Integrating Compression and Encryption based on Chaos Theory

Kwok-Wo Wong

Department of Electronic Engineering

City University of Hong Kong

Content

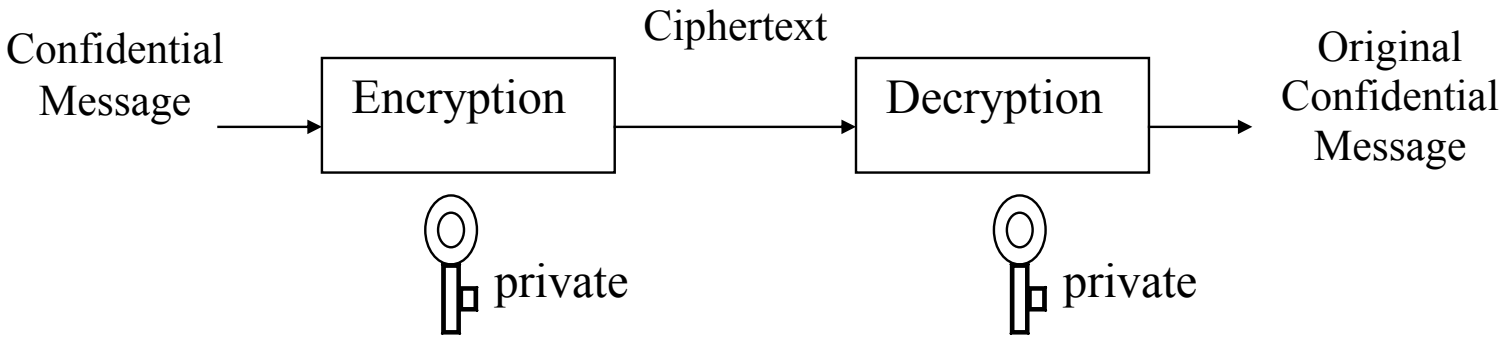
- Background
- Existing Approaches for Integrating Compression and Encryption
- Compression + Encryption using Chaos
- Conclusions

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



- Block Ciphers
  - Data Encryption Standard (DES)
  - Advanced Encryption Standard (AES)

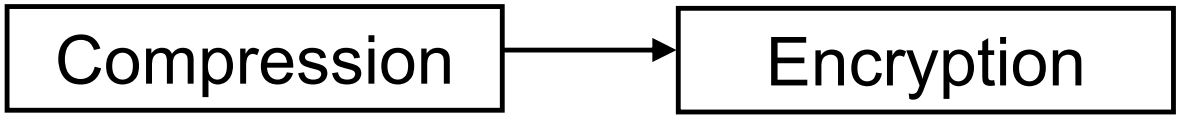
Cryptographic Standards

- Ciphertext length = Plaintext length

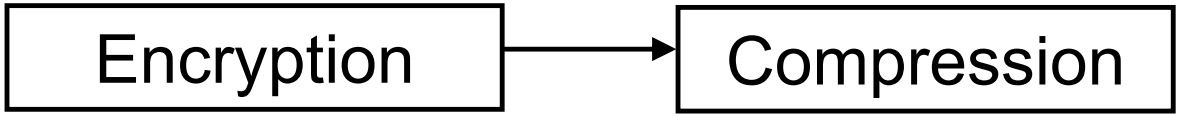
	Plaintext Block Length	Ciphertext Block Length
DES	64 bits	64 bits
AES	128 bits	128 bits

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

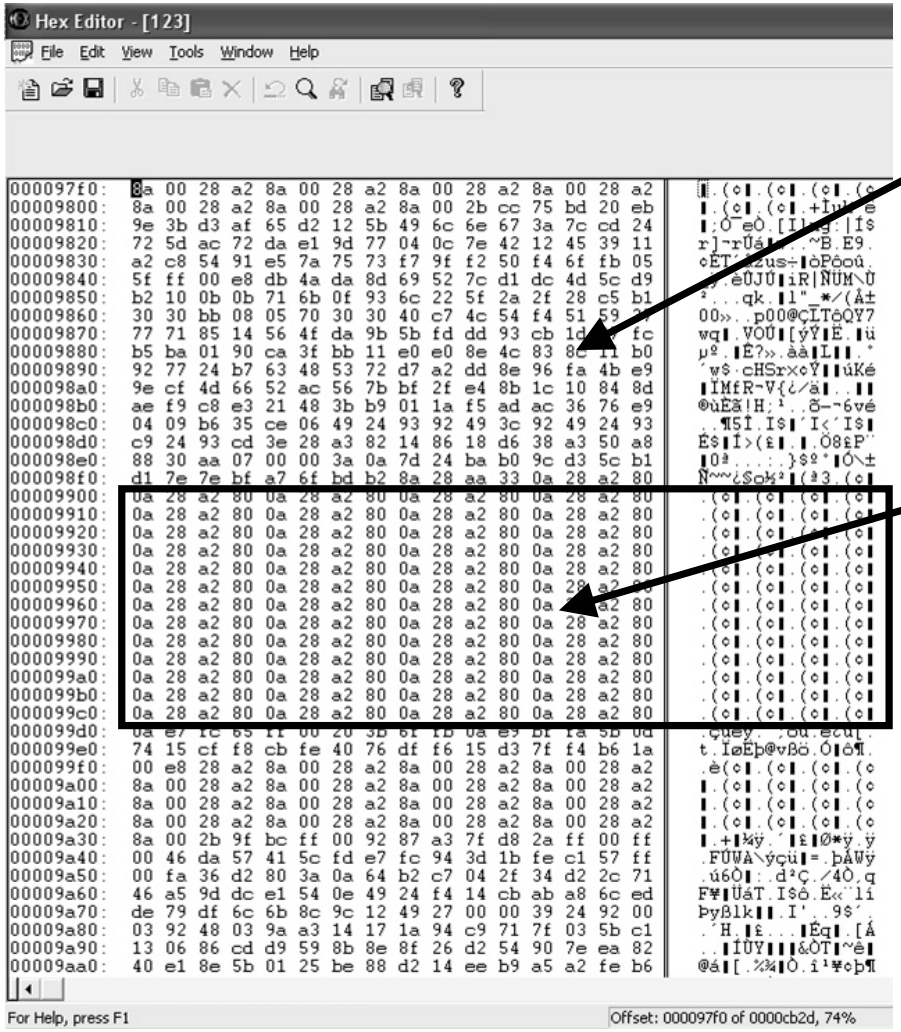
Compression + Encryption



OR



Compression First



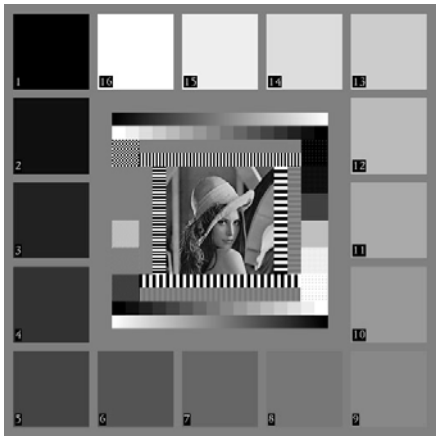
Hex value of a JPEG file

Repeated Hex value

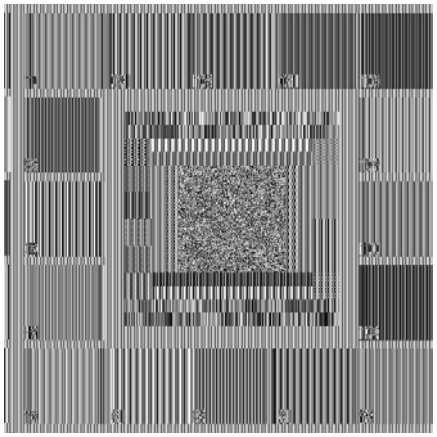
may generate repeated patterns in encrypted file

Compression First

- Example:

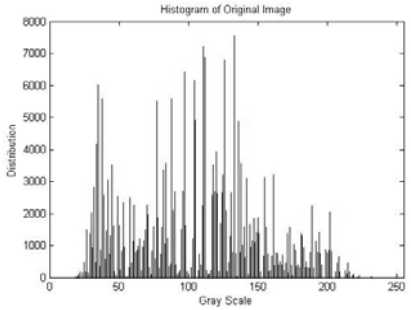


Original Image

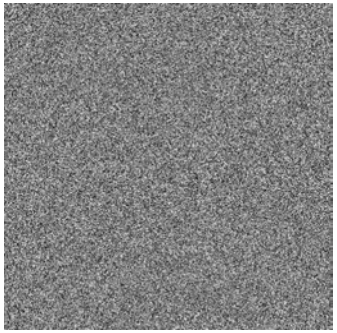


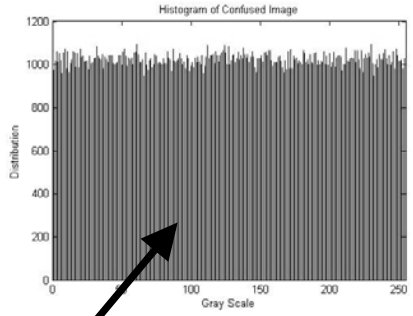
Encrypted image by Advanced Encryption Standard (AES) running in the electronic code book (ECB) mode

Encryption First



Plain Lena image &
its histogram



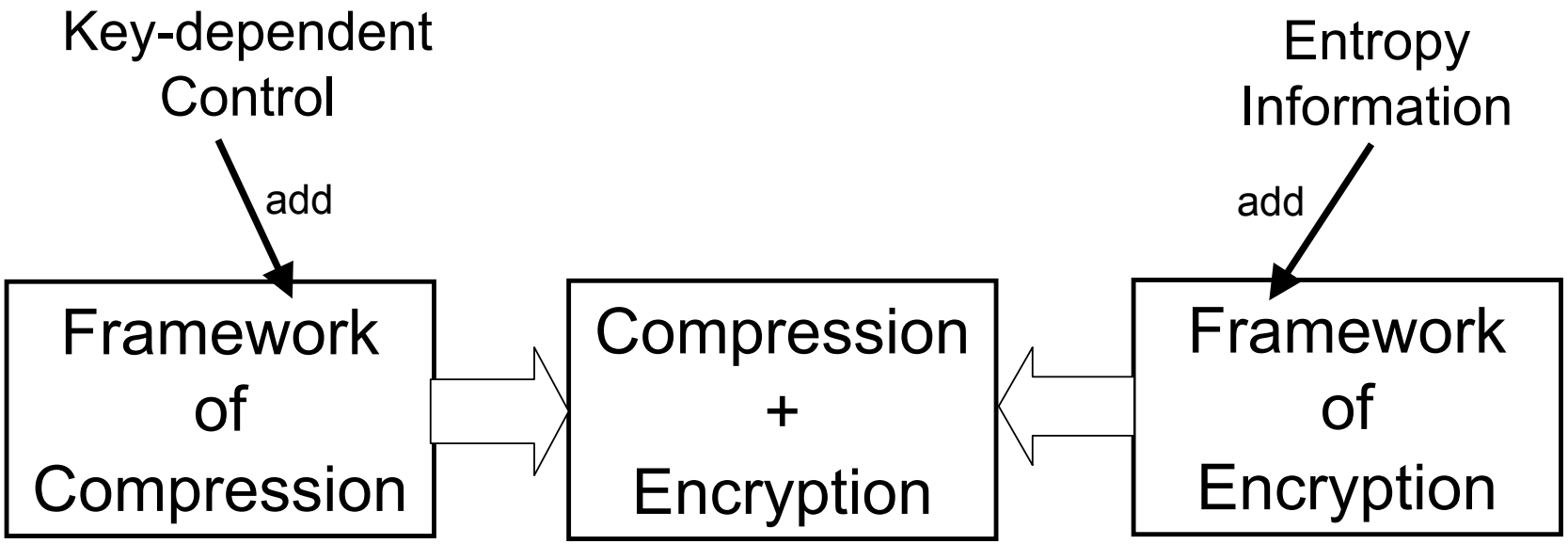


Cipher image &
its histogram

Flat Histogram ⇨ may not be compressed at all

Integrating Compression & Encryption

2 Different (if not opposite) Research Directions



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 ⇨ short code.
- Low-occurrence symbol ⇨ long code.

Reduced average code length ⇨ Compression

Based on the Framework of Compression

Embedding Encryption into Entropy Coding

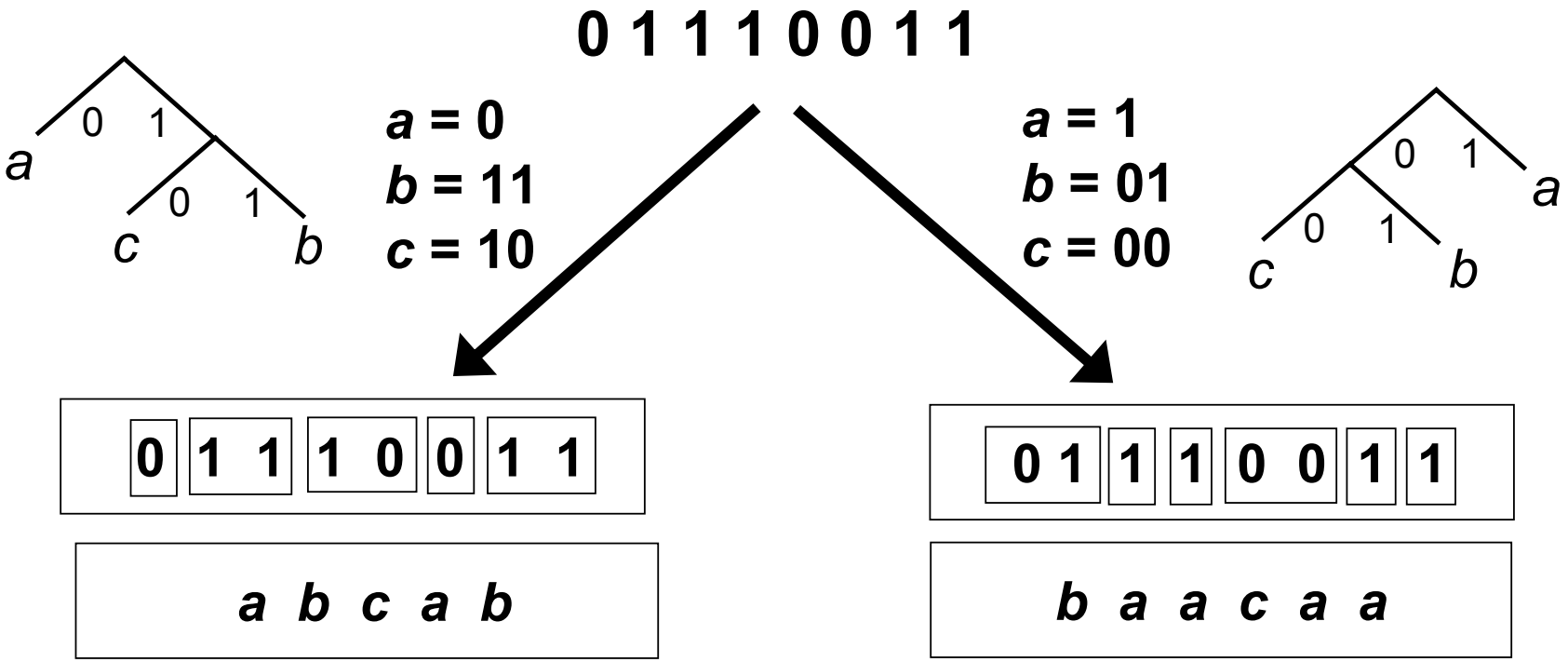
- Turn entropy coder into cryptographic cipher using multiple statistical models.
- Select a statistical model by the secret key.
- Incorrect key

⇨ Statistical model not matched

⇨ Wrong Decoding

Example

Huffman Decoding



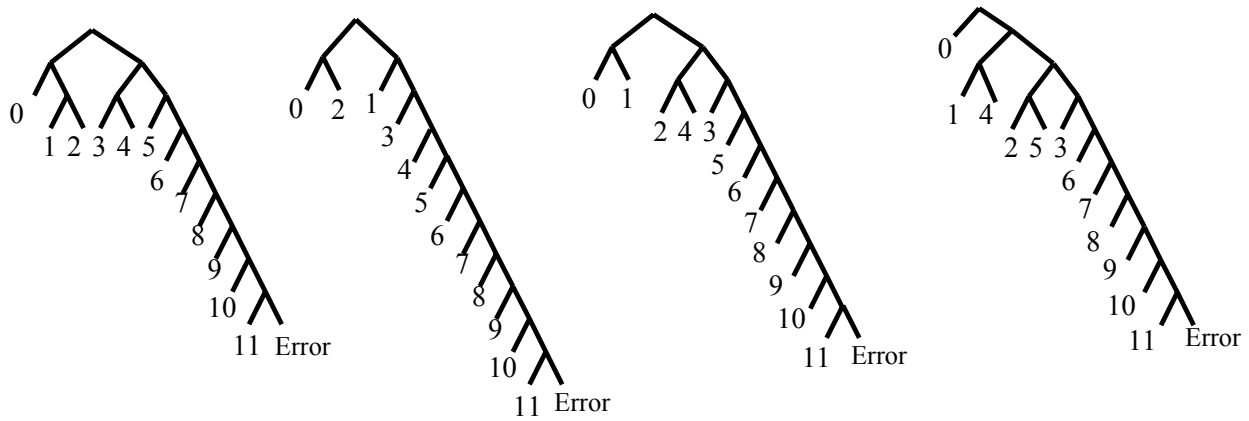
Multiple Huffman Table Approach

- C. Wu and C. Kuo, “Design of integrated multimedia compression and encryption systems,” *IEEE Trans. Multimedia*, vol. 7, no. 5, pp. 828-839, 2005.

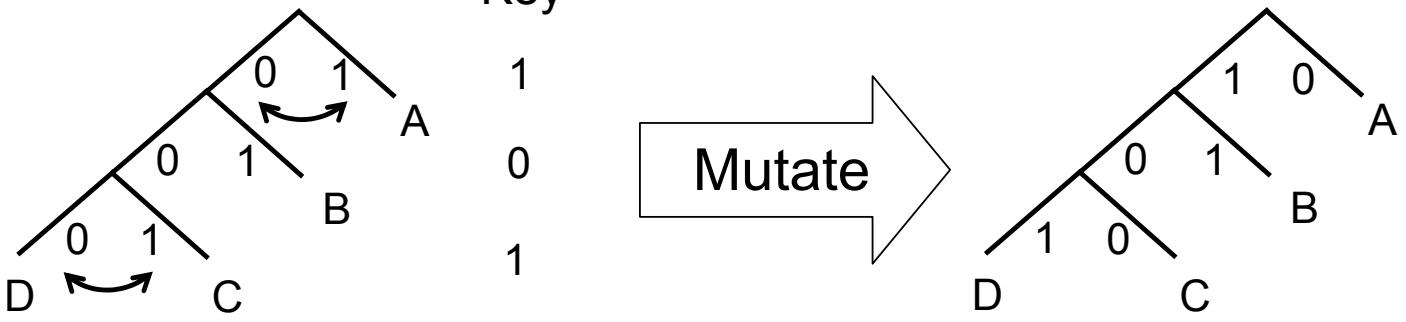
- Use a secret key to
  - Choose a table from a set of 4 basic Huffman coding tables (trees)
  - Mutate the chosen tree

Multiple Huffman Table Approach

4 Basic Huffman Trees



Tree Mutation



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.

- Jiantao Zhou, Zhiqin Liang, Yan Chen, Oscar Au, “Security Analysis of Multimedia Encryption Schemes Based on Multiple Huffman Table” *IEEE Signal Processing Letters*, vol. 14, issue 3, pp. 201 -204, March 2007.

Randomized Arithmetic Coding

- M. Grangetto, E. Magli, and G. Olmo, “Multimedia selective encryption by means of randomized arithmetic coding,” *IEEE Trans. Multimedia*, vol. 8, no. 5, pp. 905-917, 2006.

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

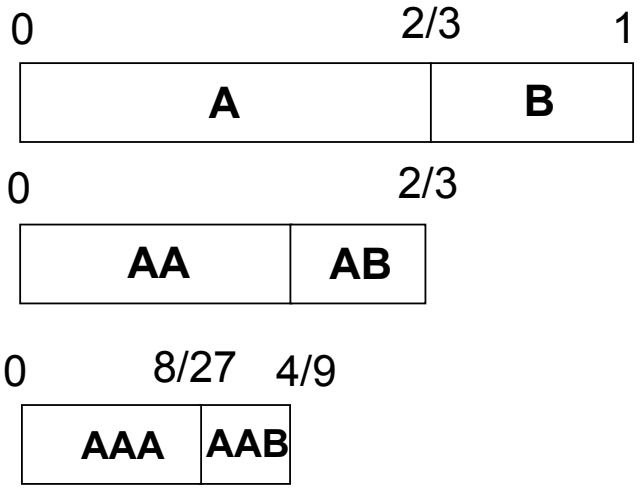
Randomized Arithmetic Coding

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

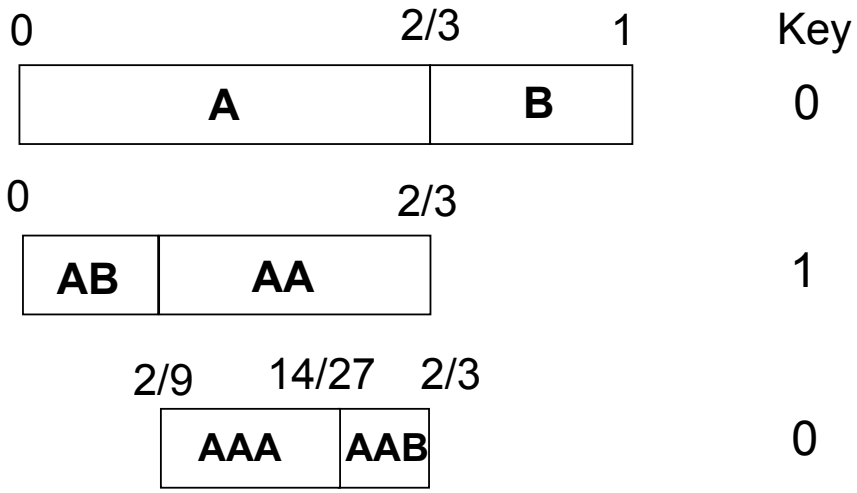
$p(A) = 2/3$

$p(B)=1/3$

Original Arithmetic Coding



Randomized Arithmetic Coding

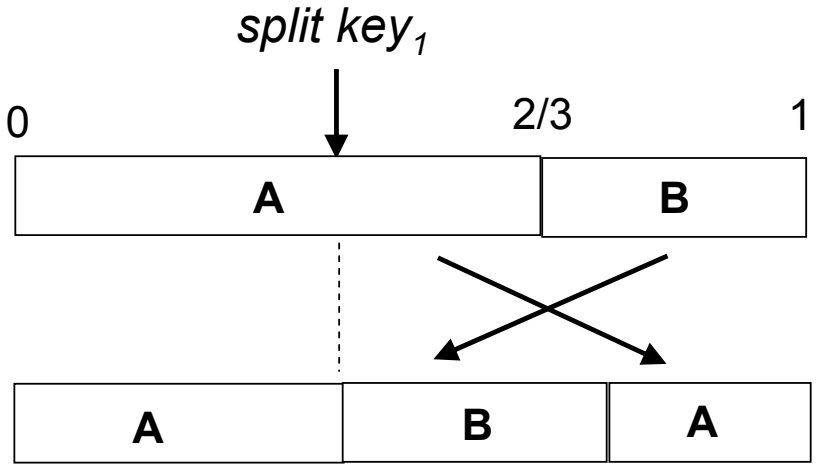


Key-based Interval Splitting in Arithmetic Coding

- J. Wen, H. Kim, and J. Villasenor, “Binary arithmetic coding with key-based interval splitting,” *IEEE Signal Processing Letters*, vol. 13, no. 2, pp. 69-72, 2006.
- Example:

$p(A) = 2/3 \qquad p(B)=1/3$

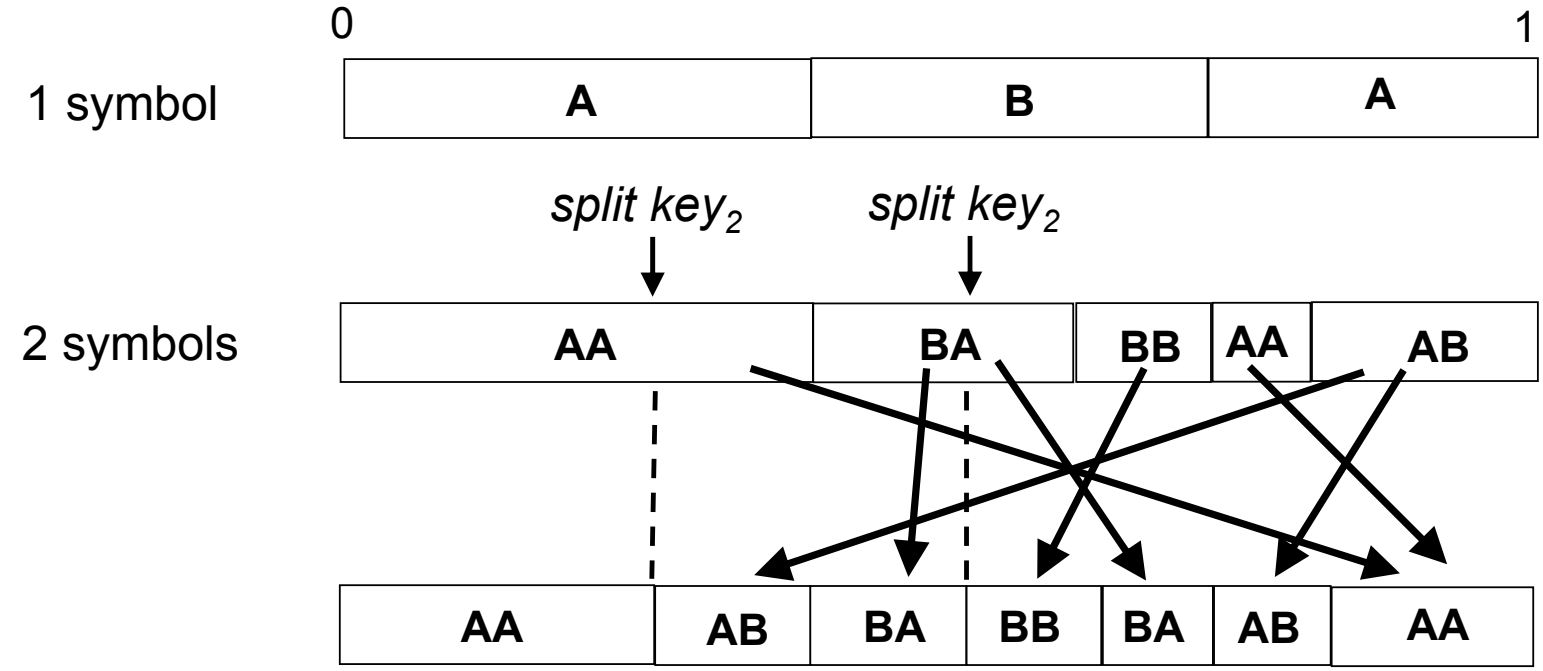
1-symbol case :



Key-based Interval Splitting in Arithmetic Coding

$p(A) = 2/3$ $p(B)=1/3$

$p(AA) = 4/9$ $p(AB)=2/9$ $p(BA) = 2/9$ $p(BB)=1/9$



Content

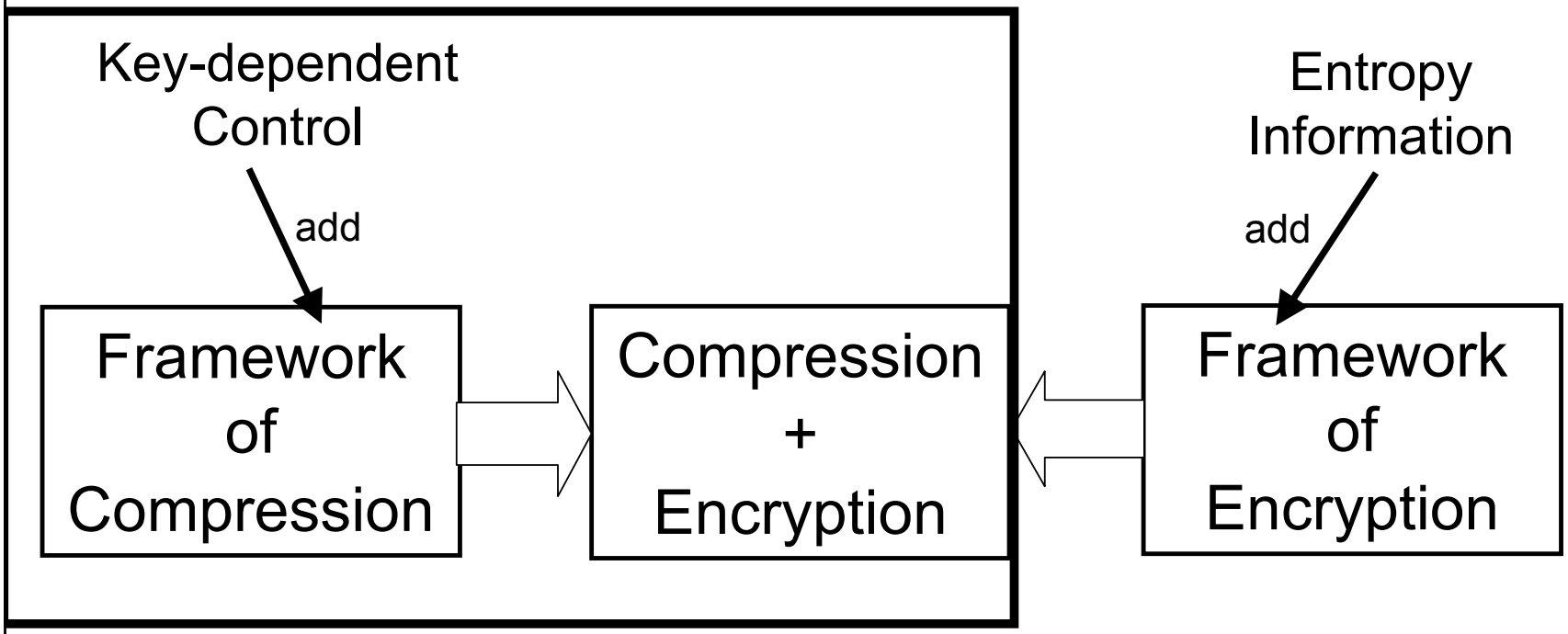
- Background
- Existing Approaches for Integrating Compression and Encryption
- Compression + Encryption using Chaos
  - Based on the framework of Compression
  - Based on the framework of Cryptography
- Conclusions

Compression + Encryption using Chaos

- Based on the framework of Compression
  - Joint work with Prof. Ron Chen and Dr. Yaobin Mao
  - DCT ⇨ lossy image compression
  - Use a 2D chaotic map to permute the DCT coefficients
  - Use 1D chaotic tent maps to mix the DCT coefficients and mask the Huffman coding sequence

- Based on the framework of Cryptography
  - Baptista-type chaotic cryptosystem
  - Incorporate entropy information in the search look-up table (the codebook) to achieve compression

Based on the Framework of Compression



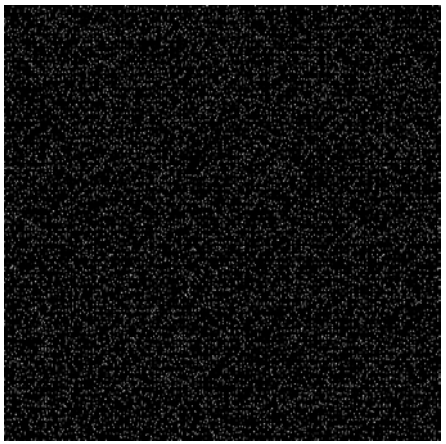
Lossy Compression + Chaotic Cryptography

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

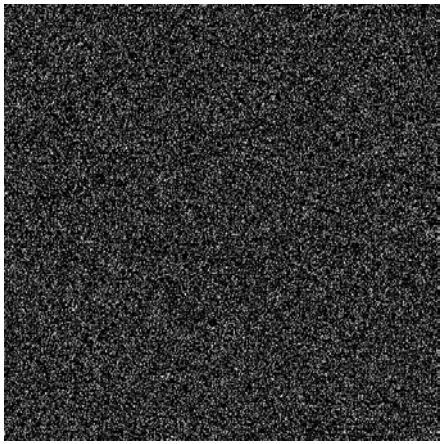
Encryption Process

DCT

2D cat map



30% mixing



DHC

110001010010010
010100011011001
0010011011……..

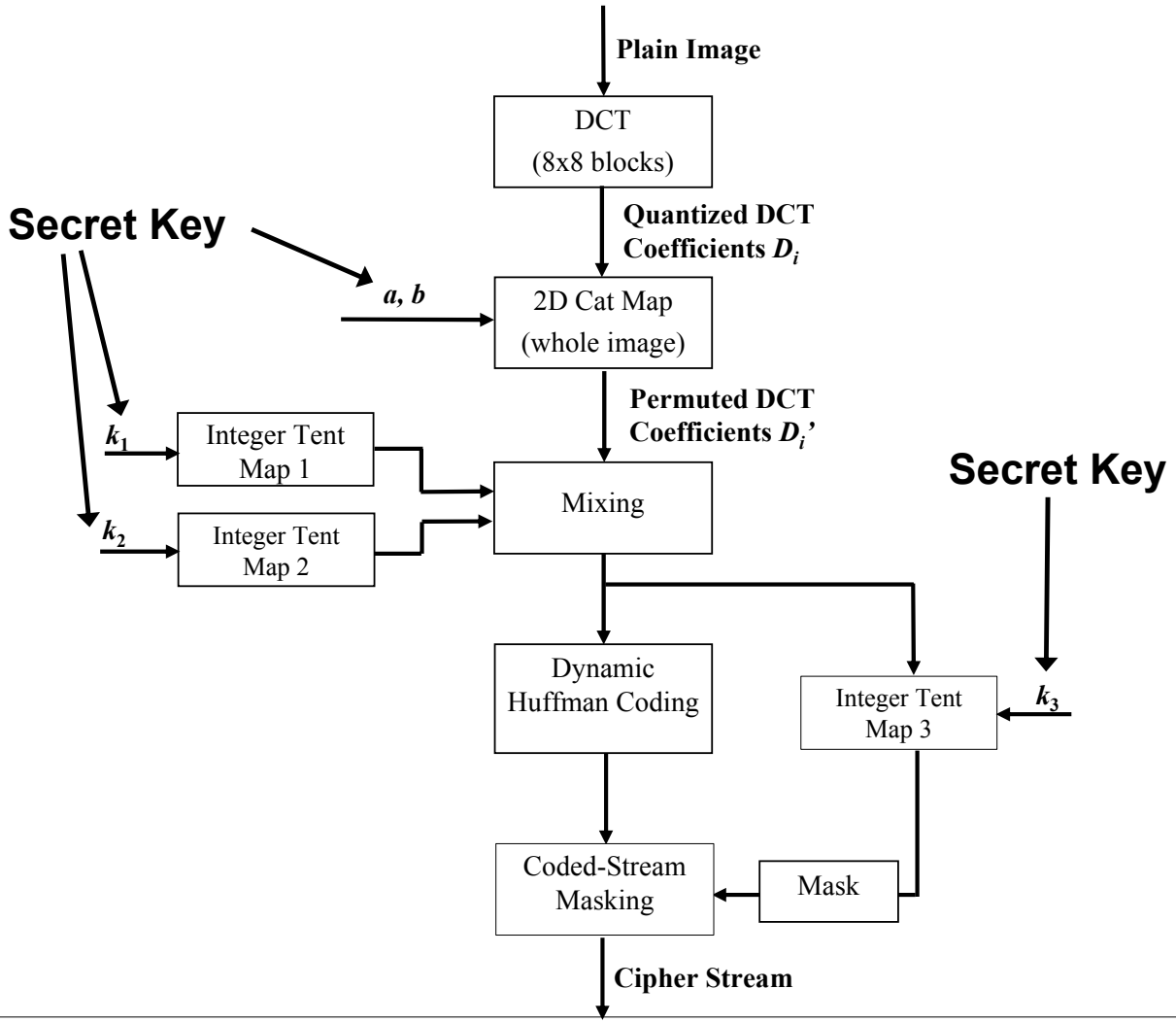
XOR

101010101101001
011010101001000
1011001101……..

variable-length Huffman code

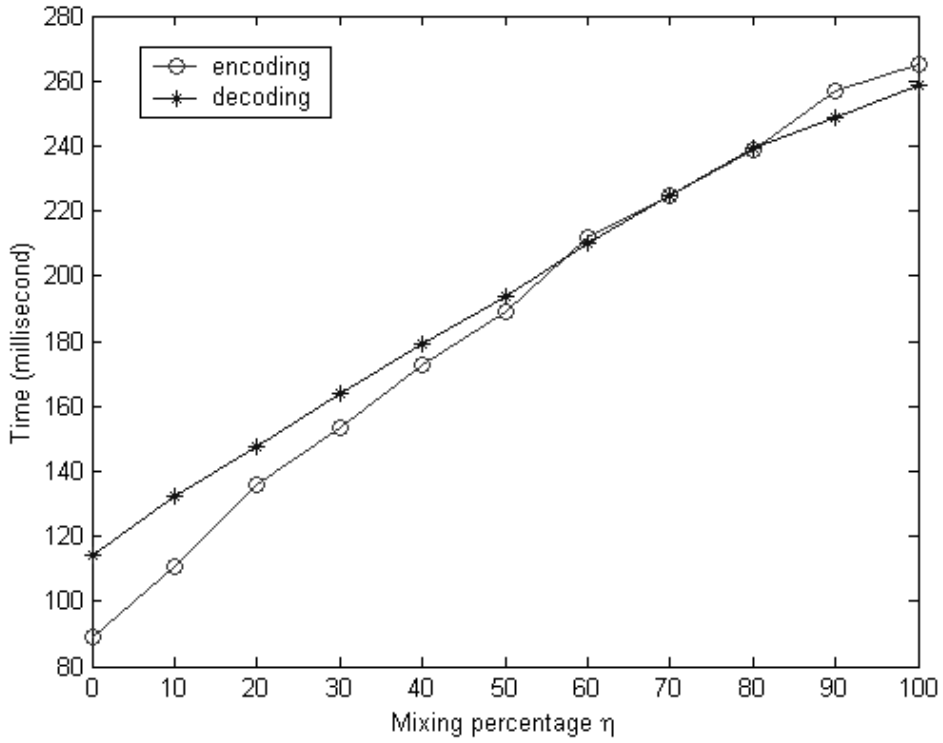
cipher stream

Lossy Compression + Chaotic Cryptography



Encryption and Decryption Speed

512 x 512 Lena with 256 grey levels



Encryption and Decryption Speed

512 x 512 Lena 256 grey levels

- Encryption Time:

  88.95 ms (no mixing) to 265.32 ms (100% mixing)

- Decryption Time:

  114.02 ms (no mixing) to 258.77 ms (100% mixing)

- Highest Data Rate:

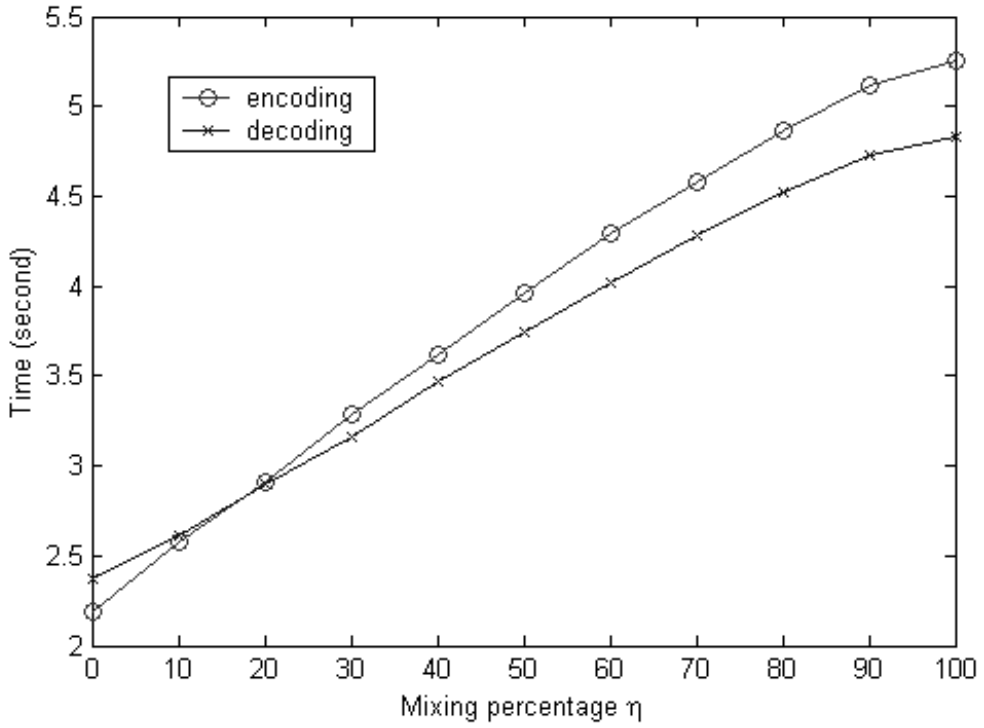
  512 x 512 x 8 / 88.95 ms = 22.48 Mbps

- Lowest Data Rate:

  512 x 512 x 8 / 265.32 ms = 7.54 Mbps

Encryption and Decryption Speed

2048 x 1536 24-bit true color image



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:

  2048 x 1536 x 24 / 2.193 s = 32.83 Mbps

- Lowest Data Rate:

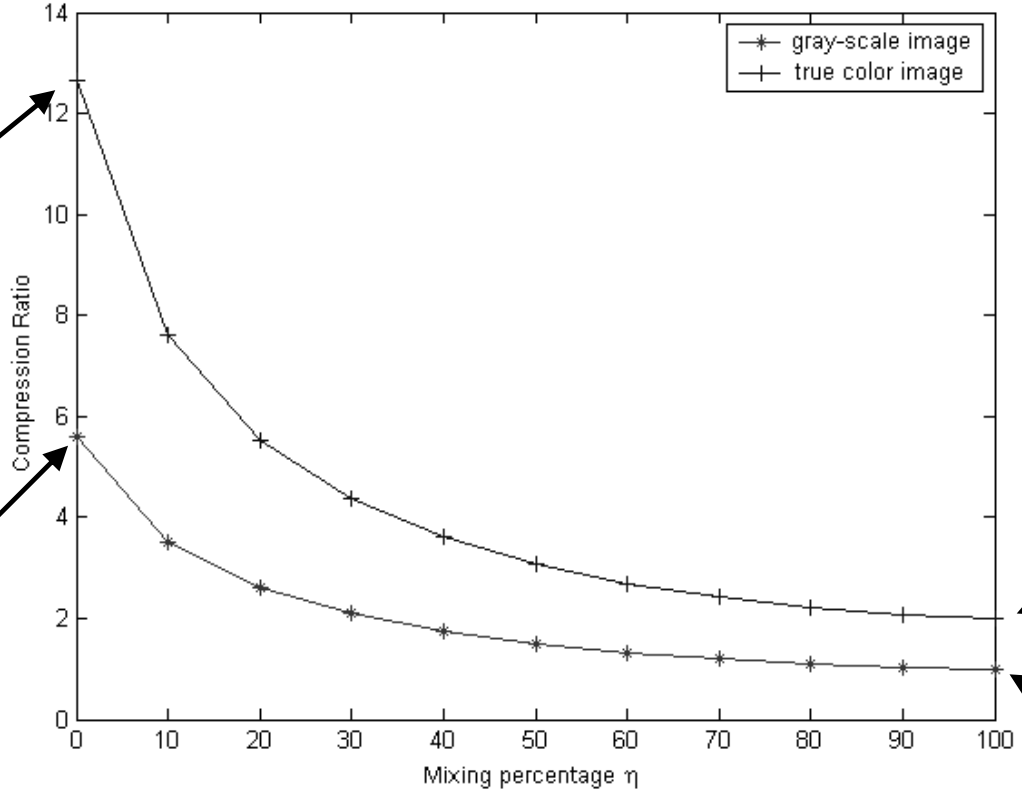
  2048 x 1536 x 24 / 5.258 s = 13.69 Mbps

Compression Ratio

No mixing

12.65 (True-color)

5.61 (Gray-scale)



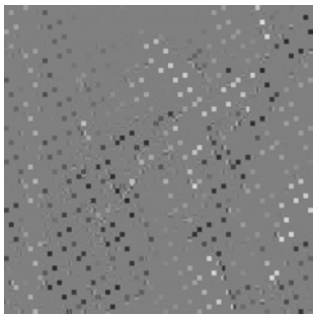
100% mixing

1.999 (True-color)

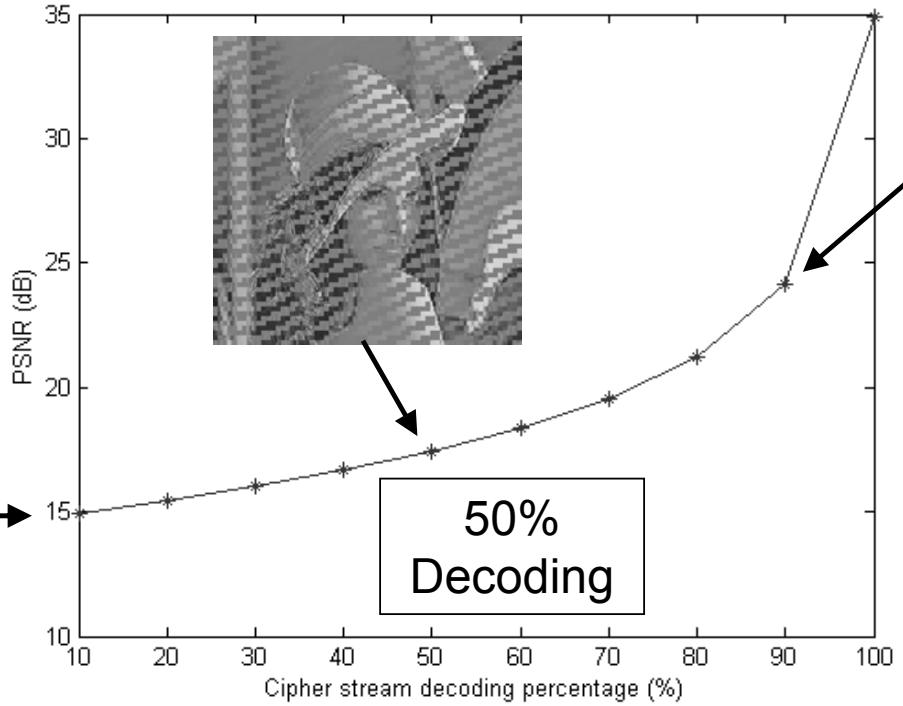
0.998 (Gray-scale)

Transmission Scalability

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



10% Decoding



50% Decoding

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)

η	**0%**	**10%**	**20%**	**30%**	**40%**	**50%**
Different Bits (%)	49.97	49.99	50.01	50.08	50.05	50.01
η	**60%**	**70%**	**80%**	**90%**	**100%**	
Different Bits (%)	50.05	49.98	50.05	49.98	50.02	

Key Sensitivity Test : Mixing Keys k_1 & k_2

- **k_1 = 0.1783929 and then k_1 = 0.1783928**

  Percentage of Different Bits = <u>50.11%</u>

- **k_2 = 0.2381211 and then k_2 = 0.2381212**

  Percentage of Different Bits = <u>50.01%</u>

- **Both are very close to 50%.**

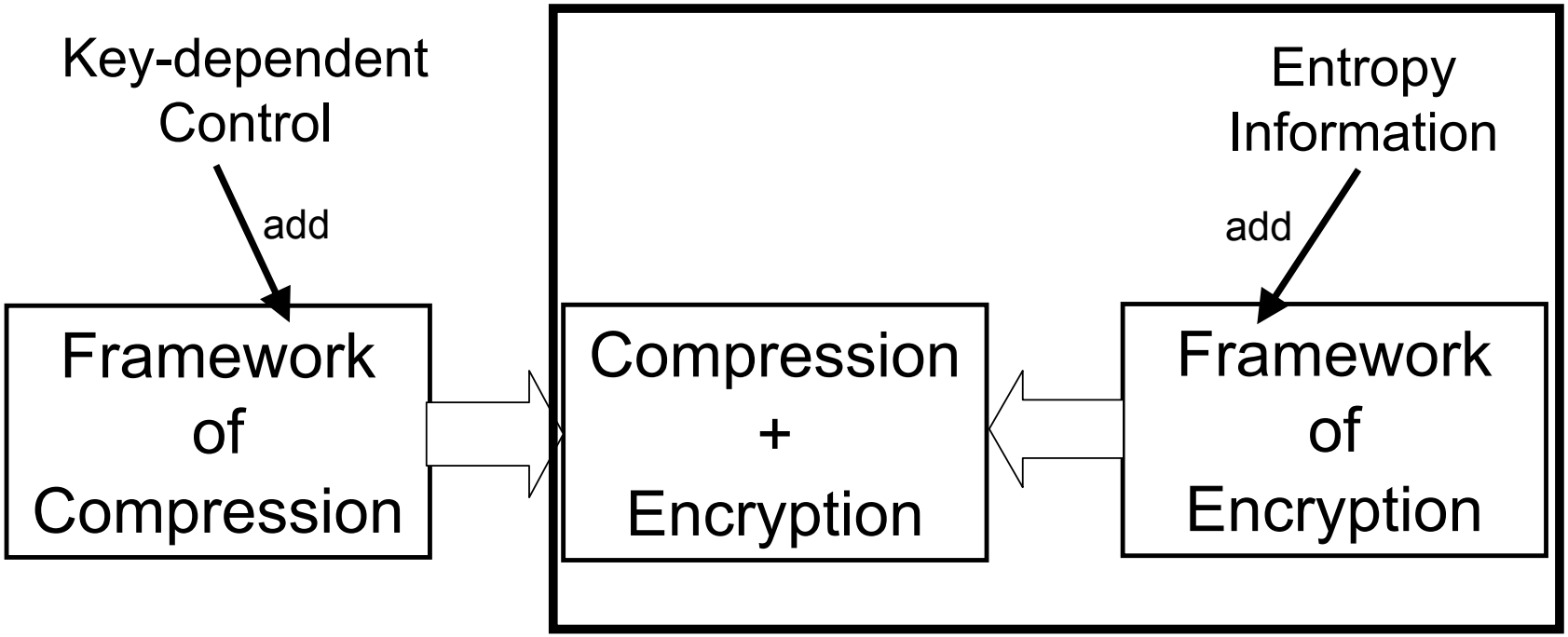
FIPS 140-2 Randomness Tests

- 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 (η = 30%)

Test Type	Result	Reference
Monobit Test	9,916	9,725 < *x* < 10,275
Poker Test	8.79	2.16 < *x* < 46.17
Run=1	2,433	2,315 < *x* < 2,685
Run=2	1,343	1,114 < *x* < 1,386
Run=3	602	527 < *x* < 723
Run=4	306	240 < *x* < 384
Run=5	177	103 < *x* < 209
Run≥6	144	103 < *x* < 209
Long Run Test (Run > 25)	0	0

Based on the Framework of Encryption



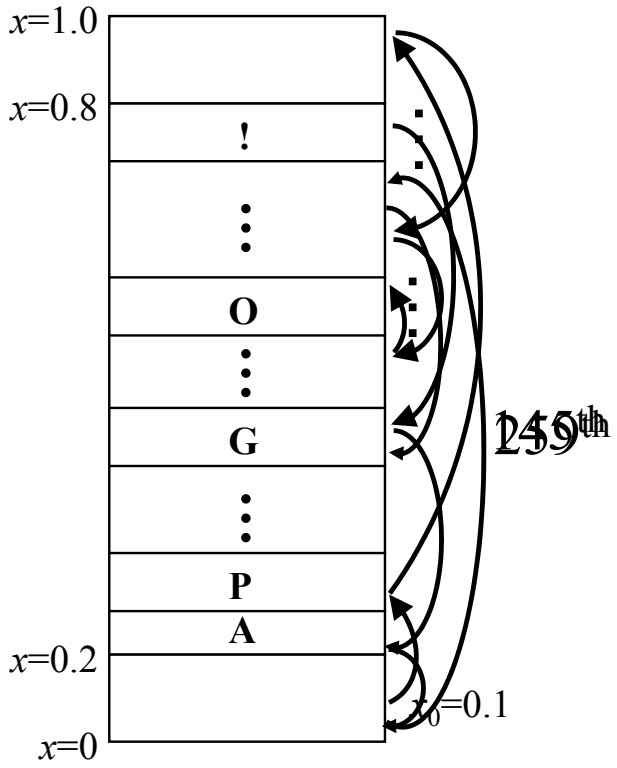
Baptista's Chaotic Cryptosystem

- M.S. Baptista, "Cryptography with chaos," *Physics Letters A*, vol. 240, no. 1-2, pp. 50-54, 1998.
- 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

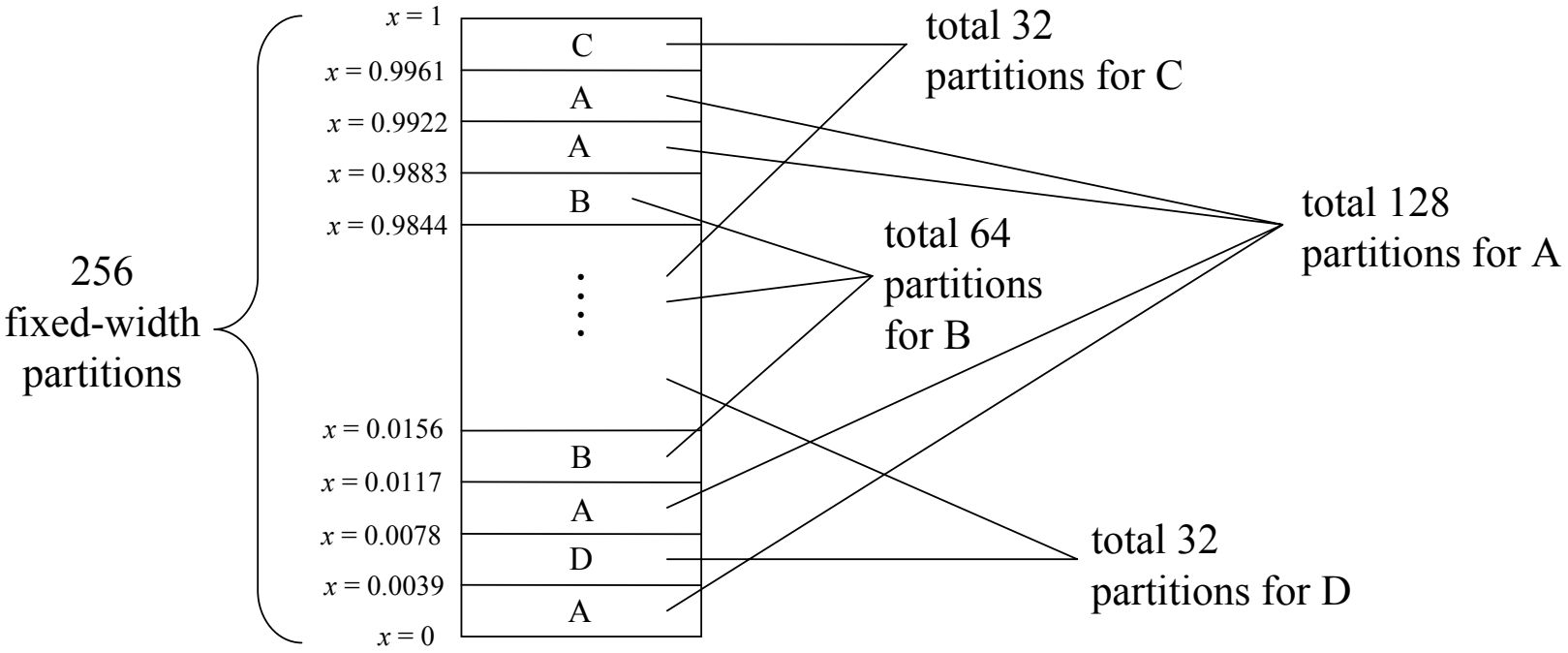
- K.W. Wong, “A fast chaotic cryptography scheme with dynamic look-up table,” *Physics Letters A*, vol. 298, pp. 238-242, 2002.
  - Use a changing look-up table to enhance the security and to increase the encryption speed.
- K.W. Wong, “A combined chaotic cryptographic and hashing scheme,” *Physics Letters A*, vol. 307, no. 5-6, pp. 292-298, 2003.
  - 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

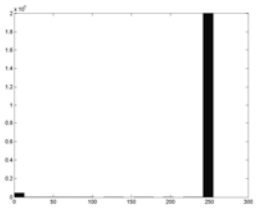
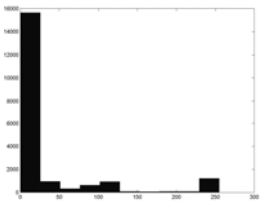
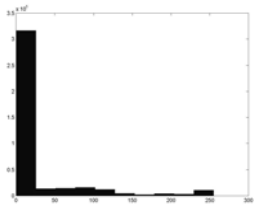
$p(A) = 1/2$ $p(B)=1/4$ $p(C) = 1/8$ $p(D)=1/8$



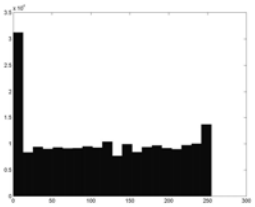
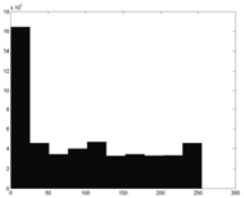
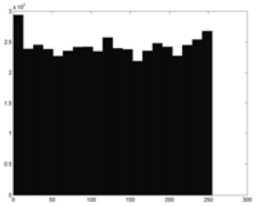
Chaotic Cryptography + Compression

- 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.

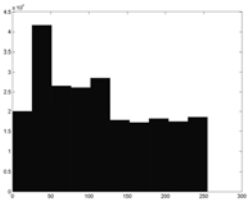
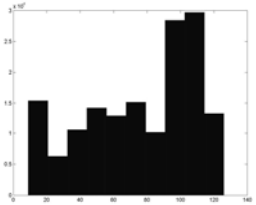
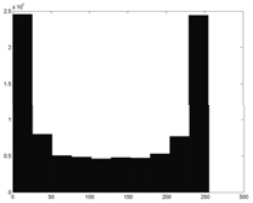
Results

File	Histogram	Percentage of non-masked symbols	Bit rate for 1-byte non-masked symbol	Ciphertext / Plaintext
graph.bmp		97.14 %	1.510	22.81 %
data.xls		73.21 %	1.910	57.68 %
db1.mdb		71.95 %	1.856	62.23 %

Results

File	Histogram	Percentage of non-masked symbols	Bit rate for 1-byte non-masked symbol	Ciphertext / Plaintext
document.doc		12.08 %	2.645	95.79 %
seminar.ppt		22.83 %	2.362	95.97 %
program.exe		3.41 %	3.065	101.91 %

Results

File	Histogram	Percentage of non-masked symbols	Bit rate for 1-byte non-masked symbol	Ciphertext / Plaintext
document.pdf		10.67 %	3.312	103 %
logo.eps		20.57 %	3.328	106.71 %
music.wav		10.04 %	2.986	106.8 %

Content

- Background
- Existing Approaches for Integrating Compression and Encryption
- Compression + Encryption using Chaos
- Conclusions

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