectly Secure Multiparty Computation and the computational overhea of cryptography, H.Gilbert(Ed.):EUROCRYPT 2010, LNCS 6110, PP. 445 -465,2010.

5) Iwan damgard, and Yuval Ishai, Scalable Secure Multiparty Computation, C.DWork(Ed.):CRYPTO 2006, LNCS 4117, PP.501-520, 2006,

6) YiSun,[1] Qiaoyan Wen,[1] Yudong Zhang,[2] Hua Zhang,[1] and Zhengping Jin[1] Efficient Secure Multiparty Computation Protocol for Sequencing Problem over Insecure Channel,Volume 2013 (2013),Article ID 172718, 5pages.