SHADE: Secure HAmming DistancE Computation from Oblivious Transfer*

Julien Bringer[1], Hervé Chabanne[1,2], and Alain Patey[1,2]

[1] Morpho
[2] Télécom ParisTech
Identity and Security Alliance (The Morpho and Télécom ParisTech Research Center)

Abstract. We introduce two new schemes for securely computing Hamming distance in the two-party setting. Our first scheme is a very efficient protocol, based solely on 1-out-of-2 Oblivious Transfer, that achieves full security in the semi-honest setting and one-sided security in the malicious setting. Moreover we show that this protocol is significantly more efficient than the previous proposals, that are either based on garbled circuits or on homomorphic encryption. Our second scheme achieves full security against malicious adversaries and is based on Committed Oblivious Transfer. These protocols have direct applications to secure biometric identification.

Keywords: Secure Multi-Party Computation, Hamming Distance, Oblivious Transfer, Biometric Identification.

1 Introduction

Secure Multiparty Computation (SMC) [35,13] enables a set of parties to jointly compute a function of their inputs while keeping the inputs private. We here focus on the 2-party case [14], also known as Secure Function Evaluation. Several generic constructions exist in this setting, which apply SMC to any function computed by two parties. In the semi-honest setting, where security is ensured against adversaries following the protocol but trying to gain more information than they should, the Yao's protocol [35,24] can be used to achieve this purpose using Oblivious Transfers and Garbled Circuits. In the malicious model, where adversaries can follow any strategy, many generic constructions have been proposed [19,23,17,28,18,25]. The problem of generic constructions is that they are often far from being optimal when one wants to securely compute specific functions of interest. However, it may happen that generic constructions can be more efficient than specific ones [15].

We here consider the secure computation of the Hamming distance. Concretely, two parties P_1 and P_2 hold bit strings of the same length n, resp. $X = (x_1, \ldots, x_n)$ and $Y = (y_1, \ldots, y_n)$ and want to jointly compute $d_H(X, Y) =$

* This work has been partially funded by the ANR SecuLar project and the European FP7 FIDELITY project (SEC-2011-284862).

A.A. Adam, M. Brenner, and M. Smith (Eds.): FC 2013, LNCS 7862, pp. 164–176, 2013.

$\Sigma_{i=1}^{n}(x_i \oplus y_i)$, without P_1 (resp. P_2) revealing X (resp. Y) to P_2 (resp. P_1). For now, let us consider this problem in the semi-honest setting. It has first been solved using additive homomorphic encryption [20,29]. Using this technique, each bit of P_1's input has to be encrypted in one Paillier ciphertext [30] and sent to the other part who can then compute a ciphertext corresponding to the Hamming distance, using homomorphic encryptions. Since Paillier ciphertexts must be at least 2048 bit-long and homomorphic encryptions are multiplications and exponentiations in large groups, this technique is inefficient. However, they also propose in [20] an adaptation of their protocol to the malicious setting. Recently, Huang *et al.* [15] showed that the generic Yao algorithm applied to Hamming distance was more efficient in terms of computation time and bandwidth consumption. Using the Yao algorithm, one needs to describe the function as a binary circuit and then "garble" every gate of this circuit to a table of 4 symmetric ciphertexts. However, using the techniques of [22] and [32], XOR gates do not need to be garbled and garbled gates can be reduced to 3 items. The circuit used in [15] is the succession of n bit-wise (free) XOR's and a *Counter* circuit that adds the results of these XOR's. This Counter circuit is the bottleneck of their protocol.

The first proposal of our paper achieves full security in the semi-honest model. We almost only rely on 1-out-of-2 oblivious transfer (OT_1^2). This primitive enables a receiver to obtain 1 out of 2 elements held by a sender without the sender learning the choice of the receiver and without the receiver learning information on the other element held by the sender. In the Yao algorithm, using OT_1^2's, party P_2 gets his input keys for a garbled circuit of the function to compute. However, the keys sent by P_1 are independent of P_1's inputs. Here we design our scheme such that, in our OT's, the elements sent by P_1 also depend on the input bits of P_1 in such a way that the element obtained by P_2 during the i^{th} OT_1^2 depends on $x_i \oplus y_i$. Moreover using the technique of [26, Third Variant], we avoid the use of a costly Counter circuit. We prove, using the OT-hybrid model [6,23,14], that our protocol is fully secure in the semi-honest setting or one-sided secure in the malicious setting, depending on the security level of the underlying OT_1^2. This protocol is significantly more efficient than the previous proposals for secure Hamming distance in the semi-honest model [20,29,15,2].

We next extend our first proposal to a second protocol that is fully secure in the malicious setting. Therefore, we use Committed Oblivious Transfer (COT) [8] instead of basic OT_1^2. In particular, we use a COT on bit strings with homomorphic commitments, as in [21]. COT enforces that the parties are committed to their inputs to the oblivious transfers and moreover that the receiver is committed to his output. The homomorphic commitment scheme enables us to guarantee that the inputs of the sender are consistent and that the computation run by the receiver on these inputs after the OT's follows the protocol.

The proofs of security of our protocol secure in the malicious setting and extensions to secure computation of weighted Hamming distance, of biometric identification and of any linear combination of bit-wise independent functions, appear in the extended version of this paper [3].

2 SMC and Oblivious Transfer

2.1 Oblivious Transfer

Oblivious Transfer was first introduced by Rabin [33] as a two-party protocol where a sender has a secret message that he sends to a receiver, which receives it with probability 1/2, without the sender knowing if the message has been received or not. This is however not the version that is now used in secure protocols, but a slightly different primitive called 1-out-of-2 Oblivious Transfer (OT_1^2). We here describe this primitive, some extensions to improve its use and a derived version called Committed Oblivious Transfer (COT) [21], used in our second proposal.

1-out-of-2 Oblivious Transfer. A 1-out-of-2 Oblivious Transfer is a cryptographic primitive that enables a receiver R to obtain 1 out of 2 elements held by a sender, without learning information on the other element and without the sender knowing which element has been chosen. This kind of protocol is stronger than a Private Information Retrieval (PIR) protocol [7] where only the choice of the receiver remains hidden from the sender. The functionality enabled by a OT_1^2 is described in Figure 1. For more details on implementations, see for instance [14, Chapter 7]. For instance, the oblivious transfers of [27] and of [31] can be used, respectively, in the semi-honest and in the malicious setting (see Section 2.2 for the security definitions).

- **Inputs:** • Sender S inputs two n-bit strings X_0 and X_1
  • Receiver R inputs a choice bit b
- **Output:** • S learns nothing on b
  • R obtains X_b but learns nothing on X_{1-b}

Fig. 1. The OT_1^2 functionality

Extensions. Several kinds of optimizations can be applied to Oblivious Transfers, independently of the implementation. Two optimizations introduced in [16] are of interest for our proposals. The first one [16, Section 3] enables, in the random oracle model, to compute many OT's with a small elementary cost from k OT's at a normal cost, where k is a security parameter. The second one [16, Appendix B] enables to reduce oblivious transfers of long strings to oblivious transfers of short strings using a pseudo-random generator.

Committed Oblivious Transfer. Committed Oblivious Transfer (COT) is a combination of OT_1^2 and bit commitment, first introduced by Crépeau [8] under the name Verifiable Oblivious Transfer. In this variant, both sender and receiver are committed to their inputs before the oblivious transfer. Moreover, the sender receives a commitment to the receiver's output, and the receiver obtains the randomness for this commitment. To our knowledge, the only scheme

- **Inputs:** • S inputs two n-bit strings X_0 and X_1 and two random values r_0 and r_1 used for commitment.
  • R inputs a choice bit b and a random r used for commitment
  • The common inputs are $Com(b, r)$, $Com(X_0, r_0)$ and $Com(X_1, r_1)$
- **Output:** • S learns nothing on b and r
  • R obtains X_b and a random u but learns nothing on X_{1-b}, r_0 and r_1.
  • Both parties obtain $Com(X_b, u)$.

Fig. 2. The *COT* functionality

that considers COT of bit strings is the one of Kiraz *et al.* [21], which uses an homomorphic cryptosystem as commitment scheme. COT is described in Figure 2, where *Com* denotes a commitment scheme.

2.2 Secure Two-Party Computation

Overview. Secure Multi-Party Computation [35] enables a set of parties to jointly compute a function of their inputs while keeping their inputs private. Different kinds of adversaries are considered:
• *semi-honest* adversaries who follow the protocols and try to gain more information than they should on the other parties' inputs,
• *malicious* adversaries who use any kind of strategy to learn information.

There also exists a notion of *covert* adversaries [1] who are malicious but averse to being caught. Notice that we only consider static adversaries.

Security Definitions. Informally, security in SMC is ensured by simulating the secure protocol in an ideal model where the inputs of both parties are sent to a trusted party who takes care of the computation and sends the outputs back to the respective parties and showing that all adversarial behaviours in a real execution are simulatable in this ideal model. Full definitions and explanations can be found in [13,14].

We quickly recall how full security is proven in the malicious setting. Let π be a protocol for computing $f(x, y) = (f_1(x, y), f_2(x, y))$. In the real world, a probabilistic polynomial-time (PPT) adversary A sends messages on behalf of the corrupted party and follows an arbitrary strategy while the honest party follows the instructions of π. In the ideal world, the honest party sends his genuine input x to a trusted party. The adversary sends any input y', of the appropriate size to the trusted party. The trusted party first sends his output $f_1(x, y')$ to the adversary and, if the adversary does not abort, also sends his output $f_2(x, y')$ to the honest party. The adversary is also allowed to abort the protocol at any time. Full Security against a malicious party P_i is ensured if, for any PPT adversary in the real world, there is a PPT adversary in the ideal world such that the distribution of the outputs in the real world is indistinguishable from the distribution of the outputs in the ideal world.

A weaker notion is *Privacy* against a malicious party P_i, for $i = 1, 2$, that guarantees that P_i cannot learn any information on the other party's input.

However, the execution in the real model might not be simulatable in the ideal model. We say that a protocol achieves *One-Sided Security* in the malicious model if it is fully-secure against a malicious P_i and private against a malicious P_{3-i}. See [14, Section 2.6] for further details.

In this paper, we prove security of our schemes in the *OT-hybrid setting* [6,23,14]. In this setting, the execution in the real model is slightly modified. The parties have access to a trusted party that computes oblivious transfers for them. We only need to prove indistinguishability between executions in this hybrid model and the ideal model to ensure security.

3 Secure Hamming Distance Computation

In the following, the $+$ and $-$ operators respectively denote modular additions and subtractions, we assume that the context is explicit enough and do not recall the moduli in the description of the algorithms. $\bar{x}$, where x is a bit value, denotes $1 - x$. The Hamming distance is denoted by d_H.

3.1 The Basic Scheme

We here introduce our new scheme based on oblivious transfers. The Yao algorithm [35] also uses oblivious transfers but the inputs of the sender are random keys that are independent of the actual inputs of the sender for the secure computation. In the protocol we propose, the inputs of the sender P_1 to the OT's depend on P_1's input bits. Consequently, the output of each oblivious transfer depends on the input bits x_i of P_1 and y_i of P_2. We adjust our scheme so that this output depends on $x_i \oplus y_i$. Then, we use a technique inspired by [26, Third

- **Inputs:**
  - P_1 inputs a n-bit string $X = (x_1, \ldots, x_n)$
  - P_2 inputs a n-bit string $Y = (y_1, \ldots, y_n)$
- **Output:**
  - 1^{st} Option: P_1 obtains $d_H(X, Y)$ and P_2 obtains nothing
  - 2^{nd} Option: P_2 obtains $d_H(X, Y)$ and P_1 obtains nothing
- **Protocol:**
  1. P_1 generates n random values $r_1, \ldots, r_n \in_R \mathbb{Z}_{n+1}$ and computes $R = \Sigma_{i=1}^{n} r_i$
  2. For each $i = 1, \ldots, n$, P_1 and P_2 engage in a OT_1^2 where
     - P_1 acts as the sender and P_2 as the receiver.
     - P_2's selection bit is y_i.
     - P_1's input is $(r_i + x_i, r_i + \bar{x}_i)$.
     - The output obtained by P_2 is consequently $t_i = r_i + (x_i \oplus y_i)$.
  3. P_2 computes $T = \Sigma_{i=1}^{n} t_i$
  4. 1^{st} Option: (1) P_2 sends T to P_1 (2)P_1 computes and outputs $T - R$
  2^{nd} Option: (1) P_1 sends R to P_2 (2) P_2 computes and outputs $T - R$

Fig. 3. The Basic Scheme

Variant] to count the number of bits such that $x_i \oplus y_i = 1$, *i.e.* to compute the Hamming distance.

We assume that parties P_1 and P_2 respectively hold inputs $X = (x_1, \ldots, x_n)$ and $Y = (y_1, \ldots, y_n)$. Party P_1 prepares n random values $r_1, \ldots, r_n \in_R \mathbb{Z}_{n+1}$ and prepares n oblivious transfers, as a sender. The inputs of the i^{th} transfer are arranged in such a way that a receiver with bit input y gets $r_i + (y \oplus x_i) \bmod n+1$. To do so, input 0 of P_1 is set to $r_i + x_i$ and input 1 to $r_i + \bar{x_i}$. Indeed, if $y = 0$, $x_i \oplus y = x_i$ and if $y = 1$, $x_i \oplus y = \bar{x_i}$. P_2 acts as a receiver for all these n OT's, with bit inputs $y_1, \ldots, y_n$ and gets $(r_i + (x_i \oplus y_i))_{i=1,\ldots,n}$. Then, P_2 adds all these values and gets $T = \Sigma_{i=1}^{n} r_i + \Sigma_{i=1}^{n}(x_i \oplus y_i) = R + d_H(X, Y)$, where $R = \Sigma_{i=1}^{n} r_i$. Finally, depending on the party that is supposed to know the output, either P_1 sends R to P_2 or P_2 sends T to P_1, the final output being $D = T - R = d_H(X, Y)$. The protocol is described in Figure 3.

Theorem 1 (Security of the Basic Scheme)
Assuming that the underlying OT_1^2 *is secure in the semi-honest setting, the Basic Scheme achieves full security in the semi-honest setting.*

Assuming that the underlying OT_1^2 *is secure in the malicious setting, the Basic Scheme achieves, in the malicious setting:*

- *one-sided security, for the* 2^{nd} *option: privacy against a malicious* P_1 *and full security against a malicious* P_2,
- *privacy against a malicious* P_2, *for the* 1^{st} *option.*

The proofs are detailed in Section 4.1.

3.2 The Fully Secure Scheme

Requirements on the Commitment Scheme. We assume that the commitment scheme used in the Committed Oblivious Transfer we use in our scheme fulfills the following requirements.

First, it must be additively homomorphic, *i.e.* there exist efficient operations $\boxplus$ and $\odot$, such that $Com(x_1, r_1) \odot Com(x_2, r_2) = Com(x_1 + x_2, r_1 \boxplus r_2)$, for any x_1, x_2, r_1, r_2.

Second, there must exist a zero-knowledge proof of knowledge π_1^2, where both parties know a commitment $C = Com(x, r)$ and two values x_1 and x_2. In this proof, the prover knows x, r and proves that x is either x_1 or x_2. Using the notations of Camenisch and Stadler [4], $\pi_1^2 = PK\{(\alpha, \beta) : C = Com(\alpha, \beta) \wedge (\alpha = x_1 \vee \alpha = x_2)\}$.

Let us consider the commitment scheme used in [21]. This commitment consists of using a (2,2)-threshold homomorphic cryptosystem, *i.e.* $Com(x, r) = Enc(x, r)$ for a homomorphic cryptosystem where the public key is known by both parties and the secret key is shared between the parties. By definition, the first condition is fulfilled (usually $\odot$ is a product and $\boxplus$ an addition). The used cryptosystem can be an additive ElGamal [11] or a Paillier [30] encryption. In both cases, the second condition can be fulfilled (see resp. [5] and [10]). This confirms that our requirements are reasonable.

More details on the COT scheme of [21] and details on the π_1^2 proofs can be found in the extended version of this paper[3].

Our Proposal. Our second scheme adapts the Basic Scheme to the malicious setting. We use a COT with a commitment scheme fulfilling the requirements previously introduced. The commitment, together with the proofs of knowledge of the inputs helps to ensure that the inputs are consistent and that the same values are used along the protocol.

First, P_1 and P_2 commit to the oblivious transfer inputs and prove that these inputs are well-formed. P_2 proves that his inputs are bits and P_1 proves that his inputs differ by 1, *i.e.* for each input pair (a_i, b_i), there exists r_i such that $(a_i, b_i) = (r_i, r_i + 1)$ or $(a_i, b_i) = (r_i + 1, r_i)$. COT's are then run with the same inputs as in the basic scheme. Party P_2 receives committed outputs, performs the addition of these outputs and a commitment to this addition, thanks to the homomorphic properties of the commitment scheme. P_2 can prove, using the commitments, that the value T obtained by adding the results of the COT's is consistent. In the same way, party P_1 can prove that the value R is consistent with his inputs to the COT's. Indeed, $\Sigma_{i=1}^n a_i + b_i = \Sigma_{i=1}^n (r_i + r_i + 1) = 2\Sigma_{i=1}^n r_i + \Sigma_{i=1}^n 1 = 2R + n$. Using the commitments to the a_i's and to the b_i's, P_2 is then able

- **Inputs:** P_1 inputs $X = (x_1, \ldots, x_n)$; P_2 inputs $Y = (y_1, \ldots, y_n)$.
- **Output:** 1^{st} (resp. 2^{nd}) Option: P_1 (resp. P_2) obtains $d_H(X, Y)$ and P_2 (resp. P_1) obtains nothing
- **Protocol:** 1. P_2 commits to all his bits y_i: he computes and publishes $Com(y_i, \chi_i)$ for each $i = 1 \ldots n$. He also proves, using π_1^2 proofs on the commitments, that $y_i = 0$ or $y_i = 1$.
  2. P_1 generates n random values $r_1, \ldots, r_n$, uniformly from the plaintext space of Com, and computes $R = \Sigma_{i=1}^n r_i$
  3. For each $i = 1, \ldots, n$, P_1 computes $(a_i, b_i) = (r_i + x_i, r_i + \bar{x}_i)$ and commits to a_i and b_i. He computes and publishes $(A_i = Com(a_i, \alpha_i))_{i=1,\ldots,n}$ and $(B_i = Com(b_i, \beta_i))_{i=1,\ldots,n}$
  4. P_1 proves to P_2, using π_1^2 proofs on the commitments, that $|b_i - a_i| = 1$, for each $i = 1, \ldots, n$.
  5. For each $i = 1, \ldots, n$, P_1 and P_2 engage in a COT where P_1 acts as the sender and P_2 as the receiver, P_2's selection bit is y_i, P_1's input is (a_i, b_i). The output obtained by P_2 is $t_i = r_i + (x_i \oplus y_i)$ and τ_i. Both parties obtain $C_i = Com(t_i, \tau_i)$
  6. P_2 computes $T = \Sigma_{i=1}^n t_i$,
  7. 1^{st} Option: (1) P_2 computes $C = Com(T, \tau) = C_1 \odot \ldots \odot C_n$ (2) P_2 sends T and a zero-knowledge proof that C commits to T to P_1 (3) P_1 computes $C = C_1 \odot \ldots \odot C_n$ and checks the proof. (4) P_1 computes and outputs $T - R$
  2^{nd} Option: (1) P_1 computes $K = Com(2R+n, \rho) = A_1 \odot \ldots \odot A_n \odot B_1 \odot \ldots \odot B_n$ (2) P_1 sends R and a zero-knowledge proof that K commits to $2R + n$ to P_2 (3) P_2 computes $K = A_1 \odot \ldots \odot A_n \odot B_1 \odot \ldots \odot B_n$ and checks that $K = Com(2R + n, \rho)$. (4) P_2 computes and outputs $T - R$.

Fig. 4. The Fully Secure Scheme

to check if the value R is consistent with the inputs of the COT's. The protocol is described in Figure 4. At any step, if a check fails, the party computing the check should halt the protocol and output $\perp$.

Theorem 2 (Security of the Fully Secure Scheme). *Assuming that the underlying COT is secure in the malicious setting, the Fully Secure Scheme achieves full security in the malicious setting.*

4 Security Proofs

4.1 The Basic Scheme

We here give the proof of security against a malicious P_2 in the case of the 2^{nd} option. The guarantees of privacy against a malicious P_2 for the 1^{st} option, or against a malicious P_1 for the 2^{nd} option are easily deduced from the privacy of the OT's, since no other messages are sent to these parties during the protocol.

Theorem 3 (Full Security against a Malicious P_2-2^{nd} option). *Assuming that the underlying OT_1^2 is secure in the malicious setting, the Basic Scheme, following the 2^{nd} option, is fully-secure against a malicious P_2 in the OT-hybrid setting.*

The following proof is partially inspired from the proofs of [26]. Indeed, our scheme can be viewed as a reduction of the third variant of their Oblivious Automata Evaluation, with only one state per line of the matrix, but where the lines of the matrix are not identical.

Proof. Let B be a PPT adversary controlling P_2 in the real world, we describe a simulator S_B who simulates the view of B in the ideal world.

S_B runs B on input Y. Since we operate in the OT-hybrid model, B sends $Y' = (y'_1, \ldots, y'_n)$ to the OT oracle. S_B sends Y' to the trusted party and obtains $D = d_H(X, Y')$. S_B picks n random values $t_1, \ldots, t_{n-1}, T \in_R \mathbb{Z}_{n+1}$ and computes $t_n = T + D - \Sigma_{i=1}^{n-1} t_i$. S_B sends the t_i's to B as results of the oblivious transfer. He then sends T. S_B then outputs whatever B outputs.

Let us now prove the indistinguishability between the real and the simulated views. Let V be a random subset of size t of $\{1, \ldots, n\}$. (V represents the bit positions where $x_i \oplus y_i = 1$.) Consider the distributions:

• (D_V): Choose n uniformly random values $\{r_1, \ldots, r_n\} \in \mathbb{Z}_{n+1}$. For every $i \in \{1, \ldots, n\}$, let $r'_i = r_i + 1$ if $i \in V$ and $r'_i = r_i$ otherwise. Output $(r'_1, \ldots, r'_n)$.

• (D'_V): Choose n uniformly random values $R, r'_1, \ldots, r'_{n-1} \in \mathbb{Z}_{n+1}$. Let $R' = R + t$ and $r'_n = R' - \Sigma_{i=1}^{n} r'_i$. Output $(r'_1, \ldots, r'_n)$.

It is easy to show that D_V and D'_V are identically distributed and that sampling from D'_V only requires the knowledge of t. The distribution D_V represents the view of B in a real execution of the protocol while our simulator S_B samples from D'_V, with the only knowledge of the final output. Thus, the view of P_2 in the real world and the simulated view of P_2 in the ideal world are indistinguishable, which ensures full security against a malicious P_2. □

Remark 1. The proofs of security in the semi-honest setting are straightforward, given the security guarantees of the Oblivious Transfer and the arguments explained in the previous proof proving that the outputs of the OT's give no information on the inputs of P_2.

4.2 The Fully Secure Scheme

We use an adaptation of the OT-hybrid model to Committed Oblivious Transfer. When the parties engage a COT in the COT-hybrid model, parties interact with each other and have access to a trusted party that computes the COT for them. Concretely, the receiver sends $b, Com(b, r)$ to the trusted party, the sender sends $x_0, Com(x_0, r_0)$ and $x_1, Com(x_1, r_1)$ to the trusted party. The trusted party sends x_b and r' back to the receiver and $Com(x_b, r')$ to both parties. This model, for a slightly different COT, has already been used in the proof of security of the binHDOT protocol [20] for malicious adversaries.

Since we use zero-knowledge proofs of knowledge, our protocol cannot be proved secure in the UC model [6] but in the stand-alone setting only. The proofs of Theorem 4 and Theorem 5 appear in the extended version of this paper [3].

Theorem 4 (Full Security Against a Malicious P_1). *Assuming that the underlying COT is secure in the malicious setting, the Fully Secure Scheme is fully-secure against a malicious P_1 in the COT-hybrid setting.*

Theorem 5 (Full Security Against a Malicious P_2). *Assuming that the underlying COT is secure in the malicious setting, the Fully Secure Scheme is fully-secure against a malicious P_2 in the COT-hybrid setting.*

5 Efficiency

5.1 The Basic Scheme

The cost of the basic scheme described in Figure 3 is essentially the cost of n OT_1^2's of inputs of $\log(n)$ bits. Using the OT extension of [16], when many OT's are performed, the workload turns out to be two evaluations of a hash function for P_1 and one for P_2 per input bit. The bandwidth requirement is then roughly $2n \cdot \log(n)$ bits.

Comparison to Previous Schemes. Let us compare our Basic Scheme to two previous protocols [15], [20,29] for semi-honest secure Hamming Distance computation, previously known as the most efficient proposals.

Other techniques, like Private Set Intersection Cardinality [9] or Private Scalar Product Computation [12] can be easily adapted to perform secure Hamming distance computation. However, in these proposals, use of homomorphic encryption and/or a linear number of exponentiations leads to schemes that are less efficient than our proposal in the semi-honest model.

We first compare to the application of the Yao algorithm to Hamming distance computation described in [15]. In this setting, the Hamming distance function has to be represented as a binary circuit. To get an idea of the cost of the computation, we need to count the number of non-XOR gates in this circuit. Let us assume that the size n of the inputs is a power of 2: $n = 2^N$. The number G of non-free gates is obtained (see the description of the Counter circuit in [15]) by $G = \Sigma_{i=1}^{N}(2^{N-i}.i) \approx 2^{N+1} = 2n$. Let k be the security parameter of the scheme. For the generation of the circuit, party P_1 has to perform $4G$ hash function evaluations. Then, P_1 sends the circuit ($3k \cdot G$ bits) and his keys for the circuit ($n \cdot k$ bits). Then P_1 and P_2 perform n OT_1^2's on k-bit strings. P_2 has then to perform G hash functions evaluations. Using the OT extension of [16], the workload of P_1 is roughly $10n$ hash functions evaluations, the workload of P_2 is $3n$ hash function evaluations and the bandwidth is $6kn$ bits. When m Hamming distances on the same input of P_2 are evaluated, all these operations but the oblivious transfers of P_2's inputs have to be computed m times.

We now evaluate the workload and bandwidth requirements of the [20,29] algorithm. The binHDOT protocol presented in [20] enables evaluation of a class of functions depending on Hamming distance. We here consider its reduction to the evaluation of the Hamming distance only. We describe the corresponding protocol in the extended version of this paper. We moreover take into account, in our evaluations, the optimizations presented in [29]. Party P_2 prepares n homomorphic ciphertexts, encrypting each of his inputs bits. These ciphertexts are sent to P_1 who homomorphically adds and subtracts them to obtain the encryption of the Hamming distance. Taking into account the optimizations of [29] (although we do not separate off-line and on-line phases), P_1 has to perform n homomorphic encryptions and P_2 n homomorphic additions. They mainly exchange n ciphertexts. When m distances are computed, with the optimizations of [29], P_2's work is almost the same and P_1 has to perform $mn/2$ homomorphic additions, once n subtractions and $3.5n$ additions are preprocessed. The bandwidth depends on the option and on the receiver of the result.

The comparison of these 3 protocols is summed up in Table 1, where **hash** means hash function evaluations and k is the security parameter of the Yao algorithm of [15]. We extrapolate to the simultaneous computation of m Hamming distances (see [3, Section 5.2]) in Table 2. In the first line of Table 2, the $(+m)$ hom. ciphertexts corresponds to the case where P_2 gets the result instead of P_1.

For concrete estimations, k should be at least 80 and Paillier ciphertexts at least 2048-bit long. It is easy to see that, for reasonable sizes of n, our scheme is more efficient and requires significantly less bandwidth. In these tables, we do not mention the k base OT's that are needed in our basic scheme and in the scheme of [15] for OT extension. They can be preprocessed.

Implementation Results. To prove our allegations regarding efficiency improvements in terms of computational workload, we ran the implementation of secure Hamming distance used in [15] and an implementation of our basic scheme using the same framework [34] on the same computer. The framework is

Table 1. Secure Computation of One Hamming Distance in the Semi-Honest Model

	P_1	P_2	Bandwidth (bits)
[20,29]	n hom.add.	n hom.enc.	n hom.ciphertexts
[15]	$10n$ **hash**	$3n$ **hash**	$6kn$
The Basic Scheme	$2n$ **hash**	n **hash**	$2n\log(n)$

Table 2. Secure Computation of m Hamming Distances in the Semi-Honest Model

	P_1	P_2	Bandwidth (bits)
[20,29]	$mn/2$ hom.add.	n hom.enc.	$n(+m)$ hom.ciphertexts
[15]	$(2+8m)n$ **hash**	$(1+2m)n$ **hash**	$(2+4m)kn$
The Basic Scheme	$2n$ **hash**	n **hash**	$2mn\log(n)$

implemented in Java and we ran it on a single computer with a 2 GHz Intel Core i7 processor and a 4 GB RAM. We think that the ratio of computation times between the protocols is more relevant than an absolute value of the time of execution of our process. This comparison is illustrated in Figure 5. For inputs with a few thousands bits size, the computation time required for our Basic scheme is approximately 22% of the time required to compute the protocol of [15].

5.2 The Fully Secure Scheme

We assume that the *COT* of the Fully Secure Scheme is the one of [21], using a threshold El-Gamal cryptosystem. According to [21], 24 exponentiations are required per *COT*, once the inputs are committed.

P_1 performs $2n$ commitments and runs n π_1^2 proofs on the commitments. He participates in n COT's as a sender. He finally computes a product of n ciphertexts (or $2n$ for the 2^{nd} option). P_2 performs n commitments and runs n π_1^2 proofs on the commitments. He participates in n COT's as a receiver.

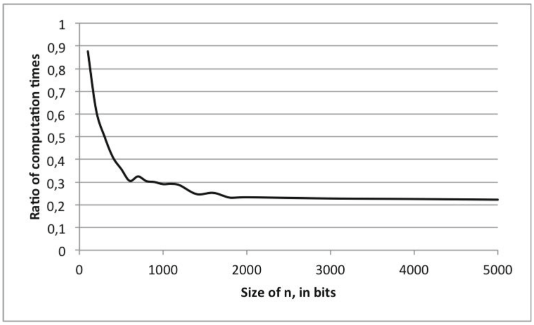

Fig. 5. Ratio computation times between our Basic Scheme and the protocol of [15]

He finally computes a product of n ciphertexts (or $2n$ for the 2^{nd} option). The bandwidth mainly comprises $3n$ commitments and n COT's.

In [20], Jarrous and Pinkas also propose an adaptation of their binHDOT protocol to the malicious setting. They also use a particular Committed Oblivious Transfer functionality, with proofs that the inputs differ by a constant number Δ, while we prove that our inputs always differ by 1. However, their protocol (for a more generic functionality) ends with an oblivious polynomial evaluation.

References

1. Aumann, Y., Lindell, Y.: Security against covert adversaries: Efficient protocols for realistic adversaries. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 137–156. Springer, Heidelberg (2007)
2. Blanton, M., Gasti, P.: Secure and efficient protocols for iris and fingerprint identification. In: Atluri, V., Diaz, C. (eds.) ESORICS 2011. LNCS, vol. 6879, pp. 190–209. Springer, Heidelberg (2011)
3. Bringer, J., Chabanne, H., Patey, A.: Shade: Secure hamming distance computation from oblivious transfer. IACR Cryptology ePrint Archive 2012, 586 (2012)
4. Camenisch, J., Stadler, M.: Efficient group signature schemes for large groups (extended abstract). In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 410–424. Springer, Heidelberg (1997)
5. Camenisch, J., Stadler, M.: Proof systems for general statements about discrete logarithms. Tech. rep., Dept. of Computer Science, ETH Zurich (1997)
6. Canetti, R.: Security and composition of multiparty cryptographic protocols. J. Cryptology 13(1), 143–202 (2000)
7. Chor, B., Goldreich, O., Kushilevitz, E., Sudan, M.: Private information retrieval. In: FOCS, pp. 41–50 (1995)
8. Crépeau, C.: Verifiable disclosure of secrets and applications (abstract). In: Quisquater, J.-J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, pp. 150–154. Springer, Heidelberg (1990)
9. Cristofaro, E.D., Gasti, P., Tsudik, G.: Fast and private computation of set intersection cardinality. IACR Cryptology ePrint Archive 2011, 141 (2011)
10. Damgård, I., Jurik, M.: A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In: Kim, K.-C. (ed.) PKC 2001. LNCS, vol. 1992, pp. 119–136. Springer, Heidelberg (2001)
11. El Gamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 10–18. Springer, Heidelberg (1985)
12. Goethals, B., Laur, S., Lipmaa, H., Mielikäinen, T.: On private scalar product computation for privacy-preserving data mining. In: Park, C.-S., Chee, S. (eds.) ICISC 2004. LNCS, vol. 3506, pp. 104–120. Springer, Heidelberg (2005)
13. Goldreich, O.: The Foundations of Cryptography. Basic Applications, vol. 2. Cambridge University Press (2004)
14. Hazay, C., Lindell, Y.: Efficient Secure Two-Party Protocols. Springer (2010)
15. Huang, Y., Evans, D., Katz, J., Malka, L.: Faster secure two-party computation using garbled circuits. In: USENIX Security Symposium (2011)
16. Ishai, Y., Kilian, J., Nissim, K., Petrank, E.: Extending oblivious transfers efficiently. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 145–161. Springer, Heidelberg (2003)

17. Ishai, Y., Prabhakaran, M., Sahai, A.: Founding cryptography on oblivious transfer – efficiently. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 572–591. Springer, Heidelberg (2008)
18. Ishai, Y., Prabhakaran, M., Sahai, A.: Secure arithmetic computation with no honest majority. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 294–314. Springer, Heidelberg (2009)
19. Jarecki, S., Shmatikov, V.: Efficient two-party secure computation on committed inputs. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 97–114. Springer, Heidelberg (2007)
20. Jarrous, A., Pinkas, B.: Secure hamming distance based computation and its applications. In: Abdalla, M., Pointcheval, D., Fouque, P.-A., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536, pp. 107–124. Springer, Heidelberg (2009)
21. Kiraz, M.S., Schoenmakers, B., Villegas, J.: Efficient committed oblivious transfer of bit strings. In: Garay, J.A., Lenstra, A.K., Mambo, M., Peralta, R. (eds.) ISC 2007. LNCS, vol. 4779, pp. 130–144. Springer, Heidelberg (2007)
22. Kolesnikov, V., Schneider, T.: Improved garbled circuit: Free XOR gates and applications. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part II. LNCS, vol. 5126, pp. 486–498. Springer, Heidelberg (2008)
23. Lindell, Y., Pinkas, B.: An efficient protocol for secure two-party computation in the presence of malicious adversaries. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 52–78. Springer, Heidelberg (2007)
24. Lindell, Y., Pinkas, B.: A proof of security of Yao's protocol for two-party computation. J. Cryptology 22(2), 161–188 (2009)
25. Lindell, Y., Pinkas, B.: Secure two-party computation via cut-and-choose oblivious transfer. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 329–346. Springer, Heidelberg (2011)
26. Mohassel, P., Niksefat, S., Sadeghian, S., Sadeghiyan, B.: An efficient protocol for oblivious DFA evaluation and applications. In: Dunkelman, O. (ed.) CT-RSA 2012. LNCS, vol. 7178, pp. 398–415. Springer, Heidelberg (2012)
27. Naor, M., Pinkas, B.: Efficient oblivious transfer protocols. In: SODA, pp. 448–457 (2001)
28. Nielsen, J.B., Orlandi, C.: LEGO for two-party secure computation. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 368–386. Springer, Heidelberg (2009)
29. Osadchy, M., Pinkas, B., Jarrous, A., Moskovich, B.: Scifi - A system for secure face identification. In: IEEE Symposium on Security and Privacy, pp. 239–254 (2010)
30. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)
31. Peikert, C., Vaikuntanathan, V., Waters, B.: A framework for efficient and composable oblivious transfer. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 554–571. Springer, Heidelberg (2008)
32. Pinkas, B., Schneider, T., Smart, N.P., Williams, S.C.: Secure two-party computation is practical. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 250–267. Springer, Heidelberg (2009)
33. Rabin, M.O.: How to exchange secrets with oblivious transfer. Tech. Rep. TR-81, Aiken Computation Lab, Harvard University (1981)
34. University of Maryland, University of Virginia: Might be evil: Privacy-preserving computing, `http://mightbeevil.com`
35. Yao, A.C.C.: How to generate and exchange secrets (extended abstract). In: FOCS, pp. 162–167 (1986)