Securing Digital Audio using Complex Quadratic Map

Suryadi MT, Tjandra Satria Gunawan and Yudi Satria
Department of Mathematics, Faculty of Mathematics & Natural Sciences, Universitas Indonesia, Depok, 16424, Indonesia
E-mail: yadi.mt@sci.ui.ac.id, tjandra.satria@sci.ui.ac.id, ysatria@sci.ui.ac.id

Abstract. In this digital era, exchanging data are common and easy to do, therefore it is vulnerable to be attacked and manipulated from unauthorized parties. One data type that is vulnerable to attack is digital audio. So, we need data securing method that is not vulnerable and fast. One of the methods that match all of those criteria is securing data using chaos function. Chaos function that is used in this research is complex quadratic map (CQM). There are some parameter value that causing the key stream that is generated by CQM function to pass all 15 NIST test, this means that the key stream that is generated using CQM is proven to be random. In addition, samples of encrypted digital sound when tested using goodness of fit test are proven to be uniform, so securing digital audio using this method is not vulnerable to frequency analysis attack. The key space is very huge about 8.1×10³¹ possible keys and the key sensitivity is very small about 10⁻¹⁰, therefore this method is also not vulnerable against brute-force attack. And finally, the processing speed for both encryption and decryption process on average about 450 times faster that its digital audio duration.

1. Introduction

The needs for securing information especially digital audio has become important and critical. For fulfilling that we need good encryption process, one of them is using chaos function.

There are many previous research discussing about securing digital multimedia using chaos function, some of them are: securing digital image that already discussed by many researcher [1,2], on securing digital audio [3,4], and securing digital video [5,6].

Chaos function [7] is usually easier to be implemented and based on research by Kocarev & Lian [8] on Table 14, encryption method using chaos function is significantly faster compared to other type of encryption like AES and DES.

Common chaos function used for many encryption process is Logistic Map that usually implemented on digital image encryption.

In this research, we are discussing other chaos function that is complex quadratic map (CQM). This function have one complex parameter \( c \) and this function map a complex number \( z \) to other complex number \( z' \) using this equation: \( z' = z^2 + c \).

Because both \( c \) and \( z \) are complex numbers, the parameter domain in CQM will be more general than the parameter domain in Logistic Map. So we predict that the key space is very big so it will not vulnerable against brute-force attack.

In this paper, we will develop new algorithm for securing digital audio using CQM function and analyzing its performance.
2. Research Method

This CQM function mapping some complex number to other complex number using this equation [9]:

\[ g_c(z) = z^2 + c \] (1)

where \( z, c \in \mathbb{C} \), but finding some parameter \( c \) such that this CQM function to be chaotic and bounded is not easy. This is CQM function in recurrence form:

\[ z_{n+1} = z_n^2 + c \] (2)

where \( z_0 = 0 + 0i \) and \( c \in \mathbb{C} \). Complex parameter \( c \) can be decomposed into two real parameter \( a, b \in \mathbb{R} \) using this equation: \( c = a + bi \). In this paper we only consider \( |a| \leq 2 \) and \( |b| \leq 2 \) as candidate parameter.

In Equation (2), if we have \( c \) we will get the sequence of complex number \( z_1, z_2, z_3, \ldots \), but because we use digital audio with integer samples, that sequence can’t be used for key stream directly. Each element in that sequence \( z_n \) should be transformed first to \( k_n \) which is integer number.

We define the transformation from \( z_n \) to \( k_n \) as follows:

- \( z_n \) will be represented as 2 real numbers \( a_n \) and \( b_n \) using this equation \( z_n = a_n + b_ni \)
- Both \( a_n \) and \( b_n \) are represented as 64 bit floating point number using double precision floating point format (DPFPF) that is defined as follows [10]:
  \[ -1^{sign}(1 + \sum_{i=1}^{52} prec_{52-i} \cdot 2^{-i}) \times 2^{exp} \times 1023 \] (3)

- Because \( k_n \) is integer and it’s used to encrypt or decrypt digital audio with max 32 bit integer, we use 32 bit unsigned integer to represent \( k_n \).
- The first 16 bit (from bit 1 to bit 16 inclusive) on \( k_i \) are taken from bit 33 to bit 48 inclusive on binary representation of \( a_i \)
- The last 16 bit (from bit 17 to bit 32 inclusive) on \( k_i \) are taken from bit 33 to bit 48 inclusive on binary representation of \( b_i \)

Using that rule to transform \( z_i \) to \( k_i \) will always make \( k_i \) become 32 bit integer. To encrypt or decrypt digital audio with bit depth \( b \) bit, and to make the resulting digital audio also with bit depth \( b \) bit, then we first should convert \( k_i \) to become \( b \) bit integer \( x_i \) with this equation: \( x_i = k_i \ mod \ 2^b \) where \( mod \) denoting modulo operation. This \( x_i \) will be XOR-ed with corresponding audio sample value on digital audio that to be either encrypted or decrypted.

Equation (2) is used for generating key stream that satisfy chaotic condition. That key stream is used indirectly to encrypt plain audio using bit exclusive or (XOR) operation and some preprocessing with some parameter and the resulting data is called cipher audio. That resulting cipher audio containing information that is too different than original plain audio and it’s very hard to get that original plain audio without knowing the parameter used to generate the key stream. To get the original plain audio back, we should do decryption process that is inverse of encryption process and we should use the exact same key parameter to generate key stream.

This encryption algorithm performance is tested based on: key sensitivity, key stream randomness based on 15 NIST test [11], ergodicity is tested using Goodness of Fit test [12], algorithm efficiency based on computation time, and key space analysis based on IEEE 754 double precision binary floating point format.

Restriction in this paper are: digital audio that is used for encryption and decryption process has (*.wav) format and pulse code modulation (PCM) data type with bit depth per sample varies from 8 bit to 32 bit (8, 16, 24, and 32 bit). The programming language that is used on this research is C language, and the program run single thread on Kubuntu 17.04 with Core i7-4710HQ @2.50 GHz processor.

3. Result and Analysis

Test data that is used on this research is digital audio that having two channels left and right (stereo), and each channel having 44100Hz sample rate, more detailed info are given on Table 1.
Table 1. Test data used in this research

| Number | Sound Type                  | File Name       | Duration     | Number of Samples      | Bit Depth |
|--------|-----------------------------|-----------------|--------------|------------------------|-----------|
| 1      | Music taken from No Copyright Sound (NCS) | Record.wav     | 9.95 seconds | 438663 samples ×2 channel  | 8         |
| 2      | Short audio recording | Ricochet.wav    | 3 minutes    | 9791479 samples ×2 channel | 16        |
| 3      | Mozart melody (Created using superposition of some sine wave) | Mozart.wav     | 9 minutes    | 24712979 samples ×2 channel | 24        |

3.1. Key Sensitivity
We use \( c = -0.47 + 0.54i \) as a key, and we use that key to encrypt test data number 10 (Mozart.wav) and then we try to decrypt the file but with different but almost equal key \( c = -0.47 - 10^{-x} + 0.54i \).
Analysis result for some value of x are given on Table 2.

Table 2. Key sensitivity analysis result

| X      | Qualitative       | Channel | \( \chi^2 \) | \( P_{value} \) |
|--------|-------------------|---------|--------------|----------------|
| 18     | Same as original  | Left    | 5903479841   | 0              |
|        |                   | Right   | 9238326871   | 0              |
| 17     | There are a little noise | Left    | 1641746228.8 | 0              |
|        |                   | Right   | 2540732224.8 | 0              |
| 16     | There are a little noise | Left    | 106923069.1  | 0              |
|        |                   | Right   | 152171262.5  | 0              |
| 15     | There are some noise | Left    | 38619867.08  | 0              |
|        |                   | Right   | 42425652.77  | 0              |
| 14     | There are many noise | Left    | 8940486.307  | 0              |
|        |                   | Right   | 8919213.488  | 0              |
| 13     | Not recognized after 2 minutes | Left    | 205585.0038  | 0              |
|        |                   | Right   | 207153.1011  | 0              |
| 12     | Not recognized after 12 seconds | Left    | 66442.89612  | 0.006624993 |
|        |                   | Right   | 66646.60339  | 0.001125034 |
| 11     | Not recognized after 1 second | Left    | 65185.36123  | 0.832887257 |
|        |                   | Right   | 65583.51007  | 0.445989412 |
| 10     | Completely not recognized | Left    | 65535.43138  | 0.498790016 |
|        |                   | Right   | 65879.51358  | 0.170604590 |
|        | Not Decrypted     | Left    | 65805.95026  | 0.226862604 |
|        |                   | Right   | 65594.53662  | 0.433983646 |

On Table 2, if we use \( c = -0.47 + 0.54i \) for encrypting data, then we need \( x \leq 10 \) or error that is larger than or equal \( 10^{-10} \) to keep the decrypted data completely not recognized. So we conclude that key sensitivity on this CQM function is about \( 10^{-10} \).

3.2. Key Stream Randomness Analysis (NIST Test)
NIST test result with level of significance \( \alpha = 0.01 \) on gey stream that is generated and converted to 32 bit integer are given on Table 3.
Table 3. NIST test result on CQM function with key parameter $c = -0.47 + 0.54i$

| Statistical Test          | Repetition | $P_{value}$ | Proportion |
|---------------------------|------------|-------------|------------|
| Frequency                 | 1          | 0.534146    | 128.00/128 |
| Block Frequency           | 1          | 0.275709    | 127.00/128 |
| Cumulative Sums           | 2          | 0.323322    | 127.50/128 |
| Runs                      | 1          | 0.378138    | 127.00/128 |
| Longest Run               | 1          | 0.070445    | 124.00/128 |
| Rank                      | 1          | 0.170294    | 127.00/128 |
| FFT                       | 1          | 0.834308    | 127.00/128 |
| Non Overlapping Template  | 148        | 0.408235    | 126.68/128 |
| Overlapping Template      | 1          | 0.057146    | 126.00/128 |
| Universal                 | 1          | 0.090936    | 126.00/128 |
| Approximate Entropy       | 1          | 0.422034    | 127.00/128 |
| Random Excursions         | 8          | 0.242362    | 107.25/108 |
| Random Excursions Variant | 18         | 0.319938    | 106.44/108 |
| Serial                    | 2          | 0.612282    | 126.00/128 |
| Linear Complexity         | 1          | 0.819544    | 128.00/128 |

It can be seen from Table 3 that CQM function with key parameter $c = -0.57 + 0.54i$ is generating completely random key stream based on all 15 NIST test. This is because for each test the $P_{value}$ are larger than the level of significance that is 0.01 and passing proportion are all larger than minimum pass rate.

The minimum pass rate for each statistical test with the exception of the random excursion (variant) test is approximately 123 for a sample size 128 binary sequences. The minimum pass rate for the random excursion (variant) test is approximately 103 for a sample size 108 binary sequences.

3.3. Ergodicity Analysis

For measuring ergodicity level, we use Goodness of Fit test on cipher audio for each test data and we use level of significance $\alpha = 0.01$. The result are given on Table 4.

Table 4. Goodness of Fit test result

| Number | Bit Depth | Degree of Freedom | Left Statistical Test | Left $P_{value}$ | Right Statistical Test | Right $P_{value}$ |
|--------|-----------|-------------------|-----------------------|------------------|------------------------|-------------------|
| 1      | 8         | 255               | 291.098               | 0.05963155       | 222.170                | 0.93207627       |
| 2      | 16        | 65535             | 65506.12642           | 0.53105557       | 65857.02580            | 0.68800259       |
| 3      | 24        | 65535             | 65215.39514           | 0.81121859       | 65202.84559            | 0.82047453       |
| 4      | 32        | 65535             | 65853.33069           | 0.18951083       | 66256.11154            | 0.02349425       |
| 5      | 8         | 255               | 278.002               | 0.15416357       | 294.036                | 0.04680851       |
| 6      | 16        | 65535             | 65626.19250           | 0.39989747       | 65209.97125            | 0.81525438       |
| 7      | 24        | 65535             | 66160.38751           | 0.04237252       | 65675.26680            | 0.34863661       |
| 8      | 32        | 65535             | 65617.66541           | 0.40901475       | 65797.15755            | 0.43113867       |
| 9      | 8         | 255               | 265.180               | 0.31761973       | 258.303                | 0.43049635       |
| 10     | 16        | 65535             | 65805.95026           | 0.22686260       | 65594.53662            | 0.43398365       |
| 11     | 24        | 65535             | 65374.86500           | 0.67034499       | 66085.37948            | 0.06452742       |
| 12     | 32        | 65535             | 65000.59374           | 0.93033663       | 65269.76015            | 0.76784941       |

It can be seen from Table 4 that for all 12 test data used on this research, all $P_{value}$ are larger than level of significance that is 0.01. That means that the cipher audio samples are uniformly distributed or satisfy the ergodicity condition.
3.4. Measuring Computation Time

For testing with key parameter \( c = -0.47 + 0.54i \), we get the measured time for encryption and decryption process that is given on Table 5.

Table 5. Computation time for encryption and decryption process using CQM function

| Number | Bit Depth | Sound Duration (seconds) | Encryption Time (seconds) | Ratio: Sound Duration per Encryption Time | Decryption Time (seconds) | Ratio: Sound Duration per Decryption Time |
|--------|-----------|--------------------------|---------------------------|------------------------------------------|---------------------------|------------------------------------------|
| 1      | 8         | 9.95                     | 0.0235722                 | 422.1073977                              | 0.0241974                 | 411.2012034                              |
| 2      | 16        | 222.03                   | 0.0225165                 | 441.8981636                              | 0.0224950                 | 442.3205157                              |
| 3      | 24        | 448.0730607              | 0.0222062                 | 448.0730607                              | 0.0221952                 | 448.2951269                              |
| 4      | 32        | 446.0103367              | 0.0223089                 | 446.0103367                              | 0.0223861                 | 444.472395                               |
| 5      | 8         | 474.4992199              | 0.4679249                 | 474.4992199                              | 0.4656103                 | 476.8580077                              |
| 6      | 16        | 446.4730080              | 0.4972977                 | 446.4730080                              | 0.4945643                 | 448.9406130                              |
| 7      | 24        | 472.309661               | 0.4700998                 | 472.309661                               | 0.4692973                 | 473.1116075                              |
| 8      | 32        | 471.9844904              | 0.4704180                 | 471.9844904                              | 0.4709581                 | 471.4432133                              |
| 9      | 8         | 452.8850155              | 1.2373781                 | 452.8850155                              | 1.2450933                 | 450.0787210                              |
| 10     | 16        | 468.8003374              | 1.1953703                 | 468.8003374                              | 1.2046227                 | 465.1996015                              |
| 11     | 24        | 444.2984566              | 1.2612918                 | 444.2984566                              | 1.2885842                 | 434.8881509                              |
| 12     | 32        | 441.3301569              | 1.2697750                 | 441.3301569                              | 1.2550427                 | 446.5107044                              |

It can be seen on Table 5 that the computation time for encryption and decryption process are nearly equal. Digital audio bit depth is not affecting the processing speed for both encryption and decryption process that is on average about 450 faster that its digital audio duration.

3.5. Key Space Analysis

Based on Equation (3) the bit value of \( \text{expo} \) is not independence because based on Equation (2) all key parameter which is used on CQM function that is \( a \) and \( b \) are all satisfy \(|a| \leq 2 \) and \(|b| \leq 2 \), therefore the variable that is independence are only \( \text{sign} \) and \( \text{prec}_1, \text{prec}_2, ..., \text{prec}_{52} \) so there are \( 2^{53} \approx 9 \times 10^{15} \) different numbers for each parameter.

There are 2 real parameters on CQM function that is \( a \) and \( b \) so based on DPFPF there are total \( (2^{53})^2 = 2^{106} \approx 8.1 \times 10^{31} \) possible keys.

4. Conclusion

Based on analysis and simulation on test data that is used on previous section, we get conclusion as follows:

a. Key sensitivity on CQM function is about \( 10^{-10} \)
b. CQM Function is passing all 15 NIST test with \( P_{\text{value}} > 0.01 \). So the key stream that is generated using this CQM function are proven to be random.
c. Generated key stream by CQM function can make the cipher audio having uniform distribution (ergodic) on both channel based on Goodness of Fit test with \( P_{\text{value}} > 0.01 \).
d. Computation time for encryption and decryption process are nearly equal. Digital audio bit depth is not affecting the processing speed for both encryption and decryption process that is on average about 450 faster that its digital audio duration.
e. Large key space that is \( \approx 8.1 \times 10^{31} \) possible keys.
Acknowledgment
The authors would like to thank Kemenristekdikti RI and Directorate of Research and Community Service (DRPM) Universitas Indonesia for the grant PTUPT 2017 who support us in this research.

References
[1] Suryadi MT, Eva Nurpeti, and Dhian Widya. (2014), Performance of Chaos-Based Encryption Algorithm for Digital Image, Journal Telecommunication Computing Electronics and Control, Vol. 12, No. 3, pp. 675-682, September 2014, ISSN: 1693-6930.
[2] Yohan Suryanto, Suryadi MT, and Kalamullah Ramli. (2016), A Secure and Robust Image Encryption Based on Chaotic Permutation Multiple Circular Shrinking and Expanding, Journal of Information Hiding and Multimedia Signal Processing, Vol. 7, No. 4, pp. 697-713, July 2016, ISSN: 2073-4212.
[3] Vishakha B. Pawar, Pritish A. Tijare, Swapnil N. Sawalkar. (2014). A Review Paper on Audio Encryption, International Journal of Research in Advent Technology, Vol. 02, No. 12, pp. 45-48, December 2014, E-ISSN: 2321-9637.
[4] Sangeeta, Vikas Dhankhar, and Amita Dhankhar. (2013), Secure LSB Based Audio Steganography, International Journal of Engineering, Applied and Management Sciences Paradigms, Vol. 05, Issue 01, pp. 131-134, July 2013, ISSN (Online): 2320-6608.
[5] Suryadi MT. (2013), New Chaotic Algorithm for Video Encryption. 4th The International Symposium on Chaos Revolution in Science, Technology and Society 2013, Jakarta, August 28-29, 2013.
[6] Brahim B, Mohamed M.L, and Salah B. (2011), Multimedia Chaos-based Encryption Algorithm, Department of Electronics, Faculty of Eng. Sciences, University Badji Mokhtar of Annaba, 23000 Annaba, Algeria.
[7] Devaney, R.L. (1989). An introduction to chaotic dynamical systems (2nd ed.). Addison-Wesley Publishing company, Inc.
[8] Kocarev, L and Lian, S. (2011), Chaos-Based Cryptography: Theory, Algorithm and Applications, Springer-Verlag, Berlin.
[9] Sarah N, Kabes. (2012), The Transition Between the Complex Quadratic Map and the Hénon Map, Department of Mathematics and Statistics, University of Minnesota, Duluth, MN 55812, July 2012.
[10] David Goldberg. (1991), What Every Computer Scientist Should Know About Floating-Point Arithmetic, ACM Computing Surveys, Vol 23, No 1, March 1991.
[11] Lawrence E Bassham III. (2010). A Statistical Test Suite for the Validation of Random Number Generator and Pseudo Random Number Generators for Cryptographic Applications, NIST Special Publication 800-22 Revision 1a.
[12] Walpole, R.E., Myers, R. H., Myers, S. L., Ye, K. (2012). Probability and Statistics for Engineers & Scientists (9th edition). Boston: Prentice Hall.