Secure Matchmaking Protocol

Byoungcheon Lee and Kwangjo Kim

Information and Communications University,
58-4, Hwaam-dong, Yusong-gu, Taejon, 305-348, Korea
{sultan,kkj}@icu.ac.kr

Abstract. Matchmaking protocol is a procedure to find matched pairs in registered groups of participants depending on their choices, while preserving their privacy. In this study we define the concept of matchmaking and construct a simple and efficient matchmaking protocol under the simple rule that two members become a matched pair only when they have chosen each other. In matchmaking protocol, participant's privacy is of prime concern, specially losers' choices should not be opened. Our basic approach to achieve privacy is finding collisions among multiple secure commitments without decryption. For this purpose we build a protocol to find collisions in ElGamal ciphertexts without decryption using Michels and Stadler's protocol [MS97] of proving the equality or inequality of two discrete logarithms. Correctness is guaranteed because all procedures are universally verifiable.

Keywords: matchmaking, secure multiparty computation, proof of knowledge, proving the equality or inequality of two discrete logarithms, finding collisions without decryption, public commitment.

1 Introduction

Consider a set of parties who trust neither other entities nor the channels by which they communicate. The parties wish to correctly compute some common function of their local inputs, while keeping their local data as private as possible. This is the problem of secure multiparty computation [Yao82], [GMW87], [CCD88], [BGW88], [Can96], [Gol98], [Cra99], which has fundamental importance in cryptography and is relevant to many distributed cryptographic applications such as electronic cash, voting, auction, and so on.

In this study we consider a problem of finding matched pairs in registered groups of participants depending on their choices, while preserving their privacy. This is an interesting example of secure multiparty computation where the common function is finding matched pairs in the registered groups of participants and local data are participants' choices.

There is a popular TV program in Korea which performs matchmaking between two registered groups of men and women. In the program all participants commit their choices to a host secretly, but in the opening stage the host opens all the choices and decides couples. The established couple members are OK for opening their choices, but losers may wish that their choices might not be opened if possible. A loser can have another chance to participate in matchmaking and in that case his or her previous choice is important private information. So secrecy of commitment is a prime security issue in this case.

Another example can be found in setting up project teams in a class. The lecturer of the class tries to permit students to form project teams of two members if they want each other. The choices of established team members are published, but losers who could not form a team may wish that their choices might not be opened if possible.

In this study we consider a typical model of secure matchmaking protocol which is used to set up couples among m male members $M_i(i = 1, ..., m)$ and n female members $F_j(j = 1, ...n)$. The basic rule of matchmaking is that in commitment stage each participant commits a single choice to TTP secretly and then in opening stage two participants are decided as a couple only when they have chosen each other. Our basic approach to provide secrecy of commitment is finding collisions in ElGamal ciphertexts without decryption. For this purpose we use Michels and Stadler's protocol [MS97] of proving the equality or inequality of two discrete logarithms. Our design provides the secrecy of commitment and guarantees the correctness of results.

This paper is organized as follows. In section 2, we describe our model of matchmaking and its security requirements. Then in section 3, we describe some building blocks such as public commitment, proving the correctness of decryption, and finding collisions in ElGamal ciphertexts. Using these primitives we construct a simple and efficient matchmaking protocol in section 4 and provide its security analysis in section 5.

2 Model of matchmaking

2.1 Definition of terms

In this paper the following terms are used in a specific sense, so we need to define them more rigorously.

Matchmaking is a protocol to find matched pairs $< a_i, b_j >$ between two registered groups of participants $A = (a_1, ..., a_m)$ and $B = (b_1, ..., b_n)$ which satisfy $< a_i, b_j > \in R$ for a special relation R we try to find.

Established couple is a matched pair $< a_i, b_j > \in R$ which is found using the matchmaking protocol.

Loser is a participant who was not established as a couple. All participants except the established couple members are losers.

Registered group of participants is the members registered in the matchmaking system who participate in the matchmaking process and try to find a partner there.

Public commitment is a commitment scheme with which a participant commits his or her choice to the public. The correctness of result should be publicly verifiable.

Collision in ciphertexts is the case that two ciphertexts are probabilistic encryptions of the same message.
Proof of coupling is a proof for the correctness of the established couple.

2.2 Participants and tools

Our matchmaking protocol has the following participants and tools. All participants are assumed to have own public/private key pairs certified by a certificate authority(CA).

Male participants: m male members $M_i (i = 1, ..., m)$ want to find partners among the female members.
Female participants: n female members $F_j (j = 1, ..., n)$ want to find partners among the male members.
TTP(Host): A trusted third party T is a host of matchmaking and mediates the matchmaking procedure, and can be modeled as a probabilistic polynomial-time Turing machine. T receives participants' public commitments as input and outputs the results of matchmaking and proofs of them. W.l.o.g., it is assumed that T does not collude with any specific participant.
Bulletin board: It is a public communication channel which can be read by anybody. Only legitimate participant is allowed to post his or her message on the specified region of the bulletin board. All participants communicate via the bulletin board.

2.3 Rule of matchmaking

In this study we consider the following simplified model of matchmaking between two registered groups of men and women.

- Each participant in a group has a single choice among the participants of the other group.
- Two participants are decided as a couple only when they have chosen each other.

In the real world there can be a variety of situations and rules for matchmaking. Some possible examples are committing multiple choices, setting up a team of multiple partners, *etc*, and each can be a good model for the study of secure multiparty computation.

2.4 Security requirements

The main security concern of matchmaking is the secrecy of participants' choices. A participant wants to keep the secrecy of his or her choice as private as possible while trying to get any information on others' choices. Even the established couple members may want to know who else have chosen him or her except the current partner. A loser may want to know who have chosen him or her, because it can be important information for the next round of matchmaking while it is also private information for the loser. Therefore secrecy of commitment should be provided such that malicious participants cannot get any partial information on others' choices.

Another scenario is that a participant A can try to help other participant B, i.e., A helps B to setup a couple $< B, C >$. If this is possible, B gets to have two choices against the fairness of the matchmaking rule. So the authenticity of commitment is required and the correctness of result should be publicly verifiable.

Still another scenario is that a member of an established couple may want to change his or her mind later and tries to repudiate the result. It is clear that this should not be allowed.

The security requirements of secure matchmaking protocol can be listed as follows.

Secrecy(privacy) The choices of participants should not be exposed to others including TTP in the whole process of matchmaking. Only the choices of established couple members are published in the opening stage.

Fairness In the commitment stage, anyone cannot be in advantageous position than others, i.e., anyone cannot have any partial information on other's choice.

Correctness The correctness of the result of matchmaking should be publicly verifiable.

Authenticity The authenticity of commitment should be provided such that each registered participant has committed a single choice can be verified.

Non-repudiation The established couple members should not be able to repudiate their commitments later.

3 Building blocks

In this section we describe some building blocks used in the proposed matchmaking protocol.

3.1 Notation

The basic cryptographic primitives used in our protocol are ElGamal public key encryption, digital signature, and zero-knowledge proof, which are commonly based on the discrete logarithm setting. Let Z_p^* denote the multiplicative group of prime modulo p. We consider a cyclic subgroup of Z_p^* of prime order q with $q|(p-1)$. Let g be the published generator of the subgroup of order q. TTP has a certified public key $y = g^x$ and its corresponding private key x. In this paper we use the following notation.

$Sig_i(m)$ is a digital signature of a participant i for a message m where the signature scheme is secure against existential forgery.

$E_j(m)$ is a probabilistic ElGamal encryption of a message m with the participant j's public key.

$Env(i, j, m) = [c||s] = [E_j(m)||Sig_i(c)]$ is an enveloped message of a plaintext m sent by a sender i to a recipient j. The plaintext m is signed by the sender i and encrypted with the recipient j's public key.

$H()$ is a collision resistant hash function. For security proof, it is considered as a random oracle.

3.2 Public commitment scheme

Commitment scheme is a digital implementation of a sealed box which is opened later. In a two party computation, commitment scheme is a two-phase protocol of commitment and reveal stages, and should satisfy the requirement of secrecy and unambiguity. The situation in matchmaking protocol is a multiparty computation, so we define a new concept of *public commitment scheme* where a message is committed to the public, revealed by the help of TTP, and is publicly verified.

Definition 1 (Public commitment scheme).
Public commitment scheme is a 3-stage protocol between multiple participants $P_1, ..., P_n$ *and a TTP which consists of:*

1. *Commitment: Multiple participants commit their secure messages to the public in a secure way such that only TTP can open it.*
2. *Reveal: TTP opens the commitments and provides the proofs of results.*
3. *Verification: Anyone verifies the correctness of the results.*

And it satisfies the following requirements:

1. *Secrecy: At the end of stage 1, anyone except the sender cannot tell what value is being sent.*
2. *Unambiguity: Given a commitment of stage 1, there is at most one value everybody may accept as valid. It means that the commitment is bound to a value.*
3. *Non-malleability: Given a commitment of stage 1, anyone except the sender cannot generate another legal commitment which has a message related with the original message.*
4. *Non-repudiation: Once a participant committed a message to the public in stage 1, he cannot repudiate it later.*

In this study we implement the public commitment scheme using the enveloped message defined above. In commitment stage, a participant P_i commits a message m_i to the public as $Env(i, T, m_i)$ which is encrypted with TTP's public key. In reveal stage, TTP recovers the message m_i and publishes it with a proof of the correctness of decryption. In verification stage, anyone verifies the correctness of the message.

Lemma 1. *Let $P_1, ..., P_n$ be multiple participants and T be a TTP. A commitment scheme using an enveloped message $Env(i, T, m) = [c||s] = [E_T(m)||Sig_i(c)]$ is a public commitment scheme between P_i and T.*

Proof. (sketch) The probabilistic public key encryption scheme provides secrecy and unambiguity assuming that T does not collude with participants. Considering the result of [TY98] and [SJ00], the enveloped message(a combination of a public key encryption and a digital signature) can be considered as a non-malleable encryption scheme. So non-malleability is satisfied. The usage of digital signature provides non-repudiation. □

3.3 Proving the correctness of decryption

Firstly we describe the zero-knowledge proof of knowledge protocol for the equality of two discrete logarithms [CP93], [CGS97]. For universal verifiability, we consider its non-interactive version using the Fiat-Shamir heuristics [FS87] and a collision resistant hash function. The 3-move interactive protocol is an honest verifier zero-knowledge and the security of the non-interactive version is obtained under the random oracle model [CGS97].

Let α and β be two independent elements of the cyclic group Z_p^* of order q, i.e., nobody knows the relative discrete logarithm $\log_\alpha \beta$.

Protocol 1. Proving the equality of two discrete logarithms

The prover wants to prove that two elements $y = \alpha^x$ and $z = \beta^x$ have the same discrete logarithm for base α and β, respectively, without exposing x, i.e., he wants to prove that $\log_\alpha y = \log_\beta z$ holds. (α, y, β, z) are given as common input.

Prover
- Chooses $k \in_R Z_q^*$.
- Computes $r_\alpha = \alpha^k$, $r_\beta = \beta^k$.
- Computes $v = H(\alpha, y, \beta, z, r_\alpha, r_\beta)$.
- Computes $s = k - vx$.
- Publishes (r_α, r_β, s).

Verifier(anyone)
- Computes $v = H(\alpha, y, \beta, z, r_\alpha, r_\beta)$.
- Verifies $r_\alpha \stackrel{?}{=} \alpha^s y^v$, $r_\beta \stackrel{?}{=} \beta^s z^v$.

Now let the public key of a recipient R be $y = g^x$. When the recipient gets an ElGamal ciphertext $(a, b) = (g^k, y^k m)$, he can recover the message m by decrypting it with his private key x and prove its correctness by exposing x. If he wants to prove the correctness of decryption without exposing x, he can do it using the following protocol.

Protocol 2. Proving the correctness of decryption

Let the public key of a recipient R be $y = g^x$. He wants to prove that m is the plaintext of a ciphertext $(a, b) = (g^k, y^k m)$ without exposing the private key x. He can prove it by showing $\log_g y = \log_a (b/m)$ using *Protocol 1.*

Lemma 2. *Protocol 2 proves the correctness of decryption.*

Proof. (sketch) The public key $y = g^x$ of the recipient R is publicly known. So showing $\log_g y = \log_a(b/m)$ is equivalent to showing $b/m = a^x$, which proves the correctness of decryption. □

3.4 Proving the equality or inequality of two discrete logarithms

Assume that the prover knows the discrete logarithm x of $y = \alpha^x$ for the base α. He wants to allow the verifier to decide whether $\log_\alpha y = \log_\beta z$ or $\log_\alpha y \neq \log_\beta z$ for given group elements β and z, such that no partial information about the logarithm x is leaked.

Protocol 1 provides only the proof of equality of two discrete logarithms. If this proof fails, all the arguments of prover are invalid. [MS97] provides the proof of equality or inequality at the same time. Zero-knowledge interactive proof system convinces only the verifier and requires interaction, but in our situation of matchmaking the prover(host of matchmaking) wants to convince all the participants, so we use the non-interactive version of their scheme which provides universal verifiability.

Protocol 3. Proving the equality or inequality of two discrete logarithms

The prover wants to convince the public whether the two elements y and z have the same discrete logarithm or not for the base α and β, respectively, without exposing the discrete logarithm, i.e., he wants to prove that either $\log_\alpha y = \log_\beta z$ or $\log_\alpha y \neq \log_\beta z$ holds. (α, β, y, z) are given as common input.

Prover(TTP)
- Chooses $k, k' \in_R Z_q^*$.
- Computes $r_\alpha = \alpha^k$, $r_\beta = \beta^k$, $r'_\alpha = \alpha^{k'}$ and $r'_\beta = \beta^{k'}$.
- Computes $v = H(\alpha, y, \beta, z, r_\alpha, r_\beta, r'_\alpha, r'_\beta)$.
- Computes $s = k - vx$ and $s' = k' - vk$.
- Publishes $(r_\alpha, r_\beta, r'_\alpha, r'_\beta, s, s')$.

Verifier(anyone)
- Computes $v = H(\alpha, y, \beta, z, r_\alpha, r_\beta, r'_\alpha, r'_\beta)$.
- Verifies $r_\alpha \stackrel{?}{=} \alpha^s y^v$, $r'_\alpha \stackrel{?}{=} \alpha^{s'} r_\alpha^v$, $r'_\beta \stackrel{?}{=} \beta^{s'} r_\beta^v$. If the verification fails, stop the verification process.
- If $r_\beta = \beta^s z^v$, then $\log_\alpha y = \log_\beta z$. Else if $r_\beta \neq \beta^s z^v$, then $\log_\alpha y \neq \log_\beta z$.

3.5 Finding collisions in ElGamal ciphertexts

When TTP has a public key $y = g^x$ and the corresponding private key x, the ElGamal encryption of a message m with TTP's public key y is given by $(a, b) = (g^k, y^k m)$ where $k \in_R Z_q^*$. Now assume that TTP has received two ciphertexts $(a_1, b_1) = (g^{k_1}, y^{k_1} m_1)$ and $(a_2, b_2) = (g^{k_2}, y^{k_2} m_2)$. Of course TTP can recover two messages m_1 and m_2 using his private key x and publish them, but TTP wants to convince the public whether the two ciphertexts have the same message or not, without exposing any partial information on the messages or his private key x. This task can be obtained as follows using *Protocol 3*.

Protocol 4. Finding collisions in ElGamal ciphertexts

Let $y = g^x$ be TTP's public key and x be his private key. Given two ElGamal ciphertexts $(a_1, b_1) = (g^{k_1}, y^{k_1}m_1)$ and $(a_2, b_2) = (g^{k_2}, y^{k_2}m_2)$, TTP wants to prove whether the two ciphertexts have the same message or not, without exposing any partial information on the messages or his private key x.

Prover(TTP)
- Computes $(a_3, b_3) \equiv (a_1/a_2, b_1/b_2) = (g^{k_1-k_2}, y^{k_1-k_2}m_1/m_2)$.
- Proves that $\log_g y = \log_{a_3} b_3$ or $\log_g y \neq \log_{a_3} b_3$ using *Protocol 3*.

Verifier(anyone)
- If $\log_g y = \log_{a_3} b_3$, then $m_1 = m_2$. Else if $\log_g y \neq \log_{a_3} b_3$, then $m_1 \neq m_2$.

Lemma 3. *Protocol 4 proves whether two ciphertexts (a_1, b_1) and (a_2, b_2) have the same plaintext or not.*

Proof. (sketch) TTP's public key $y = g^x$ is publicly known. So proving $\log_g y \stackrel{?}{=} \log_{a_3} b_3$ is equivalent to proving $a_3^x \stackrel{?}{=} b_3$, which proves $m_1 \stackrel{?}{=} m_2$. □

Protocol 4 can be used in wide range of applications where just the proof of equality or inequality of plaintext is required while the plaintext should not be recovered.

4 Proposed matchmaking protocol

In this section we describe our matchmaking protocol which is constructed using the primitives described above. The participants are m male members $M_i (i = 1, ..., m)$, n female members $F_j (j = 1, ..., n)$ and a host T. They use the bulletin board as a public communication channel.

The proposed matchmaking protocol consists of 5 stages: registration, commitment, opening, proof of coupling, and verification. In the sequel we will describe the case that M_i and F_j have chosen each other for the ease of description.

Stage 1. Registration

Any applicant who wants to participate in matchmaking should register to T. Each participant does the following steps:

- M_i chooses a random number $a_i \in_R Z_q^*$ and computes his temporal ID $m_i = g^{a_i}$. Likewise F_j chooses a random number $b_j \in_R Z_q^*$ and computes her temporal ID $f_j = g^{b_j}$.
- M_i presents to T his name, his public key with certificate, and his temporal ID. Likewise F_j presents to T her name, her public key with certificate, and her temporal ID.

– T publishes these information on the bulletin board.

Stage 2. Commitment (by participants)

In commitment stage each participant chooses a member from the other group as a possible partner, generates a couple ID for the choice, generates a public commitment by encrypting the couple ID with TTP's public key, and posts it on the bulletin board. Note that each participant is allowed to commit a single choice.

To generate the couple ID, we use the Diffie-Hellman key agreement technique. A couple ID between M_i and F_j is given by $CID(M_i, F_j) = H(m_i^{b_j}) = H(f_j^{a_i}) = H(g^{a_i b_i})$, and it can be computed only by M_i and F_j.

Each participant does the following steps:

– Chooses a possible partner from the other group.
– Computes a couple ID for the choice. If M_i has chosen F_j, he generates his couple ID with his random number a_i and the partner's published temporal ID f_j such that $CID(M_i, F_j) = H(f_j^{a_i})$.
– Generates a public commitment for the choice and posts it on the bulletin board. M_i's public commitment is given by $Env(i, T, CID(M_i, F_j))$.

Stage 3. Opening (by TTP)

When the deadline of commitment has passed, TTP tries to find couples using *Protocol 4* for every possible pairs of participants (M_i, F_j) where $i = 1, ..., m$ and $j = 1, ..., n$. TTP posts the results and their proofs on the bulletin board.

For the established couple members, TTP additionally decrypts their commitments and publishes the couple IDs, which demonstrate that the two commitments are equal. TTP proves the correctness of his decryption using *Protocol 2*.

TTP does the following steps:

– TTP tries to find couples for every possible pairs (M_i, F_j) using *Protocol 4* and posts all the results and proofs on the bulletin board.
– For the established couple members, TTP decrypts their public commitments to get the colliding couple ID and publishes them on the bulletin board. He provides proofs of correctness of the decryptions using *Protocol 2*.

Stage 4. Proof of coupling (by established couple members)

The participants who were published as a couple in the opening stage have to show that the colliding couple ID is authentic. A simple way to prove the authenticity of couple ID is showing the random number corresponding to participant's temporal ID. $CID(M_i, F_j)$ can be proven authentic by verifying $CID(M_i, F_j) = H(f_j^{a_i})$ if M_i opens a_i or by verifying $CID(M_i, F_j) = H(m_i^{b_j})$ if F_j opens b_j.

If the established couple members do not want to open their secret random numbers, the authenticity of $CID(M_i, F_j)$ can be proven using *Protocol 2*. M_i can prove $\log_g m_i = \log_{f_j}(CID(M_i, F_j))$ without exposing a_i.

One of the couple members cannot repudiate his or her commitment while the other member does not want to repudiate, because both of them can prove the authenticity of $CID(M_i, F_j)$. If both of them refuse to prove it(changed their mind together), they will not be decided as a couple. Based on their proofs of coupling, everyone can verify the correctness of coupling.

Stage 5. Verification (by anyone)
Anyone can verify the correctness of results by:

- Verify the correctness of TTP's opening in stage 3.
- Verify the authenticity of the colliding couple ID published in stage 4.
- Verify the couple member's signature for the public commitments posted in stage 2.

5 Security analysis

Our proposed matchmaking protocol satisfies all the listed security requirements.

Theorem 1. *The proposed matchmaking protocol is secure in the sense that it satisfies secrecy, fairness, correctness, authenticity, and non-repudiation.*

Proof. (sketch)

Secrecy(privacy) All the commitments are encrypted with TTP's public key. So if TTP does not help, any participant cannot get any partial information on the choices of others. Although TTP can decrypt a commitment, he cannot get any information on the choice without help of the specific couple members. Under the computational Diffie-Hellman assumption, identifying the couple (M_i, F_j) from $H(g^{a_i b_j})$ is computationally infeasible. Only the choices of established couple members are published by TTP in the opening stage.

Fairness If TTP does not help, anyone cannot be in advantageous position than others in the commitment stage. If TTP helps a participant by secretly opening other group member's commitment, he can check whether there is any choice for him by trying all the possible pairs.

Correctness Correctness of result is guaranteed because all the procedures are publicly verifiable. In opening stage, TTP tries to find couples for every possible pairs of participants and provides the proofs of result using *Protocol 4* whose correctness is given by Lemma 3. For the established couple members, TTP additionally decrypts their commitments and publishes the colliding couple ID. TTP proves the correctness of his decryption using *Protocol 2* whose correctness is given by Lemma 2. In proof of coupling stage, the established couple members prove the authenticity of the colliding couple ID by showing their random number.

Authenticity In commitment stage, public commitment is signed by the sender. It guarantees the authenticity of commitment and the fact that each registered participant has committed a single choice is verified.

Non-repudiation In commitment stage, the committed message is signed by the sender. So the established couple members cannot repudiate their commitments later. In proof of coupling stage, one of the couple member cannot repudiate the commitment while the other member does not want to repudiate because both of them can prove the authenticity of the colliding couple ID.

□

We can consider a situation when TTP is not honest. If TTP decrypts the public commitments and exposes them intentionally, each pair of participants can check whether the other participant has chosen him or her because both of them can generate the corresponding couple ID. Therefore TTP should not decrypt the commitments and expose them.

Our proposed matchmaking scheme requires $O(mn)$ computation in opening stage because TTP has to try to find couples for every possible pairs of participants. Enhancing the efficiency of the protocol is left as further study.

6 Conclusion

We introduce a secure matchmaking protocol as a new application of secure multiparty computation and construct a simple and efficient protocol under the simple rule that two participants become a couple only when they have chosen each other. To implement secure matchmaking, we use various primitives of zero-knowledge proofs: proving the correctness of decryption, proving the equality or inequality of two discrete logarithms, and finding collisions in ElGamal ciphertexts. We also define the concept of public commitment scheme and show that enveloped message(a combination of public key encryption and digital signature) can be used for public commitment. Our basic approach is that participants commit their choices secretly by encrypting them with TTP's public key and TTP tries to find collisions in ElGamal ciphertexts without decryption.

The main security issue is the honesty of TTP which was assumed in this study. If TTP colludes with a participant, he can get partial information on other participants' choices. The computational load of TTP in opening stage is $O(mn)$ because he has to try to find couples for every possible pairs of participants. In this sense more intensive study to enhance the efficiency of the protocol will be challenging very much.

To the best of our knowledge, this is the first trial which applies cryptographic primitives to the problem of matchmaking. We believe that our result can play a significant role for narrowing the gap between cryptographic theory and real multiparty applications.

Acknowledgements

We are grateful to Moti Yung for valuable comments on commitment scheme and to the anonymous referees for their helpful comments.

References

[BGW88] M. Ben-Or, S. Goldwasser and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerent distributed computation", 20th STOC, pages 1–10, 1988.

[Can96] R. Canetti, "Studies in Secure Multiparty Computation and Applications", PhD Thesis, The Weizmann Institute of Science, 1996.

[CCD88] D. Chaum, C. Crepeau and I. Damgard, "Multiparty unconditionally secure protocols", 20th STOC, pages 11–19, 1988.

[CGS97] R. Cramer, R. Gennaro, and B. Schoenmakers, "A secure an optimally efficient multi-authority election schemes", In *Advances in Cryptology – Eurocrypt '97, LNCS Vol. 1233*, pages 103–118, Springer-Verlag, 1997.

[CP93] D. Chaum and T. Pedersen, "Wallet databases with observers ", In *Advances in Cryptology – Crypto'92, LNCS Vol. 740*, pages 89–105, Springer-Verlag, 1993.

[Cra99] R. Cramer, "Introduction to Secure Computation", In Lectures on Data Security - Modern Cryptology in Theory and Practice, Ivan Damgaard (Ed.), Springer LNCS Tutorial, vol.1561, pages 16–62, 1999.

[FS87] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems", In *Advances in Cryptology - Crypto'86*, pages 186–194, Springer-Verlag, 1987.

[Gol98] O. Goldreich, "Secure Multi-Party Computation", Manuscript version 1.1, 1998.

[GMW87] O. Goldreich, S. Micali and A. Wigderson, "How to play any mental game", 19th STOC, pages 218–229, 1987.

[MS97] M. Michels and M. Stadler, "Efficient convertible undeniable signature", Proc. of 4th annual workshop on selected areas in cryptography, 1997.

[SJ00] C.P. Schnorr and M. Jakobsson, "Security of signed ElGamal encryption", In *Advances in Cryptology - Asiacrypt'2000, LNCS V. 1976*, pages 73–89, Springer-Verlag, 2000.

[TY98] Y. Tsiounis and M. Yung, "On the Security of ElGamal Based Encryption", *PKC'98, LNCS 1431*, pages 117-134, Springer-Verlag, 1998.

[Yao82] A. Yao, "Protocols for Secure Computation", 23th FOCS, pages 160–164, 1982.