Efficient Pointwise and Blockwise Encrypted Operations

Tiziano Bianchi
Department of Electronics and Telecommunications
University of Florence
Via S. Marta 3, 50139 Firenze, Italy
bianchi@lci.det.unifi.it

Alessandro Piva
Department of Electronics and Telecommunications
University of Florence
Via S. Marta 3, 50139 Firenze, Italy
piva@lci.det.unifi.it

Mauro Barni
Department of Information Engineering
University of Siena
Via Roma 56, 53100, Siena Italy
barni@dii.unisi.it

ABSTRACT

Signal processing tools working directly on encrypted data could provide an efficient solution to application scenarios where sensitive signals must be protected from an untrusted processing device. In this paper, we investigate an issue usually neglected in the proposed solutions for secure processing in the encrypted domain, that is the data expansion from the plaintext to the encrypted representation of signals, due to the use of cryptosystems operating on very large algebraic structures. A packed signal representation is proposed that allows to speed up pointwise operations on encrypted signal via parallel processing and to reduce the size of the whole encrypted signal.

Categories and Subject Descriptors

E.1 [**Data Structures**]: Arrays; E.2 [**Data Storage Representations**]: Object representation; K.4.1 [**Computers and Society**]: Public Policy Issues—*Intellectual property rights, Privacy*

General Terms

Security

Keywords

Packed representation of signals, homomorphic encryption, signal processing in the encrypted domain

1. INTRODUCTION

Most of the currently available solutions for the secure processing of signals simply apply cryptographic tools to build a secure layer on top of the signal processing modules, able to protect them from leakage of critical information against third parties not authorized to access the data. In many cases, though, the security requirements concern also the lack of trust between the data owner and the processing actors. The availability of signal processing tools that work directly on encrypted data represents a valid solution for all the applications where sensitive signals must be processed by non trusted entities. The literature on this new research field, referred to as secure signal processing or signal processing in the encrypted domain (s.p.e.d.), already proposed some solutions regarding content retrieval and content protection [5, 2]. Although the security requirements of these applications differ quite strongly, similar signal processing and cryptographic assumptions can be identified to make possible the manipulation of signals when they are in encrypted form.

First of all, encryption is applied independently on individual signal samples. This means that individual samples can be identified in the encrypted version of the signal, allowing for processing of encrypted signals on a sample-by-sample basis. This solution does not allow to explicitly hide the temporal or spatial structure of the signal; however, the use of sophisticated encryption schemes that are *semantically secure* (as the one in [3]) satisfies this property automatically.

Secondly, the encryption is carried out by resorting to *public* key cryptosystems that have particular *homomorphic* properties. For an exact definition of an homomorphic cryptosystem, let us first introduce some notations.

Given a set of possible plain texts $\mathcal{M}$, a set of cipher texts $\mathcal{C}$ and a key pair $\{pk, sk\}$ (public key and secret key), a public key encryption scheme is a couple of functions $E_{pk} : \mathcal{M} \rightarrow \mathcal{C}, D_{sk} : \mathcal{C} \rightarrow \mathcal{M}$ such that, given a plaintext $m \in \mathcal{M}$, $D_{sk}(E_{pk}(m)) = m$ and such that, given a cipher text $c \in \mathcal{C}$, it is computationally unfeasible to determine m such that $E_{pk}(m) = c$, without knowing the secret key sk.

The homomorphic property allows to carry out additions or multiplications on signal values in the encrypted domain. In the following, we will assume that a cryptosystem which is *homomorphic* with respect to the addition is adopted, i.e., there exists a map between an addition in the plaintext domain and an operation in the ciphertext domain, that usually is a modular multiplication. This means that given two plaintexts m_1 and m_2, the following equalities are then satisfied:

$$D_{sk}(E_{pk}(m_1) \cdot E_{pk}(m_2)) = m_1 + m_2 \quad (1)$$

and, as a consequence,

$$D_{sk}(E_{pk}(m)^a) = am \quad (2)$$

where a is a public integer and

$$D_{sk}(E_{pk}(m_1) \cdot E_{pk}(m_2)^{-1}) = m_1 - m_2. \quad (3)$$

MM&Sec'08, September 22–23, 2008, Oxford, United Kingdom.

The adoption of an additive homomorphic cryptosystem enables additions, subtractions and multiplications with a known (non-encrypted) value to be performed on encrypted signal samples. Thus, homomorphic encryption schemes are building blocks for many s.p.e.d. applications, since they allow to perform linear computations on encrypted data.

Moreover, the cryptosystem adopted for the encryption of signal samples need to be *probabilistic*, that is, given two encrypted values it is not possible to decide if they conceal the same value. This is fundamental, since the alphabet to which the input signal samples belong is usually limited. A widely adopted homomorphic and probabilistic cryptosystem is the Paillier cryptosystem [3], for which the operator $\phi(\cdot,\cdot)$ is a modular multiplication.

Finally, the Paillier cryptosystem and all the other public key cryptographic systems used for s.p.e.d. operations operate on very large algebraic structures. This means that signal amplitudes that were originally represented in 8 to 16 bits, will require at least 512 or 1024 bits per signal sample in their encrypted form. Though data expansion is not considered in literature, except for few examples (e.g. [4]), it may be an important impediment for practical solutions of s.p.e.d. operations.

It is the aim of this paper to discuss the issue of data expansion due to the adoption of cryptosystems operating on very large algebraic structures. In particular, a packed signal representation will be proposed that allows to speed up pointwise operations on encrypted signal via parallel processing and to reduce the size of the whole encrypted signal. Due to the linearity of the proposed representation, only linear operations on the encrypted packed data will be allowed.

2. PACKED SIGNAL REPRESENTATION

Let us consider an integer valued signal $a(n) \in \mathbb{Z}$, satisfying $|a(n)| \leq Q$, where Q is a positive integer. Given a couple of positive integers B, R, we will define the *packed* representation of $a(n)$ having order R and base B as

$$a_P(k) = \sum_{i=0}^{R-1} a_i(k) B^i \tag{4}$$

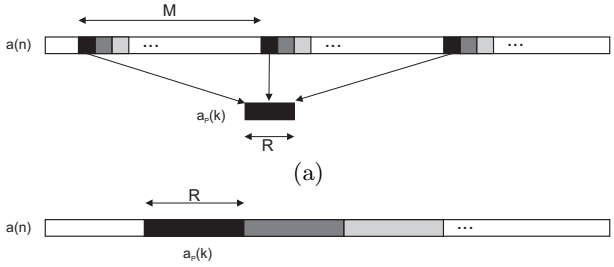
where $a_i(k)$, $i = 0, 1, \ldots, R-1$ indicate R disjoint subsequences of the signal $a(n)$. $a_P(k)$ represent a word where we can pack R signal samples, chosen according to a partitioning of the original signal samples $a(n)$. In the following, we will consider two partitions: 1) $a_i(k) = a(iM + k)$; 2) $a_i(k) = a(kR + i)$. The first case corresponds to considering the Mth order polyphase components of signal $a(n)$, i.e., dividing the signal in M periodically interleaved subsequences, and will be referred to as M-polyphase packed representation (M-PPR). The second case corresponds to partitioning the signal $a(n)$ in adjacent blocks having size R and will be referred to as block packed representation (BPR). A graphical interpretation of both M-PPR and BPR is provided in Fig. 1.

In order to use the packed representation for the parallel processing of an encrypted signal, we must first establish some properties. These are given by the following theorem:

THEOREM 1. *Let us assume*

$$B > 2Q \tag{5}$$

$$B^R \leq N \tag{6}$$



(b)

Figure 1: Graphical representation of a packed representation having order R: (a) M-polyphase packed representation; (b) block packed representation. Identically shaded boxes indicate values belonging to the same packed word.

where N is a positive integer. Then, the following holds:

$$0 \leq a_P(k) + \omega_Q < N \tag{7}$$

where $\omega_Q = Q \sum_{i=0}^{R-1} B^i = Q \frac{B^R - 1}{B-1}$. Moreover, the original samples can be obtained from the packed representation as

$$a_i(k) = \left\{ [a_P(k) + \omega_Q] \div B^i \right\} \mod B - Q. \tag{8}$$

Proof: let us express

$$a_p(k) + \omega_Q = \sum_{j=0}^{R-1} [a_j(k) + Q] B^j. \tag{9}$$

Thanks to the properties of $a(n)$ and (5), we have $0 \leq a_j(k) + Q \leq 2Q \leq B - 1$. Hence, $a_p(k) + \omega_Q$ can be considered as a positive base-B integer whose digits are given by $a_j(k) + Q$. Moreover, since $a_p(k) + \omega_Q$ has R digits, it is bounded by

$$a_p(k) + \omega_Q \leq \sum_{j=0}^{R-1} (B-1) B^j = B^R - 1 < N \tag{10}$$

where the last inequality comes from (6).

As to the second part of the theorem, let us express

$$a_p(k) + \omega_Q = B^i \sum_{j=i}^{R-1} [a_j(k) + Q] B^{j-i} + \sum_{j=0}^{i-1} [a_j(k) + Q] B^j. \tag{11}$$

Thanks to the properties of $a_j(k)+Q$, we have $\sum_{j=0}^{i-1} [a_j(k)+Q]B^j \leq B^i - 1$. Hence

$$[a_p(k) + \omega_Q] \div B^i = \sum_{j=i}^{R-1} [a_j(k) + Q] B^{j-i}$$

$$= B \sum_{j=i+1}^{R-1} [a_j(k) + Q] B^{j-i-1} + a_i(k) + Q$$

from which (8) is demonstrated.

The first part of the theorem means that $a_P(k)$ takes no more than N distinct values, so that its value can be represented modulo N without loss of information. Hence, as long as the hypotheses of the theorem hold, the packed representation can be safely encrypted using a cryptosystem defined on modulo N arithmetic.

The second part of the theorem ensures that the original signal values can be correctly computed from the packed representation. If the signal has been encrypted samplewise using an additive homomorphic cryptosystem, the conversion from samplewise to packed representation can be done in the encrypted domain by exploiting the homomorphic properties. However, the conversion from packed to samplewise representation requires rounding and division, which can not be carried out in the encrypted domain by means of homomorphic computations. Then unpacking has to be carried out by the data owner, or performed interactively.

3. EFFICIENT POINTWISE OPERATIONS

A generic packed representation can be used to speed up pointwise operations on encrypted signal via parallel processing. In the following, we will describe two applications regarding signal scaling and sum or difference of two signals.

3.1 Signal Scaling

Signal scaling is defined as the multiplication of the samples of the signal by a constant scaling factor, i.e.

$$c(n) = Ka(n). \tag{12}$$

In the following, we will assume $K \in \mathbb{Z}$.

We will define the block scaling of a signal as

$$c_P(k) = Ka_P(k). \tag{13}$$

PROPOSITION 2. *If $B > 2KQ$, then $c(n)$ can be exactly computed from the modulo N representation of $c_P(n)$.*

Proof: let us express $c_P(n) = \sum_{i=0}^{R-1}[Ka_i(k)]B^i$. It suffices to note that $|Ka_i(k)| \leq KQ$ and replace Q with KQ in the proof of Theorem 1.

3.2 Sum/Difference of two Signals

With sum/difference of two signals we denote either the sum or the difference of two signals operated sample by sample, i.e.

$$c(n) = a(n) \pm b(n). \tag{14}$$

In the following, we will assume $|b(n)| \leq Q$.

The block sum/difference of two signal is defined as

$$c_P(k) = a_P(k) \pm b_P(k) \tag{15}$$

where $b_P(k)$ is the packed representation of $b(n)$ obtained using the same partitioning rule as $a_P(k)$.

PROPOSITION 3. *If $B > 4Q$, then $c(n)$ can be exactly computed from the modulo N representation of $c_P(n)$.*

Proof: let us express $c_P(n) = \sum_{i=0}^{R-1}[a_i(k) + b_i(k)]B^i$. It suffices to note that $a_i(k)$ and $b_i(k)$ correspond to the same index n in the original sequences (same partitioning rule), $|a_i(k) + b_i(k)| \leq 2Q$ and replace Q with $2Q$ in the proof of Theorem 1.

4. BLOCKWISE TRANSFORMS

Given a finite length signal having size M, a linear transform can be defined by the following relationship

$$y(r) = \sum_{n=0}^{M-1} T_{nr}x(n) \quad r = 0, 1, \ldots, M-1 \tag{16}$$

where T_{nr} are a set of coefficients defining the particular transform. In the following, we will assume that the transform coefficients have been quantized according to some rule, i.e., $T_{nr} \in \mathbb{Z}$, and that $|T_{nr}| < Q_T$.

If the transform is implemented according to (16), the transformed signal can be bounded as $|y(r)| \leq MQ_TQ$. However, several practical transforms can be factorized by relying on the properties of the T_{nr}s (see, for example, the DFT). Such factorizations usually lead to a faster implementation, permitting to compute the whole transform as a series of smaller and very simple elementary transforms linked together by suitable scaling factors. Depending on the quantization of the intermediate steps, the final bound on the transformed signal is usually higher than for the direct implementation. Hence, in the following we will assume $|y(r)| \leq Q_S$, where Q_S is an upper bound that should be computed according to the particular implementation.

Usually, a transform is applied to the whole signal. However, when the size of a signal is not specified a-priori or it is very large, it is customary to partition the signal in adjacent blocks having some predetermined size and apply the transform to each block separately. This is the case, for example, of audio and image coding.

In our case, given the signal $a(n)$ of unspecified size, we define its blockwise transform as

$$c_i(r) = \sum_{n=0}^{M-1} T_{nr}a(iM + n) \quad r = 0, 1, \ldots, M-1. \tag{17}$$

Since the transform has a repeated structure, it is suitable for a parallel implementation.

We define the equivalent parallel blockwise transform as

$$c_P(r) = \sum_{k=0}^{M-1} T_{kr}a_P(k) \quad r = 0, 1, \ldots, M-1. \tag{18}$$

where $a_P(k)$ is the M-PPR of $a(n)$.

PROPOSITION 4. *If $B > 2Q_S$, then $c_i(r)$, $i = 0, 1, \ldots, R-1$, can be exactly computed from the modulo N representation of $c_P(r)$.*

Proof: let us consider the following equivalences

$$\begin{aligned} c_P(r) &= \sum_{k=0}^{M-1} T_{kr} \sum_{i=0}^{R-1} a(iM + k)B^i \\ &= \sum_{i=0}^{R-1} \left[\sum_{k=0}^{M-1} T_{kr}a(iM + k)\right] B^i \\ &= \sum_{i=0}^{R-1} c_i(r)B^i. \end{aligned} \tag{19}$$

Then, it suffices to note that $|c_i(r)| \leq Q_S$ and replace Q with Q_S in the proof of Theorem 1.

4.1 Extension to 2D Transforms

Given $r_1, r_2 = 0, 1, \ldots, M-1$, a blockwise 2D transform can be defined as

$$c_i(r_1, r_2) = \sum_{n_1=0}^{M-1} \sum_{n_2=0}^{M-1} T_{n_1r_1n_2r_2}a(p_iM + n_1, q_iM + n_2) \tag{20}$$

where we assumed a square $M \times M$ tiling of the original 2D signal and p_i, q_i define an indexing rule for the tiles.

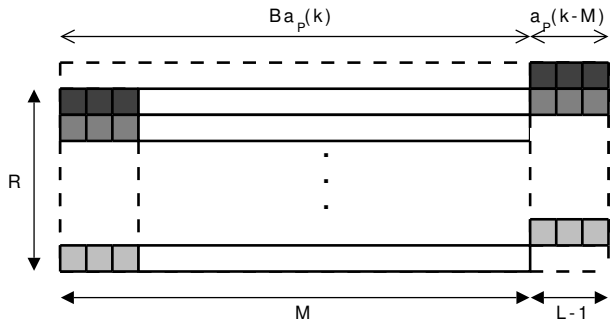


Figure 2: Graphical representation of $\tilde{a}_P(k)$. Shaded boxes indicate elements having the same value.

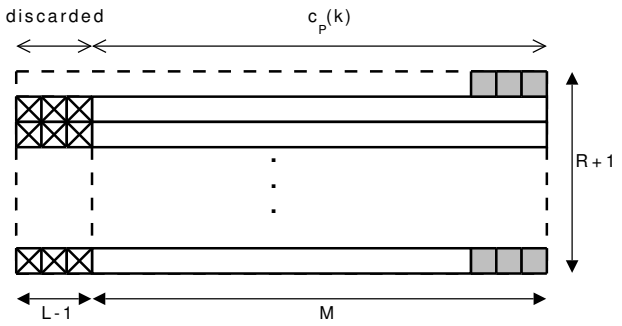


Figure 3: Graphical representation of $c_P(k)$. Blank boxes indicate valid convolution output values. Shaded boxes indicate convolution outputs at the boundaries. Crossed boxes indicate values that have to be discarded.

The parallel block transform can be easily extended to the 2D case as

$$c_P(r_1, r_2) = \sum_{k_1=0}^{M-1} \sum_{k_2=0}^{M-1} T_{n_1 r_1 n_2 r_2} a_P(k_1, k_2) \tag{21}$$

where $a_P(k_1, k_2) = \sum_{i=0}^{R-1} a(p_i M + k_1, q_i M + k_2) B^i$. Given $c_P(r_1, r_2)$, we can safely compute $c_i(r_1, r_2)$ provided that $B > 2Q_{S,2D}$. The demonstration follows the same lines as that of Proposition 4 and it has been omitted.

5. EFFICIENT CONVOLUTION

The output of a linear filter having impulse response $h(n)$ when the input is the sequence $x(n)$ is given by the convolution of two sequences, defined as

$$y(n) = \sum_{r=-\infty}^{\infty} h_r x(n-r). \tag{22}$$

In practical applications, the convolution algorithm is useful when the impulse response of the filter is finite. In our application, we consider a finite impulse response (FIR) filter of length L and we will assume that the input sequence $a(n)$ has length $P = RM$. Hence, we will consider the finite convolution

$$c(n) = \sum_{r=0}^{L-1} h_r a(n-r) \quad n = 0, 1, \ldots, P+L-2 \tag{23}$$

where we assume $h_r \in \mathbb{Z}$. Due to the properties of $a(n)$, we can bound the output as $|c(n)| \leq Q \sum_{r=0}^{L-1} |h_r| = Q_F$.

The proposed block convolution is defined as

$$c_P(k) = \sum_{r=0}^{L-1} h_r \tilde{a}_P(k-r) \quad k = 0, 1, \ldots, M+L-2 \tag{24}$$

where we define

$$\tilde{a}_P(k) = \begin{cases} Ba_P(k) & 0 \leq k < M \\ a_P(k-M) & M \leq k < M+L-1 \\ 0 & \text{elsewhere} \end{cases} \tag{25}$$

and $a_P(k)$ is the M-PPR of $a(n)$.

PROPOSITION 5. *If $B > 2Q_F$ and $B^{R+1} \leq N$, then $c(n)$ can be exactly computed from the modulo N representation of $c_P(k)$, $L-1 \leq k < M+L-1$, as*

$$c(iM+k) = \left\{ [c_P(k) + \omega_{Q_F}] \div B^{i+1} \right\} \mod B - Q_F. \tag{26}$$

where $\omega_{Q_F} = Q_F \sum_{i=0}^{R} B^i$.

Proof: let us consider $c_P(k)$. For $L-1 \leq k < M$ we have

$$\begin{aligned} c_P(k) &= \sum_{s=k-L+1}^{k} h_{k-s} \tilde{a}_P(s) \\ &= \sum_{s=k-L+1}^{k} h_{k-s} \sum_{j=0}^{R-1} a(jM+s) B^{j+1} \\ &= \sum_{s=k-L+1}^{k} h_{k-s} \sum_{i=1}^{R} a(iM-M+s) B^i \\ &= \sum_{i=1}^{R} \left[\sum_{s=k-L+1}^{k} h_{k-s} a(iM-M+s) \right] B^i \\ &= \sum_{i=1}^{R} c(iM-M+k) B^i \end{aligned} \tag{27}$$

whereas for $M \leq k < M+L-1$ we can write

$$\begin{aligned} c_P(k) &= \sum_{s=k-L+1}^{M-1} h_{k-s} \sum_{j=0}^{R-1} a(jM+s) B^{j+1} \\ &\quad + \sum_{s=M}^{k} h_{k-s} \sum_{i=0}^{R-1} a(iM-M+s) B^i \\ &= \sum_{s=k-L+1}^{M-1} h_{k-s} \sum_{i=1}^{R} a(iM-M+s) B^i \\ &\quad + \sum_{s=M}^{k} h_{k-s} \sum_{i=0}^{R-1} a(iM-M+s) B^i \\ &= \sum_{s=M}^{k} h_{k-s} a(-M+s) \\ &\quad + \sum_{i=1}^{R} \left[\sum_{s=k-L+1}^{k} h_{k-s} a(iM-M+s) \right] B^i \\ &= \sum_{s=M}^{k} h_{k-s} a(-M+s) + \sum_{i=1}^{R} c(iM-M+k) B^i. \end{aligned} \tag{28}$$

We have $|\sum_{s=M}^{k} h_{k-s}a(-M+s)| < Q_F$ and $|c_(k)| \leq Q_F$. Hence, the demonstration follows from the proof of Theorem 1 by replacing Q with Q_F and R with $R+1$.

Fig. 2 illustrates the disposition of the elements of the original sequence within $\tilde{a}_P(k)$. As can be seen, the particular concatenation of the packed words (columns) properly extends each of the R parallel subsequences (rows) at the boundaries. Fig. 3 shows the organization of the output values within $c_P(k)$. Note that the first $L-1$ packed words have to be discarded since they are affected by boundary effects.

5.1 Extension to 2D Convolutions

When a 2D signal $x(n_1, n_2)$ is considered, the corresponding 2D convolution can be defined as

$$y(n_1, n_2) = \sum_{r_1=-\infty}^{\infty} \sum_{r_2=-\infty}^{\infty} h_{r_1 r_2} x(n_1 - r_1, n_2 - r_2). \quad (29)$$

In the case of a 2D filter having finite support, the proposed block convolution can be also used to speed up 2D convolutions. The rationale is to apply the block convolution principle along one of the two dimensions. Let us consider the 2D signal $a(n_1, n_2)$ having size $P \times P$, $P = RM$, with $|a(n_1, n_2)| \leq Q$ and let us assume that $h_{r_1 r_2}$ has size $L \times L$. The 2D convolution is defined as

$$c(n_1, n_2) = \sum_{r_1=0}^{L-1} \sum_{r_2=0}^{L-1} h_{r_1 r_2} a(n_1 - r_1, n_2 - r_2). \quad (30)$$

where the output is bounded as:
$c(n_1, n_2) \leq Q \sum_{r_1=0}^{L-1} \sum_{r_2=0}^{L-1} |h_{r_1 r_2}| = Q_{F,2D}$.
Let us define the following packed representation

$$a_P(k_1, n_2) = \sum_{i=0}^{R-1} a_i(k_1, n_2) B^i \quad (31)$$

where $a_i(k_1, n_2) = a(iM + k_1, n_2)$. The block convolution can be extended to the 2D case as

$$c_P(k_1, n_2) = \sum_{r_1=0}^{L-1} \sum_{r_2=0}^{L-1} h_{r_1 r_2} \tilde{a}_P(k_1 - r_1, n_2 - r_2) \quad (32)$$

where $\tilde{a}_P(k_1, n_2)$ is defined as

$$\tilde{a}_P(k_1, n_2) = \begin{cases} B a_P(k_1, n_2) & 0 \leq k < M \\ a_P(k_1 - M, n_2) & M \leq k < M + L - 1 \\ 0 & \text{elsewhere.} \end{cases} \quad (33)$$

PROPOSITION 6. *If $B > 2Q_{F,2D}$ and $B^{R+1} \leq N$, then $c(n_1, n_2)$ can be exactly computed from the modulo N representation of $c_P(k_1, n_2)$.*

Proof: Let us express $c_P(k_1, n_2)$ as

$$c_P(k_1, n_2) = \sum_{r_2=0}^{L-1} \sum_{s=k_1-L+1}^{k_1} h_{k_1-s, r_2} \tilde{a}_P(s, n_2 - r_2). \quad (34)$$

By following the same derivations as in (27)-(28), it is easy to show that for $L-1 \leq k_1 < M$ we have

$$c_P(k_1, n_2) = \sum_{i=1}^{R} c(iM - M + k_1, n_2) B^i \quad (35)$$

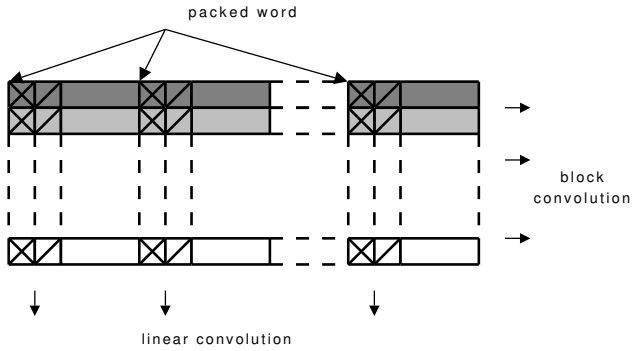


Figure 4: Packed representation for 2D convolution. Identically filled and shaded boxes indicate values belonging to the same packed word. 2D block convolution is obtained by applying the block convolution principle along one dimension.

whereas for $M \leq k_1 < M + L - 1$ we have

$$c_P(k_1, n_2) = \sum_{r_2=0}^{L-1} \sum_{s=M}^{k_1} h_{k_1-s, r_2} a(-M + s, n_2 - r_2) + \sum_{i=1}^{R} c(iM - M + k_1, n_2) B^i. \quad (36)$$

Then, the demonstration is analogous to that of Proposition 5 by noticing that $\sum_{r_2=0}^{L-1} \sum_{s=M}^{k_1} h_{k_1-s,r_2} a(-M+s, n_2 - r_2) \leq Q_{F,2D}$, $|c(n_1, n_2)| \leq Q_{F,2D}$, and replacing Q_F with $Q_{F,2D}$. A graphical interpretation of 2D block convolution is shown in Fig. 4.

6. PRACTICAL CONSIDERATIONS

The proposed packed representation permits to reduce both the computational complexity and the bandwidth usage of s.p.e.d. applications. As to block scaling, block sum/difference, and blockwise transforms, the packed representation allows us to parallely process R samples using a single s.p.e.d. operation. As to the block convolution, by computing a single output of (27) we obtain R output values of the original convolution. Therefore, the complexity of a block operation is reduced by a factor R with respect to that of the corresponding operation implemented sample-by-sample. Also the bandwidth usage is reduced by the same factor, since in all cases we pack R signal samples into a single cyphertext.

It is important to note that the constraints deriving from different applications usually lead to different choices of R. Let us consider an estimate of the number of samples that can be safely packed into a single word. A safe implementation requires at least $B = 2Q_Z + 1$, where Q_Z is a bound on the output of the computation. Since we must have $B < \sqrt[R]{N}$, this leads to

$$R_{max} = \left\lfloor \frac{\log_2 N}{\log_2(2Q_Z + 1)} \right\rfloor \approx \left\lfloor \frac{\lfloor \log_2 N \rfloor}{\log_2 Q_Z + 1} \right\rfloor = R_{U,Z}. \quad (37)$$

Moreover, the block convolution imposes a smaller B with respect to the other applications: since $R+1$ samples are to be packed in a single word, we have $R_{max}^{conv} \approx R_{U,Z} - 1$.

Table 1: Upper bounds on the number of blocks R that can be processed in parallel by a M-point s.p.e.d. block DFT. $R_{U,D}$ refers to direct DFT, $R_{U,R2}$ refers to radix-2 FFT. Q_T indicates the upper bound on quantized coefficients. We have assumed $\lfloor \log_2 N \rfloor = 1023$ and $Q = 128$.

	$Q_T = 2^7$		$Q_T = 2^{15}$		$Q_T = 2^{31}$	
M	$R_{U,D}$	$R_{U,R2}$	$R_{U,D}$	$R_{U,R2}$	$R_{U,D}$	$R_{U,R2}$
8	56	56	39	39	24	24
16	53	39	37	24	23	13
32	51	30	36	17	23	9
64	48	24	35	13	22	7
128	46	20	34	11	22	6
256	44	17	32	9	21	5
512	42	15	31	8	21	4
1024	40	13	30	7	20	3

6.1 A case study: block-wise DFT

In the following, we consider a simple case study related to the implementation of a blockwise one-dimensional DFT. Both a direct (D) DFT and a radix-2 (R2) FFT implementations are considered. As to the upper bound on the output values, we refer to the results in [1], namely

$$Q_D = M(QQ_T + Q/\sqrt{2} + Q_T/\sqrt{2} + 1/2) \tag{38}$$

for the DFT and

$$\begin{aligned} Q_{R2} =& MQQ_T^{\nu-2} + \frac{4}{\sqrt{2}}\left(2Q_T + \frac{1}{\sqrt{2}}\right)^{\nu-2} \\ &+ \sum_{k=0}^{\nu-3} \frac{2^{\nu-1-k}}{\sqrt{2}} QQ_T^{\nu-3-k}\left(2Q_T + \frac{1}{\sqrt{2}}\right)^{k} \end{aligned} \tag{39}$$

for the radix-2 FFT, where $M = 2^\nu$.

In Table 1, we give some values of $R_{U,Z}$ considering different sizes of the DFT and different upper bounds Q_T on the quantized coefficients. We have assumed 8-bit inputs, i.e., $Q = 128$, and a 1024-bit Paillier modulus.

The results demonstrate that the s.p.e.d. packed representation can effectively reduce both the bandwidth requirements and the complexity. Depending on the required precision and on the DFT size, the proposed approach allows us to process up to 50 DFTs in parallel when using a Paillier implementation satisfying standard security requirements. It is worth noting that a direct implementation allows us to increase $R_{U,Z}$ up to six times with respect to the FFT. Moreover, it is also possible that the complexity of the direct implementation is not higher than that of the fast implementation. For instance, an 32-point FFT requires about 160 multiplications versus the 1024 multiplications of a naive direct DFT. However, when $Q_T = 2^{31}$ the packed representation permits to compute 23 parallel DFTs versus 9 parallel FFTs. Hence, there can be cases in which it is preferable to employ a direct s.p.e.d. DFT, since this will reduce the bandwidth usage at a small cost in complexity.

7. CONCLUSIONS

In this paper, we have proposed a packed representation of signals that allows to group several signal samples into a single encrypted word and to perform basic linear operations on them. The aim of the packed representation is twofold: on the one hand, it permits to speed up computation via parallel processing; on the other hand, it reduces the size of the encrypted signals, having beneficial effects on both storage and bandwidth usage.

We illustrate several ways of exploiting the packed representation to perform s.p.e.d. operations, from simple pointwise operations, like signal scaling and sample-by-sample summations, to multidimensional transforms and convolutions. The proposed implementations scale the computational complexity by a factor R, where R is the number of samples that can be safely packed into a single word. Moreover, also the encrypted signal size is reduced by the same factor. We proposed a simple case study showing that up to 50 parallel DFTs can be computed by relying on the packed representation and a Paillier cryptosystem with standard security requirements.

Our preliminary results show that the packed representation can be a viable solution to the problems of both data expansion and increased complexity arising from the processing of encrypted data. Such results also suggest some interesting problems to address in future research, like the comparison of different algorithms performing the same operation (e.g. DFT and FFT).

7.1 Acknowledgments

The work reported here has been funded in part by the European Community's Sixth Framework Programme under grant number 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.

8. REFERENCES

[1] T. Bianchi, A. Piva, and M. Barni. Implementing the discrete Fourier transform in the encrypted domain. In *Proc. of ICASSP 2008*, pages 1757–1760, Las Vegas, NV, USA, 2008.

[2] Z. Erkin, A. Piva, S. Katzenbeisser, R. L. Lagendijk, J. Shokrollahi, G. Neven, and M. Barni. Protection and retrieval of encrypted multimedia content: When cryptography meets signal processing. *EURASIP Journal on Information Security*, 2007:Article ID 78943, 20 pages, doi:10.1155/2007/78943, 2007.

[3] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Lecture Notes in Computer Science*, volume 1592, pages 223–238. Springer-Verlag, 1999.

[4] J.-R. T. Pastoriza, S. Katzenbeisser, M. Celik, and A. Lemma. A secure multidimensional point inclusion protocol. In *ACM Multimedia and Security Workshop—MMSEC'07*, pages 109–120. ACM Press New York, NY, USA, 2007.

[5] A. Piva and S. Katzenbeisser. Signal processing in the encrypted domain. *EURASIP Journal on Information Security*, 2007:Article ID 82790, 1 page, doi:10.1155/2007/82790, 2007.