9-2006

Practical Private Data Matching Deterrent to Spoofing Attacks

Yanjiang YANG
Singapore Management University, yjyang@smu.edu.sg

Robert H. DENG
Singapore Management University, robertdeng@smu.edu.sg

Feng BAO
Singapore Management University, fbao@smu.edu.sg

DOI: https://doi.org/10.1145/1183614.1183763

Practical Private Data Matching Deterrent to Spoofing Attacks

Yanjiang Yang, Robert H. Deng
SIS, Singapore Management University
Singapore 178902
(yjyang, robertdeng)@smu.edu.sg

Feng Bao
Institute for Infocomm Research
Singapore 119613
baofeng@i2r.a-star.edu.sg

ABSTRACT

Private data matching between the data sets of two potentially distrusted parties has a wide range of applications. However, existing solutions have substantial weaknesses and do not meet the needs of many practical application scenarios. In particular, practical private data matching applications often require discouraging the matching parties from *spoofing* their private inputs. In this paper, we address this challenge by forcing the matching parties to "escrow" the data they use for matching to an auditorial agent, and in the "after-the-fact" period, they undertake the liability to attest the genuineness of the escrowed data.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; H.2.8 [**Information Systems**]: Database Management—*Database Applications*

General Terms: Security

Keywords: Data privacy, Secure multi-party computation.

1. INTRODUCTION

Private data matching considers the scenarios that two parties each having a set of data objects wish to determine the objects common to their data sets, while without revealing any additional information to each other. Private data matching could be either *symmetric* in the sense that both parties are interested in the matching result, or *asymmetric* in that one party gets the matching result and the other party is only interested to assist its counterpart for matching.

In general, private data matching belongs to the *secure multi-party computation* problems, where two parties with respective private inputs x and y compute a function $f(x, y)$ in such a way that each of them learns nothing other than the value of $f(x, y)$. However, private data matching in practice needs to address issues beyond the scope of secure multi-party computation. More specifically, given x and y secure multi-party computation concerns with the way $f(x, y)$ is computed that does not reveal x and y, whereas practical private data matching needs to additionally guarantee that the two matching parties' private inputs x and y are *genuine*, i.e., x and y are indeed the data owned by the two parties. As a result, while sharing certain generic properties of secure multi-party computation, private data matching in practice has an extra dimension to consider, i.e., to prevent the matching parties from spoofing their private inputs.

A weakness of many existing works on private matching such as [1, 5, 7] is that they assume the two matching parties follow semi-honest behaviors [6], which, among others, stipulates that the matching parties do not spoof their private inputs. This appears to be too strong an assumption for many data matching applications. The reference [8] gave several protocols to address input spoofing by the two matching parties, but it has changed the implication of private data matching in the sense that data objects of equal values as well as originating from the same users are matched (private data matching by definition actually refers to matching data objects of equal values).

2. OUR APPROACH

We propose an "after the fact" detection approach to deter the matching parties from spoofing their inputs. More specifically, we assume an *auditing mechanism* within our system that takes the responsibility for overseeing the data matching transactions: the matching parties may share a single auditorial agent or each matching party (or a group of matching parties) has an independent auditorial agent based on, e.g., administrative domains. For simplicity, we assume all parties have a single auditorial agent in the sequel. In the process of data matching, each matching party "escrows" the data objects it uses for matching to the auditorial agent, i.e., the party encrypts its data objects using the public key of the auditorial agent, and submits the encrypted data together with the transcript components for matching. As a result, the receiving matching party gets a data escrow that can be opened by the auditorial agent. In a later "after the fact" detection phase either under the demand of the other party or to go through certain regular oversight formalities, a party is liable to attest the genuineness of its "escrowed" data objects to the auditorial agent.

A challenge in this approach is to enable a party to convince the other party that the same data objects are used in escrow and for matching, while without revealing extra information. Our basic idea is as follows. Suppose a data object m is used by a party for matching. The matching transcript includes two main components: the first component is $h_m = m^e$ used for data matching, where e is a random number; the second is $C_m = (c_1 = m(PK)^r, c_2 = g^r)$

CIKM'06, November 5–11, 2006, Arlington, Virginia, USA.
ACM 1-59593-433-2/06/0011.

serving as the escrow of m, where PK is the public key (ElGamal encryption) of the auditorial agent, and r is another random number. We then use a zero-knowledge proof of knowledge (e.g., [2, 3, 4]) to show that h_m and c_{m1} contain the same data object m.

Adversarial model. We assume *malicious matching parties* in our system. A malicious party can act in arbitrary ways, e.g., it can submit a spoof query by adding bogus data to its private input, or terminate a protocol prematurely at any point of execution.

System setup. Each of the two matching parties Alice and Bob has a key pair for a standard digital signature algorithm. The matching parties need to sign the "escrowed" data to show their irrefutable involvement. We denote $\text{SIGN}_A(.)$ (resp. $\text{SIGN}_B(.)$) the signing by Alice (resp. Bob). The auditorial agent has a public-private key pair $\langle y = g^x \pmod p, x\rangle$ that corresponds to AlGamal encryption, where $p = 2q + 1$ is a large prime (q is also a prime), and $g \in QR_p$ with QR_p denoting the subgroup of quadratic residues in Z_p^* of order q. The system also decides on a cryptographic hash function $\mathcal{H} : \{0, 1\}^* \rightarrow QR_p$.

The protocol. For limit of space, we only present the protocol for asymmetric private data matching, where Alice eventually gets the matching result. The protocol is outlined below, where Alice has a genuine data set $DS_A = \{a_1, a_2, ..., a_{N_A}\}$ of N_A data objects, and Bob has a genuine data set $DS_B = \{b_1, b_2, ..., b_{N_B}\}$ of N_B data objects. We use $NPoK\{(\alpha) : y = g^\alpha\}$ to denote the non-interactive "zero-knowledge Proof of Knowledge of a value α such that $y = g^\alpha$ holds", following the notations in [4].

1. Local computation by Alice:
   (a) Alice picks a random number $e_A \in_R Z_q$.
   (b) For each data object a_i ($i = 1..N_A$), Alice computes $h_{a_i} = \mathcal{H}(a_i)^{e_A} \pmod p$, as well as $c_{i1} = \mathcal{H}(a_i).y^{r_i} \pmod p$ and $c_{i2} = g^{r_i} \pmod p$, where $r_i \in_R Z_q$.
   (c) Alice constructs a non-interactive zero-knowledge proof of knowledge $nPoK_A = NPoK\{(\alpha, \beta_1, ..., \beta_{N_A}) : \bigwedge_{i=1}^{N_A}(h_{a_i} = (c_{i1})^\alpha (1/y)^{\beta_i} \wedge 1 = (c_{i2})^\alpha (1/g)^{\beta_i})\}$
   (d) Alice signs $C_A = \{(c_{i1}, c_{i2}) : i = 1..N_A\}$ by computing $\sigma_A = \text{SIGN}_A(C_A)$.
   (e) $S_A = \varnothing, S_B = \varnothing$.
2. Local computation by Bob:
   (a) Bob picks a random number $e_B \in_R Z_q$.
   (b) For each data object b_j ($j = 1..N_B$), Bob computes $h_{b_j} = \mathcal{H}(b_j)^{e_B} \pmod p$, and $d_{j1} = \mathcal{H}(b_j).y^{s_j} \pmod p$ and $d_{j2} = g^{s_j} \pmod p$, where $s_j \in_R Z_q$.
   (c) Bob constructs a non-interactive zero-knowledge proof of knowledge $nPoK_B = NPoK\{(\alpha', \beta'_1, ..., \beta'_{N_B}) : \bigwedge_{j=1}^{N_B}(h_{b_j} = (d_{j1})^{\alpha'}(1/y)^{\beta'_j} \wedge 1 = (d_{j2})^{\alpha'}(1/g)^{\beta'_j})\}$.
   (d) Bob signs $D_B = \{(d_{j1}, d_{j2}) : j = 1..N_B\}$ by computing $\sigma_B = \text{SIGN}_B(D_B)$.
3. Alice sends to Bob $\{(h_{a_i}, c_{i1}, c_{i2}) : i = 1..N_A\}$ together with $nPoK_A$ and σ_A.
4. Bob does the following:
   (a) Bob checks that σ_A is a valid signature on $C_A = \{(c_{i1}, c_{i2}) : i = 1..N_A\}$.
   (b) Bob checks the validity of $nPoK_A$.
   (c) Upon completing the above validation steps, Bob computes $v_i = (h_{a_i})^{e_B} \pmod p$ for each h_{a_i}.
   (d) Bob finally returns $\{(h_{b_j}, d_{j1}, d_{j2}) : j = 1..N_B\}$, $nPoK_B$, σ_B, and $\{(h_{a_i}, v_i) : i = 1..N_A\}$ to Alice.
5. Alice does the following:
   (a) Alice checks that σ_B is a valid signature on $D_B = \{(d_{j1}, d_{j2}) : j = 1..N_B\}$.
   (b) Alice checks the validity of $nPoK_B$.
   (c) Upon completing the above validation, Alice computes $u_j = (h_{b_j})^{e_A} \pmod p$ for each h_{b_j}, and includes u_j into S_B, i.e., $S_B = S_B \cup \{u_j\}, j = 1..N_B$.
   (d) Alice includes each v_i into S_A, i.e., $S_A = S_A \cup \{v_i\}, i = 1..N_A$. Alice determines $\{h_{a_i} : v_i \in S_A \cap S_B\}$, and from h_{a_i} Alice decides the corresponding a_i. The set of a_i thus formed is the matching result.

"After the fact" detection. In the "after the fact" detection step, a matching party is required to attest the genuineness of the data objects it escrowed in matching. Let us suppose the auditorial agent gets $\{(c_{i1}, c_{i2}) : i = 1..N_A\}$ and σ_A from Bob, and asks Alice for attestation. The auditorial agent first checks whether σ_A is a valid signature on $\{(c_{i1}, c_{i2}) : i = 1..N_A\}$. Upon determining that the data are indeed from Alice, the auditorial agent decrypts each (c_{i1}, c_{i2}) to get $h'_{a_i} = c_{i1}/(c_{i2})^x \pmod p$ using its private key x. Since the "escrowed data" h'_{a_i} is a hash value, Alice is required to surrender the original data object a'_i together with the proof that vouches for the genuineness of a'_i. The auditorial agent checks whether $h'_{a_i} = \mathcal{H}(a'_i)$ and whether the proof supports that a'_i is indeed a genuine data object.

Deterrent to spoofing attack. Our method of data escrow together with the zero-knowledge proof of knowledge forces the matching parties to correctly "escrow" the data they use in matching to the auditorial agent. As a consequence, if a matching party spoofs its input, it cannot produce to the auditorial agent valid proofs for the bogus data objects. Depending on applications, failure to present correct attestation proofs for the data objects it uses in matching, a party may face compromise of reputation or punitive legal actions. We thus believe that leaving the evidence at the auditorial agent for future spoofing investigation should be a sufficient deterrent to spoofing attacks.

3. REFERENCES

[1] R. Agrawal, A. Evfimievski, R. Srikant, Information Sharing Across Private Databases, Proc. *ACM International Conference on Management of Data, SIGMOD'03*, pp. 86-97, 2003.

[2] D. Chaum, Zero-knowledge Undeniable Signatures, Proc. *Advances in Cryptology, Eurocrypt'90*, LNCS 473, pp. 458-464, 1990.

[3] D. Chaum, J. H. Evertse, J. Graaf, R. Peralta, Demonstrating Possession of a Discrete Logarithm Without Revealing it, *Proc. Advances in Cryptology, Crypto'86*, LNCS 263, pp. 200-212, 1986

[4] J. Camenisch, M. Stadler, Efficient Group Signature Scheme for Large Groups, *Advances in Cryptology, Crypto'97*, LNCS 1296, pp. 410-424, 1997.

[5] M. J. Freedman, K. Nissim, B. Pinkas, Efficient Private Matching and Set Intersection, Proc. *Advances in Cryptology, Eurocrypt'04*, LNCS 3027, pp. 1-19, 2004.

[6] O. Goldreich, Secure Multi-party Computation, Final (incomplete) draft, version 1.4, 2002.

[7] L. Kissner, D. Song, Privacy-Preserving Set Operations, *Advances in Cryptology, Crypto'05*.

[8] Y. P. Li, J. D. Tygar, J. M Hellerstein, Private Matchig, *Computer Security in the 21st Century*, Springer, pp. 25-50, 2005.