Secure Distortion Computation Among Untrusting Parties Using Homomorphic Encryption

Shantanu Rane, Wei Sun, Anthony Vetro

TR2009-070   December 2009

Abstract

Alice and Bob possess sequences xn and yn respectively and would like to compute d(xn,yn) where d(.,.) is a distortion measure. However, Alice and Bob do not trust each other and do not wish to reveal their data to each other. This paper describes and analyzes a protocol that uses homomorphic encryption for secure calculation of some special distortion functions without revealing xn and yn. The resulting distortion result is also in encrypted form. Two variants of the protocol are presented, one for the Hamming distance between binary vectors, and other for squared errors distortion between integer vectors. An application of the protocol for private biometric authentication is described in which Bob interacts with a remote encrypted fingerprint database (Alice) to achieve access control without revealing his own identity.

IEEE International Conference on Image Processing 2009

SECURE DISTORTION COMPUTATION AMONG UNTRUSTING PARTIES USING HOMOMORPHIC ENCRYPTION

Shantanu D. Rane, Wei Sun and Anthony Vetro

Mitsubishi Electric Research Laboratories
Cambridge, MA 02139
$\{$rane,weisun,avetro$\}$@merl.com

ABSTRACT

Alice and Bob possess sequences x^n and y^n respectively and would like to compute $d(x^n, y^n)$ where $d(\cdot\,,\,\cdot)$ is a distortion measure. However, Alice and Bob do not trust each other and do not wish to reveal their data to each other. This paper describes and analyzes a protocol that uses homomorphic encryption for secure calculation of some special distortion functions without revealing x^n and y^n. The resulting distortion result is also in encrypted form. Two variants of the protocol are presented, one for the Hamming distance between binary vectors, and the other for squared error distortion between integer vectors. An application of the protocol for private biometric authentication is described in which Bob interacts with a remote encrypted fingerprint database (Alice) to achieve access control without revealing his own identity.

Index Terms— Homomorphic Encryption, Paillier Encryption, Secure Multiparty Computation

1. INTRODUCTION

The process of comparing two signals to determine whether they are similar or different is a key operation in numerous electrical engineering systems. For instance, in image processing, one may be interested in determining whether a compressed image has been reconstructed to within a specified mean squared error tolerance. In security applications, comparing a received image with an original image allows forensic experts to determine the location and extent of tampering. In general, given the two signals, and the error tolerance, a computer can perform a suitable measurement and determine whether the difference between them is acceptable. However, when one or both of the signals are encrypted, this task is much more difficult because encryption obfuscates the underlying structure of the signal. For example, even if two binary vectors differ in only a single bit, their encrypted versions can be vastly different. Thus, directly measuring the Hamming distance between the encrypted versions is not useful. Such situations arise in secure multiparty computation [1], wherein the parties - Alice and Bob - that own the two signals do not trust each other, i.e., they would like to compute the distance measure without revealing the signals to each other. In this paper, we present a protocol that utilizes homomorphic encryption to enable secure calculation of Hamming distance and squared error distortion.

Homomorphic properties, which we shall elaborate on in the next section, have been used for secure voting protocols and secure auctions. More pertinent to this paper, Dåmgard *et al.* have used homomorphic encryption and secret sharing for secure comparison of two numbers, i.e., using an untrusted third party, Alice and Bob can verify whose number is larger [2]. Ravikumar *et al.* [3] present a stochastic scalar dot product protocol to approximate the Euclidean distance. In the context of secure data mining of images and biological templates, Du and Atallah [4] provide efficient protocols to approximate the Euclidean distance and the l_1 distance between two signals. Feigenbaum *et al.* [5] describe a protocol to approximate several functions including Hamming and Euclidean distance with sublinear communication overhead.

The protocols presented in this paper enable secure computation of the exact Hamming distance or Euclidean distance, and no other information about the signals is revealed. For a vector of length n, the protocol requires communication data overhead of $O(n)$ from Alice to Bob and a data overhead of $O(1)$ from Bob to Alice. The remainder of this paper is organized as follows: Section 2 reviews the concept of homomorphic encryption and describes Paillier encryption which is later employed in the protocol. Section 3 describes the protocol to determine the Hamming distance. In Section 4, the same protocol is co-opted for squared error calculation. Section 5 shows how the secure Hamming distance calculation can be utilized in remote private biometric authentication.

2. HOMOMORPHIC ENCRYPTION

Let $\mathcal{P}$ be a set of plaintexts associated with a binary operator $\cdot_{\mathcal{P}}$, and $\mathcal{H}$ be a set of ciphertexts associated with a binary operator $\cdot_{\mathcal{H}}$.

Definition 1 *A mapping $\xi : \mathcal{P} \to \mathcal{H}$ is called homomorphic if the following holds for all $a, b \in \mathcal{P}$,*

$$\xi(a \cdot_{\mathcal{P}} b) = \xi(a) \cdot_{\mathcal{H}} \xi(b).$$

Some of the public-key cryptosystems in the literature possess the homomorphic property, namely the RSA [6], El Gamal [7], Goldwasser-Micali [8] and Paillier [9] cryptosystems. The last three of these are semantically secure, in that repeated encryption of same plaintext results in different ciphertexts. The protocol described in this paper works, with small modifications, for any semantically secure homomorphic encryption scheme. For concreteness, we will employ the Paillier cryptosystem whenever encryption/decryption is required in the protocol. The Paillier cryptosystem [9] is reviewed briefly below.

- **Configuration:** Choose two large prime numbers p, q, and let $N = pq$. Denote by $\mathbb{Z}^*_{N^2} \subset \mathbb{Z}_{N^2} = \{0, 1, ..., N^2 - 1\}$ the set of non-negative integers that have multiplicative inverses modulo N^2. Select $g \in \mathbb{Z}^*_{N^2}$ such that $\gcd(L(g^\lambda \mod N^2), N) = 1$, where $\lambda = \text{lcm}(p-1, q-1)$, and $L(x) = \frac{x-1}{N}$. Let (N, g) be the public key, and (p, q) be the private key.

- **Encryption:** Let $m \in \mathbb{Z}_N$ be a plaintext. Then, the ciphertext is given by
$$\xi_r(m) = g^m \cdot r^N \mod N^2 \tag{1}$$
where $r \in \mathbb{Z}_N^*$ is a number chosen at random.
- **Decryption:** Let $c \in \mathbb{Z}_{N^2}$ be a ciphertext. Then, the corresponding plaintext is given by
$$\psi(\xi_r(m)) = \frac{L(c^\lambda \mod N^2)}{L(g^\lambda \mod N^2)} = m \mod N \tag{2}$$

Note that decryption works irrespective of the value of r used during encryption. Since r can be chosen at random for every encryption, the Paillier cryptosystem is probabilistic, and therefore semantically secure. It can now be verified that the following properties hold for the mapping (1) from the plaintext set $(\mathbb{Z}_N, +)$ to the ciphertext set $(\mathbb{Z}^*_{N^2}, \cdot)$,

$$\psi(\xi_{r_1}(m_1)\xi_{r_1}(m_2) \mod N^2) = m_1 + m_2 \mod N \tag{3}$$
$$\psi(\,[\,\xi_r(m_1)\,]^{m_2} \mod N^2) = m_1 m_2 \mod N \tag{4}$$

3. SECURE HAMMING DISTANCE CALCULATION

Suppose that Alice and Bob own two binary sequences $x^n = \{x_1, x_2, ..., x_n\}$ and $y^n = \{y_1, y_2, ..., y_n\}$ respectively and let $n \ll N$. Let $d(\cdot,\cdot)$ denote the binary Hamming distance. Then,

$$\begin{aligned} d(x^n, y^n) &= \sum_{i=1}^n (x_i \oplus y_i) = \sum_{i=1}^n (x_i + y_i - 2x_i y_i) \\ &= A + B + C \end{aligned} \tag{5}$$
$$\text{where } \; A = \sum_{i=1}^n x_i, \;\; B = \sum_{i=1}^n y_i, \;\; C = -\sum_{i=1}^n 2x_i y_i$$

Observe that Alice knows A, Bob knows B, but C contains the cross terms is unknown to both of them. For secure computation, Alice and Bob obtain the encryption key from an authentication server, but none of them possesses the decryption key. Now, the protocol for secure computation of binary Hamming distance is as follows:

1. For each $i \in 1, 2, ..., n$, Alice encrypts x_i into $\xi_{r_i}(x_i)$ according to (1). Here, r_i is chosen randomly from $\mathbb{Z}_N^*$. She transmits the encrypted results to Bob.
2. For each $i \in 1, 2, ..., n$, Bob computes
$$\tilde{y}_i = -2y_i \mod N$$
$$\xi_{r_i}(-2x_i y_i) \equiv [\,\xi_{r_i}(x_i)\,]^{\tilde{y}_i} \mod N^2$$
3. Bob computes
$$\xi_{r_c}(C) \equiv \xi_{r_c}(-\sum_{i=1}^n 2x_i y_i) \equiv \prod_{i=1}^n \xi_{r_i}(-2x_i y_i) \mod N^2$$
where $r_c = \prod_{i=1}^n r_i \mod N \in \mathbb{Z}_N^*$. Note that Bob operates solely in the encrypted domain in this step, so the values of C and r_c are unknown to him.
4. Bob chooses $r_b \in \mathbb{Z}_N^*$ at random and computes
$$\xi_{r_d}(B + C) \equiv \xi_{r_b}(B)\,\xi_{r_c}(C) \mod N^2$$
where $r_d = r_b r_c \mod N \in \mathbb{Z}_N^*$. Bob transmits this result to Alice. The value of r_d is implicit in the encryption result but is unknown to Bob.

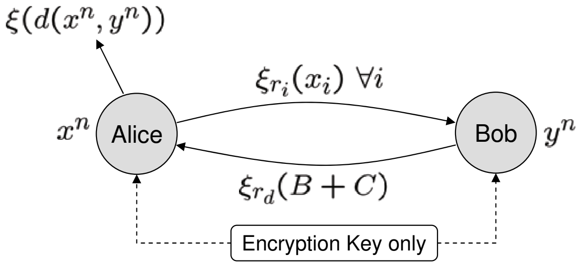

Fig. 1. Protocol for secure distortion calculation using secret sharing and homomorphic encryption.

5. Alice chooses $r_a \in \mathbb{Z}_N^*$ at random and computes
$$\begin{aligned} \xi_r(d(x^n, y^n)) &= \xi_r(A + B + C) \\ &\equiv \xi_{r_a}(A)\,\xi_{r_d}(B + C) \mod N^2 \end{aligned}$$
where $r = r_a r_d \mod N \in \mathbb{Z}_N^*$. Again, the value of r is implicit in the encryption result but unknown to Alice because she does not know r_d. If required by the application, this result is transmitted to Bob.

The transmission steps in the protocol are depicted in Fig. 1. Note that, by design, the protocol does not reveal x^n to Bob or y^n to Alice. In order to know x_i, Bob must decrypt Alice's transmissions in the absence of the decryption key. Since he is computationally bounded, Alice's inputs are computationally secure. Recall that, since Paillier encryption is semantically secure, repeated encryptions of a bit value (0 or 1) will result in different ciphertext every time, dictated by the random choice of r in (1). If Alice wants to cheat successfully, she must decrypt $\xi_{r_d}(B + C) = \xi_{r_d}(\sum_{i=1}^n (y_i - 2x_i y_i))$ in the absence of the decryption key. Since she is computationally bounded, Bob's data is computationally secure.

Table 1 contains the number of computations performed by Alice and Bob during the course of the protocol. In terms of the communication overhead, Alice transmits a maximum of $n \log N^2$ bits in Step 1 and a maximum of $\log N^2$ bits if she needs to communicate the final encrypted Hamming distance to Bob[1]. Thus Alice's maximum communication overhead is $n \log N^2 + \log N^2 < 2(n+1) \log N$ bits, while Bob transmits a maximum of $\log N^2$ bits. Since N is a constant determined by the security requirements of the encryption algorithm, the communication complexity, in terms of the vector length n, is $O(n)$ for Alice and $O(1)$ for Bob.

Party	# encryptions	# exponentiations	# multiplications in encrypted domain
Alice	n	0	1
Bob	1	n	n

Table 1. Number of computations performed by Alice and Bob during the protocol.

4. SECURE SQUARED DISTANCE CALCULATION

Suppose that Alice and Bob own two integer sequences $x^n = \{x_1, x_2, ..., x_n\}$ and $y^n = \{y_1, y_2, ..., y_n\}$ respectively and let

[1] All logarithms in this paper have base 2

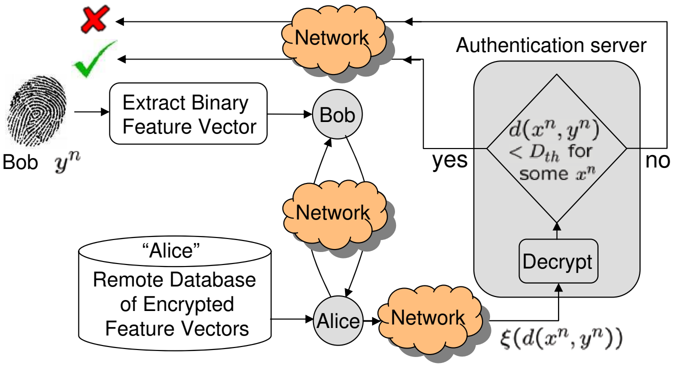
Network
Authentication server
Extract Binary Feature Vector
Bob
Bob y^n
yes
$d(x^n, y^n) < D_{th}$ for some x^n
no
Network
"Alice" Remote Database of Encrypted Feature Vectors
Decrypt
Alice
Network
$\xi(d(x^n, y^n))$

Fig. 2. Access control system based on encrypted binary fingerprint feature vectors.

$n \ll N$. Now, let $d(\cdot, \cdot)$ denote the squared error between x^n and y^n. Then, the squared Euclidean distance is given by

$$d(x^n, y^n) = \sum_{i=1}^{n} (x_i - y_i)^2 = \sum_{i=1}^{n} (x_i^2 + y_i^2 - 2x_i y_i)$$
$$= A + B + C \qquad (6)$$
$$\text{where} \quad A = \sum_{i=1}^{n} x_i^2, \; B = \sum_{i=1}^{n} y_i^2, \; C = -\sum_{i=1}^{n} 2x_i y_i$$

Again, observe that Alice knows A, Bob knows B, but C contains the cross terms and is unknown to both of them. Now, the protocol for secure computation of the squared error between x^n and y^n is exactly the same as that described in Section 3, with the updated definitions of A, B and C from (6).

5. PRIVATE FINGERPRINT AUTHENTICATION

Consider an application of the secure distortion calculation protocol to private biometric authentication on a remote server. As shown in Fig. 2, in which Bob interacts with a remote authentication server and a remote database (Alice) of legitmate fingerprints. At the sensing station, a binary feature vector is extracted from Bob's fingerprint. The remote authentication server grants access only if this feature vector matches at least one of the feature vectors in Alice's database up to a specified Hamming distortion D_{th}. However, Bob would not like to transmit his feature vector directly to the authentication server, for fear of revealing his fingerprint to the server or to an external attacker in the network. Furthermore, Alice would like to protect the feature vectors of legitimate users from Bob and from external attackers, by encrypting them and storing only the encrypted values in the database. We now describe how Bob, Alice and the remote authentication server can perform anonymous access control[2].

The authentication server possesses an encryption/decryption key pair. It assigns Bob and Alice with a common encryption key but no decryption key. Then, Bob and Alice follow the protocol in Section 3, which allows Alice to obtain the *encrypted* Hamming distance between Bob's feature vector y^n and each feature vector x^n in her database. Alice transmits this encrypted result to the remote authentication server over the network. Upon decryption, the server compares the true Hamming distance to a threshold D_{th} and transmits an "Access Granted/Denied" signal to Bob.

We implemented the system of Fig. 2 in software, and tested it with a proprietary database of 1035 fingers with 15 samples per finger. For extracting binary feature vectors from the fingerprint, we adopt the algorithm developed in [10][3]. In brief, this algorithm extracts minutiae points from a fingerprint, generates random rectangles in the minutiae space, counts the number of minutiae points in each rectangle, and binarizes these numbers based on statistics gathered from the training set. A binary feature vector extracted from each fingerprint consists of $n = 150$ bits corresponding to 150 rectangles in the minutiae space. The Hamming distance between two binary feature vectors indicates the difference between the respective underlying fingerprints. The feature extraction algorithm ensures that repeated measurements from the same finger result in a small Hamming distance while measurements from different fingers result in a large Hamming distance. The distribution of the inter-user and intra-user Hamming distance is shown in Fig. 3. The feature vectors of the legitimate users are encrypted bit-wise, using Paillier homomorphic encryption with parameters $p = 1267650600228229401496703217287$, $q = 2535301200456458802993406412663$ and $g = 2$, in which the public encryption key is $(N = pq, g)$ and the private decryption key is (p, q). We used 100-bit prime numbers for p and q but they could be larger if higher computational security is desired. As noted earlier, the encryption key is known to everyone, but the decryption key is known only to the remote authentication server. We verified that, after decryption, the Hamming distance calculated in the encrypted domain using the protocol of Section 3 is exactly equal to the true Hamming distance. Now, for the purpose of authentication, we choose different values for the distortion threshold D_{th} and plot the tradeoff between False Reject Rate (FRR) and False Accept Rate (FAR) as shown in Fig 4. The equal error rate for this tradeoff curve is 0.027, which is achieved for $D_{th} = 30$ bits.

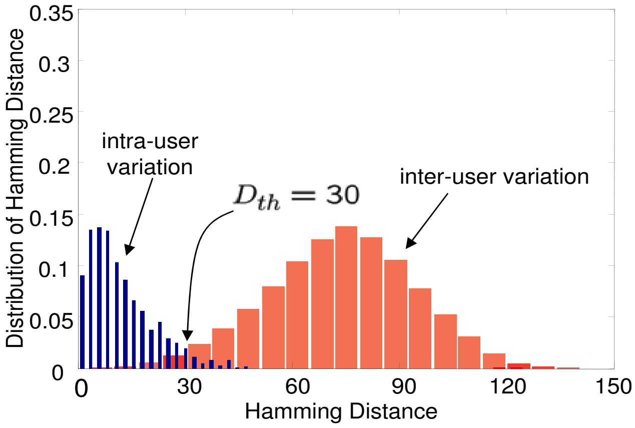

Fig. 3. Given the distributions of intra-user and inter-user Hamming distance, the choice of D_{th} determines the tradeoff between FAR and FRR.

Next, we consider the possibility of cheating. If Bob knows D_{th}, then he may try to gain access using an all-zeroes or all-ones vector

[2] An external computationally bounded attacker is unable to decrypt Bob's and Alice's data but can maliciously alter their transmissions by exploiting the malleability of homomorphic encryption. Security measures required to combat this type of attack are outside the scope of this work.

[3] The proposed system allows private (anonymous) login while the Slepian-Wolf biometric system in [10] does not.

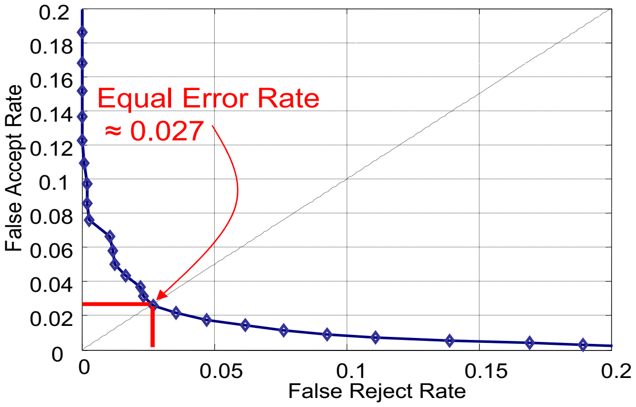

Fig. 4. The equal error rate for the feature extraction scheme in [10] is 0.027. This is achieved by selecting $D_{th} = 30$ as a decision threshold for authentication.

in encrypted form. The histogram of Hamming weights of the unencrypted feature vectors in the training set is plotted in Fig. 5. Feature vectors with Hamming weight less than D_{th} or more than $n - D_{th}$ are vulnerable to all-zeroes and all-ones attack respectively. To prevent such attacks, the server may refuse to enroll feature vectors with too small or too large Hamming weights. Alice might want to cheat by finding out which of the feature vectors stored in her database are closest to Bob's feature vector. However, the protocol only allows her to find the encrypted Hamming distance, so she is unable to cheat. The authentication server can only find out whether Bob is legitimate or not, based on the comparison between the Hamming distance and D_{th}, but has no access to Alice's or Bob's feature vectors.

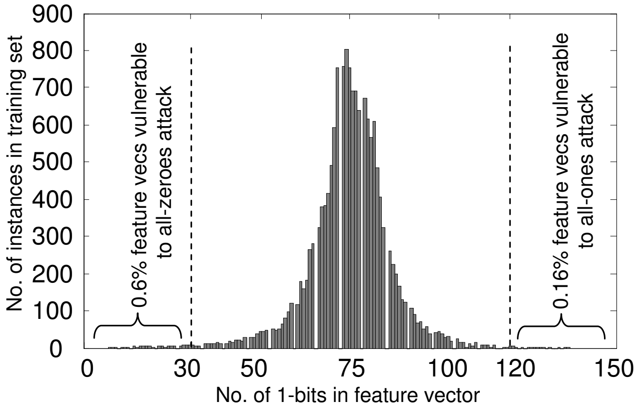

Fig. 5. An all-zeroes and all-ones attack by Bob can be prevented by refusing to enroll feature vectors with Hamming weight less than D_{th} or more than $150 - D_{th}$. For $D_{th} = 30$, this means discarding 0.76% of the feature vectors.

Finally, we consider the communication overhead due to the proposed secure Hamming distance protocol. Alice first transmits a maximum of $n \log N^2 = 60,000$ bits to Bob, where $N = pq$ is a 200-bit number, and $n = 150$. Then, Bob transmits $\log N^2 = 400$ bits to Alice during the protocol. After the protocol concludes, Alice transmits the encrypted Hamming distance to the authentication server, which consumes a maximum of $\log N^2 = 400$ bits. These numbers represent the communication overhead for matching Bob's vector with *each* encrypted vector in Alice's database. The final result of the authentication is communicated to Bob using a single bit.

6. CONCLUSION AND DISCUSSION

A two-party protocol for secure computation of Hamming distance and Euclidean distance was presented in this paper. The protocol utilizes the properties of homomorphic encryption to enable the two parties to compute the distortion in encrypted form without revealing their inputs to each other. The security, computational complexity and communication overhead of the two untrusting parties is analyzed. The secure Hamming distance protocol is applied to a private fingerprint authentication system, where a user can anonymously interact with a remote authentication server without revealing his fingerprint, while the server performs authentication without revealing the fingerprint features of legal users.

7. REFERENCES

[1] A. C-C. Yao, "How to Generate and Exchange Secrets," in *Proceedings of the 27th Annual Symposium on Foundations of Computer Science (SFCS)*, Washington, DC, USA, 1986, pp. 162–167, IEEE Computer Society.

[2] I. Damgård, M. Geisler, and M. Krøigård, "Homomorphic encryption and secure comparison," *International Journal of Applied Cryptography*, vol. 1, no. 1, pp. 22–31, Jan. 2008.

[3] P. Ravikumar, W. Cohen, and S. E. Fienberg, "A secure protocol for computing string distance metrics," in *In Workshop on Privacy, Security and Data Mining, at IEEE International Conference on Data Mining*, 2004, pp. 40–46.

[4] W. Du, M. Atallah, and F. Kerschbaum, "Protocols for secure remote database access with approximate matching," in *Seventh ACM Conference on Computer and Communications Security*, 2000, pp. 523–540.

[5] J. Feigenbaum, Y. Ishai, T. Malkin, K. Nissim, M. J. Strauss, and R. N. Wright, "Secure multiparty computation of approximations," *ACM Transactions on Algorithms*, vol. 2, no. 3, pp. 435–472, 2006.

[6] R. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978.

[7] T. El Gamal, "A Public Key Cryptosystem and A Signature Scheme Based on Discrete Logarithms," *IEEE Transactions on Information Theory*, vol. 4, pp. 469–472, Jul. 1985.

[8] S. Goldwasser and S. Micali, "Probabilistic Encryption and How to Play Mental Poker Keeping Secret All Partial Information," in *Proceedings of the 14th ACM Symposium on the Theory of Computing*, San Francisco, CA, May. 1982, pp. 365–377.

[9] P. Paillier, "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes," in *Advances in Cryptology, EUROCRYPT 99*. 1999, vol. 1592, pp. 233–238, Springer-Verlag, Lecture Notes in Computer Science.

[10] Y. Sutcu, S. Rane, J. Yedidia, S. C. Draper, and A. Vetro, "Feature Extraction for a Slepian-Wolf Biometric System Using LDPC Codes," in *Proceedings of the IEEE International Symposium on Information Theory*, Toronto, Canada, Jul. 2008.