Hindawi Publishing Corporation
EURASIP Journal on Information Security
Volume 2007, Article ID 78943, 20 pages
doi:10.1155/2007/78943

Review Article

Protection and Retrieval of Encrypted Multimedia Content: When Cryptography Meets Signal Processing

Zekeriya Erkin,[1] Alessandro Piva,[2] Stefan Katzenbeisser,[3] R. L. Lagendijk,[1] Jamshid Shokrollahi,[4] Gregory Neven,[5] and Mauro Barni[6]

[1] *Electrical Engineering, Mathematics, and Computer Science Faculty, Delft University of Technology, 2628 CD, Delft, The Netherlands*
[2] *Department of Electronics and Telecommunication, University of Florence, 50139 Florence, Italy*
[3] *Information and System Security Group, Philips Research Europe, 5656 AE, Eindhoven, The Netherlands*
[4] *Department of Electrical Engineering and Information Sciences, Ruhr-University Bochum, 44780 Bochum, Germany*
[5] *Department of Electrical Engineering, Katholieke Universiteit Leuven, 3001 Leuven, Belgium*
[6] *Department of Information Engineering, University of Siena, 53100 Siena, Italy*

Correspondence should be addressed to Zekeriya Erkin, z.erkin@tudelft.nl

Received 3 October 2007; Revised 19 December 2007; Accepted 30 December 2007

Recommended by Fernando Pérez-González

The processing and encryption of multimedia content are generally considered sequential and independent operations. In certain multimedia content processing scenarios, it is, however, desirable to carry out processing directly on encrypted signals. The field of secure signal processing poses significant challenges for both signal processing and cryptography research; only few ready-to-go fully integrated solutions are available. This study first concisely summarizes cryptographic primitives used in existing solutions to processing of encrypted signals, and discusses implications of the security requirements on these solutions. The study then continues to describe two domains in which secure signal processing has been taken up as a challenge, namely, analysis and retrieval of multimedia content, as well as multimedia content protection. In each domain, state-of-the-art algorithms are described. Finally, the study discusses the challenges and open issues in the field of secure signal processing.

1. INTRODUCTION

In the past few years,the processing of encrypted signals has emerged as a new and challenging research field. The combination of cryptographic techniques and signal processing is not new. So far, encryption was always considered as an add-on after signal manipulations had taken place (see Figure 1). For instance, when encrypting compressed multimedia signals such as audio, images, and video, first the multimedia signals were compressed using state-of-the-art compression techniques, and next encryption of the compressed bit stream using a symmetric cryptosystem took place. Consequently, the bit stream must be decrypted before the multimedia signal can be decompressed. An example of this approach is JPSEC, the extension of the JPEG2000 image compression standard. This standard adds selective encryption to JPEG2000 bit streams in order to provide secure scalable streaming and secure transcoding [1].

In several application scenarios, however, it is desirable to carry out signal processing operations directly on encrypted signals. Such an approach is called *secure signal processing*, *encrypted signal processing*, or *signal processing in the encrypted domain*. For instance, given an encrypted image, can we calculate the mean value of the encrypted image pixels? On the one hand, the relevance of carrying out such signal manipulations, that is, the algorithm, directly on encrypted signals is entirely dependent on the security requirements of the application scenario under consideration. On the other hand, the particular implementation of the signal processing algorithm will be determined strongly by the possibilities and impossibilities of the cryptosystem employed. Finally, it is very likely that new requirements for cryptosystems will emerge from secure signal processing operations and applications. Hence, secure signal processing poses a joint challenge for both the signal processing and the cryptographic community.

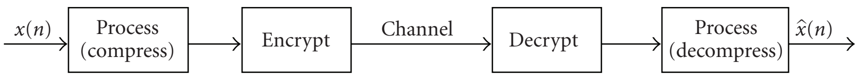

FIGURE 1: Separate processing and encryption of signals.

The security requirements of signal processing in encrypted domains depends strongly on the considered application. In this survey paper, we take an application-oriented view on secure signal processing and give an overview of published applications in which the secure processing of signal amplitudes plays an important role. In each application, we show how signal processing algorithms and cryptosystems are brought together. It is not the purpose of the paper to describe either the signal processing algorithms or the cryptosystems in great detail, but rather focus on possibilities, impossibilities, and open issues in combining the two. The paper includes many references to literature that contains more elaborate signal processing algorithms and cryptosystem solutions for the given application scenario. It is also crucial to state that the scenarios in this survey can be implemented more efficiently by using trusted third entities. However, it is not always easy to find trusted entities with high computational power, and even if one is found, it is not certain that it can be applicable in these scenarios. Therefore, the trusted entities either do not exist or have little role in discussed scenarios in this paper.

In this paper, we will survey applications that directly manipulate encrypted signals. When scanning the literature on secure signal processing, it becomes immediately clear that there are currently two categories under which the secure signal processing applications and research can be roughly classified, namely, content retrieval and content protection. Although the security objectives of these application categories differ quite strongly, similar signal processing considerations and cryptographic approaches show up. The common cryptographic primitives are addressed in Section 2. This section also discusses the need for clearly identifying the security requirements of the signal processing operations in a given scenario. As we will see, many of the approaches for secure signal processing are based on homomorphic encryption, zero-knowledge proof protocols, commitment schemes, and multiparty computation. We will also show that there is ample room for alternative approaches to secure signal processing towards the end of Section 2. Section 3 surveys secure signal processing approaches that can be classified as "content retrieval," among them secure clustering and recommendation problems. Section 4 discusses problems of content protection, such as secure watermark embedding and detection. Finally, Section 5 concludes this survey paper on secure protection and retrieval of encrypted multimedia content.

2. ENCRYPTION MEETS SIGNAL PROCESSING

2.1. Introduction

The capability to manipulate signals in their encrypted form is largely thanks to two assumptions on the encryption strategies used in all applications discussed. In the first place, encryption is carried out independently on individual signal samples. As a consequence, individual signal samples can be identified in the encrypted version of the signal, allowing for processing of encrypted signals on a sample-by-sample basis. If we represent a one-dimensional (e.g., audio) signal $\mathbf{X}$ that consists of M samples as

$$\mathbf{X} = \left[x_1, x_2, x_3, \ldots, x_{M-1}, x_M\right]^T, \tag{1}$$

where x_i is the amplitude of the ith signal sample, then the encrypted version of $\mathbf{X}$ using key k is given as

$$E_k(\mathbf{X}) = \left[E_k(x_1), E_k(x_2), E_k(x_3), \ldots, E_k(x_{M-1}), E_k(x_M)\right]^T. \tag{2}$$

Here the superscript "T" refers to vector transposition. Note that no explicit measures are taken to hide the temporal or spatial structure of the signal, however, the use of sophisticated encryption schemes that are *semantically secure* (as the one in [2]) achieves this property automatically.

Secondly, only *public* key cryptosystems are used that have particular *homomorphic* properties. The homomorphic property that these public key cryptographic system provide will be concisely discussed in Section 2.2.1. In simple terms, the homomorphic property allows for carrying out additions or multiplications on signal amplitudes in the encrypted domain. Public key systems are based on the intractability of some computationally complex problems, such as

(i) the discrete logarithm in finite field with a large (prime) number of elements (e.g., ElGamal cryptosystem [3]);
(ii) factoring large composite numbers (e.g., RSA cryptosystem [4]);
(iii) deciding if a number is an nth power in $\mathbb{Z}_N$ for large enough composite N (e.g., Paillier cryptosystem [2]).

It is important to realize that public key cryptographic systems operate on very large algebraic structures. This means that signal amplitudes x_i that were originally represented in 8-to-16 bits will require at least 512 or 1024 bits per signal sample in their encrypted form $E_k(x_i)$. This data expansion is usually not emphasized in literature but this may be an important hurdle for practical applicability of secure signal processing solutions. In some cases, however, several signal samples can be packed into one encrypted value in order to reduce the size of the whole encrypted signal by a linear factor [5].

A characteristic of signal amplitudes x_i is that they are usually within a limited range of values, due to the 8-to-16 bits amplitude representation format of sampled signals. If a deterministic encryption scheme would be used, each signal amplitude would always give rise to the same encrypted value, making it easy for an adversary to infer information

Table 1: Some (probabilistic) encryption systems and their homomorphisms.

Encryption system	$f_1(\cdot,\cdot)$	$f_2(\cdot,\cdot)$
Multiplicatively Homomorphic El-Gamal [3]	Multiplication	Multiplication
Additively Homomorphic El-Gamal [13]	Addition	Multiplication
Goldwasser-Micali [14]	XOR	Multiplication
Benaloh [15]	Addition	Multiplication
Naccache-Stern [16]	Addition	Multiplication
Okamoto-Uchiyama [17]	Addition	Multiplication
Paillier [2]	Addition	Multiplication
Damgård-Jurik [18]	Addition	Multiplication

about the signal. Consequently, probabilistic encryption has to be used, where each encryption uses a randomization or blinding factor such that even if two signal samples x_i and x_j have the same amplitude, their encrypted values $E_{\text{pk}}[x_i]$ and $E_{\text{pk}}[x_j]$ will be different. Here, pk refers to the public key used upon encrypting the signal amplitudes. Public key cryptosystems are constructed such that the decryption uses only the private key sk, and that decryption does not need the value of the randomization factor used in the encryption phase. All encryption schemes that achieve the desired strong notion of *semantic security* are necessarily probabilistic.

Cryptosystems operate on (positive) integer values on finite algebraic structures. Although sampled signal amplitudes are normally represented in 8-to-16 bits (integer) values when they are stored, played, or displayed, intermediate signal processing operations often involve noninteger signal amplitudes. Work-arounds for noninteger signal amplitudes may involve scaling signal amplitudes with constant factors (say factors of 10 to 1000), but the unavoidable successive operations of rounding (quantization) and normalization by division pose significant challenges for being carried out on encrypted signal amplitudes.

In Section 2.2, we first discuss four important cryptographic primitives that are used in many secure signal processing applications, namely, homomorphic encryption, zero-knowledge proof protocols, commitment schemes, and secure multiparty computation. In Section 2.3, we then consider the importance of scrutinizing the security requirements of the signal processing application. It is meaningless to speak about secure signal processing in a particular application if the security requirements are not specified. The security requirements as such will also determine the possibility or impossibility of applying the cryptographic primitives. As we will illustrate by examples—and also in more detail in the following sections—some application scenarios simply cannot be made secure because of the inherent information leakage by the signal processing operation because of the limitations of the cryptographic primitives to be used, or because of constraints on the number of interactions between parties involved. Finally, in Section 2.4, we briefly discuss the combination of signal encryption and compression using an approach quite different from the ones discussed in Sections 3 and 4, namely, by exploiting the concept of coding with side information. We discuss this approach here to emphasize that although many of the currently existing application scenarios are built on the four cryptographic primitives discussed in Section 2.2, there is ample room for entirely different approaches to secure signal processing.

2.2. *Cryptographic primitives*

2.2.1. *Homomorphic cryptosystems*

Many signal processing operations are linear in nature. Linearity implies that multiplying and adding signal amplitudes are important operations. At the heart of many signal processing operations, such as linear filters and correlation evaluations, is the calculation of the inner product between two signals $\mathbf{X}$ and $\mathbf{Y}$. If both signals (or segments of the signals) contain M samples, then the inner product is defined as

$$\langle \mathbf{X}, \mathbf{Y} \rangle = \mathbf{X}^T\mathbf{Y} = \left[x_1, x_2, \dots, x_M\right] \cdot \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{bmatrix} = \sum_{i=1}^{M} x_i y_i. \qquad (3)$$

This operation can be carried out directly on an encrypted signal $\mathbf{X}$ and plain text signal $\mathbf{Y}$ if the encryption system used has the additive homomorphic property, as we will discuss next.

Formally, a "public key" encryption system $E_{\text{pk}}(\cdot)$ and its decryption $D_{\text{sk}}(\cdot)$ are homomorphic if those two functions are maps between the message group with an operation $f_1(\cdot)$ and the encrypted group with an operation $f_2(\cdot)$, such that if x and y are taken from the message space of the encryption scheme, we have

$$f_1(x, y) = D_{\text{sk}}\left(f_2\left(E_{\text{pk}}(x), E_{\text{pk}}(y)\right)\right). \qquad (4)$$

For secure signal processing, multiplicative and additive homomorphisms are important. Table 1 gives an overview of encryption systems with additive or multiplicative homomorphism. Note that those homomorphic operations are applied to a modular domain (i.e., either in a finite field or in a ring $\mathbb{Z}_N$)—thus, both addition and multiplication are taken modulo some fixed value. For signal processing applications, which usually require integer addition and multiplication, it is thus essential to choose the message space of the encryption scheme large enough so that overflows due to modular arithmetic are avoided when operations on encrypted data are performed.

Another important consideration is the representation of the individual signal samples. As encryption schemes usually operate in finite modular domains (and all messages to be encrypted must be represented in this domain), a mapping is required which quantizes real-valued signal amplitudes and translates the signal samples of $\mathbf{X}$ into a vector of modular numbers. In addition to the requirement that the computations must not overflow, special care must be taken to represent negative samples in a way which is compatible with the homomorphic operation offered by the cryptosystem. For the latter problem, depending on the algebraic structure of the cipher, one may either encode the negative value $-x$ by the modular inverse x^{-1} in the underlying algebra of the message space or by avoiding negative numbers entirely by using a constant additive shift.

In the context of the above inner product example, we require an additively homomorphic scheme (see Table 1). Hence, f_1 is the addition, and f_2 is a multiplication:

$$x + y = D_{\text{sk}}(E_{\text{pk}}(x) \cdot E_{\text{pk}}(y)), \tag{5}$$

or, equivalently,

$$E_{\text{pk}}(x + y) = E_{\text{pk}}(x) \cdot E_{\text{pk}}(y). \tag{6}$$

Note that the latter equation also implies that

$$E_{\text{pk}}(c \cdot x) = (E_{\text{pk}}(x))^c \tag{7}$$

for every integer constant c. Thus, every additively homomorphic cryptosystem also allows to multiply an encrypted value with a constant available or known as clear text.

The Paillier cryptosystem [2] provides the required homomorphism if both addition and multiplication are considered as modular. The encryption of a message m under a Paillier cryptosystem is defined as

$$E_{\text{pk}}(m) = g^m r^N \bmod N^2, \tag{8}$$

where $N = pq$, p and q are large prime number, $g \in \mathbb{Z}^*_{N^2}$ is a generator whose order is a multiple of N, and $r \in \mathbb{Z}^*_N$ is a random number (blinding factor). We then easily see that

$$\begin{aligned} E_{\text{pk}}(x)E_{\text{pk}}(y) &= (g^x r_x^N)(g^y r_y^N) \bmod N^2 \\ &= g^{x+y}(r_x r_y)^N \bmod N^2 \\ &= E_{\text{pk}}(x + y). \end{aligned} \tag{9}$$

Applying the additive homomorphic property of the Paillier encryption system, we can evaluate (3) under the assumption that $\mathbf{X}$ is an encrypted signal and $\mathbf{Y}$ is a plain text signal:

$$E_{\text{pk}}\langle \mathbf{X}, \mathbf{Y}\rangle = E_{\text{pk}}\left(\sum_{i=1}^{M} x_i y_i\right) = \prod_{i=1}^{M} E_{\text{pk}}(x_i y_i) = \prod_{i=1}^{M} E_{\text{pk}}(x_i)^{y_i}. \tag{10}$$

Here, we implicitly assume that x_i, y_i are represented as integers in the message space of the Paillier cryptosystem, that is, $x_i, y_i \in \mathbb{Z}_N$. However, (10) essentially shows that it is possible to compute an inner product directly in case one of the two vectors is encrypted. One takes the encrypted samples $E_{\text{pk}}(x_i)$, raises them to the power of y_i, and multiplies all obtained values. Obviously, the resulting number itself is also in encrypted form. To carry out further useful signal processing operations on the encrypted result, for instance, to compare it to a threshold, another cryptographic primitive is needed, namely, zero knowledge proof protocols, which is discussed in the next section.

In this paper, we focus mainly on public-key encryption schemes, as almost all homomorphic encryption schemes belong to this family. The notable exception is the one-time pad (and derived stream ciphers), where messages taken from a finite group are blinded by a sequence of uniformly random group elements. Despite its computationally efficient encryption and decryption processes, the application of a one-time pad usually raises serious problems with regard to key distribution and management. Nevertheless, it may be used to temporarily blind intermediate values in larger communication protocols. Finally, it should be noted that some recent work in cryptography (like searchable encryption [6] and order-preserving encryption [7]) may also yield alternative ways for the encryption of signal samples. However, these approaches have not yet been studied in the context of media encryption.

To conclude this section, we observe that directly computing the inner product of *two* encrypted signals is not possible since this would require a cryptographic system that has both multiplicative and additive (i.e., algebraic) homomorphism. Recent proposals in that direction like [8, 9] were later proven to be insecure [10, 11]. Therefore, no *provably secure* cryptographic system with these properties is known to date. The construction of an algebraic privacy homomorphism remains an open problem. Readers can refer to [12] for more details on homomorphic cryptosystems.

2.2.2. Zero-knowledge proof protocols

Zero-knowledge protocols are used to prove a certain statement or condition to a verifier, without revealing any "knowledge" to the verifier except the fact that the assertion is valid [19]. As a simple example, consider the case where the prover Peggy claims to have a way of factorizing large numbers. The verifier Victor will send her a large number and Peggy will send back the factors. Successful factorization of several large integers will decrease Victor's doubt in the truth of Peggy's claim. At the same time Victor will learn "no knowledge of the actual factorization method."

Although simple, the example shows an important property of zero-knowledge protocol proofs, namely, that they are interactive in nature. The interaction should be such that with increasing number of "rounds," the probability of an adversary to successfully prove an invalid claim decreases significantly. On the other hand, noninteractive protocols (based on the random oracle model) also do exist. A formal definition of interactive and noninteractive proof systems, such as zero-knowledge protocols, falls outside the scope of this paper, but can be found, for instance, in [19].

As an example for a commonly used zero-knowledge proof, consider the proof of knowing the discrete logarithm

x of an element y to the base g in a finite field [20]. Having knowledge of discrete logarithm x is of interest in some applications since if

$$y = g^x \bmod p, \tag{11}$$

then given p (a large prime number), g, and y (the calculation of the logarithm x) are computationally infeasible. If Peggy (the prover) claims she knows the answer (i.e., the value of x), she can convince Victor (the verifier) of this knowledge without revealing the value of x by the following zero-knowledge protocol. Peggy picks a random number $r \in \mathbb{Z}_p$ and computes $t = g^r \bmod p$. She then sends t to Victor. He picks a random challenge $c \in \mathbb{Z}_p$ and sends this to Peggy. She computes $s = r - cx \bmod p$ and sends this to Victor. He accepts Peggy's knowledge of x if $g^s y^c = t$, since if Peggy indeed used the correct logarithm x in calculating the value of s, we have

$$g^s y^c \bmod p = g^{r-cx}(g^x)^c \bmod p = g^r = t \bmod p. \tag{12}$$

In literature, many different zero-knowledge proofs exist. We mention a number of them that are frequently used in secure signal processing:

(i) proof that an encrypted number is nonnegative [21];
(ii) proof that shows that an encrypted number lies in a certain interval [22];
(iii) proof that the prover knows the plaintext x corresponds to the encryption $E(x)$ [23];
(iv) proofs that committed values (see Section 2.2.3) satisfy certain algebraic relations [24].

In zero-knowledge protocols, it is sometimes necessary for the prover to commit to a particular integer or bit value. Commitment schemes are discussed in the next section.

2.2.3. Commitment schemes

An integer or bit commitment scheme is a method that allows Alice to commit to a value while keeping it hidden from Bob, and while also preserving Alice's ability to reveal the committed value later to Bob. A useful way to visualize a commitment scheme is to think of Alice as putting the value in a locked box, and giving the box to Bob. The value in the box is hidden from Bob, who cannot open the lock (without the help of Alice), but since Bob has the box, the value inside cannot be changed by Alice; hence, Alice is "committed" to this value. At a later stage, Alice can "open" the box and reveal its content to Bob.

Commitment schemes can be built in a variety of ways. As an example, we review a well-known commitment scheme due to Pedersen [25]. We fix two large primes p and q such that $q \mid (p-1)$ and a generator g of the subgroup of order q of $\mathbb{Z}_p^*$. Furthermore, we set $h = g^a \bmod p$ for some random secret a. The values p, q, g, and h are the public parameters of the commitment scheme. To commit to a value m, Alice chooses a random value $r \in \mathbb{Z}_q$ and computes the commitment $c = g^m h^r \bmod p$. To open the commitment, Alice sends m and r to Bob, who verifies that the commitment c received previously indeed satisfies $c = g^m h^r \bmod p$. The scheme is hiding due to the random blinding factor r; furthermore, it is binding unless Alice is able to compute discrete logarithms.

For use in signal processing applications, commitment schemes that are additively homomorphic are of specific importance. As with homomorphic public key encryption schemes, knowledge of two commitments allows one to compute—without opening—a commitment of the sum of the two committed values. For example, the above-mentioned Pedersen commitment satisfies this property: given two commitments $c_1 = g^{m_1} h^{r_1} \bmod p$ and $c_2 = g^{m_2} h^{r_2} \bmod p$ of the numbers m_1 and m_2, a commitment $c = g^{m_1+m_2} h^{r_1+r_2} \bmod p$ of $m_1 + m_2$ can be computed by multiplying the commitments: $c = c_1 c_2 \bmod p$. Note that the commitment c can be opened by providing the values $m_1 + m_2$ and $r_1 + r_2$. Again, the homomorphic property only supports additions. However, there are situations where it is not possible to prove the relation by mere additive homomorphism as in proving that a committed value is the square of the value of another commitment. In such circumstances, zero-knowledge proofs can be used. In this case, the party which possesses the opening information of the commitments computes a commitment of the desired result, hands it to the other party, and proves in zero-knowledge that the commitment was actually computed in the correct manner. Among others, such zero-knowledge proofs exist for all polynomial relations between committed values [24].

2.2.4. Secure multiparty computation

The goal of secure multiparty computation is to evaluate a public function $f(x^{(1)}, x^{(2)}, \dots, x^{(m)})$ based on the secret inputs $x^{(i)}$, $i = 1, 2, \dots, m$ of m users, such that the users learn nothing except their own input and the final result. A simple example, called Yao's Millionaire's Problem, is the comparison of two (secret) numbers in order to determine if $x^{(1)} > x^{(2)}$. In this case, the parties involved will only learn if their number is the largest, but nothing more than that.

There is a large body of literature on secure multiparty computation; for example, it is known [26] that any (computable) function can be evaluated securely in the multiparty setting by using a general circuit-based construction. However, the general constructions usually require a large number of interactive rounds and a huge communication complexity. For practical applications in the field of distributed voting, private bidding and auctions, and private information retrieval, dedicated lightweight multiparty protocols have been developed. An example relevant to signal processing application is the multiparty computation known as Bitrep which finds the encryption of each bit in the binary representation of a number whose encryption under an additive homomorphic cryptosystem is given [27]. We refer the reader to [28] for an extensive summary of secure multiparty computations and to [29] for a brief introduction.

2.3. Importance of security requirements

Although the cryptographic primitives that we discussed in the previous section are useful for building secure signal processing solutions, it is important to realize that in each application the security requirements have to be made explicit right from the start. Without wishing to turn to formal definition, we choose to motivate the importance of what to expect from secure signal processing with three simple yet illustrative two-party computation examples.

The first simple example is the encryption of a (say audio) signal **X** that contains M samples. Due to the sample-by-sample encryption strategy as shown in (2), the encrypted signal $E_{\text{pk}}(\mathbf{X})$ will also contain M encrypted values. Hence, the *size M* of the plain text signal cannot be hidden by the approaches followed in secure signal processing surveyed in this paper.

In the second example, we consider the linear filtering of the signal **X**. In an (FIR) linear filter, the relation between the input signal amplitudes **X** and output signal amplitudes **Y** is entirely determined by the impulse response $(h_0, h_1, \ldots, h_r)$ through the following convolution equation:

$$y_i = h_0 x_i + h_1 x_{i-1} + \cdots + h_r x_{i-r} = \sum_{k=0}^{r} h_k x_{i-k}. \qquad (13)$$

Let us assume that we wish to compute this convolution in a secure way. The first party, Alice, has the signal **X** and the second party, Bob, has the impulse response $(h_0, h_1, \ldots, h_r)$. Alice wishes to carry out the convolution (13) using Bob's linear filter. However, both Bob and Alice wish to keep secret their data, that is, the impulse response and the input signal, respectively. Three different setups can now be envisioned.

(1) Alice encrypts the signal **X** under an additive homomorphic cryptosystem and sends the encrypted signal to Bob. Bob then evaluates the convolution (13) on the encrypted signal as follows:

$$\begin{aligned} E_{\text{pk}_A}(y_i) &= E_{\text{pk}_A}\left(\sum_{k=0}^{r} h_k x_{i-k}\right) \\ &= \prod_{k=0}^{r} E_{\text{pk}_A}(h_k x_{i-k}) \\ &= \prod_{k=0}^{r} E_{\text{pk}_A}(x_{i-k})^{h_k}. \end{aligned} \qquad (14)$$

Notice that the additive homomorphic property is used in the above equation and that, indeed, individually encrypted signal samples should be available to Bob. Also notice that the above evaluation is only possible if both **X** and $(h_0, h_1, \ldots, h_r)$ are integer-valued, which is actually quite unlikely in practice. After computing (14), Bob sends the result back to Alice who decrypts the signal using her private key to obtain the result **Y**. In this setup, Bob does not learn the output signal **Y**.

(2) Bob encrypts his impulse response $(h_0, h_1, \ldots, h_r)$ under a homomorphic cryptosystem and sends the result to Alice. Alice then evaluates the convolution (13) using the encrypted impulse response as follows:

$$\begin{aligned} E_{\text{pk}_B}(y_i) &= E_{\text{pk}_B}\left(\sum_{k=0}^{r} h_k x_{i-k}\right) \\ &= \prod_{k=0}^{r} E_{\text{pk}_B}(h_k x_{i-k}) \\ &= \prod_{k=0}^{r} E_{\text{pk}_B}(h_k)^{x_{i-k}}. \end{aligned} \qquad (15)$$

Alice then sends the result to Bob, who decrypts to obtain the output signal **Y**. In this solution, Bob learns the output signal **Y**.

(3) Alice and Bob engage in a formal multiparty protocol, where the function $f(x_1, x_2, \ldots, x_M, h_0, h_1, \ldots, h_r)$ is the convolution equation, Alice holds the signal values x_i and Bob the impulse response h_i as secret inputs. Both parties will learn the resulting output signal **Y**.

Unfortunately, none of the above three solutions really provides a solution to the secure computation of a convolution due to inherent algorithm properties. For instance, in the first setup, Alice could send Bob a signal that consists of all-zero values and a single "one" value (a so-called "impulse signal"). After decrypting the result $E_{\text{pk}_A}(y_i)$ that she obtains from Bob, it is easy to see that **Y** is equal to $(h_0, h_1, \ldots, h_r)$, hence Bob's impulse response is subsequently known to Alice. Similar attacks can be formulated for the other two cases. In fact, even for an arbitrary input, both parties can learn the other's input by a well-known signal processing procedure known as "deconvolution." In conclusion, although in some cases there may be a need for the secure evaluation of convolutions, the inherent properties of the algorithm make secure computing in a two-party scenario meaningless. (Nevertheless, the protocols have value if used as building blocks in a large application where the output signal **Y** is not revealed to the attacker.)

The third and final example is to threshold a signal's (weighted) mean value in a secure way. The (secure) mean value computation is equivalent to the (secure) computation of the inner product of (3), with **X** the input signal and **Y** the weights that define how the mean value is calculated. In the most simple case, we have $y_i = 1$ for all i, but other definitions are quite common. Let us assume that Alice wishes Bob to determine if the signal's mean value is "critical," for instance, above a certain threshold value T_c, without revealing **X** to Bob. Bob, on the other hand, does not want to reveal his expert knowledge, namely, the weights **Y** and the threshold T_c. Two possible solutions to this secure decision problem are the following.

(i) Use secure multiparty computation, where the function $f(x_1, x_2, \ldots, x_M, y_1, y_2, \ldots, y_M, T_c)$ is a combination of the inner product and threshold comparison. Both parties will only learn if the mean value is critical or not.

(ii) Alice sends Bob the signal **X** under additively homomorphic encryption. Bob securely evaluates the inner product using (10). After encrypting T_c using Alice's public key, Bob computes the (encrypted version of the) difference between the computed mean and

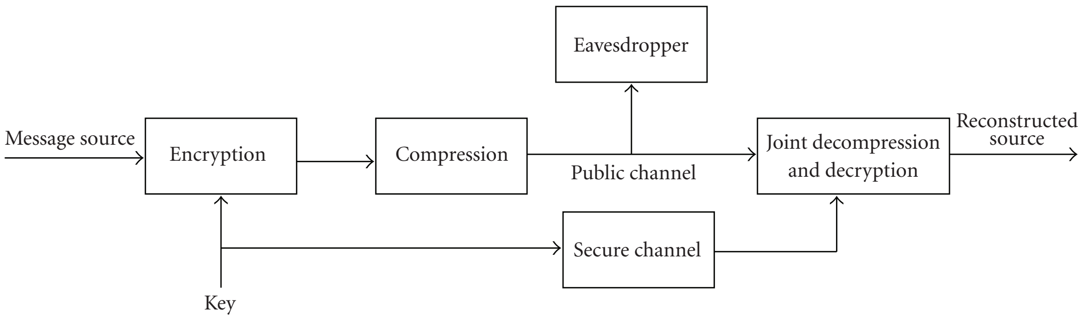

Figure 2: Compression of an encrypted signal from [30].

threshold T_c. Bob sends the result to Alice, who decrypts the result using her secret key and checks if the value is larger or smaller than zero.

Although the operations performed are similar to the second example, in this example the processing is secure since Bob learns little about Alice's signal and Alice will learn little about the Bob's expert knowledge. In fact, in the first implementation, the entire signal processing operation is ultimately condensed into a single bit of information; the second implementation leaks more information, namely, the distance between the correlation value from the threshold. In both cases, the result represents a high information abstraction level, which is insufficient for launching successful signal processing-based attacks. In contrast, in the example based on (13), the signal processing operation led to an enormous amount of information—the entire output signal—to be available to either parties, making signal processing-based attacks quite easy.

As we will see in Sections 3 and 4, many of the two-party secure signal processing problems eventually include an information condensation step, such as (in the most extreme case) a binary decision. We postulate that for two-party linear signal processing operations in which the amount of plain text information after processing is in the same order of magnitude as before processing, no secure solutions exist purely based on the cryptographic primitives discussed in the previous section, due to inherent properties of the signal processing problems and the related application scenario. For that reason, entirely other approaches to secure signal processing are also of interest. Although few results can be found in literature on approaches not using homomorphic encryption, zero-knowledge proofs, and multiparty computation protocols, the approach discussed in the next section may well show a possible direction for future developments.

2.4. Compression of encrypted signals

When transmitting signals that contain redundancy over an insecure and bandwidth-constrained channel, it is customary to first compress and then encrypt the signal. Using the principles of coding with side information, it is, however, also possible to interchange the order of (lossless) compression and encryption, that is, to compress *encrypted* signals [30]. The concept of swapping the order of compression and encryption is illustrated in Figure 2. A signal from the message source is first encrypted and then compressed. The compressor does *not* have access to the secret key used in the encryption. At the decoder, decompression and decryption are performed jointly. From classical information theory, it would seem that only minimal gain could be obtained as the encrypted signal has maximal entropy, that is, no redundancy is left after encryption. However, the decoder can use the cryptographic key to *decode and decrypt* the compressed and encrypted bit stream. This brings opportunities for efficient compression of encrypted signals based on principle of coding with side information. In [30], it was shown that neither compression performance nor security need to be negatively impacted under some reasonable conditions.

In source coding with side information, the signal $\mathbf{X}$ is coded under the assumption that the decoder—but not the encoder—has statistically dependent information $\mathbf{Y}$, called the side information, available. In conventional coding scenarios, the encoder would code the difference signal $\mathbf{X}-\mathbf{Y}$ in some efficient way, but in source coding with side information, this is impossible since we assume that $\mathbf{Y}$ is only known at the decoder. In the Slepian-Wolf coding theory [31], the crucial observation is that the side information $\mathbf{Y}$ is regarded as a degraded version of $\mathbf{X}$. The degradations are modeled as "noise" on the "virtual channel" between $\mathbf{X}$ and $\mathbf{Y}$. The signal $\mathbf{X}$ can then be recovered from $\mathbf{Y}$ by the decoder if sufficient error-correcting information is transmitted over the channel. The required bit rate and amount of entropy are related as $R \geq H(\mathbf{X} \mid \mathbf{Y})$. This shows that, at least theoretically, there is no loss in compression efficiency since the lower bound $H(\mathbf{X} \mid \mathbf{Y})$ is identical to the scenario in which $\mathbf{Y}$ is available at the encoder. Extension of the Slepian-Wolf theory exists for lossy source coding [32]. In all practical cases of interests, the information bits that are transmitted over the channel are parity bits or syndromes of channel coding methods such as Hamming, Turbo or LDPC codes.

In the scheme depicted in Figure 2, we have a similar scenario as in the above source coding with side information case. If we consider the encrypted signal $E_k(\mathbf{X})$ at the input of the encoder, then we see that the decoder has the key k available, representing the "statistically dependent side information." Hence, according to the Slepian-Wolf viewpoint, the encrypted signal $E_k(\mathbf{X})$ can be compressed to a rate that is the same as if the key k would be available during the source encoding process, that is, $R \geq H(E_k(\mathbf{X}) \mid k) = H(\mathbf{X})$. This clearly says that the (lossless) coding of the encrypted signal $E_k(\mathbf{X})$ should be possible with the same efficiency as the (lossless) coding of $\mathbf{X}$. Hence, using the side information key k, the decoder can recover first $E_k(\mathbf{X})$ from the compressed channel bit stream and subsequently decode $E_k(\mathbf{X})$ into $\mathbf{X}$.

A simple implementation of the above concept for a binary signal $\mathbf{X}$ uses a pseudorandomly generated key. The key k is in this case a binary signal $\mathbf{K}$ of the same dimension M as the signal $\mathbf{X}$. The encrypted signal is computed as follows:

$$\begin{aligned} E_k(\mathbf{X}) &= \mathbf{X} \oplus \mathbf{K}, \\ E_k(x_i) &= x_i \oplus k_i, \quad i = 1, 2, \ldots, M. \end{aligned} \tag{16}$$

The encrypted signal $E_k(\mathbf{X})$ is now input to a channel coding strategy, for instance, a Hamming coding. The strength of the Hamming code is dependent on the dependency between $E_k(\mathbf{X})$ and the side information $\mathbf{K}$ at the decoder. This strength obviously depends solely on the properties of the original signal $\mathbf{X}$. This does, however, require the message source to inform the source encoder about the entropy $H(\mathbf{X})$, which represents a small leak of information. The encoder calculates parity check bits over binary vectors of some length L created by concatenating L bits of the encrypted signal $E_k(\mathbf{X})$, and sends *only these parity check bits* to the receiver.

The decoder recovers the encrypted signal by first appending to $\mathbf{K}$ the parity check bits, and then error correcting the resulting bit pattern. The success of this error correction step depends on the strength of the Hamming code, but as mentioned, this strength has been chosen sufficiently with regards to the "errors" in $\mathbf{K}$ on the decoding side. Notice that in this particular setup the "errors" represent the bits of the original signal $\mathbf{X}$. If the error correction step is successful, the decoder obtains $E_k(\mathbf{X})$, from which the decryption can straightforwardly take place:

$$\begin{aligned} \mathbf{X} &= E_k(\mathbf{X}) \oplus \mathbf{K}, \\ x_i &= E_k(x_i) \oplus k_i, \quad i = 1, 2, \ldots, M. \end{aligned} \tag{17}$$

The above example is too simple for any practical scenario for a number of reasons. In the first place, it uses only binary data, for instance, bit planes. More efficient coding can be obtained if the dependencies between bit planes are considered. This effectively requires an extension of the bit plane coding and encryption approach to coding and encryption of symbol values. Secondly, the decoder lacks a model of the dependencies in $\mathbf{X}$. Soft decoders for Turbo or LDPC codes can exploit such message source models, yielding improved performance. Finally, the coding strategy is lossless. For most continuous or multilevel message sources, such as audio, images, and video, lossy compression is desirable.

3. ANALYSIS AND RETRIEVAL OF CONTENT

In the today's society, huge quantities of personal data are gathered from people and stored in databases for various purposes ranging from medical researches to online personalized applications. Sometimes, providers of these services may want to combine their data for research purposes. A classical example is the one where two medical institutions wish to perform joint research on the union of their patients data. Privacy issues are important in this scenario because the institutions need to preserve their private data during their cooperation. Lindell and Pinkas [33] and Agrawal and Srikant [34] proposed the notion of privacy preserving data mining, meaning the possibility to perform data analysis from distributed database, under some privacy constraints. Privacy preserving data mining [35–38] deals with mutual untrusted parties that on the one hand wish to cooperate to achieve a common goal but, on the other hand, are not willing to disclose their knowledge to each other.

There are several solutions that cope with exact matching of data in a secure way. However, it is more common in signal processing to perform inexact matching, that is, learning the distance between two signal values, rather than exact matching. Consider two signal values x_1 and x_2. Computing the distance between them or checking if the distance is within a threshold is important:

$$|x_1 - x_2| < \epsilon. \tag{18}$$

This comparison or *fuzzy matching* can be used in a variety of ways in signal processing. One example is quantizing data which is of crucial importance for multimedia compression schemes. However, considering that these signal values are encrypted—thus the ordering between them is totally destroyed, there is not any efficient way known to *fuzzy* compare two values.

In the following sections, we give a summary of techniques that focus on extracting some information from protected datasets. Selected studies mostly use homomorphic encryption, zero-knowledge proofs, and, sometimes, multiparty computations. As we will see, most solutions still require substantial improvements in communication and computation efficiency in order to make them applicable in practice. Therefore, the last section addresses a different approach that uses other means of preserving privacy to show that further research on combining signal processing and cryptography may result in new approaches rather than using encryption schemes and protocols.

3.1. *Clustering*

Clustering is a well-studied combinatorial problem in data mining [39]. It deals with finding a structure in a collection of unlabeled data. One of the basic algorithms of clustering is the K-means algorithm that partitions a dataset into K clusters with a minimum error. We review the K-means algorithm and its necessary computations such as distance computation and finding the cluster centroid, and show that cryptographic protocols can be used to provide user's privacy in clustering for certain scenarios.

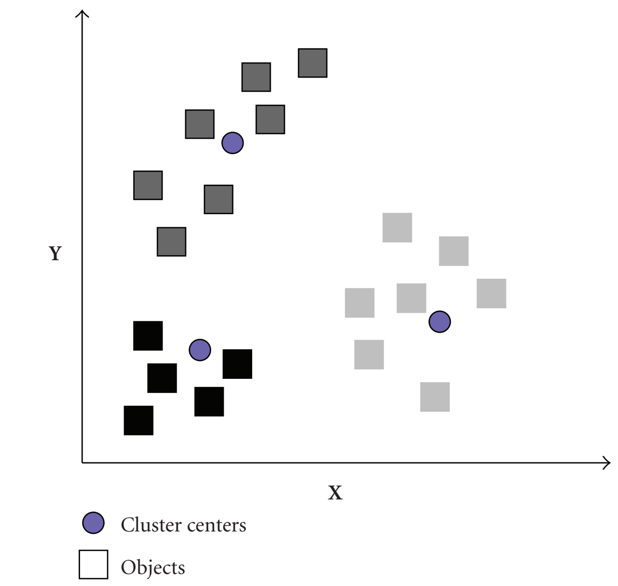

Figure 3: Clustered dataset. Each object is a point in the 2-dimensional space. K-means clustering algorithm assigns each object to the cluster with the smallest distance.

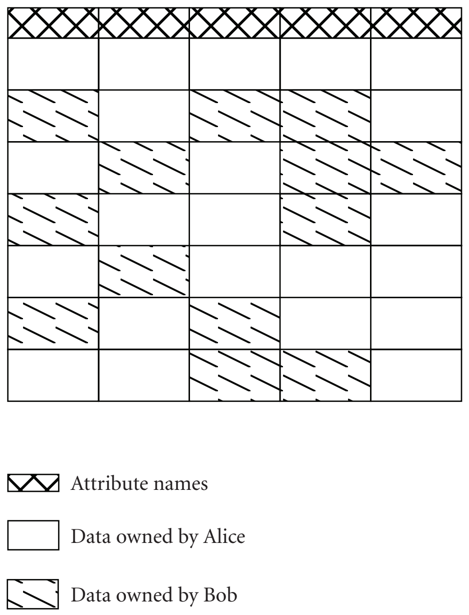

Figure 4: Shared dataset on which K-means algorithm is run.

```
(1) select $K$ random objects representing the $K$
    initial centroid of the clusters.
(2) assign each object to the cluster with the
    nearest centroid.
(3) recalculate the centroids for each cluster.
(4) repeat step 2 and 3 until centroids do not
    change or a certain threshold achieved.
```

Algorithm 1: The K-means clustering algorithm

3.1.1. K-means clustering algorithm

The K-means clustering algorithm partitions a dataset D of "objects" such as signal values or features thereof into K disjoint subsets, called clusters. Each cluster is represented by its center which is the centroid of all objects in that subset.

As shown in Algorithm 1, the K-means algorithm is an iterative procedure that refines the cluster centroids until a predefined condition is reached. The algorithm first chooses K random points as the cluster centroids in the dataset D and assigns the objects to the closest cluster centroid. Then, the cluster centroid is recomputed with recently assigned objects. When the iterative procedure reaches the termination condition, each data object is assigned to the closest cluster (Figure 3). Thus to carry out the K-means algorithm, the following quantities needs to be computed:

(i) the cluster centroid, or the mean of the data objects in that cluster,
(ii) the distance between an object and the cluster centroid,
(iii) the termination condition which is a distance measurement compared to a threshold.

In the following section, we describe a secure protocol that carries out secure K-means algorithm on protected data objects.

3.1.2. Secure K-means clustering algorithm

Consider the scenario in which Alice and Bob want to apply the K-means algorithm on their joint datasets as shown in Figure 4, but at the same time they want to keep their own dataset private. Jagannathan and Wright proposed a solution for this scenario in [40].

In the proposed method, both Alice and Bob get the final output but the values computed in the intermediate steps are unknown to the both parties. Therefore, the intermediate values such as cluster centroids are uniformly shared between Alice and Bob in such a way that for a value x, Alice gets a random share a and Bob gets another random share b, where $(a+b) \bmod N = x$ and N is the size of the field in which all operations take place. Alice and Bob keep their private shares of the dataset secret.

The secure K-means clustering algorithm is separated into subprotocols where Alice and Bob computes the followings (Algorithm 2).

(i) *Distance measurement and finding the closest cluster:* the distance between each object and cluster centroid is computed by running a secure scalar product protocol by Goethals et al. [41]. The closest cluster centroid is determined by running Yao's circuit evaluation protocol [42] with the shared data of Alice and Bob.
(ii) *New cluster centroid:* the new cluster centroid requires to determine an average computation over shared values of Alice and Bob. This function of the form $(a+b)/(m+n)$ can be computed by applying Yao's protocol where Alice knows a and m and Bob knows b and n.

```
Randomly select $K$ objects from the dataset $D$ as initial
cluster centroids
Randomly share the cluster centroid between Alice
and Bob
*repeat*
  *for all* object $d_k$ in dataset $D$ *do*
    Run the secure closest cluster protocol
    Assign to $d_k$ to the closest cluster
  *end for*
  Alice and Bob computes the random shares for the new
  centroids of the clusters.
*until* cluster centroids are close to each other with an error
of $\epsilon$.
```

Algorithm 2: Privacy preserving K-means clustering algorithm.

(iii) *Termination condition:* the termination condition of the algorithm is computed by running the Yao's circuit evaluation protocol [42].

The squared distance between an object $\mathbf{X}_i = (x_{i,1}, \dots, x_{i,M})$ and a cluster centroid μ_j is given by the following equation:

$$(\text{dist}(\mathbf{X}_i, \mu_j))^2 = (x_{i,1} - \mu_{j,1})^2 + (x_{i,2} - \mu_{j,2})^2 + \cdots + (x_{i,M} - \mu_{j,M})^2. \quad (19)$$

Considering that the clusters centroids are shared between Alice and Bob, (19) can be written as

$$(\text{dist}(\mathbf{X}_i, \mu_j))^2 = (x_{i,1} - (\mu^A_{j,1} + \mu^B_{j,1}))^2 + \cdots + (x_{i,M} - (\mu^A_{j,M} + \mu^B_{j,M}))^2, \quad (20)$$

where μ^A_j is Alice's share and μ^B_j is Bob's share such that the jth-cluster centroid is $\mu_j = \mu^A_j + \mu^B_j$. Then, (20) can be written as

$$(\text{dist}(\mathbf{X}_i, \mu_j))^2 = \sum_{k=1}^{M} x^2_{i,k} + \sum_{k=1}^{M} (\mu^A_{j,k})^2 + \sum_{k=1}^{M} (\mu^B_{j,k})^2 + 2\sum_{k=1}^{M} \mu^A_{j,k}\mu^B_{j,k} - 2\sum_{k=1}^{M} \mu^A_{j,k} x_{i,k} - 2\sum_{k=1}^{M} x_{i,k}\mu^B_{j,k}. \quad (21)$$

Equation (21) can be computed by Alice and Bob jointly. As the first term of the equation is shared between them, Alice computes the sum of components of her share while Bob computes the rest of the components. The second term and third term can be computed by Alice and Bob individually, and the rest of the terms are computed by running a secure scalar product protocol between Alice and Bob, much similar to the evaluation of (3) via the secure form of (10). Alice first encrypts her data $E_{\text{pk}_A}(\mu^A_j) = (E_{\text{pk}_A}(\mu^A_{j,1}), \dots, E_{\text{pk}_A}(\mu^A_{j,M}))$ and sends it to Bob who computes the scalar product of this data with his own by using the additive homomorphic property of the encryption scheme as follows:

$$E_{\text{pk}_A}(\mu^A_j)^{\mu^B_j} = \left(E_{\text{pk}_A}(\mu^A_{j,1})^{\mu^B_{j,1}}, \dots, E_{\text{pk}_A}(\mu^A_{j,M})^{\mu^B_{j,M}}\right). \quad (22)$$

Then, multiplying the encrypted components gives the encrypted scalar product of Alice's and Bob's data

$$E_{\text{pk}_A}\left(\sum_{k=1}^{M} \mu^A_{j,k}\mu^B_{j,k}\right) = \prod_{k=1}^{M} E_{\text{pk}_A}(\mu^A_{j,k})^{\mu^B_{j,k}}. \quad (23)$$

The computed distances between the objects and the cluster centroids can later be the input to the Yao's circuit evaluation protocol [42] in which the closest cluster centroid is determined. We refer readers to [41, 42] for further details on this part.

Once the distances and the closest clusters to the objects are determined, each object is labeled with the nearest cluster index. At the end of each iteration, it is necessary to compute the new cluster centroids. Alice computes the sum of the corresponding coordinates of all object s_j and the number of objects n_j within each of the K clusters for j, $1 \le j \le M$. As shown in Figure 4, Alice has only some of the attributes of the objects, thus she treats these missing values as zero. Bob also applies the same procedure and determines the sum of coordinates t_j and the number of objects m_j in the clusters. Given s_j, t_j, n_j, and m_j, the jth component of the ith cluster is

$$\mu_{i,j} = \frac{s_j + t_j}{n_j + m_j}. \quad (24)$$

Since there are only four values, this equation can be computed efficiently by using Yao's circuit evaluation protocol [42] with Alice's shares s_j and n_j and Bob's shares t_j and m_j.

In the last step of the K-means algorithm, the iteration is terminated if there is no further improvement between the previous and current cluster centroids. In order to do that, a distance is computed between the previous and current cluster centroids. This is done in the same way as computing distances between an object and a cluster centroid but in addition, this distance is compared to a threshold value ϵ. Considering that the cluster centroids are shared between Alice and Bob, the result of the computation of the squared distance of cluster centroids for the kth and $(k+1)$th iterations is again random shares for Alice and Bob:

$$(\text{dist}(\mu^{A,k+1}_j + \mu^{B,k+1}_j, \mu^{A,k}_j + \mu^{B,k}_j))^2 = \alpha_j + \beta_j, \quad (25)$$

where α and β are the shares of Alice and Bob. Alice and Bob then apply Yao's protocol on their K-length vectors $(\alpha_1, \dots, \alpha_K)$ and $(\beta_1, \dots, \beta_K)$ to check if $\alpha_j + \beta_j < \epsilon$ for $1 \le j \le K$.

3.2. Recommender systems

Recommender services play an important role in applications like e-commerce and direct recommendations for multimedia contents. These services attempt to predict items that a user may be interested in by implementing a signal processing algorithm known as *collaborative filtering* on user preferences to find similar users that share the same taste (likes or dislikes). Once similar users are found, this information can be used in variety ways such as recommending restaurants, hotels, books, audio, and video.

Recommender systems store user data, also known as preferences, in servers, and the collaborative filtering algorithms work on these stored preferences to generate recommendations. The amount of data collected from each user directly affects the accuracy of the predictions. There are two concerns in collecting information from the users in such systems. First, in an ordinary system they are in the order of thousands items, so that it is not realistic for the users to rate all of them. Second, users would not like to reveal too much privacy sensitive information that can be used to track them.

The first problem, also known as the sparseness problem in datasets, is addressed for collaborative filtering algorithms in [43–45]. The second problem on user privacy is of interest to this survey paper since users tend to not give more information about themselves for privacy concerns and yet they expect more accurate recommendations that fit their taste. This tradeoff between privacy and accuracy leads us to an entirely new perspective on recommender systems. Namely, how can privacy of the users be protected in recommender systems without loosing too much accuracy?

We describe two solutions that address the problem of preserving privacy of users in recommender systems. In the first approach, user privacy is protected by means of encryption and the recommendations are still generated by processing these encrypted preference values. In the second approach, protecting the privacy of the users is possible without encryption but by means of perturbation of user preference data.

3.2.1. Recommendations by partial SVD on encrypted preferences

Canny [46] addresses the user privacy problem in recommender systems and proposes to encrypt user preferences. Assume that the recommender system applies a collaborative filtering algorithm on a matrix **P** of users versus item ratings. Each row of this matrix represents the corresponding user's taste for the corresponding items. Canny proposes to use a collaborative filtering algorithm based on dimension reduction of **P**. In this way, an approximation matrix of the original preference matrix is obtained in a lower dimension that best represents the user taste for the overall system. When a new user enters the system, the recommendations are generated by simply reprojecting the user preference vector, which has many unrated items, over the approximation matrix. As a result, a new vector will be obtained which contains approximated values for the unrated items [43, 46].

The ratings in recommender systems are usually integer numbers within a small range and items that are not rated are usually assigned to zero. To protect the privacy of the users, the user preferences vector $\mathbf{X} = [x_1, x_2, \dots, x_M]$ is encrypted individually as $E_{\text{pk}}(\mathbf{X})$. To reduce the dimension of the preference matrix **P** singular value decomposition (SVD) is an option. The SVD allows **P** to be written as

$$\mathbf{P} = \mathbf{U}\mathbf{D}\mathbf{V}^T, \tag{26}$$

where the columns of **U** are the left singular vectors, **D** is a diagonal matrix containing the singular values, and $\mathbf{V}^T$ has rows that are the right singular vectors.

Once the SVD of the preference matrix **P** is computed, an approximation matrix in a lower-dimension subspace can be computed easily. Computing the SVD on **P** that contains encrypted user preferences is, however, more complicated.

Computing the decomposition of the users' preference matrix requires sums of products of vectors. If the preference vector of each user is encrypted, there is no efficient way of computing sums of products of vectors since this would require an algebraic homomorphic cryptosystem. Using secure multiparty computation protocols on this complex function is costly considering the size of the circuit necessary for the complex operation.

Instead of straightforward computation of SVD, Canny [46] proposed to use an iterative approximation algorithm to obtain a partial decomposition of the user preference matrix. The conjugate gradient algorithm is an iterative procedure consisting merely of *additions* of vectors which *can* be done under homomorphically encrypted user preference vectors. Each iteration in the protocol has two steps, that is, users compute (1) their contribution to the current gradient and (2) scalar quantities for the optimization of the gradient. Both steps require only additions of vectors thus we only explain the first step.

For the first step of the iterations, each user computes his contribution $\mathbf{G}_k$ to the current gradient **G** by the following equation:

$$\mathbf{G}_k = \mathbf{A}\mathbf{X}_k^T\mathbf{X}_k(\mathbf{I} - \mathbf{A}^T\mathbf{A}), \tag{27}$$

where matrix **A** is the approximation of the preference matrix **P** and it is initialized as a random matrix before the protocol starts. Each user encrypts his own gradient vector $\mathbf{G}_k$ with the public key of the user group by following the Pedersen's threshold scheme [47] that uses El Gamal cryptosystem which is modified to be additively homomorphic. All contributions from the users are then added up to form the encrypted gradient $E_{\text{pk}}(\mathbf{G})$ by using the additive homomorphic property of the cryptosystem:

$$E_{\text{pk}}(\mathbf{G}) = E_{\text{pk}}\left(\sum_{k\in\text{users}} \mathbf{G}_k\right) = \prod_{k\in\text{users}} E_{\text{pk}}(\mathbf{G}_k). \tag{28}$$

This resulting vector $E_{\text{pk}}(\mathbf{G})$ is then jointly decrypted and used to update the approximated matrix **A** which is publicly known and used to compute the new gradient for the next iteration.

Although the protocol is based on addition of vectors, zero-knowledge proof protocols play an important role. The validity of the user inputs, that is, the encrypted preference vector elements lie in a certain range, are verified by zero-knowledge proofs. Moreover, the partial encryption results

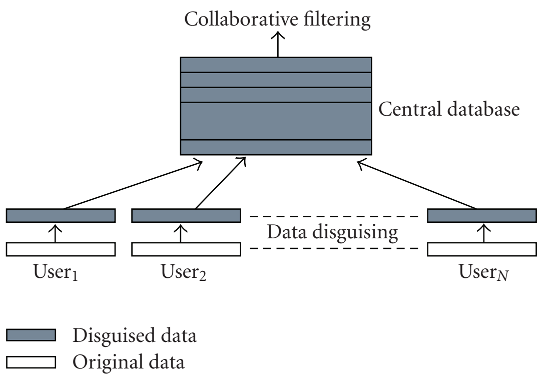

Figure 5: Privacy preserving collaborative filtering with user preference perturbation.

from the users are also proved valid by running a zero-knowledge proof protocol. Both group of zero-knowledge proofs are checked by a subgroup of users of whose majority is necessary for the validation.

Canny [48] also applies this approach to a different collaborative filtering method, namely, expectation maximization- (EM-) based factor analysis. Again this algorithm involves simple iterative operations that can be implemented by vector additions. In both recommender system solutions, multiple iterations are necessary for the algorithm to converge and in each iteration, users need to participate in the cryptographic computations as in joint decryption and zero-knowledge proofs for input validation. These computations are interactive and thus, it is imperative for the users to be online and synchronized.

3.2.2. Randomized perturbation to protect preferences

Previous section showed that homomorphic cryptosystems, zero-knowledge proof protocols, and secure multiparty computations play an important role in providing solutions for processing encrypted data. However, there are other ways to preserve privacy. In the following, we discuss preserving privacy in recommender systems by perturbation of user data.

Randomized perturbation technique was first introduced in privacy-preserved data-mining by Agrawal and Srikant [34]. Polat and Du [49, 50] proposed to use this randomization-based technique in collaborative filtering. The user privacy is protected by simply randomizing user data while certain computations on aggregate data can still be done. Then, the server generates recommendations based on the blinded data but can not derive the user's private information (Figure 5).

Consider the scalar product of two vectors $\mathbf{X}$ and $\mathbf{Y}$. These vectors are blinded by $\mathbf{R} = [r_1, \dots, r_M]$ and $\mathbf{S} = [s_1, \dots, s_M]$ such that $\mathbf{X}' = \mathbf{X} + \mathbf{R}$ and $\mathbf{Y}' = \mathbf{Y} + \mathbf{S}$. Here r_i's and s_i's are uniformly distributed random values with zero mean. The scalar product of $\mathbf{X}$ and $\mathbf{Y}$ can be estimated from $\mathbf{X}'$ and $\mathbf{Y}'$:

$$\mathbf{X}' \cdot \mathbf{Y}' = \sum_{k=1}^{M} (x_k y_k + x_k s_k + r_k y_k + r_k s_k) \approx \sum_{k=1}^{M} x_k y_k. \tag{29}$$

Since $\mathbf{R}$ and $\mathbf{S}$ are independent and independent of $\mathbf{X}$ and $\mathbf{Y}$, we have $\sum_{k=1}^{M} x_k s_k \approx 0$, $\sum_{k=1}^{M} r_k y_k \approx 0$, and $\sum_{k=1}^{M} r_k s_k \approx 0$. Similarly, the sum of the elements of any vector $\mathbf{A}$ can be estimated from its randomized form $\mathbf{A}'$. Polat and Du used these two approximations to develop a privacy-preserving collaborative filtering method [49, 50].

This method works if the number of users in the system is significantly large. Only then the computations based on aggregated data can still be computed with sufficient accuracy. Moreover, it is also pointed out in [51, 52] that the idea of preserving privacy by adding random noise might not preserve privacy as much as it had been believed originally. The user data can be reconstructed from the randomly perturbed user data matrix. The main limitation in the original work of Polat and Du is shown to be the item-invariant perturbation [53]. Therefore, Zhang et al. [53] propose a two-way communication perturbation scheme for collaborative filtering in which the server and the user communicates to determine perturbation guidance that is used to blind user data before sending to the server. Notwithstanding these approaches, the security of such schemes based on perturbation of data is not well understood.

4. CONTENT PROTECTION

4.1. Watermarking of content

In the past decade, content protection measures have been proposed based on digital watermarking technology. Digital watermarking [54, 55] allows hiding into a digital content information that can be detected or extracted at a later moment in time by means of signal processing operations such as correlation. In this way, digital watermarking provides a communication channel multiplexed into original content through which it is possible to transmit information. The type of information transmitted from sender to receiver depends on the application at hand. As an example, in a forensic tracing application, a watermark is used to embed a unique code into each copy of the content to be distributed, where the code links a copy either to a particular user or to a specific device. When unauthorized published content is found, the watermark allows to trace the user who has redistributed the content.

Secure signal processing needs to be performed in case watermark detection or embedding is done in untrusted devices; watermarking schemes usually rely on a symmetric key for both embedding and detection, which is critical to both the robustness and security of the watermark and thus needs to be protected.

For the application of secure signal processing in content protection, three categories can be identified, namely, distribution models, customer rights protection, and secure watermark detection. The first two categories are relevant to forensic tracing (fingerprinting) applications. In classical distribution models, the watermark embedding process is carried out by a trusted server before releasing the content to the user. However this approach is not scalable, and in large-scale distribution systems, the server may become overloaded. In addition, since point-to-point communication channels are

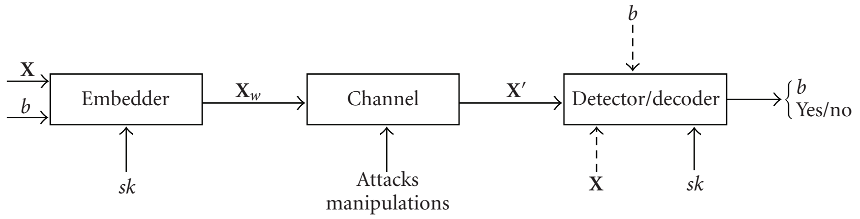

FIGURE 6: A digital watermarking model.

required, bandwidth requirements become prohibitive. A proposed solution is to use client-side watermark embedding. Since the client is untrusted, the watermark needs to be embedded without the client having access to the original content and watermark.

The customer's rights problem relates to the intrinsic problem of ambiguity when watermarks are embedded at the distribution server: a customer whose watermark has been found on unauthorized copies can claim that he has been framed by a malicious seller who inserted his identity as watermark in an arbitrary object. The mere existence of this problem may discredit the accuracy of the forensic tracing architecture. Buyer-seller protocols have been designed to embed a watermark based on the encrypted identity of the buyer, making sure that the watermarked copy is available only to the buyer and not to the seller.

In the watermark detection process, a system has to prove to a verifier that a watermark is present in certain content. Proving the presence of such a watermark is usually done by revealing the required detection information to the verifying party. All current applications assume that the verifier is a trusted party. However, this is not always true, for instance, if the prover is a consumer device. A cheating verifier could exploit the knowledge acquired during watermark detection to break the security of the watermarking system. Cryptographic protocols, utilizing zero-knowledge proofs, have been constructed in order to mitigate this problem.

We will first introduce a general digital watermarking model to define the notation that will be useful in the remainder of the section. An example of a watermarking scheme is proposed, namely, the one proposed by Cox et al. [56] since this scheme is adopted in many of the content protection applications.

4.1.1. *Watermarking model*

Figure 6 shows a common model for a digital watermarking system [57]. The inputs of the system are the original host signal **X** and some application dependent to-be-hidden information, here represented as a binary string $\mathbf{B} = [b_1, b_2, \dots, b_L]$, with b_i taking values in $\{0, 1\}$. The embedder inserts the watermark code **B** into the host signal to produce a watermarked signal $\mathbf{X_w}$, usually making use of a secret key sk to control some parameters of the embedding process and allow the recovery of the watermark only to authorized users.

The watermark channel takes into account all processing operations and (intentional or non-intentional) manipulations the watermarked content may undergo during distribution and use. As a result, the watermarked content $\mathbf{X_w}$ is modified into the "received" version $\mathbf{X}'$. Based on $\mathbf{X}'$, either a detector verifies the presence of a specific message given to it as input, thus only answering *yes* or *no*, or a decoder reads the (binary) information conveyed by the watermark. Detectors and decoders may need to know the original content **X** in order to retrieve the hidden information (non-blind detector/decoder), or they do not require the original content (blind or oblivious detector/decoder).

4.1.2. *Watermarking algorithm*

Watermark information is embedded into host signals by making imperceptual modifications to the host signal. The modifications are such that they convey the to-be-hidden information **B**. The hidden information can be retrieved afterwards from the modified content by detecting the presence of these modifications. Embedding is achieved by modifying the set of features $\mathbf{X} = [x_1, x_2, \dots, x_M]$. In the most simple case, the features are simple signal amplitudes. In more complicated scenarios, the features can be DCT or wavelet coefficients. Several watermarking schemes make use of a spread-spectrum approach to code the to-be-hidden information **B** into $\mathbf{W} = [w_1, w_2, \dots, w_M]$. Typically, **W** is a realization of a normally distributed random signal with zero mean and unit variance.

The most well-known spread-spectrum techniques was proposed by Cox et al. [56]. The host signal is first transformed into a discrete cosine transform (DCT) representation. Next the largest magnitude DCT coefficients are selected, obtaining the set of features **X**. The multiplicative watermark embedding rule is defined as follows:

$$x_{w,i} = x_i + \gamma w_i x_i = x_i(1 + \gamma w_i), \tag{30}$$

where $x_{w,i}$ is the ith component of the watermarked feature vector and γ is a scaling factor controlling the watermark strength. Finally, an inverse DCT transform yields the watermarked signal $\mathbf{X_w}$.

To determine if a given signal **Y** contains the watermark **W**, the decoder computes the DCT of **Y**, extracts the set $\mathbf{X}'$ of largest DCT coefficients, and then computes the correlation $\rho_{\mathbf{X}'\mathbf{W}}$ between the features $\mathbf{X}'$ and the watermark **W**. If the correlation is larger than a threshold T, that is,

$$\rho_{\mathbf{X}'\mathbf{W}} = \frac{\langle \mathbf{X}', \mathbf{W} \rangle}{\langle \mathbf{X}', \mathbf{X}' \rangle} \geq T, \tag{31}$$

the watermark is considered present in **Y**.

4.2. *Client-side watermark embedding*

Client-side watermark embedding systems transmit the same encrypted version of the original content to all the clients but a client-specific decryption key allows to decrypt the content and at the same time implicitly embed a watermark. When the client uses his key to decrypt the content, he obtains a uniquely watermarked version of the content. The security properties of the embedding scheme usually guarantees that obtaining either the watermark or the original content in the clear is of comparable hardness as removing the watermark from the personalized copy.

In literature, several approaches for secure embedding can be found. In [58], a pseudorandom mask is blended over each frame of a video. Each client is given a different mask, which, when subtracted from the masked broadcast video, leaves an additive watermark in the content. The scheme is not very secure because since the same mask is used for all frames of a video, it can be estimated by averaging attacks.

In broadcast environments, stream switching [59, 60] can be performed. Two differently watermarked signals are chopped up into small chunks. Each chunk is encrypted by a different key. Clients are given a different set of decryption keys that allow them to selectively decrypt chunks of the two broadcast streams such that each client obtains the full stream decrypted. The way the full stream is composed out of the two broadcast versions encodes the watermark. This solution consumes considerable bandwidth, since the data to be broadcast to the clients is twice as large as the content itself.

A second solution involves partial encryption, for instance, encrypting the signs of DCT coefficients of a signal [61]. Since the sign bits of DCT coefficients are perceptually significant, the partially encrypted version of the signal is heavily distorted. During decryption, each user has a different key that decrypts only a subset of these coefficients, so that some signs are left unchanged. This leaves a detectable fingerprint in the signal. A similar approach was used in [62] to obtain partial encryption-based secure embedding solutions for audiovisual content.

A third approach is represented by methods using a stream-cipher that allows the use of multiple decryption keys, which decrypt the same cipher text to slightly different plain-texts. Again, the difference between the original and the decrypted content represents the embedded watermark. The first scheme following this approach was proposed by Anderson and Manifavans [63] who designed a special stream cipher, called Chameleon, which allows to decrypt Chameleon-encrypted content in slightly different ways. During encryption, a key and a secure index generator are used to generate a sequence of indices, which are used to select four entries from a look-up-table (LUT). These entries are XORed with the plaintext to form a word of the cipher text. The decryption process is identical to encryption except for the use of a decryption LUT, which is obtained by properly inserting bit errors in some entries of the encryption LUT. Decryption superimposes these errors onto the content, thus leaving a unique watermark. Recently, Adelsbach et al. [64] and Celik et al. [65] proposed generalizations of Chameleon, suitable for embedding robust spread-spectrum watermarks. The schemes operate on LUTs composed of integers from $\mathbb{Z}_p$ and replace the XOR operation by a (modular) addition.

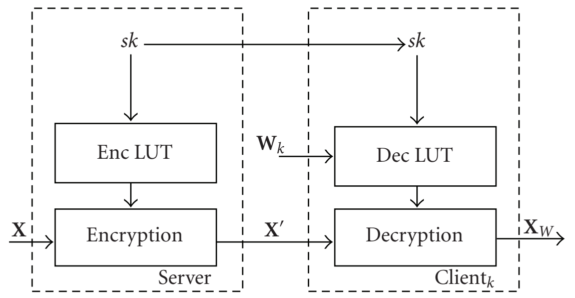

Figure 7: Encryption and following joint decryption and watermarking procedure proposed in [65].

In more detail, the secure embedding solution works as follows. The distribution server generates a long-term master encryption LUT $\mathbf{E}$ of size L, whose entries properlygenerated random samples; $\mathbf{E}$ will be used to encrypt the content to be distributed to the clients. Next, for the kth client, the server generates a personalized watermark LUT $\mathbf{W}_k$ according to a desired probability distribution, and builds a personalized decryption LUT $\mathbf{D}_k$ by combining the master LUT and the watermark LUT:

$$\mathbf{D}_k[i] = -\mathbf{E}[i] + \mathbf{W}_k[i]. \qquad (32)$$

The personalized LUTs are then transmitted once to each client over a secure channel. Let us note that the generation of the LUTs is carried out just once at the setup of the application. A content $\mathbf{X}$ is encrypted by adding to it a pseudorandom sequence obtained by selecting some entries of the LUT with a secure pseudorandom sequence generator driven by a session key sk. Each client receives the encrypted content $\mathbf{X}'$ along with the session key sk and decrypts it using some entries of his/her personalized decryption LUT $\mathbf{D}_k$ (again chosen according to sk), with the final effect that a spread-spectrum watermark sequence is embedded into the decrypted content. This process is summarized in Figure 7. In detail, driven by the session key sk, a set of indices t_{ij} is generated, where $0 \leq i \leq M-1$, $0 \leq j \leq S-1$, $0 \leq t_{ij} \leq L-1$. Each feature of the content x_i is encrypted by adding S entries of the encryption LUT, obtaining the encrypted feature x_i' as follows:

$$x_i' = x_i + \sum_{j=0}^{S-1} E[t_{ij}]. \qquad (33)$$

Joint decryption and watermarking is accomplished by reconstructing with the session key *sk* the same set of indices t_{ij} and by adding S entries of the decryption LUT to each encrypted feature x_i':

$$x_{w,i} = x_i' + \sum_{j=0}^{S-1} \mathbf{D}[t_{ij}] = x_i + \sum_{j=0}^{S-1} \mathbf{W}[t_{ij}] = x_i + w_i. \qquad (34)$$

4.3. Buyer seller protocols

Forensic tracing architectures which perform watermark embedding at the distribution server are vulnerable against a dishonest seller. The mere fact that a seller may fool a buyer may have an impact on the credibility of the whole tracing system. (Note that a seller may in fact have an incentive to fool a buyer: a seller who acts as an authorized reselling agent may be interested in distributing many copies of a work containing the fingerprint of a single buyer to avoid paying the royalties to the author by claiming that such copies were illegally distributed or sold by the buyer.)

A possible solution consists in resorting to a trusted third party, responsible for both embedding and detection of watermarks; however, such an approach is not feasible in practical applications because the TTP could easily become a bottleneck for the whole system. The Buyer-Seller Protocol relies on cryptographic primitives to perform watermark embedding [66]; the protocol assures that the seller does not have access to the watermarked copy carrying the identity of the buyer, hence he cannot distribute or sell these copies. In spite of this, the seller can identify the buyer from whom unauthorized copies originated, and prove it by using a proper dispute resolution protocol.

We describe the protocol by Memon and Wong [66] in more detail. Let Alice be the seller, Bob the buyer, and WCA a trusted watermark certification authority in charge of generating legal watermarks and sending them to any buyer upon request. The protocol uses a public key cryptosystem which is homomorphic with respect to the operation used in the watermark embedding equation (i.e., the cryptosystem will be multiplicatively homomorphic if watermark embedding is multiplicative, like in Cox's scheme); moreover, Alice and Bob possess a pair of public/private keys denoted by pk_A, pk_B (public keys) and sk_A, sk_B (private keys).

In the first part of the protocol, on request of Bob, the WCA generates a valid watermark signal $\mathbf{W}$ and sends it back to Bob, encrypted with Bob's public key $E_{\mathrm{pk}_B}(\mathbf{W})$, along with its digital signature $S_{\mathrm{WCA}}(E_{\mathrm{pk}_B}(\mathbf{W}))$, to prove that the watermark is valid.

Next, Bob sends to Alice $E_{\mathrm{pk}_B}(\mathbf{W})$ and $S_{\mathrm{WCA}}(E_{\mathrm{pk}_B}(\mathbf{W}))$, so that Alice can verify that the encrypted watermark has been generated by the WCA. Alice performs two watermark embedding operations. First, she embeds (with any watermarking scheme) into the original content $\mathbf{X}$ a watermark $\mathbf{V}$, which just conveys a distinct ID univocally identifying the transaction, obtaining the watermarked content $\mathbf{X}'$. Next, a second watermark is built by using $E_{\mathrm{pk}_B}(\mathbf{W})$: Alice permutes the watermark components through a secret permutation σ:

$$\sigma(E_{\mathrm{pk}_B}(\mathbf{W})) = E_{\mathrm{pk}_B}(\sigma(\mathbf{W})), \tag{35}$$

and inserts $E_{\mathrm{pk}_B}(\sigma(\mathbf{W}))$ in $\mathbf{X}'$ directly in the encrypted domain, obtaining the final watermarked content $\mathbf{X}''$ in encrypted form; $\mathbf{X}''$ is thus unknown to her. This is possible due to the homomorphic property of the cipher:

$$E_{\mathrm{pk}_B}(\mathbf{X}'') = E_{\mathrm{pk}_B}(\mathbf{X}') \cdot E_{\mathrm{pk}_B}(\sigma(\mathbf{W})). \tag{36}$$

When Bob receives $E_{\mathrm{pk}_B}(\mathbf{X}'')$, he decrypts it by using his private key sk_B, thus obtaining $\mathbf{X}''$, where the watermarks $\mathbf{V}$ and $\sigma(\mathbf{W})$ are embedded. Note that Bob cannot read the watermark $\sigma(\mathbf{W})$, since he does not know the permutation σ. The scheme is represented in Figure 8.

In order to recover the identity of potential copyright violators, Alice first looks for the presence of $\mathbf{V}$. Upon detection of an unauthorized copy of $\mathbf{X}$, say $\mathbf{Y}$, she can use the second watermark to effectively prove that the copy is originated from Bob. To do so, Alice must reveal to judge the permutation σ, the encrypted watermark $E_{\mathrm{pk}_B}(\mathbf{W})$ and $S_{\mathrm{WCA}}(E_{\mathrm{pk}_B}(\mathbf{W}))$. After verifying $S_{\mathrm{WCA}}(E_{\mathrm{pk}_B}(\mathbf{W}))$, the judge asks Bob to use his private key sk_B to compute and reveal $\mathbf{W}$. Now it is possible to check $\mathbf{Y}$ for the presence of $\sigma(\mathbf{W})$: if such a presence is verified, then Bob is judged guilty, otherwise, Bob's innocence has been proven. Note that if $\sigma(\mathbf{W})$ is found in $\mathbf{Y}$, Bob cannot state that $\mathbf{Y}$ originated from Alice since to do so Alice should have known either $\mathbf{W}$ to insert it within the plain asset $\mathbf{X}$, or sk_B to decrypt $E_{\mathrm{pk}_B}(\mathbf{X}'')$ after the watermark was embedded in the encrypted domain.

As a particular implementation of the protocol, [66] proposed to use Cox's watermarking scheme and a multiplicatively homomorphic cipher (despite its deterministic nature, authors use RSA). More secure and less complex implementations of the Buyer Seller Protocol have been proposed in [67–70].

4.4. Secure watermark detection

To tackle the problem of watermark detection in the presence of an untrusted verifier (to whom the watermark secrets cannot be disclosed), two approaches have been proposed: one approach called *asymmetric watermarking* [71, 72] uses different keys for watermark embedding and detection. Whereas a watermark is embedded using a private key, its presence can be detected by a public key. In such schemes, the knowledge of the public detection key must not enable an adversary to remove the embedded watermark; unfortunately, none of the proposed schemes is sufficiently robust against malicious attacks [73]. Another approach is represented by *zero-knowledge watermark detection*.

Zero-knowledge watermark detection (ZKWD) uses a cryptographic protocol to wrap a standard symmetric watermark detection process. In general, a zero-knowledge watermark detection algorithm is an interactive proof system where a prover tries to convince a verifier that a digital content $\mathbf{X}'$ is watermarked with a given watermark $\mathbf{B}$ without disclosing $\mathbf{B}$. In contrast to the standard watermark detector, in ZKWD the verifier is given only properly encoded (or encrypted) versions of security-critical watermark parameters. Depending on the particular protocol, the watermark code, the watermarked object, a watermark key, or even the original unmarked object is available in an encrypted form to the verifier. The prover runs the zero-knowledge watermark detector to demonstrate to the verifier that the encoded watermark is present in the object in question, without removing the encoding. A protocol run will not leak any information except for the unencoded inputs and the watermark presence detection result.

Early approaches for zero-knowledge watermark detection used permutations to conceal both the watermark and

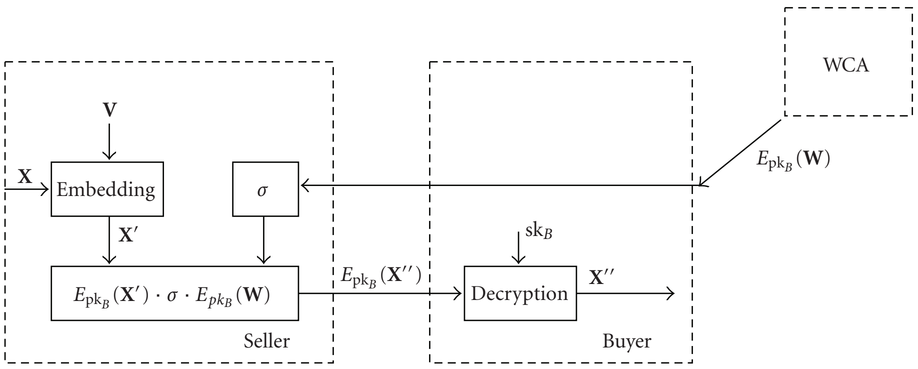
WCA
V
X
Embedding
σ
$E_{\mathrm{pk}_B}(\mathbf{W})$
$\mathbf{X}'$
sk_B
$E_{\mathrm{pk}_B}(\mathbf{X}') \cdot \sigma \cdot E_{pk_B}(\mathbf{W})$
$E_{\mathrm{pk}_B}(\mathbf{X}'')$
Decryption
$\mathbf{X}''$
Seller
Buyer

Figure 8: The scheme of the Buyer Seller Protocol proposed in [66].

the object in which the watermark is to be detected [74]; the protocol assures that the permuted watermark is detected in the permuted content and that both the watermark and the object are permuted in the same manner. Craver [75] proposed to use ambiguity attacks as a central tool to construct zero-knowledge detectors; such attacks allow to compute a watermark that is detectable in a content but never has been embedded there. To use ambiguity attacks in a secure detector, the real watermark is concealed within a number of fake marks. The prover has to show that there is a valid watermark in this list without revealing its position. Now, the adversary (equipped solely with a watermark detector) cannot decide which of the watermarks is not counterfeit. Removal of the watermark is thus sufficiently more difficult.

Another proposal is to compute the watermark detection statistic in the encrypted domain (e.g., by using additive homomorphic public-key encryption schemes or commitments) and then use zero-knowledge proofs to convince the verifier that the detection statistic exceeds a fixed threshold. This approach was first proposed by Adelsbach and Sadeghi [76], who use a homomorphic commitment scheme to compute the detection statistic; the approach was later refined in [77].

Adelsbach and Sadeghi [76] propose a zero-knowledge protocol based on the Cox's watermarking scheme. In contrast to the original algorithm, it is assumed that the watermark and DCT-coefficients are integers and not real numbers (this can be achieved by appropriate quantization). Moreover, for efficiency reasons, the correlation computation in (31) is replaced by the detection criterion:

$$\begin{aligned} C &:= \left(\langle \mathbf{X}', \mathbf{W}\rangle\right)^2 - \langle \mathbf{X}', \mathbf{X}'\rangle \cdot T^2 \\ &:= (A)^2 - B \geq 0; \end{aligned} \tag{37}$$

the latter detection criterion is equivalent to the original one, provided that the factor A is positive.

The following zero-knowledge detection protocol has been designed to allow the prover to prove to a verifier that the watermark committed to in the commitment $\mathrm{com}(\mathbf{W})$ is present in the watermarked content $\mathbf{X}'$, without revealing any information about $\mathbf{W}$. In the protocol, the authors employ an additively homomorphic commitment scheme (such as the one proposed by Damgård and Fujisaki [78]). Let p_{pub}, $\mathbf{X}'$, $\mathrm{com}(\mathbf{W})$, T be the common inputs of prover and verifier and let p_{sec} be the private input of the prover. First, both prover and verifier select the watermarked features $\mathbf{X}'$ and compute the value B of (37); the prover sends a commitment $\mathrm{com}(B)$ to the verifier and opens it immediately, allowing him to verify that the opened commitment contains the same value B he computed himself. Now both compute the commitment

$$\mathrm{com}(A) = \prod_{i=1}^{M} \mathrm{com}(w_i)^{x_i'} \tag{38}$$

by taking advantage of the homomorphic property of the commitment scheme. Subsequently, the prover proves in zero-knowledge that $A \geq 0$. Next, the prover computes the value A^2, sends a commitment $\mathrm{com}(A^2)$ to the verifier, and gives him a zero-knowledge proof to prove that $\mathrm{com}(A^2)$ really contains the square of the value contained in $\mathrm{com}(A)$. Being convinced that $\mathrm{com}(A^2)$ really contains the correctly computed value A^2, the two parties compute the commitment $\mathrm{com}(C) := \mathrm{com}(A^2)/\mathrm{com}(B)$ on the value C. Finally, the prover proves to the verifier, with a proper zero-knowledge protocol, that $\mathrm{com}(C) \geq 0$. If this proof is accepted, then the detection algorithm ends with true, otherwise, with false.

While early protocols addressed only correlation-based watermark detectors, the approach has recently be extended to Gaussian maximum likelihood detectors [79] and Dither modulation watermarks [80, 81].

5. CONCLUSION AND DISCUSSION

The availability of signal processing algorithms that work directly on the encrypted data would be of great help for application scenarios where "valuable" signals must be produced, processed, or exchanged in digital format. In this paper, we have broadly referred to this new class of signal processing techniques operating in the encrypted domain as signal processing in the encrypted domain. We mainly review the state-of-the-art, describing the necessary properties of the cryptographic primitives and highlighting the limits of current solutions that have an impact on processing in the encrypted domain.

Concerning the use of cryptographic primitives for signal processing in the encrypted domain, we can observe that treating the digital content as a binary data is not realistic and eliminates the possibility of further processing. Concerning the basic encryption primitives that make processing in the encrypted domain possible, for the particular case when it is necessary to compress an encrypted signal, a possibility is to resort to the theory of coding with side information; this primitive, however, seems to be applicable only to this kind of problem.

The general cryptographic tools that allow to process encrypted signals are homomorphic cryptosystems since they allow performing linear computations on the encrypted data. In order to implement necessary signal processing operations, it seems crucial to have an algebraic cryptosystem. However, such a system does not exist and despite the fact that there is no formal proof, it is highly believed that such a system will be insecure due to preserving too much structure. Yet, homomorphic cryptosystems are the key components in signal processing in the encrypted domain. Another property, important for signal processing in the encrypted domain, is probabilistic encryption: since signal samples are usually 8-bit or 16-bit in length, encrypting such values with a deterministic cryptosystem will result in reoccurring encrypted values which significantly reduces the search space for brute-force attacks. A probabilistic scheme, which does not encrypt two equal plain texts into the same cipher text, eliminates such an issue. However, once the data is encrypted, the probabilistic encryption makes it impossible to check if the encrypted value represents a valid input for the purposes of the subsequent processing. Similarly, the output of a function that is computed with encrypted data may need to be compared with another value. In such situations, cryptography provides a solution known as zero-knowledge proofs. Moreover, when nonlinear function needs to be computed, homomorphic encryption cannot help; in such a case, it is possible to resort to interactive protocols (e.g., the secure multiparty computation). The limit of these protocols is that a general solution is infeasible for situations where the parties own huge quantities of data or the functions to be evaluated are complex, as it happens in signal processing scenarios.

Though the possibility of processing encrypted data has been advanced several years ago, processing encrypted signals poses some new problems due to the peculiarities of signals with respect to other classes of data more commonly encountered in the cryptographic literature, for example, alphanumeric strings or bit sequences. One property of signals is that in many signal processing applications, there is interest on the way the signal varies with time rather than the single values it assumes. Moreover, the arithmetic used to represent the signal samples has to be carefully taken into account. If the signal samples are represented by means of fixed-point arithmetic, we need to ensure that no overflow occurs; for signal processing in the encrypted domain, it is necessary that such a condition is ensured a priori by carefully constraining the properties of the signals we operate on and the type and number of operations we want to perform on them. Moreover, keeping the distinction between the integer and the fractional part of a number is a difficult task, given that, once again, it calls for the possibility of comparing encrypted numbers. If the signals are represented by means of floating point arithmetic, working in the encrypted domain is a very difficult task due to the necessity of implementing operations such as comparisons and right shifts in the encrypted domain, for which efficient (noninteractive) solutions are not known yet.

ACKNOWLEDGMENTS

The work reported here has been funded in part by the European Community's Sixth Framework Programme under Grant no. 034238, SPEED project—Signal Processing in the Encrypted Domain. The work reported reflects only the authors' views; the European Community is not liable for any use that may be made of the information contained herein.

REFERENCES

[1] JPSEC, International standard, ISO/IEC 15444-8, 2007.

[2] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT '99)*, vol. 1592 of *Lecture Notes in Computer Science*, pp. 223–238, Springer, Prague, Czech Republic, May 1999.

[3] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in *Proceedings of the 4th Annual International Cryptology Conference (CRYPTO '84)*, vol. 196 of *Lecture Notes in Computer Science*, pp. 10–18, Springer, Santa Barbara, Calif, USA, August 1985.

[4] R. L. 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, 1978.

[5] J. R. Troncoso-Pastoriza, S. Katzenbeisser, M. Celik, and A. Lemma, "A secure multidimensional point inclusion protocol," in *Proceedings of the 9th Workshop on Multimedia & Security (MM&Sec '07)*, pp. 109–120, ACM Press, Dallas, Tex, USA, September 2007.

[6] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT '04)*, vol. 3027 of *Lecture Notes in Computer Science*, pp. 506–522, Springer, Interlaken, Switzerland, May 2004.

[7] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order preserving encryption for numeric data," in *Proceedings of the ACM International Conference on Management of Data (SIGMOD '04)*, pp. 563–574, ACM Press, Paris, France, June 2004.

[8] J. Domingo-Ferrer, "A new privacy homomorphism and applications," *Information Processing Letters*, vol. 60, no. 5, pp. 277–282, 1996.

[9] J. Domingo-Ferrer, "A provably secure additive and multiplicative privacy homomorphism," in *Proceedings of the 5th International Conference on Information Security (ISC '02)*, vol. 2433 of *Lecture Notes in Computer Science*, pp. 471–483, Springer, Sao Paulo, Brazil, September-October 2002.

[10] J. H. Cheon and H. S. Nam, "A cryptanalysis of the original Domingo-Ferrer's algebraic privacy homomophism," Cryptology ePrint Archive, Report 2003/221, 2003.

[11] D. Wagner, "Cryptanalysis of an algebraic privacy homomorphism," in *Proceedings of the 6th International Conference on Information Security (ISC '03)*, vol. 2851 of *Lecture Notes in Computer Science*, pp. 234–239, Bristol, UK, October 2003.

[12] C. Fontaine and F. Galand, "A survey of homomorphic encryption for nonspecialists," *EURASIP Journal on Information Security*, vol. 2007, Article ID 13801, 10 pages, 2007.

[13] B. Schoenmakers and P. Tuyls, "Practical two-party computation based on the conditional gate," in *Proceedings of the 10th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT '04)*, vol. 3329 of *Lecture Notes in Computer Science*, pp. 119–136, Jeju Island, Korea, December 2004.

[14] S. Goldwasser and S. Micali, "Probabilistic encryption," *Journal of Computer and System Sciences*, vol. 28, no. 2, pp. 270–299, 1984.

[15] J. Benaloh, *Verifiable secret-ballot elections*, Ph.D. thesis, Department of Computer Science, Yale University, New Haven, Conn, USA, 1988.

[16] D. Naccache and J. Stern, "A new public key cryptosystem based on higher residues," in *Proceedings of the 5th ACM Conference on Computer and Communications Security (CCS '98)*, pp. 59–66, San Francisco, Calif, USA, November 1998.

[17] T. Okamoto and S. Uchiyama, "A new public-key cryptosystem as secure as factoring," in *Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT '98)*, vol. 1403 of *Lecture Notes in Computer Science*, pp. 308–318, Springer, Espoo, Finland, May-June 1998.

[18] I. Damgård and M. Jurik, "A generalisation, a simplification and some applications of Paillier's probabilistic public-key system," in *Proceedings of the 4th International Workshop on Practice and Theory in Public Key Cryptosystems (PKC '01)*, vol. 1992 of *Lecture Notes In Computer Science*, pp. 119–136, Springer, Cheju Island, Korea, February 2001.

[19] O. Goldreich, *Foundations of Cryptography I*, Cambridge University Press, Cambridge, UK, 2001.

[20] C. P. Schnorr, "Efficient identification and signatures for smart cards," in *Proceedings of the 9th Annual International Cryptology Conference (CRYPTO '89)*, vol. 435 of *Lecture Notes in Computer Science*, pp. 239–252, Springer, Santa Barbara, Calif, USA, August 1990.

[21] H. Lipmaa, "On diophantine complexity and statistical zero-knowledge arguments," in *Proceedings of the 9th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT '03)*, vol. 2894 of *Lecture Notes in Computer Science*, pp. 398–415, Springer, Taipei, Taiwan, November-December 2003.

[22] F. Boudot, "Efficient proofs that a committed number lies in an interval," in *Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT '00)*, vol. 1807 of *Lecture Notes in Computer Science*, pp. 431–444, Springer, Bruges, Belgium, May 2000.

[23] E. Fujisaki and T. Okamoto, "Statistical zero-knowledge protocols to prove modular polynomial relations," in *Proceedings of the 17th Annual International Cryptology Conference (CRYPTO '97)*, vol. 1294 of *Lecture Notes in Computer Science*, pp. 16–30, Springer, Santa Barbara, Calif, USA, August 1997.

[24] J. Camenisch and M. Michels, "Proving in zero-knowledge that a number is the product of two safe primes," in *Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT '99)*, vol. 1592 of *Lecture Notes in Computer Science*, pp. 107–122, Springer, Prague, Czech Republic, May 1999.

[25] T. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Proceedings of the 11th Annual International Cryptology Conference (CRYPTO '91)*, vol. 576 of *Lecture Notes in Computer Science*, pp. 129–140, Springer, Santa Barbara, Calif, USA, August 1992.

[26] A. C. Yao, "Protocols for secure computations," in *Proceedings of 23rd IEEE Symposium on Foundations of Computer Science (FOCS '82)*, pp. 160–164, Chicago, Ill, USA, November 1982.

[27] B. Schoenmakers and P. Tuyls, "Efficient binary conversion for Paillier encrypted values," in *Proceedings of the 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT '06)* , vol. 4004 of *Lecture Notes in Computer Science*, pp. 522–537, Springer, St. Petersburg, Russia, May-June 2006.

[28] O. Goldreich, *Foundations of Cryptography II*, Cambridge University Press, Cambridge, UK, 2004.

[29] S.-C. S. Cheung and T. Nguyen, "Secure multiparty computation between distrusted networks terminals," *EURASIP Journal on Information Security*, vol. 2007, Article ID 51368, 10 pages, 2007.

[30] M. Johnson, P. Ishwar, V. Prabhakaran, D. Schonberg, and K. Ramchandran, "On compressing encrypted data," *IEEE Transactions on Signal Processing*, vol. 52, no. 10, pp. 2992–3006, 2004.

[31] D. Slepian and J. K. Wolf, "Noiseless coding of correlated information sources," *IEEE Transactions on Information Theory*, vol. 19, pp. 471–480, 1973.

[32] S. Pradhan and K. Ramchandran, "Distributed source coding using syndromes (DISCUS): design and construction," *IEEE Transactions on Information Theory*, vol. 49, no. 3, pp. 626–643, 2003.

[33] Y. Lindell and B. Pinkas, "Privacy preserving data mining," in *Proceedings of the 20th Annual International Cryptology Conference (CRYPTO '00)*, vol. 1880 of *Lecture Notes in Computer Science*, pp. 36–54, Santa Barbara, Calif, USA, August 2000.

[34] R. Agrawal and R. Srikant, "Privacy-preserving data mining," *ACM SIGMOD Record*, vol. 29, no. 2, pp. 439–450, 2000.

[35] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Y. Zhu, "Tools for privacy preserving distributed data mining," *ACM SIGKDD Explorations Newsletter*, vol. 4, no. 2, pp. 28–34, 2002.

[36] M. Kantarcioglu and J. Vaidya, "Privacy preserving naive Bayes classifier for horizontally partitioned data," in *Procceedings of the IEEE Workshop on Privacy Preserving Data Mining (ICDM '03)*, pp. 3–9, Melbourne, Fla, USA, November 2003.

[37] B. Pinkas, "Cryptographic techniques for privacy-preserving data mining," *ACM SIGKDD Explorations Newsletter*, vol. 4, no. 2, pp. 12–19, 2002.

[38] V. S. Verykios, E. Bertino, I. N. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis, "State-of-the-art in privacy preserving data mining," *ACM SIGMOD Record*, vol. 33, no. 1, pp. 50–57, 2004.

[39] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

[40] G. Jagannathan and R. N. Wright, "Privacy-preserving distributed k-means clustering over arbitrarily partitioned data," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '05)*, pp. 593–599, ACM Press, Chicago, Ill, USA, August 2005.

[41] B. Goethals, S. Laur, H. Lipmaa, and T. Mielikainen, "On secure scalar product computation for privacy-preserving data

mining," in *Proceedings of the 7th Annual International Conference in Information Security and Cryptology (ICISC '04)*, vol. 3506 of *Lecture Notes in Computer Science*, pp. 104–120, Springer, Seoul, Korea, December 2004.

[42] A. C. Yao, "How to generate and exchange secrets," in *Proceedings of the 27th Annual Symposium on Foundations of Computer Science*, pp. 162–167, Toronto, Ontario, Canada, October 1986.

[43] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: a constant time collaborative filtering algorithm," *Information Retrieval*, vol. 4, no. 2, pp. 133–151, 2001.

[44] J. D.M. Rennie and N. Srebro, "Fast maximum margin matrix factorization for collaborative prediction," in *Proceedings of the 22nd International Conference on Machine Learning (ICML '05)*, pp. 713–720, Bonn, Germany, August 2005.

[45] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Application of dimensionality reduction in recommender systems," in *Proceedings of the Web Mining for E-Commerce—Challenges and Opportunities (WEBKDD '00)*, Boston, Mass, USA, August 2000.

[46] J. F. Canny, "Collaborative filtering with privacy," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 45–57, Berkeley, Calif, USA, May 2002.

[47] T. Pedersen, "A threshold cryptosystem without a trusted party," in *Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques (EUROCRYPT '91)*, vol. 547 of *Lecture Notes in Computer Science*, pp. 522–526, Brighton, UK, April 1991.

[48] J. F. Canny, "Collaborative filtering with privacy via factor analysis," in *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '02)*, pp. 238–245, ACM Press, Tampere, Finland, August 2002.

[49] H. Polat and W. Du, "Privacy-preserving collaborative filtering using randomized perturbation techniques," in *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM '03)*, pp. 625–628, IEEE Computer Society, Melbourne, Fla, USA, November 2003.

[50] H. Polat and W. Du, "SVD-based collaborative filtering with privacy," in *Proceedings of the 20th Annual ACM Symposium on Applied Computing (SAC '05)*, vol. 1, pp. 791–795, Santa Fe, NM, USA, March 2005.

[51] Z. Huang, W. Du, and B. Chen, "Deriving private information from randomized data," in *Proceedings of the ACM International Conference on Management of Data (SIGMOD '05)*, pp. 37–48, ACM Press, Baltimore, Md, USA, June 2005.

[52] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar, "On the privacy preserving properties of random data perturbation techniques," in *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM '03)*, pp. 99–106, Melbourne, Fla, USA, November 2003.

[53] S. Zhang, J. Ford, and F. Makedon, "A privacy-preserving collaborative filtering scheme with two-way communication," in *Proceedings of the 7th ACM Conference on Electronic Commerce (EC '06)*, pp. 316–323, Ann Arbor, Mich, USA, June 2006.

[54] M. Barni and F. Bartolini, *Watermarking Systems Engineering: Enabling Digital Assets Security and Other Applications*, Marcel Dekker, New York, NY, USA, 2004.

[55] I. J. Cox, M. L. Miller, and J. A. Bloom, *Digital Watermarking*, Morgan Kaufmann, San Francisco, Calif, USA, 2001.

[56] I. J. Cox, J. Kilian, F. T. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Transactions on Image Processing*, vol. 6, no. 12, pp. 1673–1687, 1997.

[57] M. Barni and F. Bartolini, "Data hiding for fighting piracy," *IEEE Signal Processing Magazine*, vol. 21, no. 2, pp. 28–39, 2004.

[58] S. Emmanuel and M. Kankanhalli, "Copyright protection for MPEG-2 compressed broadcast video," in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME '01)*, pp. 206–209, Tokyo, Japan, August 2001.

[59] J. Crowcroft, C. Perkins, and I. Brown, "A method and apparatus for generating multiple watermarked copies of an information signal," WO Patent No. 00/56059, 2000.

[60] R. Parviainen and P. Parnes, "Large scale distributed watermarking of multicast media through encryption," in *Proceedings of the IFIP TC6/TC11 International Conference on Communications and Multimedia Security Issues of the New Century*, vol. 192, pp. 149–158, Darmstadt, Germany, May 2001.

[61] D. Kundur and K. Karthik, "Video fingerprinting and encryption principles for digital rights management," *Proceedings of the IEEE*, vol. 92, no. 6, pp. 918–932, 2004.

[62] A. Lemma, S. Katzenbeisser, M. Celik, and M. van der Veen, "Secure watermark embedding through partial encryption," in *Proceedings of the 5th International Workshop on Digital Watermarking (IWDW '06)*, vol. 4283 of *Lecture Notes in Computer Science*, pp. 433–445, Jeju Island, Korea, November 2006.

[63] R. J. Anderson and C. Manifavas, "Chameleon—a new kind of stream cipher," in *Proceedings of the 4th International Workshop on Fast Software Encryption (FSE '97)*, vol. 1267, pp. 107–113, Springer, Haifa, Israel, January 1997.

[64] A. Adelsbach, U. Huber, and A.-R. Sadeghi, "Fingercasting—joint fingerprinting and decryption of broadcast messages," in *Proceedings of the 11th Australasian Conference on Information Security and Privacy (ACISP '06)*, vol. 4058 of *Lecture Notes in Computer Science*, pp. 136–147, Springer, Melbourne, Australia, July 2006.

[65] M. Celik, A. Lemma, S. Katzenbeisser, and M. van der Veen, "Secure embedding of spread spectrum watermarks using look-up tables," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP '07)*, vol. 2, pp. 153–156, IEEE Press, Honolulu, Hawaii, USA, April 2007.

[66] N. Memon and P. W. Wong, "A buyer-seller watermarking protocol," *IEEE Transactions on Image Processing*, vol. 10, no. 4, pp. 643–649, 2001.

[67] F. Ahmed, F. Sattar, M. Y. Siyal, and D. Yu, "A secure watermarking scheme for buyer-seller identification and copyright protection," *EURASIP Journal on Applied Signal Processing*, vol. 2006, Article ID 56904, 15 pages, 2006.

[68] M. Kuribayashi and H. Tanaka, "Fingerprinting protocol for images based on additive homomorphic property," *IEEE Transactions on Image Processing*, vol. 14, no. 12, pp. 2129–2139, 2005.

[69] C.-L. Lei, P.-L. Yu, P.-L. Tsai, and M.-H. Chan, "An efficient and anonymous buyer-seller watermarking protocol," *IEEE Transactions on Image Processing*, vol. 13, no. 12, pp. 1618–1626, 2004.

[70] J. Zhang, W. Kou, and K. Fan, "Secure buyer-seller watermarking protocol," *IEE Proceedings—Information Security*, vol. 153, no. 1, pp. 15–18, 2006.

[71] J. J. Eggers, J. K. Su, and B. Girod, "Public key watermarking by eigenvectors of linear transforms," in *Proceedings of the European Signal Processing Conference (EUSIPCO '00)*, Tampere, Finland, September 2000.

[72] T. Furon and P. Duhamel, "An asymmetric public detection watermarking technique," in *Proceedings of the 3rd International Workshop on Information Hiding (IH '99)*, vol. 1768 of

Lecture Notes in Computer Science, pp. 88–100, Springer, Dresden, Germany, September-October 2000.

[73] J. J. Eggers, J. K. Su, and B. Girod, "Asymmetric watermarking schemes," in *Proceedings of the Sicherheit in Mediendaten, GMD Jahrestagung*, Berlin, Germany, September 2000.

[74] S. A. Craver and S. Katzenbeisser, "Security analysis of public-key watermarking schemes," in *Mathematics of Data/Image Coding, Compression, and Encryption IV, with Applications*, vol. 4475 of *Proceedings of SPIE*, pp. 172–182, San Diego, Calif, USA, July 2001.

[75] S. Craver, "Zero knowledge watermark detection," in *Proceedings of the 3rd International Workshop on Information Hiding (IH '99)*, vol. 1768 of *Lecture Notes in Computer Science*, pp. 101–116, Springer, Dresden, Germany, September-October 1999.

[76] A. Adelsbach and A.-R. Sadeghi, "Zero-knowledge watermark detection and proof of ownership," in *Proceedings of the 4th International Workshop on Information Hiding (IH '01)*, vol. 2137 of *Lecture Notes in Computer Science*, pp. 273–288, Springer, Pittsburgh, Pa, USA, April 2001.

[77] A. Adelsbach, M. Rohe, and A.-R. Sadeghi, "Non-interactive watermark detection for a correlation-based watermarking scheme," in *Proceedings of the 9th IFIP TC-6 TC-11International Conference on Communications and Multimedia Security (CMS '05)*, vol. 3677 of *Lecture Notes in Computer Science*, pp. 129–139, Springer, Salzburg, Austria, September 2005.

[78] I. Damgård and E. Fujisaki, "A statistically-hiding integer commitment scheme based on groups with hidden order," in *Proceedings of the 8th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT '02)*, Y. Zheng, Ed., vol. 2501 of *Lecture Notes in Computer Science*, pp. 125–142, Springer, Queenstown, New Zealand, December 2002.

[79] J. R. Troncoso-Pastoriza and F. Pérez-González, "Efficient non-interactive zero-knowledge watermark detector robust to sensitivity attacks," in *Security,Steganography, and Watermarking of Multimedia Contents IX*, P. W. Wong and E. J. Delp, Eds., vol. 6505 of *Proceedings of SPIE*, pp. 1–12, San Jose, CA, USA, January 2007.

[80] M. Malkin and T. Kalker, "A cryptographic method for secure watermark detection," in *Proceedings of the 8th International Workshop on Information Hiding (IH '06)*, vol. 4437 of *Lecture Notes in Computer Science*, pp. 26–41, Springer, Alexandria, Va, USA, July 2006.

[81] A. Piva, V. Cappellini, D. Corazzi, A. De Rosa, C. Orlandi, and M. Barni, "Zero-knowledge ST-DM watermarking," in *Security, Steganography, and Watermarking of Multimedia Contents VIII*, vol. 6072 of *Proceedings of SPIE*, pp. 291–301, San Jose, Calif, USA, January 2006.

Special Issue on Advances in Quality and Performance Assessment for Future Wireless Communication Services

Call for Papers

Wireless communication services are evolving rapidly in tandem with developments and vast growth of heterogeneous wireless access and network infrastructures and their potential. Many new, next-generation, and advanced future services are being conceived. New ideas and innovation in performance and QoS, and their assessment, are vital to the success of these developments. These should be open and transparent, with not only network-provider-driven but also service-provider-driven and especially user-driven, options on management and control to facilitate always best connected and served (ABC&S), in whatever way this is perceived by the different stake holders. To wireless communication services suppliers and users, alike the complexity and integrability of the immense, diverse, heterogeneous wireless networks' infrastructure should add real benefits and always appear as an attractive user-friendly wireless services enabler, as a wireless services performance enhancer and as a stimulant to wireless services innovation. Effecting the integration of services over a converged IP platform supported by this diverse and heterogeneous wireless infrastructure presents immense QoS and traffic engineering challenges. Within this context, a special issue is planned to address questions, advances, and innovations in quality and performance assessment in heterogeneous wireless service delivery.

Topics of interest include, but are not limited to:

- Performance evaluation and traffic modelling
- Performance assessments and techniques at system/flow level, packet level, and link level
- Multimedia and heterogeneous service integration-performance issues, tradeoffs, user-perceived QoS, and quality of experience
- Network planning; capacity; scaling; and dimensioning
- Performance assessment, management, control, and solutions: user-driven; service-provider-driven; network-provider-driven; subscriber-centric and consumer-centric business model dependency issues
- Wireless services in support of performance assessment, management, and control of multimedia service delivery
- Performance management and assessment in user-driven live-access network change and network-driven internetwork call handovers
- Subscriber-centric and consumer-centric business model dependency issues for performance management, control, and solutions
- Simulations and testbeds

Before submission, authors should carefully read over the journal's Author Guidelines, which are located at http://www.hindawi.com/journals/wcn/guidelines.html. Prospective authors should submit an electronic copy of their complete manuscript through the journal Manuscript Tracking System at http://mts.hindawi.com/, according to the following timetable:

Manuscript Due	August 1, 2009
First Round of Reviews	November 1, 2009
Publication Date	February 1, 2010

Lead Guest Editor

Máirtín O'Droma, Telecommunications Research Centre, University of Limerick, Ireland; mairtin.odroma@ul.ie

Guest Editors

Markus Rupp, Institute of Communications and Radio-Frequency Engineering, Vienna University of Technology, Gusshausstrasse 25/389, 1040 Vienna, Austria; mrupp@nt.tuwien.ac.at

Yevgeni Koucheryavy, Department of Communication Engineering, Tampere University of Technology, Korkeakoulunkatu 10, 33720 Tampere, Finland; yk@cs.tut.fi

Andreas Kassler, Computer Science Department, University of Karlstad, Universitetsgatan, 65188 Karlstad, Sweden; kassler@ieee.org

Special Issue on Enhancing Privacy Protection in Multimedia Systems

Call for Papers

The right to privacy has long been regarded as one of the basic universal human rights. In the last thirty years, advances in computing technologies have brought dramatic improvement in collecting storing and sharing personal information among government agencies and private sectors. The combination of ubiquitous sensors, wireless connectivity, and powerful recognition algorithms make it easier than ever to monitor every aspect of our daily activities. From the use of sophisticated pattern recognition in surveillance video to the theft of biometric signals and personal multimedia contents—people have become increasingly worried about the privacy of their multimedia data. To mitigate public concern about privacy violation, it is imperative to make privacy protection a priority in current and future multimedia systems.

Even though research on privacy enhancing technologies (PETs) began twenty years ago, most of the existing schemes focus on textual or categorical data and are inadequate to protect multimedia. The particular challenges include but are not limited to the difficulty in extracting semantic information for protection, the ability to apply cryptographic primitives to high data-rate multimedia streams, basic signal processing algorithms for protecting privacy without destroying the perceptual quality of the signal, and privacy models for governing and handling privacy rights in multimedia systems. In the last few years, there has been much exciting new theoretical and practical work to tackle these challenges by combining expertise from multimedia, pattern recognition, cryptography, and computer security. This work has the potential of not only providing enhanced level of privacy, but also revolutionizing the research frontier in the fundamental studies of multimedia and security. The goal of this special issue is to collect cutting-edge research work in privacy protection technologies for multimedia, and to provide a high-quality forum for researchers from different areas to explore future opportunities in this area.

We seek submissions from academia and industry presenting novel research and field experiments on topics which include, but are not limited to:

- Privacy in multimedia database systems
- Privacy in multimodal biometric systems
- Privacy in multimodal surveillance systems
- Privacy in mobile multimedia systems
- Privacy preserving digital right management systems
- Privacy preserving feature extraction
- Privacy preserving pattern recognition
- Privacy threat and attack models
- Signal-based obfuscation
- Reversibility in signal-based obfuscation
- Signal processing in encrypted domains
- Subject identification for privacy protection
- Location and tracking privacy in multimedia signals
- Multimedia sensor protocols that preserve anonymity/ privacy
- Application of multimedia scrambling and data hiding for privacy protection
- Usability issues in privacy-protected multimedia systems
- Legality and economics of privacy in multimedia systems

Authors should follow the EURASIP Journal on Information Security manuscript format described at the journal site http://www.hindawi.com/journals/is/. Prospective authors should submit an electronic copy of their complete manuscript through the journal Manuscript Tracking System at http://mts.hindawi.com/ according to the following timetable:

Manuscript Due	March 1, 2009
First Round of Reviews	June 1, 2009
Publication Date	September 1, 2009

Guest Editors

Sen-ching Samson Cheung, Center for Visualization and Virtual Environments, Department of Electrical and Computer Engineering, University of Kentucky, Lexington, KY 40507, USA; sccheung@ieee.org

Deepa Kundur, Electrical and Computer Engineering Department, Texas A&M University, College Station, TX 77843, USA; deepa@ece.tamu.edu

Andrew Senior, Google Inc., NY 10011, USA; andrewsenior@google.com