Secure and Private Outsourcing of Shape-Based Feature Extraction

Shumiao Wang[1,⋆], Mohamed Nassar[2], Mikhail Atallah[1], and Qutaibah Malluhi[2]

[1] Computer Science Department, Purdue University, West Lafayette, USA
wang845@purdue.edu, mja@cs.purdue.edu
[2] Computer Science and Engineering, Qatar University, Doha, Qatar
meb.nassar@gmail.com, qmalluhi@qu.edu.qa

Abstract. There has been much recent work on secure storage outsourcing, where an organization wants to store its data at untrusted remote cloud servers in an encrypted form, such that its own employees can query the encrypted data using weak devices (both computationally and storage-wise). Or a weak client wants to outsource an expensive computational task without revealing to the servers either the inputs or the computed outputs. The framework requires that the bulk of the computational burden of query-processing be placed on the remote servers, without revealing to these servers anything about the data. Most of the existing work in this area deals with non-image data that is keyword based, and the present paper is to deal with raw image data (without any keyword annotations). We demonstrate that shape-based image feature extraction, a particularly computationally intensive task, can be carried out within this framework, by presenting two schemes for doing so, and demonstrating their viability by experimentally evaluating them. Our results can be used in a number of practical situations. In one scenario the client has images and wants to securely outsource shape-based feature extraction on them, in another the server has encrypted images and the client wants a feature-extracted representation of those that are feature-rich.

Keywords: Secure Outsourcing, Feature Extraction, Cloud Service.

1 Introduction

One of the major impediments to larger-scale use of cloud services is concern for confidentiality of the data and the queries carried out on it. This has motivated much of the recent work on secure storage and computational outsourcing. In the storage outsourcing setting that interests us, a data owner wants to store its

⋆ Portions of this work were supported by National Science Foundation Grants CPS-1329979, CNS-0915436, CNS-0913875, Science and Technology Center CCF-0939370; by an NPRP grant from the Qatar National Research Fund; and by sponsors of the Center for Education and Research in Information Assurance and Security. The statements made herein are solely the responsibility of the authors.

S. Qing et al. (Eds.): ICICS 2013, LNCS 8233, pp. 90–99, 2013.

data in encrypted form at untrusted remote cloud servers, after which trusted clients can query it using weak devices (both computationally and storage-wise) in such a way that the bulk of the computational burden of the query-processing is placed on the remote servers without leaking the data to the servers. The main technical challenges are (i) how to get the untrusted servers to do the query-processing and associated computational work without leaking the data to them (the security issue), and (ii) how to lighten the clients' computational burden and lessen the number of rounds in the client-server interaction (the efficiency issue). Many problems have been considered in previous work, for text, numerical data, spatial data, etc (as we will review later), there has been almost no work that deal with raw image data. The present paper addresses the problem of secure and private outsourcing of feature extraction of image data. Feature extraction from an image is a fundamental operation in image processing, and has the goal of producing a reduced representation of the image that is more computationally tractable than working with the raw image. In shape-based feature extraction, the image is reduced to a collection of based shapes (line segments, circles, etc) for subsequent processing, and namely, these shapes (represented by their parameters) are a set of features of the input image. It is a computationally expensive operation, especially for massive image data (such as satellite images), and is therefore an ideal candidate for outsourcing. To be specific, we are outsourcing the Hough Transform method, which is a widely used shape-based feature extraction method in computer vision [9]. The essential idea of this method is that parametric shapes in an image are detected by looking for accumulation points in the parameter space (the detailed detection process will be reviewed in Section 3). The input images are used to produce a set of features in each by the servers, after which any trusted client can query images at the servers for the features.

We assume honest-but-curious untrusted servers (as is done in most of the research work in the secure outsourcing setting). And our privacy model is that the nature of the basic shapes (line segment, circle, etc) is not hidden from the servers, and what is confidential is the positions at which they occur and how they fit together to form complex patterns. Two approaches based on the Hough Transform method are proposed in this paper: the preliminary approach is simple, easy to implement, and has good performance, while it may leak some minor information about the input image; the second approach is provably secure, but it needs the implementation of more expensive cryptographic tools, such as the garbled circuit protocol [15]. The performances of the two approaches are evaluated by experiments, that quantify their security/efficiency trade-offs.

2 Related Work

There is much previous work in the area of secure outsourcing, and we lack the space for a comprehensive review of all the related research problems, so we give examples of what has been achieved. Paper [14] presents an implementation of Oblivious RAM which allows clients with limited (logarithmic) local storage to

store their data on untrusted storage and retrieve it securely without leaking either the data or the access pattern to the server. Paper [5] shows how to securely outsource modular exponentiation with two untrusted servers. Papers [1] and [2] deal with the problem of securely outsourcing matrix computations.

In the area of secure image processing, related work exists in outsourcing of feature extraction in images directed towards face recognition. [11] and [13] address the problem of comparing subject faces with a database of faces and, at the same time, preserving the privacy of the subject faces and the confidentiality of the database. They are based on specific algorithms to extract features from face images, e.g, Eigenfaces. [7] works on extracting another kind of features, namely scale-invariant feature transform (SIFT). While these contributions are similar to ours, they are not shape-based and thus cannot be used to identify and analyze the structural components of images.

3 Building Blocks

Hough Transform. The Hough Transform (HT) method was introduced by P.V.C. Hough in [6]. One input to HT is a binary image with each pixel either 1 or 0, where 1 is the pixel representing data and 0 is the background pixel. Another input is the based shape (the nature of the features to be extracted), which can be represented using a number of parameters in cartesian coordinates. Generally, any parameterized shape can be represented by a vector of parameters as $\overrightarrow{p}$ in the equation of the form $f(\overrightarrow{p}, \overrightarrow{x}) = 0$, where $\overrightarrow{x}$ is the coordinates vector in cartesian coordinates.

Given the based shape (e.g, straight line), HT first quantizes the parameters and initializes an array with a cell for each possible parameter vector of the shape in the quantized parameter space (e.g, we use a two dimension array for the ρ-θ space of straight lines, described in Table 1). In the rest of the paper, we may use the index of a cell to represent its corresponding parameter vector. Then for each parameter vector in the parameter space, HT counts how many data pixels are lying on its corresponding shape instance, and records this count in the corresponding cell in the array. This is the accumulation process, and the resulting array is called the accumulation array or accumulation matrix. The cells in the accumulation array with high counts, *and* which are local maxima, correspond to shape instances. If a $\overrightarrow{p}$'s cell in the accumulation array is (i) a local maximum, and (ii) greater than a pre-determined threshold value t, then that cell is considered to be the parameter vector of an instance of the based shape. The set of all the indexes of such cells gives all the parameter vectors of occurrences of the based shape in the image, which are the features extracted.

Paillier's Homomorphic Encryption. The Paillier's Homomorphic Encryption[12] possesses the following properties. (i) It's a public key scheme. (ii) It's probabilistic. (iii) It possesses the homomorphic property that $E(M_1) * E(M_2) = E(M_1 + M_2)$ holds for any M_1 and M_2, where E denotes the encryption, $+$ is modular addition, and $*$ is modular multiplication.

Gaussian Blur. Gaussian blur [4] is a technique to blur an image by a Gaussian function. In this work, we do *not* use such blurring on any image, but we do use it

on the accumulation matrix in computing the local maxima, which preserves local maxima with high probability. A Gaussian Blurring function specifies a group of integer weights $w_0, w_1, \ldots, w_8$ to compute a weighted average of a cell (with weight w_0) and its 8 neighbors (with weight $w_1, \ldots, w_8$ respectively), and the weights satisfy the constraint $w_0 \geq \sum_{i=1}^{m} w_m$. And by blurring an accumulation matrix, we mean to compute the weighted sum of each cell and its neighbors with the weights specified in the function as the resulting cell.

Blind and Permute. The input to the Blind and Permute (BP) protocol is a sequence of data items $S = (s_1, s_2, \cdots, s_n)$ whose values are component-wise additively split between party A who has $S' = (s'_1, s'_2, \cdots, s'_n)$ and party B who has $S'' = (s''_1, s''_2, \cdots, s''_n)$, where $s_i = s'_i + s''_i$ for $i = 1, \cdots, n$. The output is a sequence $\hat{S}$ (also additively split between A and B) obtained from S by (i) permuting the entries of S according to a random permutation π that is known to neither A nor B; and (ii) modifying the additive split of the entries of S so that neither A nor B can use their share of it to gain any information about π. A BP protocol (adapted from [3]) is used in the our secure approach.

Garbled Circuits. Garbled Circuits, first presented by Yao in [15], is a cryptographic technique for securely evaluating two-party functionalities. A two-party functionality can be written as $f(x, y) = (f_1(x, y), f_2(x, y))$, where x and y are the private inputs from the two parties, and after the evaluation of the function, one party receives $f_1(x, y)$ and the other one receives $f_2(x, y)$ as the outputs. Neither of the two parties should learn anything about the other's input other than what can be inferred from his own input and output. We refer to [10] for a review of Yao's protocol and a rigorous security proof.

4 Approaches

A preliminary approach using blurring method is provided to prevent possible leakage of information in Section 4.1 with a provably secure approach provided in Section 4.2. The outsourcing framework consists of four parties: the Data Owner DO, the Clients C, the first and second cloud server S_1 and S_2.

4.1 A Preliminary Approach with Homomorphic Encryption and Blurring

We first give an overview of this approach. For each image, DO specifies the shape(s) to be based on, encrypts each pixel in the image by the homomorphic encryption scheme whose decryption key is shared with S_2, and then sends the encrypted image to S_1 for analysis. With the homomorphic property, S_1 generates an encrypted accumulation array for each shape under detection without decryption, and associates each cell in the array with its encrypted index and some neighboring information in order to allow S_2 to check whether it's a local maximum cell, then permutes the cells and sends them to S_2. S_2 decrypts the cells by the homomorphic encryption key, finds out the local maxima after thresholding, and stores the qualified indexes(encrypted) as the set of features.

(The main challenge here is that S_2 should find the local maxima without knowing the index information.) The scheme is described in detail in the following, using straight line as an example of the based shape(s)to simplify the description.
Initialization. *DO* initializes the scheme by specifying the shape(s) to be based on and the global parameters (e.g, the ρ-θ space) to be used by the servers. *DO* generates a public and private key pair (K_E, K_D) for Paillier's homomorphic encryption scheme, publishes the public key K_E, and sends the private key K_D only to S_2. *DO* generates a key K for a symmetric encryption scheme (e.g. AES), and shares it with S_1 and C, which is used to encrypt the indexes of cells in the accumulation array as mentioned in the overview. In addition, if *DO* wants to hide the image *id* or to share more information of the images with C which are not supposed to be seen by the servers, he generates another symmetric key K_{DC} shared only to C, and uses it to encrypt the image *id* and other information.
Analyzing a New Image. To analyze a new image I with *id* and add it to the existing database, *DO* encrypts its *id* into $E_{K_{DC}}(id)$ to hide it, encrypts the image pixel by pixel with K_E, and gets an encrypted image, denoted by $E_{K_E}(I)$. *DO* sends $(E_{K_{DC}}(id), E_{K_E}(I))$ to S_1.

Accumulation. In this step, S_1 uses the homomorphic property of the Pallier's scheme, to generate the encrypted accumulation array *ACC* for the shape of interest, and we only consider the straight line here. Recall that HT counts the pixels with value 1 on a straight line for the cell in *ACC* corresponding to its parameter vector. In this scheme, for every possible parameter vector in ρ-θ space, S_1 calculates the sum of every pixel value on its corresponding straight line by multiplying the encryptions of them. After this step, the encrypted counts for all possible parameter vectors are obtained in the matrix *ACC*.

Processing Local Information and Permutation. Before sending *ACC* to S_2, S_1 should randomly permute all the cells and associate each cell $ACC[i][j]$ with its encrypted index $E_K(i, j)$, and provide enough information for S_2 to find the local maxima without seeing the indexes. S_1 computes the gradients for each cell in *ACC* before permutation and associate them with the cell, so that S_2 could check whether it is a local maximum cell after decryption. Here we define the gradients of a cell $ACC[i, j]$ as the difference value between it and its neighbors, and subtractions on the plaintexts could be performed by divisions on the ciphertexts according to the homomorphic property. In order to break the symmetry of the gradients between two neighbors, before computing the gradients, S_1 chooses two different simplified Gaussian functions, performs the two Gaussian blurring processes separately on *ACC* and gets ACC_1 and ACC_2 as the resulting matrices respectively. Gaussian blur preserves the local maxima with overwhelming probability due to the heavy weight of the central cell, so the local maxima in *ACC* will be preserved in ACC_1 and ACC_2 very likely. To compute the gradients for each cell $ACC[i][j]$, instead of using the exact values from *ACC* and performing the subtractions, if $i + j$ is odd, S_1 uses the values from corresponding cells of ACC_1; otherwise, uses the values from ACC_2. For each cell $ACC[i][j]$, S_1 creates a tuple $(E_K(i, j), ACC[i, j]$, its Gradients), and

permutates all the tuples in the array randomly. S_1 sends all the tuples in an array along with $E_{K_{DC}}(id)$ to S_2.

Detection and Storage. In this step, S_2 receives the data, and for each tuple, he decrypts the gradients and checks whether it's a local maximum. If not, just discard it; otherwise, he decrypts $ACC[i, j]$ and if it's also beyond a pre-fixed threshold, the index is corresponding to an occurrence of the based shape. S_2 saves all the qualified indexes of occurrences of the based shape detected in I, and stores them with $E_{K_{DC}}(id)$.

Querying Phase. C queries S_2 with the encrypted image id $E_{K_{DC}}(id)$, gets back all the encrypted indexes of occurrences of the based shape, and decrypts them to recover the parameters as the features of the image.

4.2 Secure Approach with Additive Splitting and Garbled Circuit

The overall idea of this approach is that DO additively splits the image randomly into two shares, and sends one share to S_1, the other to S_2, so that the two servers can perform the accumulation process locally, and then collaborate to detect the parameters for the occurring based shapes without seeing any information about the original image. To add an image in this approach, instead of encrypting each pixel by encryption as in the preliminary approach, the DO additively splits each pixel into two secret shares: DO first chooses a modulo m, say 2^{32}, which should be larger than any value in the accumulation array; then for each pixel value v, DO randomly chooses v_1 over $[0, m-1]$, and splits v as v_1 and $v_2 = v - v_1 \bmod m$. And it can be proved that if I' and I'' are the two shares obtained by pixel-wise additive splitting of an image I, which means $I = I' + I'' \bmod m$, then the accumulation matrices produced by I' and I'' are a pair of additive splitting shares of the accumulation matrix produced by I. With this property, S_1 and S_2 could perform the accumulation process for I' and I'' separately without knowing I. Then they work together to detect the local maximum cells in the accumulation matrix of I, and which is the main challenge when designing this scheme and will be handled later. After the detection, one server could store the results in encrypted form and serve the clients for queries.

We first present our solution for the problem that two parties share an additive splitting of an accumulation matrix $M = M' + M'' \bmod m$, say A has M' and B has M'', and they want to compute the indexes of local maximum cells in M which are also beyond a given threshold t. In this protocol, we consider local maximum cells as those which are greater than its 8 neighbors within the radius equal to 1, while this radius can be adjusted as discussed in Table 1, and so is the protocol. The computation should not leak one party's share to the other and neither should see the result indexes.

Let M, M' and M'' be matrices of size $p \times q$, and the size is public known to A and B. For each cell in M' (resp, M''), A (resp, B) constructs a 3×3 square matrix which is the 3×3 submatrix of M' (resp, M'') centering at this cell (for those cells who do not have 8 neighbors, pad 0's to make a square matrix). A (resp, B) orders all the square matrices sequentially as a sequence $S' = s'_1, \cdots, s'_n$

(resp, $S'' = s''_1, \cdots, s''_n$), where s'_i (resp, s''_i) is the square matrix corresponding to the cell $M'[j][k]$ (resp, $M''[j][k]$) with $j = \lfloor i/q \rfloor, k = i - q * j$.

Note that now A and B together have the local information for each cell to determine whether it's a local maximum cell in M, and both of them know the indexes of the cells. We start with a variation of the BP protocol to allow them to permute their sequences of square matrices with the same permutation, which is known to neither of them but could be recovered by the clients.

Initialization. Both of A and B initialize a public homomorphic encryption scheme, denoted by E_A and D_A (resp, E_B and D_B) in this protocol. (The decryption key of B should be shared with the clients, and we will explain this later.) We use $E_A(s_i)$ or $D_A(s_i)$ to denote encrypting/decrypting a 3×3 square matrix s_i cell by cell.

One Direction Blind and Permute.

1. A computes and sends $E_A(s'_1), \cdots, E_A(s'_n)$ to B.
2. B generates n 3×3 random matrices $r_1, \cdots, r_n$, and for $i = 1, \cdots, n$ he computes $E_A(-r_i)$ and cell-wisely multiplies it to $E_A(s'_i)$, thereby obtaining $E_A(s'_i - r_i)$. B associates $E_B(i)$ to the matrix $E_A(s'_i - r_i)$ as an index field.
3. B generates a random permutation π_B and applies it to the sequence of $E_A(s'_i - r_i)$'s computed in the previous step, obtaining a sequence of the form $E_A(v'_1), \cdots, E_A(v'_n)$ that he sends to A. He also applies π_B to the sequence $s''_1 + r_1, \cdots, s''_n + r_n$, obtaining a sequence $V'' = v''_1, \cdots, v''_n$ as his new share.
4. A decrypts the n items $E_A(v'_1), \cdots, E_A(v'_n)$ received from B, obtaining the sequence $V' = v'_1, \cdots, v'_n$ as the new share.

The Other Direction Blind and Permute. A and B repeat the one direction BP protocol by changing their roles with their new sequences V' and V'' as inputs instead of S' and S''. The result of this step is they both possess a sequence which two together form an additive splitting of the original sequence S after permutation $\pi_A(\pi_B)$. Now each square matrix in A's sequence is associated with an index filed which is the encryption of its original index in S as $E_B(i)$, and each one in B's sequence is associated with $E_A(\pi_B(i))$.

Detection of Local Maxima. For each aligned pair of square matrices in their sequence, A and B perform the garbled circuit protocol to determine (i) whether its central cell in the original matrix M is greater than all its neighbors and (ii) is beyond the given threshold t, and reveal the answer to party A. If it satisfies the two conditions, A adds the encrypted index to a result set. Let s'_i, held by A, and s''_i, held by B, be a pair of aligned square matrices, which are a pair of additive splitting shares modulo by m. Let

$$s'_i = \begin{pmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ a_7 & a_8 & a_9 \end{pmatrix}, s''_i = \begin{pmatrix} b_1 & b_2 & b_3 \\ b_4 & b_5 & b_6 \\ b_7 & b_8 & b_9 \end{pmatrix}. \tag{1}$$

We define the functionality to check the two conditions as:

$$f(s'_i, s''_i) = equal(add(a_5, b_5), \max_{j=1}^{9}(add(a_j, b_j))) \wedge gt(add(a_5, b_5), t) \tag{2}$$

A and B perform a garbled circuit protocol for f on each pair of the input square matrices, and enable A to learn the result. If the result is 1, which means the cell of the input square matrix is a local maximum point and beyond t, A stores the associated encrypted index for later queries. Now we are ready to present the main scheme for outsourcing feature extraction.

Initialization. *DO* initializes the scheme by specifying the shape(s) to be based on and the global parameters (e.g, the ρ-θ space) to be used by the servers. *DO* generates the homomorphic key pairs for S_1 and S_2 to be used in the BP protocol and shares the decryption key with the clients.

Analyzing a New Image.

1. To add a new binary image I to the database, *DO* additively splits it as $I = I' + I''$, and shares I' (resp, I'') to S_1 (resp, S_2).
2. S_1 (resp, S_2) performs Hough accumulation on I' (resp, I'') for the based shape specified by *DO*, obtaining the accumulation matrix M' (resp, M'').
3. S_1 and S_2 collaborate to detect the indexes (encrypted) for local maxima in $M = M' + M''$ which are also beyond a threshold, using the method described above. WLOG, assume S_1 plays the role of A.
4. S_1 stores the set of indexes (encrypted) of occurrences of the based shape in the image as the extracted features.

Querying Phase. The query part of this scheme is similar to the preliminary approach, except that the clients interact with S_1 for querying, and decrypt the results with the key of S_2.

Analysis. From the view of S_1 and S_2, either is receiving a random image due to the property of additive splitting secret sharing. They perform accumulation on their own share, which gives them no more information, after that they perform the provable secure BP protocol as used in [3], and then interact for detection under the garbled circuit protocol, the security of which has been proved in [10].

5 Experiment

In this section we evaluate the performance of the two approaches per party and per activity. All the parties are run on a local machine having Windows OS, Intel i5 four cores 2.67 GHz CPU, 4GB memory. For secure circuit evaluation, we adopt the approach in [8] and the tool (GCParser) from `http://www.mightbeevil.com`. The performance measurements included next are only indicative.

The most important input parameters are shown in Table 1. For this experiment we use the image shown in Fig. 1(a). The detected lines in Fig. 1(b) show a good accuracy example. However, a standalone study of the accuracy of the Hough transform is outside the scope of this paper (as it is an image processing issue). We focus more on the trade off between the accuracy and the computational demand.

For the preliminary approach, we show the breakdown by party in Fig. 2(a) and by activity in Fig. 2(b); and for the secure one, we show the breakdown by

Table 1. Discussion on the Input Parameters

Parameter	Description
ρ-θ space size	Sampling of the ρ dimension and the θ dimension. Large size means better accuracy but is more performance demanding.
Local maximization radius	Choosing a large radius may enhance the accuracy because it helps detecting only one line for a group of line segments that are close in the ρ-θ space.
Threshold	Since the parties are agnostic of the real votes for the lines, we can not filter a subset of the lines based on the threshold.

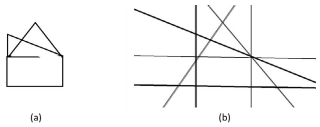

Fig. 1. The image used for experiments (a) and the extracted line-based features of the image shown as the detected lines (b)

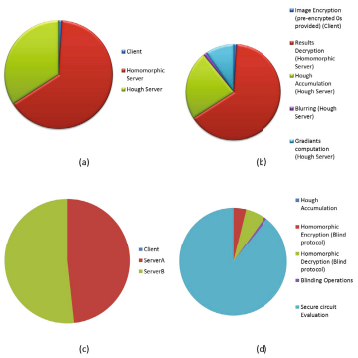

Fig. 2. Computation Break Down - First Approach by Party (a) and by Activity (b); Second Approach by Party (c) and by Activity (d)

party in Fig. 2(c) and by activity in Fig. 2(d). The second approach runs about 50 times slower than the first approach on the same set of parameters. This big difference is mainly caused by the secure circuit evaluation and the blind protocols in the second approach. Note that the time taken by both approaches are mainly dependent on the ρ-θ space size, which is the size of the ACC matrix and independent of how complex the original image is.

For the first approach Fig. 2(a) shows that the homomorphic server carries more load than the Hough server. This is due to the homomorphic cryptography operations as perceived in Fig. 2(b). For the second approach Fig. 2(c) shows that the load is distributed symmetrically between the two servers (Still the server starting the BP protocol does less homomorphic cryptography). The main bottleneck is the use of garbled circuits as perceived in Fig. 2(d).

6 Conclusion

In this paper, we presented two schemes for the secure outsourcing of shape-based feature extraction of images, one is more practical and easier to implement, while the other is provable secure, and experimentally demonstrated their viability and quantified their security and efficiency trade-offs.

References

1. Atallah, M.J., Frikken, K.B.: Securely outsourcing linear algebra computations. In: ASIACCS, pp. 48–59 (2010)
2. Atallah, M.J., Frikken, K.B., Wang, S.: Private outsourcing of matrix multiplication over closed semi-rings. In: SECRYPT, pp. 136–144 (2012)
3. Atallah, M.J., Kerschbaum, F., Du, W.: Secure private sequence comparisons. In: Proceedings of the 2003 ACM Workshop on Privacy in the Electronic Society (WPES 2003), pp. 39–44 (2003)
4. Ballard, D., Brown, C.: Computer Vision. Prentice Hall (1982)
5. Hohenberger, S., Lysyanskaya, A.: How to securely outsource cryptographic computations. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 264–282. Springer, Heidelberg (2005)
6. Hough, P.: Method and means for recognizing complex patterns (1962)
7. Hsu, C.-Y., Lu, C.-S., Pei, S.-C.: Image feature extraction in encrypted domain with privacy-preserving sift. IEEE Transactions on Image Processing 21(11), 4593–4607 (2012)
8. Huang, Y., Evans, D., Katz, J., Malka, L.: Faster secure two-party computation using garbled circuits. In: USENIX Security Symposium (2011)
9. Leavers, V.F.: Shape Detection in Computer Vision Using the Hough Transform. Springer-Verlag New York, Inc., Secaucus (1992)
10. Lindell, Y., Pinkas, B.: A proof of security of yao's protocol for two-party computation. J. Cryptol. 22(2), 161–188 (2009)
11. Osadchy, M., Pinkas, B., Jarrous, A., Moskovich, B.: Scifi - a system for secure face identification. In: 2010 IEEE Symposium on Security and Privacy (SP), pp. 239–254 (2010)
12. 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)
13. Sadeghi, A.-R., Schneider, T., Wehrenberg, I.: Efficient privacy-preserving face recognition. In: Lee, D., Hong, S. (eds.) ICISC 2009. LNCS, vol. 5984, pp. 229–244. Springer, Heidelberg (2010)
14. Williams, P., Sion, R.: Single round access privacy on outsourced storage. In: Proceedings of the 2012 ACM Conference on Computer and Communications Security, CCS 2012, pp. 293–304. ACM, New York (2012)
15. Yao, A.C.-C.: How to generate and exchange secrets. In: SFCS 1986, pp. 162–167. IEEE Computer Society, Washington, DC (1986)