Secure Data Transmission with Variable File Types

Hnin Wut Yee Lai, Khin Than Mya
University of Computer Studies, Yangon
hninwutyee6@gmail.com

Abstract

Encryption is used to securely transmit data in open network. Each type of data has its own features, therefore different techniques should be used to protect confidential image data from unauthorized access. Most of the available encryption algorithms are mainly used for textual data and may not be suitable for multimedia data. Cryptographic algorithms is divided into symmetric key algorithm and asymmetric key algorithm.. The Ronal Rivest symmetric key algorithm (RC4) is a fast stream cipher with variable keylengths I to 256 bytes (8 to 2948 bits). Integrity check algorithm (CRC32) is a hash function to detect raw data. In this proposed system, we use Ronald Rivest symmetric key algorithm (RC4) for confidentiality and Integrity Check Algorithm (CRC32) for integrity for any format data such as text, image, sound, video and so on.

1. Introduction

Encryption is the process of transforming plaintext data into ciphertext in order to conceal its meaning and so preventing any unauthorized recipient from retrieving the original data. Cryptography is a tool that can be used to keep information confidential and to ensure its integrity and authenticity [18]. Many cryptographic algorithms use complex transformations involving substitutions and permutations to transform the plaintext into the ciphertext. Cryptography algorithms can be divided into symmetric-key algorithms and public-key algorithms. Symmetric-key algorithms mangle the bits in a series of rounds parameterized by the key to urn the plaintext into the ciphertext [18]. Public-key algorithms have the property that different key are used for encryption and decryption and that the decryption key cannot be derived from the encryption key.

2. Methods of Encryption

The two methods of producing ciphertext are stream cipher and block cipher. The two methods are similar except for the amount of data each encrypts on each pass.

2.1. Stream Cipher

Stream cipher is one of the simplest methods of encrypting data where each bit of the data is sequentially encrypted using one bit of the key as shown in Figure 1. The main advantage of the stream cipher is that it is faster and more suitable for streaming application but its main disadvantage is that it is not suitable in some architecture.

2.2. Block Cipher

In cryptography, a **block cipher** is a symmetric key cipher which operates on fixed-length groups of bits, termed blocks, with an unvarying transformation. When encrypting, a block cipher might take (for example) a 128-bit block of plaintext as input, and output a corresponding 128-bit block of ciphertext. The exact transformation is controlled using a second input the secret key. Decryption is similar: the decryption algorithm takes, in this example, a 128-bit block of ciphertext together with the secret key, and yields the original 128-bit block of plaintext. To encrypt messages longer than the block size (128 bits in the above example), a mode of operation is used.

Block ciphers can be contrasted with stream ciphers; a stream cipher operates on individual digits one at a time and the transformation varies during the encryption. The distinction between the two types is not always clear-cut: a block cipher, when used in certain modes of operation, acts effectively as a stream cipher.

3. Related Work

Cryptographers prefer using another cipher into that has been proven secure such as Advanced Encryption Standard (AES) [16], Block Cipher International Data Encryption algorithm (IDEA), Triple DES or SHA-256 as the Pseudo Random Number Generator (PRNG) in the cryptosystems.

Lae Lae Khin presented RC4 is a stream cipher, the key must never to used twice .It had attracted considerable attention in the research community since it was first propose [10]. Andrew Roos [2] developed one of the first such attacks on RC4 a set of weak keys causing bias in the initial output. Fluhrer, Mantin and Shamir [7] presented several

weaknesses in the KSA. Knudsen, Mier, Preneel, Rijimen and Verdoolaege [13] presented the swap operation makes the recovery of the table S very difficult. They developed and attack on simplified version of RC4, where the swap operation occurs less often. RC4 without the swap operation is useless as a key stream generator. D.R. Stinson presented the hash function (CRC32) is CRC calculates a short, fixed-length binary sequence, known as the CRC code or just CRC [16]. This proposed system is combined the stream cipher RC4 algorithm for confidentiality and integrity check algorithm CRC32 for integrity for texts, images and media files..

4. Background Theory

The background theory shows the combination of the stream cipher RC4 algorithm and CRC32 checksum of texts, images and media files.

4.1. RC4 Algorithm

RC4 is a stream cipher that was designed in 1984 by Ronald Rivest for RSA Data Security. RC4 is used in many data communication and networking protocols, including SSL/TLS and the IEEE802.11 wireless LAN standard. RC4 is a byte-oriented stream cipher, symmetric key algorithm, in which a byte (8 bits) of a plaintext is execlusive-ored with a byte of key to produce a byte of a ciphertext.

The same algorithm is used for both encryption and decryption as the stream is simply XORed with the generated key sequence .The secrete key , from which the one-byte keys in the key stream are generated, can contain anywhere from 1 to 256 bytes.RC4 is based on the concept of a state . At each moment, a state of 256 bytes is active, from which one of the bytes is randomly selected to serve as the key for encryption.

Array of bytes: S[0] S[1] S[2] … S[255]

The elements range between 0 and 255. The content of each element is also a byte (8 bits) that can be interpreted as an integer between 0 to 255.

Initialization: Initialization is done in two steps;

(i) In first step, the state is initialized to values0,1,…,255. A key arrays, K[0],K[1],…,K[255] is also created. If the secrete key has exactly 256 bytes, the bytes are copied to the K array; otherwise, the bytes are repeated until the K array is filled.

```
for (i= 0 to 255)
{
        S[i] ← i
        K[i] ← Key [i mod Key Length]
}
```

(ii) In the second step, the initialized state goes through a permutation (swapping the elements) based on the value of the bytes in K[*i*].The key byte is used only in this step to define which elements are to be swapped.

After this step, the state bytes are completely shuffled.

```
        j ← 0
        for (i = 0 to 255)
{
        j ← (j +S[i] + K[i]) mod 256
        swap ← (S[i] , S[j])
}
```

Key Stream Generation: The keys in the key stream, the *k*'s are generated, one by one.

(i) First, the state is permutated based on the values of state elements and the values of two individual variables, *i* and *j*.

(i) Second, the values of two statements in positions *i* and *j* are used to define the index of the state element that serves as *k.* The following code is repeated for each byte of the plaintext to create a new key element in the key stream.

The variable *i* and *j* are initialized to 0 before the first iteration, but the values are copied from one iteration to the next.

```
i ← (i + 1) mod 256
j ← (j + S[i]) mod 256
swap (S[i] , S[j])
k ← S [(S[i] + S[j]) mod 256]
```

Encryption or Decryption: After k has been created, the plaintext byte is encrypted with k to create the ciphertext byte. Decryption is the reverse process.

4.2. Integrity Check Algorithm (CRC32)

A cyclic redundancy check (CRC32) is non-secure hash function to detect raw computer data and in digital networks. It is easy to generate message. CRC calculates a short, fixed-length binary sequence, known as the CRC code or just CRC. It accepts data streams of any lengths as input but always outputs a fixed length code. IEEE-recommended 32-bit CRC used in Ethernet.

A Basic Algorithm for CRC Computation

Let P be the binary representation of the divisor polynomial, of degree N, and so N+1 bits long. Let B be a message, with N extra zero bits added. R will be the remainder, N bits long (really it should be N+1 but it's always safe to ignore the last bit).

(1) let R be 0.

(2) If there are no more bits in B, the answer is R. Stop.
(3) Shift R left by one. The new bit 0 We should be the next bit of the Message also shift a bit out of the left of R.
(4) If this bit is 0 go back to 2.
(5) Otherwise it is 1. XOR R with the bottom N bits of P and go back to step2 .

Its computation resembles a long division operation in which the quotient is discarded and the remainder becomes the result, with the important distinction that the arithmetic used is the carry-less arithmetic of a finite field. The Length of the remainder is always less than or equal to the length of the divisor. CRCs can be constructed any finite field, all commonly used CRCs employ the finite field GF(2).

4.3. The Proposed System

The proposed system combines the symmetric key algorithm RC4 stream cipher and integrity check algorithm CRC32 checksum to get more secure data for variable file types such as text file, image file and media file. This system can propose variety of file types text files (.pdf, .doc, .rtf, .ptt, .txt), image files (.jpeg, .bmp, .png) and media file (mp3) and so on. If sender send the cipher to the receiver, we will be defined the shared keys is used as the input to the RC4 key scheduling-algorithm to initialize RC4's internal pseudo-random number generator. The generator outputs key stream to be used for XOR'ing with the plaintext. The output is the ciphertext. The decryption process is vice versa. In order to validate the authentically of data, a 4-byte value called the ICV (integrity check value) is added into the plaintext before going through the encryption process. The system involves two main parts; sender side and receiver side.

On the sender side, the RC4 gets re-initialized the fixed shared key. The plaintext is hashed by CRC32 hash algorithm and generates the hash values with 8 hexacode. And then that plaintext is encrypted by RC4 encryption algorithm. The output is the ciphertext. The ciphertext including CRC code that is beginning of the ciphertext is sent to the receiver. This process is depicted in Figure 4.1.

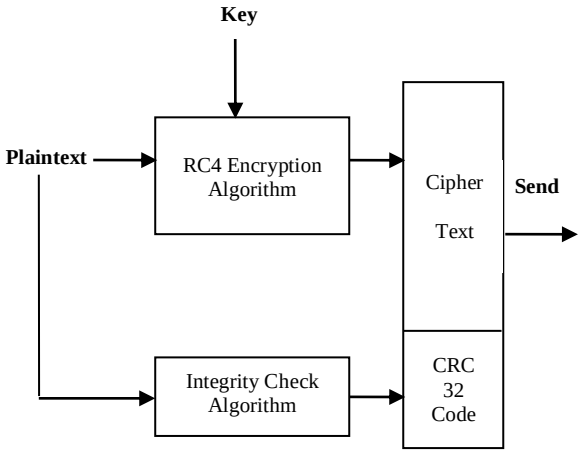


Figure 1. The proposed system of Encryption Process.

On the receiver side, the receiver receives that cipher and hash value. Firstly, the receiver computes the hash value for cipher file by CRC32 algorithm. And then the cipher is decrypted by RC4 decryption algorithm. The keys of RC4 are used both the sender and receiver for RC4 random number generator, and the same keystreams are generated. The original file is received by XOR'ing these keystreams with the cipher file. If the ICV in encryption process is notequal to the ICV in decryption process, the error message is appeared. Else if the original file is obtained. The process is depicted in Figure 2.

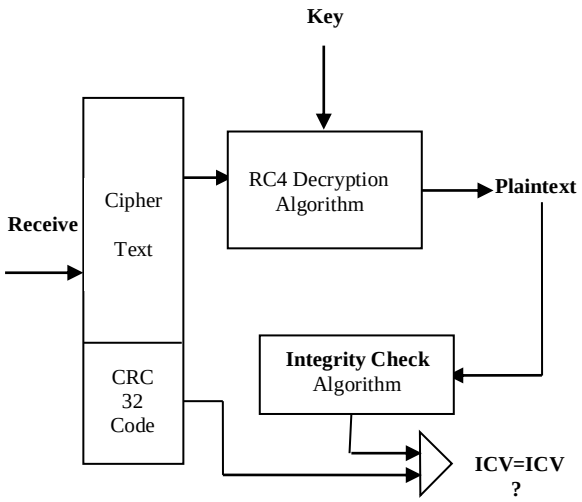


Figure 2. The proposed system of Decryption Process

5. Experimental Results

This system is tested on the encrypting and decrypting of text files, image files and media files. The experimental results show that the performance

of the combination of RC4 symmetric encryption and CRC32 Checksum algorithm. The performance of system is tested on the Desktop computer Intel(R) Core2Duo CPU 2.80GHZ, 160HDD, 1GB of RAM, Microsoft Window XP Professional Version 2002 Service Park 2. Following tables is shown before encryption and after encryption in milliseconds as an example. The table 1 shows the encryption and decryption times (in milliseconds) for text files(pdf).Then, the results of the encryption and decryption times for image files is shown in table 2 (jpg).

Table 1. The Encryption and Decryption Times for Text Files (PDF).

File Sizes	Encryption Times(ms)	Decryption Times(ms)
10KB	15	15
100KB	62	65
200KB	93	97
300KB	156	162
400KB	187	195
500KB	250270	270
1MB	765	830

Table 2. The Encryption and Decryption Times for Image Files (JPEG).

File Sizes	Encryption Times(ms)	Decryption Times(ms)
10KB	218	237
100KB	343	375
200KB	457	487
300KB	484	531
400KB	4140	4500
500KB	5004	5200
1MB	5580	6109

The table 3 shows the encryption and decryption times for media files (mp3).

Table 3. The Encryption and Decryption Times for Media Files (Mp3).

File Sizes	Encryption Times(ms)	Decrypton Times(ms)
10KB	45	46
100KB	78	81
200KB	93	103
300KB	156	169
400KB	185	198
500KB	234	254
1MB	672	694

The experimental results of the encryption times is shown in figure 3.

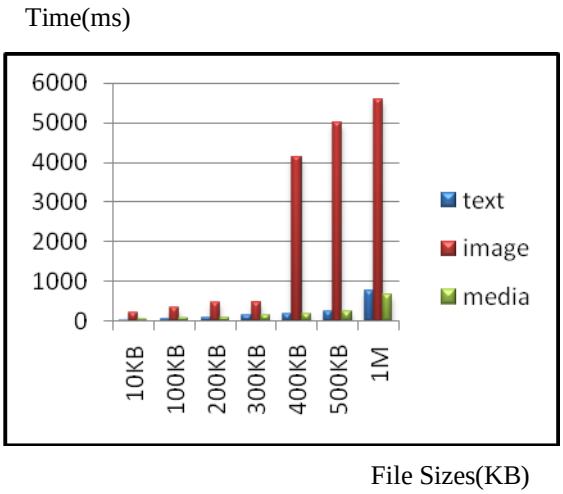
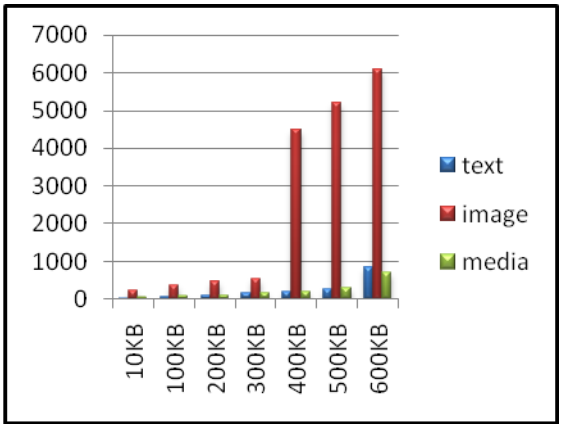


Figure 3. The Experiential results of the encryption times for text, image and media files.

The (x) axis is the size of file to be encrypted in kilo bytes (KB) and (y) axis is the time for encryption in milliseconds. The figure 4 involves the experimental result of the decryption times for texts, images and media files by analyzing with this proposed system.

Time(ms)



File Sizes(KB)

Figure 4. The Experimental results of the decryption times for text, image and media files.

The (x) axis is the size of file to be decrypted in kilo bytes (KB) and (y) axis is the time for encryption in milliseconds. The performance of RC4 is tested based on the processing time under certain variable file types.

6. Conclusion

The symmetric key algorithm RC4 is a shared key stream cipher algorithm, which requires a secure exchange of a shared key that is outside the specification. The algorithm is serial as it requires successive exchanges of state entire based on the key sequence. This algorithm process with a key size of up to 2048 bits (256bytes), to be a relatively fast and strong cipher. The design of the CRC polynomial depends on what is the maximum total length of blocks to be protected (data combine CRC bits). The performance of system may vary according to the processor and software used to implement the system. The performance of the RC4 is tested based on the processing time under certain variable file types. By analyzing the experimental results, the encryption and decryption time of the system are depends on the file sizes. If file sizes are large, the processing time is large. The system combines the stream cipher RC4 for confidentiality and CRC32 checksum for integrity to be more secure for any format file types.

References

[1] Allam Mouse and Ahmad Hamad, "Evaluation of the RC4 Algorithm for Data Encryption," Electrical Engineering Department Dn-Najah University, Nablus, Palestine, Systems Engineer Pal Tel Company Nablus Palestine.

[2] Andrew Roos, "A Class of Weak Keys in the RC4 Stream Cipher" Preliminary Draft, 22 September,1995.

[3] B.A Forouzan, "Cryptography and Network Security", International Edition, 2008.

[4] Donghoon Chang , Kishan Chand Gupta and Mridul Nandi,"RC4_Hash:A New Hash Function based on RC4 (Extended Abstract)" Center of Information Security Technologies(CIST), Korea University ,Korea, Department of Combinations and Optimization, University of Waterloo, Canada, David R. Cheriton School of Computer Science, University of Waterlo, Canada.

[5] D.R. Stinson, "Cryptography: Theory and Practice CRC Press ", INC.,Boca Raton , FL, USA ,1993.

[6] Dr. Wen-Ping Ying ,"Key Hopping _ A Security Enhancement Scheme For IEE802. 11Wep Standards" Director of Software Development, February 2002.

[7] Fhlurer, I Martin, A Shamir, "Weakness in the key Scheduling Algorithm of RC4" Selected Areasin Cryptography, Springer–Verlag, 2001.

[8] Grosul A.L, Wallach D.S, "A related key cryptanalysisofRC4", 2000.

[9] http://www. Wikipidea.org.

[10] Lae Lae Khin, "A New Variant of RC4 Stream Cipher".

[11] M. Liskov, R.Rivest and D. Wager, "The weakable Block Cipher,: Crypto 2002 PDF.

[12] Mantin I. Shamir A. "A practical attack on broadcast RC4" Proceeding of FSE. 2001.

[13] Mathew E. NcKague, "Design and Analysis of RC4 like Stream Ciphers", Mathew E.McKague, 2005.

[14] Michael Pseudorandomness and Cryptographic Applications, Princeton Uni Press, 1996.

[15] Moon K . Chetry and W . B. Vasantha, "A Note On Self-Shrinking Lagged Fibonacci Generator" ,Department of Mathematics Indian Institute of Technology , Madras.

[16] Ronalrd Rivest, Email encryption using AES.

[17] Sean Whalen, "Analysis of Wep and Rlike Key Stream Grnerator".

[18] W.Stalling, "Cryptography and Network Security", Principles and Practices, Fourth Edition, 2006.

.