Axel Mertens[1], Georgio Nicolas[1], Sergi Rovira[2]
[1] COSIC, KU Leuven, Leuven, Belgium
[2] WiSeCom, UPF, Barcelona, Spain
FHE.org 2024, Toronto, Canada, 24 Mar 2024

Convolution-friendly Image Compression in FHE

Our Contribution

- We propose a practical FHE-friendly image compression and processing pipeline.
- An image can be compressed and encrypted on the client-side, sent to a server which decompresses it homomorphically and then performs image processing in the encrypted domain before returning the encrypted result to the client.
- This pipeline is designed to be compatible with existing image-processing techniques in FHE, such as pixel-wise processing and convolutional filters.
- Using this technique, a high-definition (1024 × 1024) image can be homomorphically decompressed, processed with a convolutional filter and re-compressed in under 24.7s, while using 8GB memory.

Pipeline

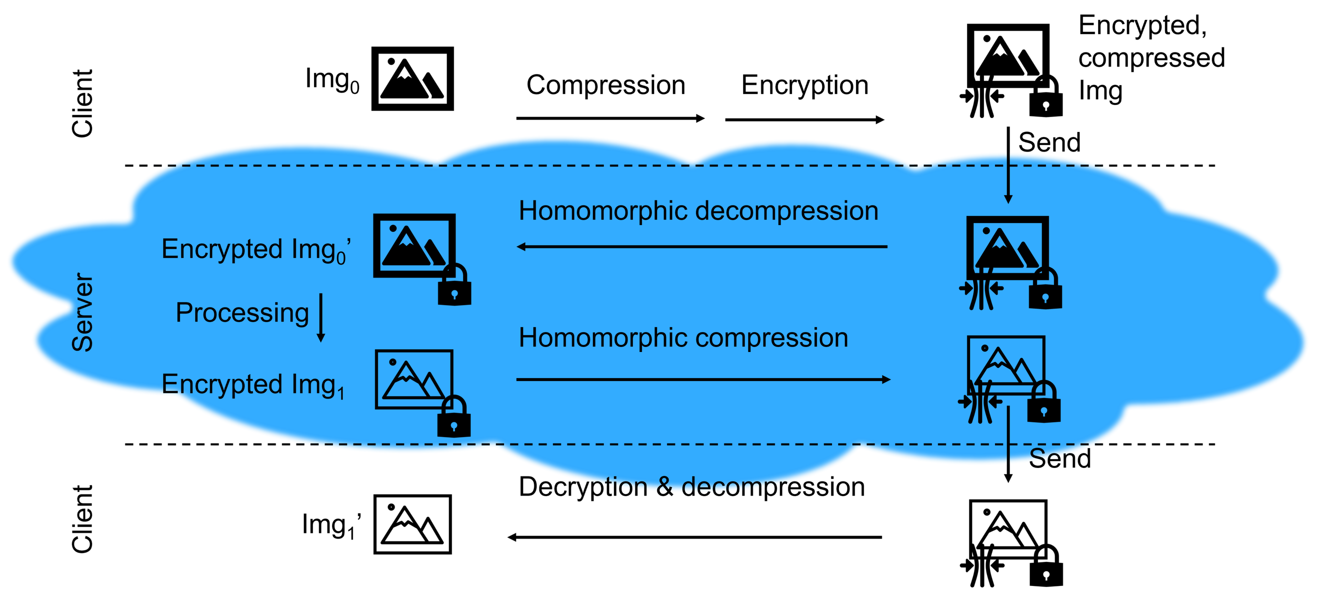

Figure 1: Sketch of our pipeline

Encoding

Algorithm 1 Encode

Input: Blocks $\mathbf{a} = [a_0, a_1, \ldots, a_{n-1}]$ with $|a_i| = l_a$
Input: Cutoff point c
Output: Blocks $\mathbf{b} = [b_0, b_1, \ldots, b_{n-1}]$ with $|b_i| = l_b$ and $l_a \geq l_b$
  for $i \in [0, n-1]$ **do**
    $b_i \leftarrow a_i[0 : c]$
  end for
  return b

Compress

Algorithm 2 Compress

Input: Encrypted image **A**: split into n $m \times m$ blocks: $\mathbf{A} = [A_0, A_1, \ldots, A_{n-1}]$ with $A_i \in \mathbb{Z}^{m\times m}$
Input: Cutoff point c
Output: Encrypted, compressed image $\mathbf{B} = [b_0, b_1, \ldots, b_{n-1}]$ with $b_i \in \mathbb{Z}^c$
  for $l \in [0, n-1]$ **do**
    $C_l \leftarrow Level(A_l)$
    $D_l \leftarrow T_m C_l T_m^\top$
    for $(i, j) \in \mathbb{N}_m^2$ **do**
      $E_{l,i,j} \leftarrow \lfloor(\frac{D_{l,i,j}}{Q_{m,i,j}})\rceil$
    end for
    $\mathbf{b}_l \leftarrow \text{ZigZag}(E_l)[0..c]$
  end for
  return $[b_0, b_1, \ldots, b_{n-1}]$

Decompress

Algorithm 3 Decompress

Input: Block size m, which can be equal to either 8 or 16
Input: Encrypted, compressed image $\mathbf{B} = [b_0, b_1, \ldots, b_{n-1}]$ with $b_i \in \mathbb{Z}^c$
Output: Encrypted image $\mathbf{A} = [A_0, A_1, \ldots, A_{n-1}]$ with $A_i \in \mathbb{Z}^{m\times m}$
  for $l \in [0, n-1]$ **do**
    for $i \in [c, m^2[$ **do** $b_{l,i} \leftarrow Enc(0)$
      $C_l \leftarrow \text{InverseZigZag}(b_l)$
    end for
    for $(i, j) \in \mathbb{N}_m^2$ **do**
      $D_{l,i,j} \leftarrow C_{l,i,j} \cdot Q_{l,i,j}$
    end for
    $E_l \leftarrow T_m^\top D_l T_m$
    $A_l \leftarrow \text{Unlevel}(E_l)$
  end for
  return $[A_0, A_1, \ldots, A_{n-1}]$

Packing

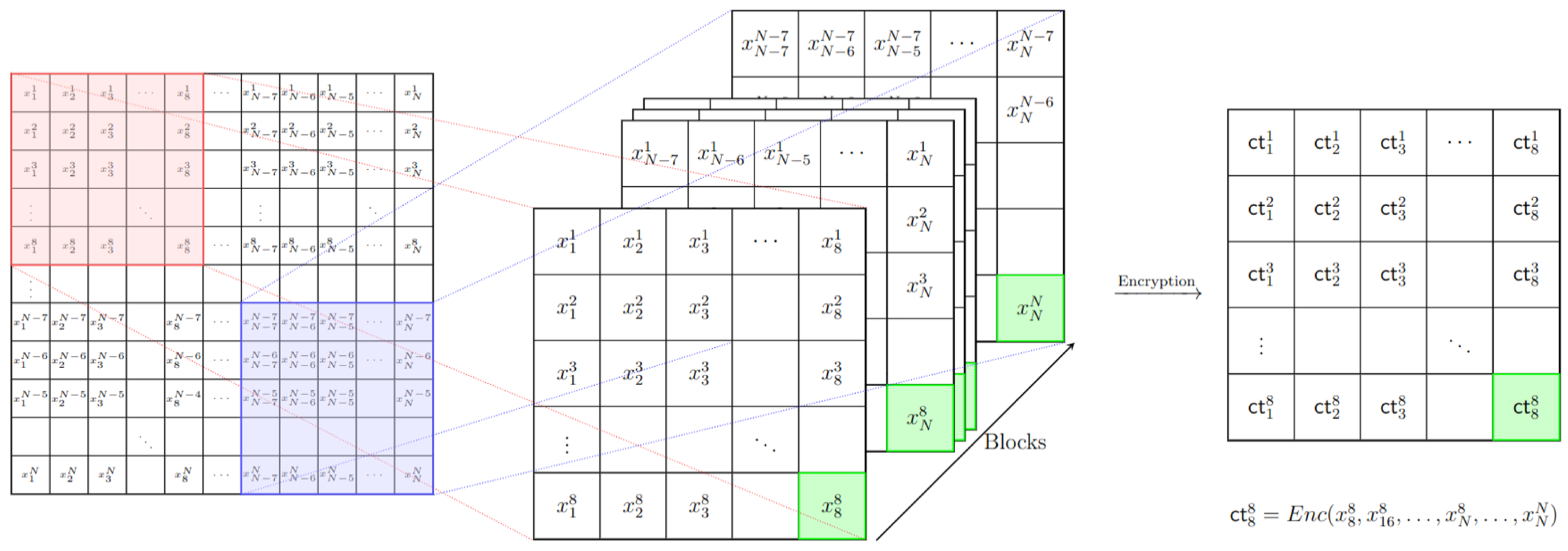

Figure 2: Illustration of our ciphertext packing

An Example: Edge Detection

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \longrightarrow$$

Experimental Results

We ran tests for four different image dimensions, and four algorithms each. The four algorithms are

1. Pixel-wise processing on 8 × 8 blocks.
2. Convolutional processing on 8 × 8 blocks.
3. Pixel-wise processing on 16 × 16 blocks.
4. Convolutional processing on 16 × 16 blocks.

image size	setting	decompression	processing	compression	total	SSI	compression ratio
256 × 256	1	8s	< 1s	5s	13s	0.95	100 : 34.4
256 × 256	2	11.3s	4s	9s	24.5s	0.935	100 : 83.3
252 × 252	3	39.5	1s	29s	69.5s	0.91	100 : 24.6
252 × 252	4	60s	19s	62s	131s	0.905	100 : 35.7
512 × 512	1	7.5s	< 1s	5s	13.5s	0.95	100 : 34.4
512 × 512	2	11s	4s	9s	24ss	0.935	100 : 83.3
504 × 504	3	39s	< 1s	29s	68s	0.92	100 : 24.6
504 × 504	4	50s	18.5s	62s	130.5s	0.92	100 : 35.7
1024 × 1024	1	8s	< 1s	5s	13s	0.98	100 : 34.4
1024 × 1024	2	11s	4s	9.7s	24.7s	0.975	100 : 83.3
1022 × 1022	3	41s	< 1s	29s	70s	0.96	100 : 24.6
1022 × 1022	4	61.5s	19s	62s	142.5s	0.97	100 : 35.7
2048 × 2048	1	107s	1s	70s	178s	0.98	100 : 34.4
2048 × 2048	2	107s	36.5s	79s	222.5s	0.98	100 : 83.3
2044 × 2044	3	137s	< 1s	99.5s	236.5s	0.975	100 : 24.6
2044 × 2044	4	136s	41.5s	136s	313.5s	0.975	100 : 35.7

Table 1: Experimental results (server-side). Each timing is for one 8-bit greyscale image of mentioned dimensions. The values are averaged over a few randomly selected images: 8 images of 512 × 512, 4 images for other dimensions.

Future directions

- Move towards more complex image processing, such as face recognition and object detection.
- An interesting question is whether it is possible to prove that entropy encoding is impractical in combination with FHE. If not, it should be possible to further improve the compression ratio we achieve.

Acknowledgement

This work was supported by CyberSecurity Research Flanders with reference number VR20192203, the FWO under an Odysseus project GOH9718N. The 3rd listed author is partially supported by the Spanish grant PID2019-110224RB-I00.

Contact

- axel.mertens@esat.kuleuven.be
- georgio.nicolas@esat.kuleuven.be
- sergi.rovira@upf.edu