Integrating Compression and Encryption based on Chaos Theory

Kwok-Wo Wong

Department of Electronic Engineering
City University of Hong Kong
## Content

| • Background |
| • Existing Approaches for Integrating Compression and Encryption |
| • Compression + Encryption using Chaos |
| • Conclusions |
Content

• Background
  – Cryptographic Standards
  – Compression + Encryption

• Existing Approaches for Integrating Compression and Encryption

• Compression + Encryption using Chaos

• Conclusions
Cryptographic Standards

• Private Key (Symmetric Key) Cryptography

  Confidential Message \rightarrow \text{Encryption} \rightarrow \text{Ciphertext} \rightarrow \text{Decryption} \rightarrow \text{Original Confidential Message}

  \text{private} \rightarrow \text{private}

• Block Ciphers
  – Data Encryption Standard (DES)
  – Advanced Encryption Standard (AES)
Cryptographic Standards

- Ciphertext length = Plaintext length

<table>
<thead>
<tr>
<th></th>
<th>Plaintext Block Length</th>
<th>Ciphertext Block Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>DES</td>
<td>64 bits</td>
<td>64 bits</td>
</tr>
<tr>
<td>AES</td>
<td>128 bits</td>
<td>128 bits</td>
</tr>
</tbody>
</table>

- No compression
- Not suitable for the direct encryption of bulk multimedia data (image / video)
- Need [ Compression + Encryption ]
Compression + Encryption

OR

Compression → Encryption

OR

Encryption → Compression
Compression First

Hex value of a JPEG file

Repeated Hex value

may generate repeated patterns in encrypted file
Compression First

- Example:

<table>
<thead>
<tr>
<th>Original Image</th>
<th>Encrypted image by Advanced Encryption Standard (AES) running in the electronic code book (ECB) mode</th>
</tr>
</thead>
</table>

8
Encryption First

Plain Lena image & its histogram

Cipher image & its histogram

Flat Histogram \rightarrow may not be compressed at all
Integrating Compression & Encryption

2 Different (if not opposite) Research Directions

- Key-dependent Control
- Entropy Information

Framework of Compression → Compression + Encryption → Framework of Encryption

(add)
Content

• Background

• Existing Approaches for Integrating Compression and Encryption
  – Based on Huffman Coding
  – Based on Arithmetic Coding

• Compression + Encryption using Chaos

• Conclusions
Based on the Framework of Compression

## Entropy Coding

- Make use of the entropy (average information) of source symbols.
- A statistical model is required.
- High-occurrence symbol $\rightarrow$ short code.
- Low-occurrence symbol $\rightarrow$ long code.

Reduced average code length $\rightarrow$ **Compression**
<table>
<thead>
<tr>
<th>Based on the Framework of Compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embedding Encryption into Entropy Coding</td>
</tr>
<tr>
<td>• Turn entropy coder into cryptographic cipher using multiple statistical models.</td>
</tr>
<tr>
<td>• Select a statistical model by the secret key.</td>
</tr>
<tr>
<td>• Incorrect key</td>
</tr>
<tr>
<td>✔️ Statistical model not matched</td>
</tr>
<tr>
<td>✔️ Wrong Decoding</td>
</tr>
</tbody>
</table>
Huffman Decoding

Example

```
0 1 1 1 0 0 1 1
```

```
a = 0
b = 11
c = 10
```

```
a = 1
b = 01
c = 00
```

```
0 1 1 1 0 0 1 1
```

```
a b c a b
```

```
b a a c a a
```

```
0 1 1 1 0 0 1 1
```

```
0 1 1 1 0 0 1 1
```

```
a b c a b
```

```
b a a c a a
```
Multiple Huffman Table Approach


- Use a secret key to
  - Choose a table from a set of 4 basic Huffman coding tables (trees)
  - Mutate the chosen tree
Multiple Huffman Table Approach

4 Basic Huffman Trees

Tree Mutation

Key

Mutate
## Multiple Huffman Table Approach

- The mutation process does not change the structure of the basic Huffman table, but only swaps the bit 0 and bit 1 in the Huffman code.

- The codeword length remains unchanged.

  ➔ favor chosen-plaintext attack.

Randomized Arithmetic Coding


- The intervals may swap, controlled by the secret key.
Randomized Arithmetic Coding

- Secret key bit = 0  ➔  No swap
- Secret key bit = 1  ➔  Swap

\[ p(A) = \frac{2}{3} \quad p(B) = \frac{1}{3} \]

### Original Arithmetic Coding

<table>
<thead>
<tr>
<th>0</th>
<th>2/3</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>2/3</td>
<td></td>
</tr>
<tr>
<td>AA</td>
<td>AB</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>8/27</td>
<td>4/9</td>
</tr>
<tr>
<td>AAA</td>
<td>AAB</td>
<td></td>
</tr>
</tbody>
</table>

### Randomized Arithmetic Coding

<table>
<thead>
<tr>
<th>0</th>
<th>2/3</th>
<th>1</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>2/3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AB</td>
<td>AA</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>2/9</td>
<td>14/27</td>
<td>2/3</td>
<td></td>
</tr>
<tr>
<td>AAA</td>
<td>AAB</td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>
Key-based Interval Splitting in Arithmetic Coding


- Example:

\[ p(A) = \frac{2}{3} \quad p(B) = \frac{1}{3} \]

1-symbol case:

Split key: 1

```
  0  2/3  1
  \begin{array}{c}
   A   \end{array}
\begin{array}{c}
 \begin{array}{c}
   B   \end{array}
\end{array}
```

```
  0  2/3  1
  \begin{array}{c}
   A   \end{array}
\begin{array}{c}
 \begin{array}{c}
   B   \end{array}
\end{array}
```

```
  0  2/3  1
  \begin{array}{c}
   A   \end{array}
\begin{array}{c}
   B   \end{array}
\begin{array}{c}
   A   \end{array}
```
Key-based Interval Splitting in Arithmetic Coding

\[ p(A) = \frac{2}{3} \quad p(B) = \frac{1}{3} \]

\[ p(\text{AA}) = \frac{4}{9} \quad p(\text{AB}) = \frac{2}{9} \quad p(\text{BA}) = \frac{2}{9} \quad p(\text{BB}) = \frac{1}{9} \]

<table>
<thead>
<tr>
<th>1 symbol</th>
<th>0</th>
<th>A</th>
<th>B</th>
<th>A</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>2 symbols</th>
<th>AA</th>
<th>BA</th>
<th>BB</th>
<th>AA</th>
<th>AB</th>
</tr>
</thead>
</table>

**split key**

21
## Content

- **Background**

- **Existing Approaches for Integrating Compression and Encryption**

- **Compression + Encryption using Chaos**
  - Based on the framework of Compression
  - Based on the framework of Cryptography

- **Conclusions**
## Compression + Encryption using Chaos

<table>
<thead>
<tr>
<th><strong>Based on the framework of Compression</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>- Joint work with Prof. Ron Chen and Dr. Yaobin Mao</td>
</tr>
<tr>
<td>- DCT ↔ lossy image compression</td>
</tr>
<tr>
<td>- Use a 2D chaotic map to permute the DCT coefficients</td>
</tr>
<tr>
<td>- Use 1D chaotic tent maps to mix the DCT coefficients and mask the Huffman coding sequence</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Based on the framework of Cryptography</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>- Baptista-type chaotic cryptosystem</td>
</tr>
<tr>
<td>- Incorporate entropy information in the search look-up table (the codebook) to achieve compression</td>
</tr>
</tbody>
</table>
Based on the Framework of Compression

- Key-dependent Control
- Framework of Compression
- Compression + Encryption
- Framework of Encryption
- Entropy Information

(add)
**Lossy Compression + Chaotic Cryptography**

1. Partition the plain-image into 8x8 blocks
2. Transform each block by Discrete Cosine Transform (DCT)
3. Permute the DCT-coefficients of all the blocks using 2D cat map
4. Mix the permuted DCT-coefficients to change the statistical structure in the transformed domain
5. Code the mixed and permuted DCT-coefficients using Dynamic Huffman Coding (DHC)
6. Mask the Huffman codes to form the final ciphertext
Encryption Process

101010101101001
011010101001000
1011001101001

DCT

2D cat map

30% mixing

110001010010010
010100011011001
00100110110101

cipher stream

XOR

variable-length Huffman code

DHC
Lossy Compression + Chaotic Cryptography

Plain Image

DCT
(8x8 blocks)

Quantized DCT
Coefficients $D_i$

2D Cat Map
(whole image)

Permutated DCT
Coefficients $D'_i$

Mixing

Integer Tent
Map 1

Integer Tent
Map 2

Dynamic
Huffman Coding

Coded-Stream
Masking

Cipher Stream

Secret Key

$k_1$

$a, b$

$k_2$

$k_3$

$k_3$

Secret Key
Encryption and Decryption Speed

512 x 512 Lena with 256 grey levels
## Encryption and Decryption Speed

<table>
<thead>
<tr>
<th>Image Size</th>
<th>Grey Levels</th>
<th>Encryption Time</th>
<th>Decryption Time</th>
<th>Highest Data Rate</th>
<th>Lowest Data Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>512 x 512 Lena</td>
<td>256</td>
<td>88.95 ms (no mixing) to 265.32 ms (100% mixing)</td>
<td>114.02 ms (no mixing) to 258.77 ms (100% mixing)</td>
<td>512 x 512 x 8 / 88.95 ms = 22.48 Mbps</td>
<td>512 x 512 x 8 / 265.32 ms = 7.54 Mbps</td>
</tr>
</tbody>
</table>
Encryption and Decryption Speed

2048 x 1536 24-bit true color image

![Graph showing time (seconds) vs. mixing percentage (η) for encoding and decoding processes.]
## Encryption and Decryption Speed

2048 x 1536 24-bit true color image

- **Encryption Time:**
  
  2.193 s (no mixing) to 5.258 s (100% mixing)

- **Decryption Time:**
  
  2.375 s (no mixing) to 4.833 s (100% mixing)

- **Highest Data Rate:**

  \[
  \frac{2048 \times 1536 \times 24}{2.193 \text{ s}} = 32.83 \text{ Mbps}
  \]

- **Lowest Data Rate:**

  \[
  \frac{2048 \times 1536 \times 24}{5.258 \text{ s}} = 13.69 \text{ Mbps}
  \]
Compression Ratio

No mixing

12.65 (True-color)

5.61 (Gray-scale)

100% mixing

1.999 (True-color)

0.998 (Gray-scale)
Transmission Scalability

- A measure of scalable coding
- Decrypt only the first fixed percentage of cipher stream

![Graph showing PSNR (dB) vs. Cipher stream decoding percentage (%) with key points at 10%, 50%, and 90% decoding.]
Key Sensitivity Test Cat Map Parameters \((a,b)\)

- in terms of Percentage of Different Bits

\((a,b) = (23,182)\) and then \((a,b) = (23,181)\)

<table>
<thead>
<tr>
<th>(\eta)</th>
<th>0%</th>
<th>10%</th>
<th>20%</th>
<th>30%</th>
<th>40%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Different Bits (%)</td>
<td>49.97</td>
<td>49.99</td>
<td>50.01</td>
<td>50.08</td>
<td>50.05</td>
<td>50.01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(\eta)</th>
<th>60%</th>
<th>70%</th>
<th>80%</th>
<th>90%</th>
<th>100%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Different Bits (%)</td>
<td>50.05</td>
<td>49.98</td>
<td>50.05</td>
<td>49.98</td>
<td>50.02</td>
</tr>
</tbody>
</table>
Key Sensitivity Test : Mixing Keys $k_1$ & $k_2$

- $k_1 = 0.1783929$ and then $k_1 = 0.1783928$
  
  Percentage of Different Bits = **50.11%**

- $k_2 = 0.2381211$ and then $k_2 = 0.2381212$
  
  Percentage of Different Bits = **50.01%**

- Both are very close to 50%.
FIPS 140-2 Randomness Tests

- 100 cipher stream segments, each of length 20,000 bits, are subject to the FIPS 140-2 randomness test.
- They all pass the test.

A typical set of test results ($\eta = 30\%$ )

<table>
<thead>
<tr>
<th>Test Type</th>
<th>Result</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monobit Test</td>
<td>9,916</td>
<td>$9,725 &lt; x &lt; 10,275$</td>
</tr>
<tr>
<td>Poker Test</td>
<td>8.79</td>
<td>$2.16 &lt; x &lt; 46.17$</td>
</tr>
<tr>
<td>Run=1</td>
<td>2,433</td>
<td>$2,315 &lt; x &lt; 2,685$</td>
</tr>
<tr>
<td>Run=2</td>
<td>1,343</td>
<td>$1,114 &lt; x &lt; 1,386$</td>
</tr>
<tr>
<td>Run=3</td>
<td>602</td>
<td>$527 &lt; x &lt; 723$</td>
</tr>
<tr>
<td>Run=4</td>
<td>306</td>
<td>$240 &lt; x &lt; 384$</td>
</tr>
<tr>
<td>Run=5</td>
<td>177</td>
<td>$103 &lt; x &lt; 209$</td>
</tr>
<tr>
<td>Run(\geq6)</td>
<td>144</td>
<td>$103 &lt; x &lt; 209$</td>
</tr>
<tr>
<td>Long Run Test (Run &gt; 25)</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Based on the Framework of Encryption

Key-dependent Control

Framework of Compression

Entropy Information

Compression + Encryption

Framework of Encryption
Baptista’s Chaotic Cryptosystem


- Follow a chaotic orbit to search in the multiple-partitioned state space (lookup-table, or codebook)

- The number of iterations required to reach the target plaintext symbol is the ciphertext.
Baptista’s Chaotic Cryptosystem

Logistic Map: $x_{n+1} = b x_n (1 - x_n)$

Secret Keys: $b=3.99 \ x_0=0.1$

Plaintext: GO!

145th iteration => Bingo

Generate a random number between 0 and 1.

If $>$ a preset threshold, take 145 as ciphertext.

Otherwise continue the iteration.

Hit again at 259th iteration.
Random number $>$ preset threshold.

Ciphertext: 259 344 127

G O !
Some Modifications by K.W. Wong

  - Use a changing look-up table to enhance the security and to increase the encryption speed.

  - The final look-up table depends on the plaintext, and can be considered as a hash.
# How to Compress?

- Original and modified Baptista-type chaotic cryptosystem: a byte of plaintext is encrypted as an integer (number of iterations).

- As the required number of iterations to reach the target is usually larger than 256, the ciphertext is usually longer than the plaintext (about 1.5 – 2 times).

- **Problem**: All plaintext symbols occupy the same area in the state space.

- **Idea**: High occurrence symbols occupy a larger area than low occurrence symbols.
Chaotic Cryptography + Compression

\[ \begin{align*}
\{ &p(A) = 1/2 \\
&\text{total 32 partitions for C} \\
&x = 1 \\
&x = 0.9961 \\
&x = 0.9922 \\
&x = 0.9883 \\
&x = 0.9844 \\
&\ldots \\
&x = 0.0156 \\
&x = 0.0117 \\
&x = 0.0078 \\
&x = 0.0039 \\
&x = 0
\end{align*} \]

\[ \begin{align*}
\{ &p(B) = 1/4 \\
&\text{total 64 partitions for B} \\
&x = 1 \\
&x = 0.9961 \\
&x = 0.9922 \\
&x = 0.9883 \\
&x = 0.9844 \\
&\ldots \\
&x = 0.0156 \\
&x = 0.0117 \\
&x = 0.0078 \\
&x = 0.0039 \\
&x = 0
\end{align*} \]

\[ \begin{align*}
\{ &p(C) = 1/8 \\
&\text{total 32 partitions for D} \\
&x = 1 \\
&x = 0.9961 \\
&x = 0.9922 \\
&x = 0.9883 \\
&x = 0.9844 \\
&\ldots \\
&x = 0.0156 \\
&x = 0.0117 \\
&x = 0.0078 \\
&x = 0.0039 \\
&x = 0
\end{align*} \]

\[ \begin{align*}
\{ &p(D) = 1/8 \\
&\text{total 128 partitions for A} \\
&x = 1 \\
&x = 0.9961 \\
&x = 0.9922 \\
&x = 0.9883 \\
&x = 0.9844 \\
&\ldots \\
&x = 0.0156 \\
&x = 0.0117 \\
&x = 0.0078 \\
&x = 0.0039 \\
&x = 0
\end{align*} \]
Chaotic Cryptography + Compression

- Scan the whole plaintext sequence once
- Select $N$ (e.g., 16) popular plaintext symbols.
- Generate the lookup-table (the codebook)
- If current symbol = one of the popular symbols,
  - encrypted by Baptista-type chaotic cryptosystem
  - use Huffman coding to encode the number of iterations.
- Otherwise, encrypt by masking plaintext block with a pseudo-random key stream generated by the chaotic map.
## Results

<table>
<thead>
<tr>
<th>File</th>
<th>Histogram</th>
<th>Percentage of non-masked symbols</th>
<th>Bit rate for 1-byte non-masked symbol</th>
<th>Ciphertext / Plaintext</th>
</tr>
</thead>
<tbody>
<tr>
<td>graph.bmp</td>
<td><img src="image1" alt="Histogram" /></td>
<td>97.14 %</td>
<td>1.510</td>
<td>22.81 %</td>
</tr>
<tr>
<td>data.xls</td>
<td><img src="image2" alt="Histogram" /></td>
<td>73.21 %</td>
<td>1.910</td>
<td>57.68 %</td>
</tr>
<tr>
<td>db1.mdb</td>
<td><img src="image3" alt="Histogram" /></td>
<td>71.95 %</td>
<td>1.856</td>
<td>62.23 %</td>
</tr>
</tbody>
</table>
## Results

<table>
<thead>
<tr>
<th>File</th>
<th>Histogram</th>
<th>Percentage of non-masked symbols</th>
<th>Bit rate for 1-byte non-masked symbol</th>
<th>Ciphertext / Plaintext</th>
</tr>
</thead>
<tbody>
<tr>
<td>document. doc</td>
<td><img src="image1.png" alt="Histogram" /></td>
<td>12.08 %</td>
<td>2.645</td>
<td>95.79 %</td>
</tr>
<tr>
<td>seminar. ppt</td>
<td><img src="image2.png" alt="Histogram" /></td>
<td>22.83 %</td>
<td>2.362</td>
<td>95.97 %</td>
</tr>
<tr>
<td>program. exe</td>
<td><img src="image3.png" alt="Histogram" /></td>
<td>3.41 %</td>
<td>3.065</td>
<td>101.91 %</td>
</tr>
</tbody>
</table>
## Results

<table>
<thead>
<tr>
<th>File</th>
<th>Histogram</th>
<th>Percentage of non-masked symbols</th>
<th>Bit rate for 1-byte non-masked symbol</th>
<th>Ciphertext / Plaintext</th>
</tr>
</thead>
<tbody>
<tr>
<td>document.pdf</td>
<td><img src="image1.png" alt="Histogram" /></td>
<td>10.67 %</td>
<td>3.312</td>
<td>103 %</td>
</tr>
<tr>
<td>logo.eps</td>
<td><img src="image2.png" alt="Histogram" /></td>
<td>20.57 %</td>
<td>3.328</td>
<td>106.71 %</td>
</tr>
<tr>
<td>music.wav</td>
<td><img src="image3.png" alt="Histogram" /></td>
<td>10.04 %</td>
<td>2.986</td>
<td>106.8 %</td>
</tr>
<tr>
<td>Content</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Background</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Existing Approaches for Integrating Compression and Encryption</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Compression + Encryption using Chaos</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Conclusions</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Conclusions

- Introduced the topic of integrating compression and encryption.
- Reviewed some approaches to embed encryption into compression.
- Based on the framework of compression, proposed a lossy image cryptosystem utilizing DCT and chaotic maps.
- Based on the framework of chaotic cryptography, suggested an approach to achieve compression in Baptista-type chaotic cryptosystem.
Thank You!

Q & A