EUSIPCO 2013 1569742051

PRIVACY PRESERVING DISTRIBUTED BEAMFORMING BASED ON HOMOMORPHIC ENCRYPTION

Richard C. Hendriks and Zekeriya Erkin

Delft University of Technology, The Netherlands
{R.C.Hendriks, Z.Erkin}@tudelft.nl

Timo Gerkmann

University of Oldenburg, Germany
timo.gerkmann@uni-oldenburg.de

ABSTRACT

The last few years have shown a growing interest in distributed noise reduction for speech enhancement in wireless acoustic sensor networks. With this new generation of speech processing algorithms, part of the processing is done at processors not necessarily owned by the user. As a consequence, privacy of the users might be at risk. One risk that underlies such distributed speech enhancement systems is the risk of eavesdropping by untrusted parties. To address these privacy issues, we present how a distributed beamformer can be turned into a privacy preserving distributed beamformer using so-called homomorphic encryption techniques and semi-trusted third parties.

Index Terms— Distributed speech enhancement, encryption, privacy.

1. INTRODUCTION

The use of speech processing applications like mobile phones, hearing aids and voice controlled devices is fully integrated in our daily life. As a result, people expect these applications to work anywhere and at anytime. However, many daily life environments are rather noisy in character, degrading the speech quality and resulting in a reduced speech intelligibility. To widen the range where these speech processors can successfully be used, they are often equipped with a noise reduction algorithm, see [1–3] for an overview. A powerful means to increase both quality and intelligibility of noisy speech is to employ the spatial diversity and use a multi-microphone noise reduction algorithm. The performance of these multi-microphone noise reduction algorithms is to a large extent determined by the number of microphones and the array topology. However, due to physical constraints (e.g., size or battery power) mobile devices are heavily constrained with respect to the number and positions of the microphones, which limits the performance of multi-microphone noise reduction.

The recent advances in the area of wireless acoustic sensor networks (WASNs) allow to use a much larger number of microphones at positions that are not constrained by the device itself. However, conventional multi-microphone noise reduction algorithms are characterized by a centralized setup (see e.g. [2, 3]). This requires to transmit all data to a single node for processing. To fully employ the power of WASNs, there has recently been an increased interest to develop distributed noise reduction algorithms, see for example [4–6].

An important difference between conventional centralized and distributed speech enhancement is the fact in the former situation all processing is done on one device, owned by one user, while in the latter situation, the processing is performed by multiple processors potentially owned by different users. This can lead to severe privacy issues and allows potentially untrusted people to eavesdrop conversations.

Distributed processing in WASNs is expected to play an important role in future speech communication systems. However, even though privacy and security are important values in our society, they are potentially at risk with the new generation of distributed speech processing algorithms. An important condition for the success and acceptance of WASNs for speech communication is the development of algorithms that can guarantee the privacy of the users. However, until recently, privacy and security considerations for (distributed) speech processing received hardly any attention.

An interesting development in the context of signal processing and privacy preservation is the development of secure signal processing (SSP) [7], which has recently also been introduced to the area of speech processing [8]. With SSP, data is encrypted using a homomorphic encryption scheme [9]. The decryption key is only available to the person who generates the keys. This means that the data is inaccessible by other parties. However, the homomorphic property of the encryption technique allows to perform linear operations on the encrypted data like scaling and addition without necessarily knowing the content of the ciphertext. With the help of the homomorphism property, it is possible to realize privacy preserving noise reduction in a distributed fashion.

In [10] we introduced the problem of distributed noise reduction for speech enhancement and privacy considerations. We demonstrated how a simplified version of the distributed beamformer from [6] can be turned into a privacy preserving algorithm using homomorphic encryption. The scenario in [10] concerned the case where a user does not want to share to which source he intents to listen, but still wants to use other

entities to estimate this source signal in a distributed manner.

In the current paper we target a different scenario for privacy preservation. An important risk that underlies distributed speech enhancement algorithms is the risk of eavesdropping. This becomes more prominent than in the conventional centralized way of processing, as (intermediate) estimates of the target signal might be available at other entities in the network. This facilitates eavesdropping by untrusted parties. We consider the situation that the processing entities in the WASN are semi-trusted third parties. This means that they fully collaborate with the user in order to perform SSP, however, they will not be provided with the decryption key and will only provide data to other entities in encrypted format. This makes eavesdropping by untrusted parties impossible, as computations done in the encrypted domain and only encrypted data is exchanged.

2. PROBLEM FORMULATION AND NOTATION

As an example of distributed beamforming, we consider the situation where a target source degraded by noise is estimated using the beamformer from [6]. This beamformer estimates the target signal on a frame-by-frame basis in the Fourier domain. Assuming an additive noise model, we can write

$$Y_i(k,m) = S_i(k,m) + N_i(k,m), \tag{1}$$

where $Y_i(k,m)$, $S_i(k,m)$ and $N_i(k,m)$ are the noisy speech, clean speech and noise discrete Fourier transform (DFT) coefficient for microphone i, frequency bin k and time frame m. Further we assume that all sources are mutually uncorrelated.

The target and noise DFT coefficients are assumed to be independent across time and frequency, which allows to omit the time and frequency indices for notational convenience. Further, we use a stacked vector notation, i.e., $\mathbf{Y} = [Y_1, \cdots, Y_M]^T$, with M the number of microphones and where $(\cdot)^T$ denotes transposition of a vector or a matrix. The speech and noise vector $\mathbf{S}$ and $\mathbf{N}$ are defined similarly as $\mathbf{Y}$. Let $\mathbf{d} = [d_1, \cdots, d_M]^T$ be the acoustic transfer function from the speech source to all sensor nodes. In order to concentrate on the privacy preserving context, we assume that $\mathbf{d}$ is known by the user or can be computed given the knowledge of the sensor positions and the position of interest. Although the exact value of d_i also depends on the room acoustics, this will be neglected in this paper for simplicity and we assume a free field situation. In that case, d_i is completely specified by the location of the target source and the microphones. For an overview on sensor network self localization and source localization algorithms see [11, Ch. 13] and [3, Ch. 8], respectively. Altogether we can write

$$\mathbf{Y} = \mathbf{S} + \mathbf{N} = S\mathbf{d} + \mathbf{N}, \tag{2}$$

with S the target DFT coefficient at the target location.

The clean speech estimate $\hat{S}$ is obtained by taking a linear combination of the elements in $\mathbf{Y}$, that is, $\hat{S} = \mathbf{w}^*\mathbf{Y}$, where $(\cdot)^*$ indicates Hermitian transposition and $\mathbf{w}$ denotes the spatial filter coefficients.

2.1. Centralized Beamformer

The filter coefficients for the minimum variance distortionless response (MVDR) beamformer are obtained by minimizing the output power of $\hat{S}$, subject to the constraint of no speech distortion, that is,

$$\min_{\mathbf{w}} \mathbf{w}^*\mathbf{R}_{\mathbf{YY}}\mathbf{w}, \text{ subject to } \mathbf{w}^*\mathbf{d} = 1, \tag{3}$$

with $\mathbf{R}_{\mathbf{YY}} = E[\mathbf{Y}\mathbf{Y}^*]$. Assuming that the noise DFT coefficients $N_i \,\forall\, i$ are zero-mean with power spectral density (PSD) $\sigma^2_{N_i}$, spatially uncorrelated, and uncorrelated with the speech DFT coefficients S, we can write the estimate $\hat{S}$ under the constraint optimization problem expressed by Eq. (3) as [6]

$$\hat{S} = \frac{\frac{1}{M}\sum_{i=1}^{M} d_i^* \sigma_{N_i}^{-2} Y_i}{\frac{1}{M}\sum_{i=1}^{M} |d_i|^2 \sigma_{N_i}^{-2}}. \tag{4}$$

2.2. Randomized Gossip Based Distributed Beamformer

The randomized gossip (RG) algorithm [12] can be used to compute averages in a distributed way. Although the RG algorithm is iterative and convergence depends on the number of sensors, it has the advantage that it puts no constraints on the underlying network topology (apart from being connected). Notice that in the case that the network topology is known and the order of communication can be coordinated, a non-randomized Gossip strategy could be used as well. However, in this paper we only consider the RG strategy. In [6] it was proposed to use RG in order compute the estimator in Eq. (4) in a distributed fashion. The estimator in Eq. (4) is therefore on purpose written as a ratio of two averages. With the RG algorithm these two averages are estimated iteratively after which the estimate $\hat{S}$ is obtained by computing the ratio between these averages.

To do so, each node needs the initial values $\widetilde{Y}_i(0) = d_i^* \sigma_{N_i}^{-2} Y_i$ and $\widetilde{d}_i(0) = |d_i|^2 \sigma_{N_i}^{-2}$. Given these initial values, the RG algorithm can be employed to compute the two averages in the numerator and denominator of Eq. (4). To compute these initial values, each node is in need of d_i, Y_i and $\sigma^2_{N_i}$. The DFT coefficient Y_i is computed at node i and the noise PSD $\sigma^2_{N_i}$ can be estimated at each node i using a noise PSD estimation algorithm, e.g., [13]. Given the initial values $\widetilde{Y}_i(0)$ and $\widetilde{d}_i(0)$, in each iteration two randomly selected neighboring nodes in the network exchange their information and compute

$$\widetilde{Y}_i(t) = \widetilde{Y}_j(t) = \left(\widetilde{Y}_i(t-1) + \widetilde{Y}_j(t-1)\right)/2, \tag{5}$$

$$\widetilde{d}_i(t) = \widetilde{d}_j(t) = \left(\widetilde{d}_i(t-1) + \widetilde{d}_j(t-1)\right)/2. \tag{6}$$

After a specified number of iterations, $\hat{S}$ can then be computed as $\hat{S} = \widetilde{Y}_i(t)/\widetilde{d}_i(t)$.

3. HOMOMORPHIC ENCRYPTION

One approach to protect private data against malicious entities is to keep it secret by means of encryption. However, if the data is to be used for further processing, traditional encryption schemes require the data to be decrypted first. Clearly, this approach does not provide an effective protection mechanism, particularly against the party who processes the secret data.

Fortunately, a number of cryptosystems preserves *some* structure after encryption that can be exploited. These cryptosystems, known as homomorphic cryptosystems, allow to perform simple mathematical computations like addition and multiplication on encrypted data. In this work we consider the homomorphic Paillier encryption system [14]. The Paillier encryption scheme is from a family of so-called asymmetric encryption that uses two different keys: a public key (PK) that can be used for encryption and known by everyone, and a secret key (SK) that can be used for decryption and known by the owner only.

To illustrate how homomorphism can be used for providing a protection mechanism, we first introduce the Paillier encryption scheme in detail. Let $\mathcal{E}_{pk}$ and $\mathcal{D}_{sk}$ denote the encryption and decryption operations with the public and the secret key, respectively.

The Paillier cryptosystem is *additive*, meaning that multiplication of two Pailler encrypted numbers yields the encryption of the sum of the numbers, that is,

$$\mathcal{D}_{sk}(\mathcal{E}_{pk}(m_1) \cdot \mathcal{E}_{pk}(m_2)) = m_1 + m_2. \tag{7}$$

As a consequence of the additive homomorphism, a number can also be scaled by exponentiating its encryption by a constant, that is,

$$\mathcal{D}_{sk}(\mathcal{E}_{pk}(m)^c) = m \cdot c. \tag{8}$$

The Paillier encryption function for encrypting the message $m \in \mathbb{Z}_n$ is given by,

$$c = \mathcal{E}_{pk}(m) = g^m \cdot r^n \bmod n^2, \tag{9}$$

where $n = p \cdot q$ is a product of two large prime numbers p and q, g is a generator of the group with order n (i.e., $g^n = 1 \bmod n^2$) and can always be chosen as $g = n + 1$, and r is randomly chosen from a specific set of numbers that are co-prime[1] with n. The private key is the tuple (p, q), from which the public key follows as the tuple (n, g). The decryption function for encryption c is defined as

$$\mathcal{D}_{sk}(c) = \frac{L(c^\lambda \bmod n^2)}{L(g^\lambda \bmod n^2)} \bmod n, \tag{10}$$

where $L(u) = (u-1)/n$ and $\lambda = \mathrm{lcm}(p-1, q-1)$, with $\mathrm{lcm}(a, b)$ the least common multiple of two integers a and b, i.e., the smallest positive integer that is divisible by both a and b. Notice that as we work with modular arithmetic, a^{-1} is the modular inverse, defined as, $a \cdot a^{-1} = 1 \bmod n$.

[1]Two numbers are co-prime if there is no positive divisor that can divide both, except the number one.

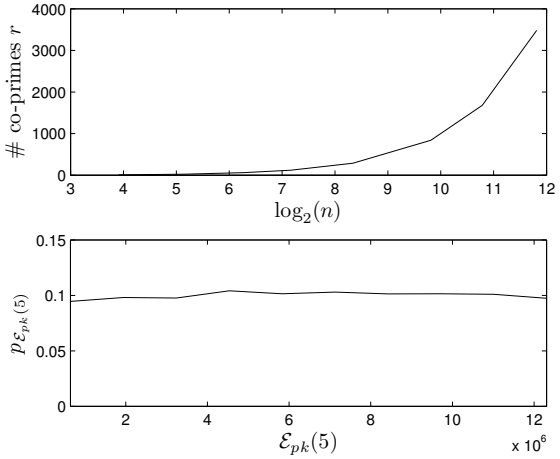

Fig. 1. (a) Number of co-primes as a function of the size of n in bits. (b) Distribution $p_{\mathcal{E}_{pk}(5)}$ of the encryption $\mathcal{E}_{pk}(5)$.

As an example, we consider the encryption of the number $m = 5$. In this example, the secret key is chosen as $(p, q) = (3, 5)$ leading to a public key $(n, g) = (15, 16)$. The numbers p and q in this example are relatively small for demonstration purpose. In practice they are of a much higher order for security purposes. Even with this small number $n = 15$, there are already eight numbers r that are co-prime with n, e.g., $r = 7$ or $r = 8$. Encryption of the number 5 can then lead to eight different possible outcomes, e.g.,

$$\mathcal{E}_{pk}(5) = 16^5 7^{15} \bmod 15^2 = 193$$

or

$$\mathcal{E}_{pk}(5) = 16^5 8^{15} \bmod 15^2 = 32.$$

To decrypt the encryption 193, we need to compute λ, which is in this example given by $\mathrm{lcm}(2, 4) = 4$. Then,

$$\mathcal{D}_{sk}(193) = \frac{L(193^4 \bmod 15^2)}{L(16^4 \bmod 15^2)} \bmod 15 = \frac{5}{4} \bmod 15 = 5,$$

with $\frac{1}{4}$ the modular inverse of 4, that is, $\frac{1}{4} = 4 \bmod 15$.

In practice, the number n is very large (more than 1024 bits), to make the system secure as these cryptosytems rely on one of the mathematical hard problems like nth residues and factorization that requires using very large numbers. As the numbers are very large, it is practically infeasible to determine the plain data m without access to the secret key. This is due to the difficulty for the adversaries with bounded computational resources to solve the nth residue problem that will help them to recover the secret key. Another key point is that the adversaries cannot distinguish the encryptions due to the *probabilistic* nature of the encryption function. Notice that even if the same message is encrypted, the resulting ciphertext will be different due to the random number introduced in the encryption function. This can be explained in the following way: with increasing n, the amount of numbers that are

co-prime with n increases exponentially, as shown in Fig. 1(a) as a function of the size of n in bits. This makes it, for sufficiently large n, infeasible to guess or learn the encrypted number m from the encryptions $\mathcal{E}_{pk}(m)$. Moreover, as r is chosen uniformly from the set of numbers that are co-prime with n and all calculations are modulo n^2, the outcomes $\mathcal{E}_{pk}(m)$ tend to be uniformly distributed. This is shown in Fig. 1(b), where the histogram $p_{\mathcal{E}_{pk}(5)}$ is shown based on repeatedly encryption (10^4 times) of the number $m = 5$. The fact that the encryptions $\mathcal{E}_{pk}(5)$ are uniformly distributed implies no information is leaked about the encrypted number $m = 5$.

An important aspect of this encryption scheme is the fact that it operates in a modular domain and that the message m needs to be representable in this domain. This means that m should be an integer in the range $m \in [0, n-1]$. Negative numbers can be represented using a constant shift, or, recoding the negative number, say $-m$, using its modular inverse.

4. PRIVACY-PRESERVING DISTRIBUTED SPEECH ENHANCEMENT

To overcome eavesdropping by untrusted third parties, we apply SSP based on homomorphic encryption [9]. We consider a scenario where the processing entities in the network are considered to be semi-trusted. This means that these entities will not be provided with the decryption key, but, they are trusted in the sense that they will collaborate with the user and perform all computations with microphone data in the encrypted domain using a public key and only exchange encrypted processed microphone signals. As a consequence, eavesdroppers will have no access to the microphone signals and will not be able to combine them into a beamformer, eventhough the acoustic transfer function might be available. Moreover, none of the network entities can listen to the estimated signal, apart from the owner of the decryption key.

4.1. Privacy Preserving Algorithm

The core of the RG based distributed beamformer depends on the iterative operations expressed by Eqs. (5) and (6) in order to compute the two averages that form Eq. (4), i.e., $Y_{avg} = \frac{1}{M}\sum_{i=1}^{M} d_i^* \sigma_{N_i}^{-2} Y_i$ and $d_{avg} = \frac{1}{M}\sum_{i=1}^{M} |d_i|^2 \sigma_{N_i}^{-2}$, in distributed fashion. Such distributed computations imply that estimates of $\hat{S}$ are available in the WASN and can easily be accessed by other untrusted entities. To provide the required security, we assume the network consists of entities that are semi-trusted. This means that these entities collaborate with the user and process data according to certain rules. In order to prevent eavesdropping of conversations, the estimate $Y_{avg} = \frac{1}{M}\sum_{i=1}^{M} d_i^* \sigma_{N_i}^{-2} Y_i$ is not computed using standard distributed processing, but using distributed processing in the encrypted domain using homomorphic encryption.

First, a user generates a secret key and a public key. Together with their d_i, all semi-trusted entities are provided with the public key. As the d_i's are unencrypted, d_{avg} is computed using standard RG without any encryption, see Sec. 2.2. The public key can then be used to encrypt Y_i and compute Y_{avg} in a distributed and secure way using homomorphic encryption. As Y_i is complex and homomorphic encryption works on real integer valued numbers, all variables must be split into real and imaginary parts and quantized to integers. We will use the subscripts $\Re$ and $\Im$ to denote the real and imaginary part of a variable, and let $[\,\cdot\,]$ denote the scaling and quantization operation. Encryption of the scaled and quantized quantities at entity i then leads to $\mathcal{E}_{pk}([Y_{i\Re}])$ and $\mathcal{E}_{pk}([Y_{i\Im}])$.

Next, the encrypted version of the initialization $\widetilde{Y}_i(0) = d_i^* \sigma_{N_i}^{-2} Y_i$ can be computed by each entity. Splitting $\widetilde{Y}_i(0)$ into its real and imaginary parts we obtain

$$\widetilde{Y}_i(0)_{\Re} = d_{i\,\Re}^* \sigma_{N_i}^{-2} Y_{i\Re} - d_{i\,\Im}^* \sigma_{N_i}^{-2} Y_{i\Im}, \quad (11)$$

$$\widetilde{Y}_i(0)_{\Im} = d_{i\,\Im}^* \sigma_{N_i}^{-2} Y_{i\Re} + d_{i\,\Re}^* \sigma_{N_i}^{-2} Y_{i\Im}. \quad (12)$$

Notice that $d_{i\,\Im}^*$ indicates to first take the conjugate of d_i followed by taking the imaginary part of that.

Given the encryptions $\mathcal{E}_{pk}([Y_{i\Re}])$ and $\mathcal{E}_{pk}([Y_{i\Im}])$, Eqs. (11) and (12) can be transformed into

$$\mathcal{E}_{pk}(\widetilde{Y}_i(0)_{\Re}) = \mathcal{E}_{pk}([Y_{i\Re}])^{\left[\sigma_{N_i}^{-2} d_{i\,\Re}^*\right]} \mathcal{E}_{pk}([Y_{i\Im}])^{-\left[\sigma_{N_i}^{-2} d_{i\,\Im}^*\right]},$$

$$\mathcal{E}_{pk}(\widetilde{Y}_i(0)_{\Im}) = \mathcal{E}_{pk}([Y_{i\Re}])^{\left[\sigma_{N_i}^{-2} d_{i\,\Im}^*\right]} \mathcal{E}_{pk}([Y_{i\Im}])^{\left[\sigma_{N_i}^{-2} d_{i\,\Re}^*\right]}.$$

Given these initial values, the iterations expressed by Eq. (5) can be performed in the encrypted domain. Given two communicating nodes i and j, these iterations are translated in the encrypted domain as

$$\mathcal{E}_{pk}(\widetilde{Y}_i(t)_{\Re}) = \mathcal{E}_{pk}(\widetilde{Y}_i(t-1)_{\Re})^{2^{-1}} \mathcal{E}_{pk}(\widetilde{Y}_j(t-1)_{\Re})^{2^{-1}},$$

$$\mathcal{E}_{pk}(\widetilde{Y}_i(t)_{\Im}) = \mathcal{E}_{pk}(\widetilde{Y}_i(t-1)_{\Im})^{2^{-1}} \mathcal{E}_{pk}(\widetilde{Y}_j(t-1)_{\Im})^{2^{-1}},$$

where power 2^{-1} denotes the modular inverse of 2. As all numbers in the encrypted domain should remain integers, special care needs to be taken when exponentiating encrypted numbers by a modular inverse as 2^{-1}, as this implies a devision by 2 potentially leading to non-integers. To overcome this, the scaling and quantization operation $[\,\cdot\,]$ should be a multiple of two, related to the number of applied iterations.

Finally, after sufficient iterations, the secret key owner can compute the estimate $\hat{S} = (\widetilde{Y}_i(t)_{\Re} + j\widetilde{Y}_i(t)_{\Im})/\widetilde{d}_i(t)$ by decryption and construct the time-domain waveform by computing an inverse DFT followed by an overlap-add.

5. CONCLUDING REMARKS

In this paper we addressed the fact that with distributed beamforming, privacy might be at risk. To overcome eavesdropping of conversations, we presented how a distributed beamformer can be turned into a privacy preserving distributed beamformer based on homomorphic encryption.

Homomorphic encryption is an elegant technique that allows to perform basic mathematical operations on encrypted data. However, existing homomorphic encryption schemes are based on mathematically difficult problems and thus, require to use large numbers. This introduces two main drawbacks: 1) data expansion, and 2) computational complexity due to the large numbers. The former results in an increased bit-rate, while the latter is directly related to available computational power, which is an important consideration in mobile devices.

To reduce computational complexity, secret sharing can be used [15]. With secret sharing, data is split among a number of parties such that construction is only possible when a subset of these parties combines their shares. While additions are significantly efficient, multiplications require more computational resources. Even though the size of the data becomes larger (linear in the number of shares), the overall data expansion can be smaller than using homomorphic encryption.

A second approach to consider is to use garbled circuits, which realize any function with private inputs in a secure manner [16]. The main idea is to represent the function with a Boolean circuit and evaluate each logic gate securely. In literature, there are already efficient solutions for garbled circuits that realize secure computation of non-linear functions [17]. When combined with homomorphic encryption, this can significantly reduce the computational complexity as it uses symmetric encryption algorithms, which are significantly faster than homomorphic cryptosystems. However, this approach heavily depends on pre-computed values that are assumed to be computed before the actual protocol begins. Therefore, the right choice of the approach is directly related to the application setting.

6. REFERENCES

[1] R. C. Hendriks, T. Gerkmann, and J. Jensen, *DFT-Domain Based Single-Microphone Noise Reduction for Speech Enhancement: A Survey of the State of the Art*, Morgan & Claypool, 2013.

[2] J. Benesty, M. M. Sondhi, and Y. Huang (Eds), *Springer handbook of speech processing*, Springer, 2008.

[3] M. Brandstein and D. Ward (Eds.), *Microphone arrays: signal processing techniques and applications*, Springer, 2001.

[4] A. Bertrand and M. Moonen, "Distributed adaptive node-specific signal estimation in fully connected sensor networks – part I: Sequential node updating," *IEEE Trans. Signal Processing*, vol. 58, no. 10, pp. 5277 – 5291, Oct. 2010.

[5] S. Markovich-Golan, S. Gannot, and I. Cohen, "Distributed multiple constraints generalized sidelobe canceler for fully connected wireless acoustic sensor networks," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 21, no. 2, pp. 343 –356, feb. 2013.

[6] Y. Zeng and R. C. Hendriks, "Distributed delay and sum beamformer for speech enhancement in wireless sensor networks via randomized gossip," in *IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2012.

[7] R. L. Lagendijk, Z. Erkin, and M. Barni, "Encrypted signal processing for privacy protection," *IEEE Signal Process. Mag.*, pp. 82–105, Jan. 2013.

[8] M. A. Pathak, B. Raj, S. D. Rane, and P. Smaragdis, "Privacy-preserving speech processing: cryptographic and string-matching frameworks show promise," *IEEE Signal Process. Mag.*, vol. 30, no. 2, pp. 62–74, 2013.

[9] C. Fontaine and F. Galand, "A survey of homomorphic encryption for nonspecialists," *EURASIP Journal on Information Security*, vol. 2007, 2007.

[10] R. C. Hendriks, Z. Erkin, and T. Gerkmann, "Privacy-preserving distributed speech enhancement for wireless sensor networks by processing in the encrypted domain," in *IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2013.

[11] S. Haykin and K. J. R. Liu, *Handbook on array processing and sensor networks*, Wiley Online Library, 2009.

[12] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. on Information Theory*, vol. 52, no. 6, pp. 2508 – 2530, June 2006.

[13] T. Gerkmann and R. C. Hendriks, "Unbiased MMSE-based noise power estimation with low complexity and low tracking delay," *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 4, pp. 1383–1393, 2012.

[14] P. Paillier, "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes," in *Advances in Cryptology — EUROCRYPT '99*, J. Stern, Ed. May 1999, vol. 1592 of *LNCS*, pp. 223–238, Springer.

[15] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[16] A. C. Yao, "Protocols for secure computations," in *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, 1982, pp. 160–164.

[17] A. Sadeghi and T. Schneider, "Generalized universal circuits for secure evaluation of private functions with application to data classification," in *11th International Conference on Information Security and Cryptology (ICISC 2008)*, Seoul, Korea, 3-5 December 2008, pp. 336–353, Springer-Verlag.