INFO-F-514 - Year Report

Université Libre de Bruxelles

Hakim Boulahya

June 18, 2018

Contents

1 Secure computation **2**
1.1 Introduction . . . 2
1.2 Secure computation . . . 2
1.3 Oblivious Transfer . . . 2
1.3.1 Definition and variants . . . 2
1.3.2 1-out-of-2 Oblivious Transfer protocol . . . 3
1.4 An application: Yao's millionaires problem . . . 3
1.4.1 Ioannidis and Ananth solution . . . 4
1.4.2 Lin and Tzend solution . . . 4
1.5 Conclusion . . . 5

2 Conference #1: FAB Framework **6**
2.1 Introduction . . . 6
2.2 Blockchain . . . 6
2.3 Scalability . . . 6
2.4 FAB Overview . . . 7

3 Conference #2: Empowering the cashless society **8**
3.1 Introduction . . . 8
3.2 Fundamentals of security . . . 8
3.3 Cryptographic protocols and key management . . . 8
3.3.1 DUKPT . . . 8
3.3.2 Hardware Security Modules . . . 9

1 Secure computation

1.1 Introduction

In this paper we will propose formal definitions of secure computation, also refered as secure multiparty computation. We will also gives known example in the litterature that are defined following the logical of secure computation. We will then recall state of the art techniques and protocols that allow to resolve secure computations.

1.2 Secure computation

A secure multi-party computation problem, is a problem where a computation, or a result, must be computed but the input that each party must used is confidential and not share between all parties. Such problem can be defined as a function $f(\cdot)$, that takes n parameters. The idea is to be able to compute the function $f(x_0, .., x_n)$ where the input x_i can only be accessed by the party i. The final results is accessible to everyone.

The first secure computation problem was first introduced by Yao in 1985 [Yao82], with the millionnaires problem. This problem is a secure two-party computation, a subproblem of multi-party computation problem. Unlike other cryptographic protocols, the malicious behaviour come from the participant in the exchange. Indeed in secure computation we can define two party behaviour. A *semi-honest adversary* is a party in the protocol that will always follow the steps that must be performed as stated in the protocol. That is, a semi-honest adversary will always send well-formed messages. It is not fully honest because such adversaries will try to learn other participants secrets by analyzing their protocol messages. A *malicious adversary* can behave in the worst way possible. That is it can deviate from the protocol, send mal-formed message, and can use any other possible way to find the other parties secrets. When implementing such protocols, it is necessary to take in consideration the behaviour of the parties, that is the security of the protocol may rely on the honesty of the participants.

1.3 Oblivious Transfer

1.3.1 Definition and variants

The Oblivious Transfer introduced by Rabin during in 1982 [Rab]. The Oblivious Transfer has many application and has been first introduced has a protocol to resolve the Exchange Of Secret problem. The oblivious transfer protocol is defined as follow: a sender want to send a message to a receiver, but it must not be able to tell if the receiver got the message, that is there is a probability of $\frac{1}{2}$, that the message has been sent to the receiver.

In the context of secure computation, The 1-out-of-2 Oblivious Transfer (OT_2^1), an another approach to the original Oblivious Transfer, is used. It is the problem that for a sender and a receiver, one of two message must be sent from the receiver to the sender. The message receive can be chosen by the receiver. Two constraints are that the sender must never know which message has been chosen, and the receiver must not know the content of the other message.

There exists also 1-out-of-n Oblivious Transfer (OT_n^1) is an extension of OT_2^1, where the sender has n messages to send and the receiver must choose one of them. Those two protocols are theorically equivalent has proven in [Cre88, Cac98].

1.3.2 1-out-of-2 Oblivious Transfer protocol

On possible protocol for the OT_2^1 problem is by using a pair of key using the RSA protocol, first proposed in [EGL85]. Let Alice be the sender and Bob the receiver. Alice has two messages m_0, m_1, and in addition to that a public RSA key (e, d, n). The protocol is a multi-step communication between the two parties using the RSA public key of Alice.

The first step is for Alice to send the public key and two random values, that is the public key (e, n) and two random values x_0, x_1 contains in the domain $[1, n-1]$. Now that Bob has those inputs, he will generates on his side two other random values. The first one is the bit b which value is either 0 or 1, and is used to choose which random inputs received from Alice, that is x_b would be either x_0 or x_1. The second generated random value of Bob is a value k in the domain $[1, n-1]$.

The second step is for Bob to return his response. Since we don't want Alice to know which value has been chosen, Bob will encrypt the value x_b by blinding it using the random value k that he generated. That is Bob will send to Alice the value $v = (x_b + k^e) \mod n$. Upon receival of v, Alice will decrypt v two times, by removing the random values. That is, Alice will have two values k_0, k_1 where $k_i = (v - x_i)^d \mod n$.

Finally, the last step if for Alice to send back the real message. Since v was blinded by Bob with the value k, Alice doesn't know which random x_i has been chosen. By computing the two k_i based on both values, one of them will be identical to the k value of Bob. The last inputs that Alice will send to Bob are the two messages m'_0, m'_1 where $m'_i = m_i + k_i$. Upon receival, Bob will have to decrypt the message with k, that is $m_b = m'_b - k$. Since Bob only has the k_i value associate to his message, he will not be able to decrypt the other message.

1.4 An application: Yao's millionaires problem

Oblivious Transfer provides a protocol to share inputs without giving to much informations to the other parties. It is still necessary to provide a protocol that will compute the function for the secure computation, and share the results among parties. With this objective, we will focus on a well known problem in secure computation, that is Yao's millionaires problem.

The millionaires problem introduced by Yao in [Yao82], is the problem that for two millionaires they both want to know which one of them is the richer, but they don't want to know the difference. In this problem, the computation function is the usual comparison $<$, and the inputs are the incomes of the individuals. We will present two solutions of the problem, where both use Oblivious Transfer protocol to securely exchange informations. The first solution is Iaonnidis and Grama [IG03] where they propose a solution using 1-out-of-2 oblivious transfer that construct parallel computation of the comparison, using XOR operations. The second solution is the one proposed by Lin and Tzeng [LT05], that uses homomorphic encryption scheme.

1.4.1 Ioannidis and Ananth solution

Ioannidis and Ananth protocol to resolve the millionaires problem [IG03] is divided in five major steps, and make used of the 1-out-of-2 oblivious transfer protocol.

Alice and Bob wants to know which one is richer that the other. Let a be the number of millions she posses and b the amount of Bob. Before the computation, they first both agree that $a, b < d$, for $d \in \mathbb{N}$.

Protocol The goal of this protocol is for Alice to first create a matrix where all elements are values of k bits, where k is the size of the RSA key of Alice, used in the Oblivous Transfer. The first step for alice is to set the matrix to specific values depending on the key size k, and bits of her value a. The second major steps is to generate $d-1$ k bits random numbers S_i. The create another value S_d so that all the bits are random except for the the last, that is $k-1$ and $k-2$. Those two last bits are set using the bitwise XOR operation, based on the S_i random values and the generated matrix. Then a rotation of the elements in the two first column is made, by rotating the the cell with the second random value generated by Alice at the beginning. The resulting matrix is send to Bob, which will resend using the oblivious transfer protocol for each line i, the value at the column $b_i + 1$, where b_i is the i^{th} bit of b. At reception, Alice will rotate the S_i values using her generated random numbers, and send back the result to Bob. This will allow Bob to scan the value from the matrix at indices $b_i + 1$ (as explained above) and the S_i rotated values received from Alice. The scan should reveal a large sequence of zero bits. If the bit to the right of the sequence is equals to 1 then $a \geq b$, otherwise $a < b$.

1.4.2 Lin and Tzend solution

Preliminaries Their protocol is based on multiplicative homomorphic encryption scheme, and on binary encodings, that are 0-encoding and 1-enconding.

A multiplicative homomorphic encryption function is function that when computed with a specific operator $\circ$, the results is equal to the encryption of the multiplication of the message, that is:

$$E(m_1) \circ E(m_2) = E(m_1 \times m_2)$$

Regarding the encoding, a 0-encoding for a binary number b of length n is a set S_s^0 of binary strings such that $S_s^0 = \{s_n s_{n-1} ... s_{i+1} 1 | s_i = 0, 1 \leq i \leq n\}$. A 1-encoding of s is the set $S_s^1 = \{s_n s_{n-1} ... s_i | s_i = 1, 1 \leq i \leq n\}$. The goal of using such encoding is to be able to reduce the millionaires problem to the set intersection problem. Indeed, Let S_x^1, S_y^0 be respectively the 1-encoding of a binary number x and the 0-encoding of a binary number y, $x \leq y$ if and only if $S_x^1 \cap S_y^0 = \emptyset$.

Protocol The secure computation protocol is divided in 3 steps. Let a be Alice private input, and b Bob privates input, both binary number of size n. They want to know if $a < b$ without sharing their inputs. Alice will choose an homomorphic encryption scheme (G, E, D), respectively a random generator, an encryption function and a decryption function. They will do the following exchange:

1. Alice: Generate a key pair from G (pk, sk) for E and D. Generate a square matrix T of size n where the elements at positions $(a_i, i) = E(1)$ and $T(\overline{x}_i, i) = E(r_i)$, where r_i is a random number generated by G. This matrix is send to Bob.

2. Bob: For all elements s in S_b^0, he will compute a value $c_s = T[t_n][n] \circ T[t_{n-1}][n-1]... \circ T[t_i][i]$. Then he will scalarize and randomly permute c_s values. Send to Alice all c_s values.

3. Alice: Decrypt $D(c_i) = m_i$ for all $1 \leq i \leq n$, and determize $x < y$ if and only if any $m_i = 1$.

1.5 Conclusion

We have presented secure computation and the different protocols that were implemented to resolve such problem. We summarize two solutions, one using the oblivious transfer protocol, and one using homomorphic encryption scheme. There exists other solutions for the protocol, including the one introduced in the original paper [Yao82] that presented secure computation. Because the goal of the paper is to introduce secure computation and summarize solution to the problem, we only presented some of the *recent* efficient solutions.

2 Conference #1: FAB Framework

2.1 Introduction

During the conference of 28 March, Mohammed El Kandri presented Fast Access Blockchain, a framework that allow to overcome the scalability of blockchain technology. In this paper, we will first introduce the blockchain and the bitcoin applications. We will the explain what is the scalability problem. Finally we will present the Fast Access Blockchain framework and how it propose to deal with the scalability problem.

2.2 Blockchain

Blockchain is a technology that allow to decentralize a system to avoid using a thrusted third-party, such as a bank. The blockchain technology was first introduced by Satoshi Nakamoto [Nak08] within the framework of the Bitcoin blockchain, a decentralized network that allow to exchange coins in form of transactions.

The different techniques first presented in [Nak08], are a combination of different cryptographic functions that allow this massive scale decentralization. The goal is to make sure that a transaction that is acknowledge by the network, can be thrusted where no users can be thrusted. The different cryptographic protocols used allow to insure the will of a user to send his coins, avoid double-spending and insure that nobody in the network can change any past transactions.

Let's summarize the Bitcoin protocol. A coin is represented by a sequence of transactions. When Alice wants to send Bob a coin, she musts signed a SHA-256 hash, using the ECSDA digital signature algorithm, composed of the last transaction of the coin and the one that she wants to create. This new transaction will be send to the decentralized network, usually a peer-to-peer network. Miners in the network will provide a computational work called proof-of-work. The goal of the proof-of-work is to mint a block, a file containing multiple transactions, using a cost-function of the hashcash [Bac02] (the proof-of-work) algorithm. A cost-function should be easy to verify but hard to compute. This mechanism allow to create a Blockchain, that can be seen as a linked list of blocks, where the proof-of-work make it hard to edit the blockchain because each added block is linked to the previous on in the blockchain.

2.3 Scalability

One problem that arise using this protocol is the scalability problem. Because the number of transactions per block is fixed (the block creation frequency is also fixed) and the proof-of-work takes time, the amount of transactions that the bitcoin decentralized network can process is limited.

The debit of transaction is a major concern of large scale entreprise. In order to be able to use blockchain technology for different applications, such as banking, real estate, energy and others industry, it is important to be able to process multiple transactions concurrently. What has been proposed during the conference is the Fast Access Blockchain. A *public blockchain which overcomes the scalability challenge*. The challenge is stated as following: keeping the characteristics of a blockchain that is, permissionless, decentralized and open, and allowing multiple transactions to be acknowledge at the same time.

2.4 FAB Overview

The solution process by Fast Access Blockchain presented during the conference and by the developers [Pap17] is to divide the network with 3 main components: a foundation chain, additional chains, called annex chains and an open storage architecture, which should have a theoritical potential to process a million transactions, and resolve the scalability problem.

Foundation The foundation chain, is the core of the system and the functionnality is to contain a root ledger, process the transactions and take the final decision. It is the core of the decentralized architecture. A technical implementation provided by FAB within the foundation blockchain is a tool called KanBan. The main goal of KanBan is to avoid the foundation blockchain to be overloaded, and to add scalability to the processing, that is redirecting some transactions to the annex chains.

Annex chains The annex chains are multiple blockchains that process the majority of the transactions, through the KanBan. It allows for business to join the public blockchain, therefore it means that an annex chain is used for one specific business applications. It is used with the SCAR (Smart Contract Address Router) tool, that is used to execute the transactions between the annex itself and the externals, that is the KanBan component.

OSA The last component, the Open Storage, is used to, as the name states, to store all the public data of the blockchains, and allows fast access to it. The data stored are the one stored in the public blockchain and are already very thrustworthy, as it depends on the KanBan tool.

3 Conference #2: Empowering the cashless society

3.1 Introduction

During the conference of 28 March, Cedric Meuter talk about his activity in Atos Worldline company. The subject of the conference was the use of cryptographic technology within cashless banking payment. In this paper, we will propose a summary of the content of the conference. We will first describe fundamentals of security. Then we will propose a description of the important cryptographic protocol uses in the domain. Then an important part of the job in banking is to handle the keys used during the communication of the devices during a payment, we will then describe the key management as presented by Atom Worldline company.

3.2 Fundamentals of security

It is important for a company to analyse the risk of the infrastrucutre deployed. Two important questions must be asked for (1) How likely an attacker will try to compromise you and acquire your data (2) What's the impact of the compromisation of your data. A good example of of an attack that highlights the importance of such risk analysis is the Sony Playstation Network Attack [Rai12]. Sony acknowledge that personal informations were stolen, including credit cards number. There are three important fundamentals of security: (1) Confidentiality (2) Integrity (3) Authenticity. With this informations, the goal of the protocols presented in the next section is to respect those fundamentals.

3.3 Cryptographic protocols and key management

The Payment Card Industry (PCI) must rely on standard to make sure that the transactions follow the fundamentals of security. Those standards used by the PCI are defined by different institution, such as NIST, FIPS or ANSI. An important protocol used in terminal payment is the DUKPT protocol defined in the ANSI X9.24 standard [ANS14].

3.3.1 DUKPT

DUKPT or Derived Unique Key Per Transaction, is at it's named says to generate a unique key per transaction. The goal of such a protocol, if a single transaction is compromised, it will not compromised the all infrastructure. To do so, it is necessary to have a centralized server, usually a Hardware Security Module, a centralized server that stores the keys and provides cryptographic functions. The algorithms need a secret key, called a Base Derivation Key or BDK, to be able to generate a key for each terminal. Storing the BDK is very costly because if they are compromised, they can compromised a lot of devices. Therefore store new keys, developer must request to add a new key, and this will be done during a Key Ceremony, that will add the key to the HSM.

Each terminal as a secret key, based on the serial number, generate at factory by the HSM. To encrypt a transaction and process secret information, such as a PIN code, the terminal generate a number of key, and each key will only be used for only one transactions. Then the terminal send the encrypted data and it's serial number to the HSM. The HSM will be able to retrieve the key from the terminal, and regenerate the key used for the transaction, since it provides the cryptographic functions, and the key of the terminal was generated by the HSM.

3.3.2 Hardware Security Modules

The HSM is the devices (server), that allows to generate key for the terminals, and provides crypto-functions to handle the payements. The HSM stores multiple Base Deriviation Key, and with those will be able to generate per terminal keys, called Initial Key (IK). Those keys are loaded in factory in each terminal and will allow the terminals to generated the Trasanctions Keys.

References

[ANS14] Ansi x9.24 - pin security requirements. Standard, American National Standard Institute, 2014.

[Bac02] Adam Back. *Hashcash - A Denial of Service Counter-Measure*. 2002.

[Cac98] Christian Cachin. On the foundations of oblivious transfer. In *Advances in Cryptology — EUROCRYPT'98*, volume 1403, pages 361–374. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[Cre88] Claude Crepeau. Equivalence Between Two Flavours of Oblivious Transfers. In *Advances in Cryptology — CRYPTO '87*, volume 293, pages 350–354. Springer Berlin Heidelberg, Berlin, Heidelberg, 1988.

[EGL85] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. *Communications of the ACM*, 28(6):637–647, June 1985.

[IG03] I. Ioannidis and A. Grama. An efficient protocol for Yao's millionaires' problem. page 6 pp. IEEE, 2003.

[LT05] Hsiao-Ying Lin and Wen-Guey Tzeng. An Efficient Solution to the Millionaires' Problem Based on Homomorphic Encryption. In *Applied Cryptography and Network Security*, volume 3531, pages 456–466. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[Nak08] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. 2008.

[Pap17] Fast Access Blockchain White Paper. *https://medium.com/@fabcoin/fast-access-blockchain-fab-white-paper-1025016d3595*, 2017.

[Rab] Michael O. Rabin. How to Exchange Secrets with Oblivious Transfer.

[Rai12] Costin Raiu. Cyber-threat evolution: the past year. *Computer Fraud & Security*, 2012(3):5–8, March 2012.

[Yao82] Andrew C. Yao. Protocols for secure computations. pages 160–164. IEEE, November 1982.