A Novel Approach Cryptography by using Residue Number System

Ahmad Habibizada Navin
Department of Computer Engineering
Islamic Azad University Tabriz Branch
Tabriz, Iran
ah_habibi@iaut.ac.ir

Asghar Shahrzad Khashandarag
Young researchers Club Tabriz Branch
Islamic Azad University Tabriz Branch
Tabriz, Iran
a.shahrzad@iaut.ac.ir

Amin Rahimi Oskuei*
Department of Computer Engineering
Islamic Azad University Tabriz Branch
Tabriz, Iran
rahimi.amin@iaut.ac.ir

Mirkamal Mirnia
Department of Mathematics Science
University of Tabriz
Tabriz, Iran
mirnia-kam@tabrizu.ac.ir

Abstract—in this research, we aim to encrypt secret information with high security. In our approach, Residue Number System (RNS) is used to encrypt and Huffman coding and Lempel–Ziv–Welch (LZW) compression algorithm are used to compress information. In the embedding process, Data Encryption Standard (DES) algorithm is used to earn high security.

Index Terms—Cryptography, Residue Number System (RNS), Huffman coding, Data Encryption Standard (DES), LZW technique.

I. INTRODUCTION

Now a day, information security is very important. Information that you sent, how and under what circumstances to reach the receiver, is not clear to us. Therefore, information security will be important to you. Due to growth and progress of science in the world, computers and the Internet to send data are used. Therefore the information, as more digital data will be. Today, Image, sound, and etc are known as digital data and we can in addition to information and the digital data to hide the main recipient, we send. Cryptography is the practice and study of techniques for secure communication in the presence of third parties (called adversaries) [1]. In the Figure 1, the structure of cryptography is shown. Cryptography techniques [2-7] are used to encrypt digital data so that no one can find it. Steganography techniques [8-10] are used to hide digital data in coverage media (such as image and sound) so that no one can find it. More generally, it is about constructing and analyzing protocols that overcome the influence of adversaries and which are related to various aspects in information security such as data confidentiality, data integrity, and authentication. Modern cryptography intersects the disciplines of mathematics, computer science, and electrical engineering. Applications of cryptography include ATM cards, computer passwords, and electronic commerce [1].

Cryptography prior to the modern age was almost synonymous with encryption, the conversion of information from a readable state to apparent nonsense. The sender retained the ability to decrypt the information and therefore avoid unwanted persons being able to read it. Since World War I and the advent of the computer, the methods used to carry out cryptology have become increasingly complex and its application more widespread [1]. Modern cryptography follows a strongly scientific approach, and designs cryptographic algorithms around computational hardness assumptions, making such algorithms hard to break by an adversary. Such systems are not unbreakable in theory but it is infeasible to do so by any practical means. These schemes are therefore computationally secure. There exist information-theoretically secure schemes that probably cannot be broken—an example is the one-time pad—but these schemes are more difficult to implement than the theoretically breakable but computationally secure mechanisms [1].

II. RELATED WORKS

Lejla Batina and her cooperators by using RNS techniques and Hardware Architectures have proposed a method for Public Key Cryptography [11].

A. Residue Number System

A residue number system (RNS) [12-13] represents a large integer using a set of smaller integers, so that computation may be performed more efficiently. It relies on the Chinese remainder theorem of modular arithmetic for its operation, a mathematical idea from Sun Tzu Sun-Ching (Master Sun’s Arithmetic Manual) in the 4th century AD. In the residue number system, a set of moduli which are independent of each other is given. An integer is represented by the residue of each modulus and the arithmetic operations are based on the residues individually. The arithmetic operations based on residue number system can be performed on various moduli independently to avoid the carry in addition, subtraction and multiplication, which is usually time consuming.
However, the comparison and division are more complicated and the fraction number computation is immature. Due to this, a residue number system is not yet popular in general-purpose computers, though it is extremely useful for digital-signal-processing applications. This thesis deals with the design, simulation and microcontroller implementation of some (residue number system based) building blocks for applications in the field of digital signal processing. The building blocks which have been studied are binary to residue converter, residue to binary converter, residue adder and residue multiplier. New algorithms were also introduced. A residue number system is defined by a set of N integer constants, 

\[
\{m_1, m_2, m_3, \ldots, m_N\},
\]

referred to as the moduli. Let M be the least common multiple of all the \(m_i\). Any arbitrary integer \(X\) smaller than \(M\) can be represented in the defined residue number system as a set of \(N\) smaller integers 

\[
\{x_1, x_2, x_3, \ldots, x_N\}
\]

with 

\[
x_i = X \mod m_i,
\]

representing the residue class of \(X\) to that modulus. Note that for maximum representational efficiency it is imperative that all the moduli are coprime; that is, no modulus may have a common factor with any other. \(M\) is then the product of all the \(m_i\) [12].

B. Huffman coding

In computer science and information theory, Huffman coding is an entropy encoding algorithm used for lossless data compression [14]. The term refers to the use of a variable-length code table for encoding a source symbol (such as a character in a file) where the variable-length code table has been derived in a particular way based on the estimated probability of occurrence for each possible value of the source symbol. It was developed by David A. Huffman while he was a Ph.D. student at MIT, and published in the 1952 paper "A Method for the Construction of Minimum-Redundancy Codes".

Huffman coding uses a specific method for choosing the representation for each symbol, resulting in a prefix code (sometimes called "prefix-free codes", that is, the bit string representing some particular symbol is never a prefix of the bit string representing any other symbol) that expresses the most common source symbols using shorter strings of bits than are used for less common source symbols. Huffman was able to design the most efficient compression method of this type: no other mapping of individual source symbols to unique strings of bits will produce a smaller average output size when the actual symbol frequencies agree with those used to create the code. A method was later found to design a Huffman code in linear time if input probabilities (also known as weights) are sorted [14].

For a set of symbols with a uniform probability distribution and a number of members which is a power of two, Huffman coding is equivalent to simple binary block encoding, e.g., ASCII coding. Huffman coding is such a widespread method for creating prefix codes that the term "Huffman code" is widely used as a synonym for "prefix code" even when such a code is not produced by Huffman's algorithm [14].

C. LZW Compression algorithm

Lempel–Ziv–Welch (LZW) is a universal lossless data compression algorithm created by Abraham Lempel, Jacob Ziv, and Terry Welch. It was published by Welch in 1984 as an improved implementation of the LZ78 algorithm published by Lempel and Ziv in 1978. The algorithm is designed to be fast to implement [15].

A common choice is to provide 4096 entries in the table. In this case, the LZW encoded data consists entirely of 12 bit codes, each referring to one of the entries in the code table. Uncompressing is achieved by taking each code from the compressed file, and translating it through the code table to find what character or characters it represents. Codes 0-255 in the code table are always assigned to represent single bytes from the input file. For example, if only these first 256 codes were used, each byte in the original file would be converted into 12 bits in the LZW encoded file, resulting in a 50% larger file size. During uncompressing, each 12 bit code would be translated via the code table back into the single bytes. Of course, this wouldn't be a useful situation [15].

D. Data Encryption Standard

The Data Encryption Standard (DES) is a block cipher that uses shared secret encryption [16]. It was selected by the National Bureau of Standards as an official Federal Information Processing Standard (FIPS) for the United States in 1976 and which has subsequently enjoyed widespread use internationally. It is based on a symmetric-key algorithm that uses a 56-bit key.
The algorithm was initially controversial because of classified design elements, a relatively short key length, and suspicions about a National Security Agency (NSA) backdoor. DES consequently came under intense academic scrutiny which motivated the modern understanding of block ciphers and their cryptanalysis. DES is now considered to be insecure for many applications. This is chiefly due to the 56-bit key size being too small; in January, 1999, distributed.net and the Electronic Frontier Foundation collaborated to publicly break a DES key in 22 hours and 15 minutes (see chronology) [16].

III. PROPOSED METHOD

A. Data-embedding algorithm

A cryptography scheme must be extremely secure. Within the context of any application-to-application communication, there are some specific security requirements, including:

- Authentication
- Privacy/confidentiality
- Integrity
- Non-repudiation

Cryptography, then, not only protects data from theft or alteration, but can also be used for user authentication.

The overall concealing process of our proposed scheme is shown in Figure 2. The Secret information (SI) is compression by LZW technique. The Result of compression stages, with pseudo-random number is Exclusive-or (⊕), and then bit streams are split into three parts ($x_1$, $x_2$, and $x_3$) by using RNS technique and then DES is used for security. Finally they are merged and huffman algorithm (HF) is used to compress the result.

In Tables 1, two strings as secret information are shown. In this example, $m_1$ and $m_2$ and $m_3$ equals 17, 18 and 19 are assumed.

In string 1 (#1) before LZW compression:

$$bits = 16 \times 8 = 128$$

Number 16 is number of characters. After LZW compression:

$$bits = 7 \times 12 = 84$$

Number 7 is number of characters.

B. Inverse Data-embedding algorithm

The process of inverse data embedding is shown in Figure 3. Note that, in the inverse data-embedding algorithm, $m_1$ and $m_2$ and $m_3$ are public key.
TABLE I
The Example of Embedding Algorithm.

<table>
<thead>
<tr>
<th>#</th>
<th>Secret Information (SI)</th>
<th>The result of LZW stage</th>
<th>PSN</th>
<th>(\oplus)</th>
<th>(x_1)</th>
<th>(x_2)</th>
<th>(x_3)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>aaabbbbbbaaba</td>
<td></td>
<td>011</td>
<td>1111</td>
<td>5</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1111111000111</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>BABAABAAA</td>
<td></td>
<td>011</td>
<td>1111</td>
<td>6</td>
<td>9</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1111111000111</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CONCLUSIONS
In this research, we aimed to encrypt secret information with high security. In our approach, Residue Number System (RNS) is used to encrypt and Huffman coding and Lempel–Ziv–Welch (LZW) compression algorithm are used to compress information. In the embedding process, Data Encryption Standard (DES) algorithm is used to earn high security.

REFERENCES