Journal of Artificial Intelligence and Engineering Applications

Website: https://ioinformatic.org/

15th October 2023. Vol. 3. No. 1; e-ISSN : 2808-4519

Digital Signature Security Analysis By Applying The Elgamal Algorithm And The Idea Method

Rizka Khairani Lubis[1], A M H Pardede [2], Husnul Khair [3]

[1,2,3] Information Systems, STMIK Kaputama
Jl. Veterans No. 4A-9A, Binjai, North Sumatra, Indonesia
rizkakhairanilbs@gmail.com[1], akimmhp@live.com [2], husnul.khair@gmail.com [3]*

Abstract

The development of an all-digital era, all activities use digital technology. Including signatures, hands are no longer manual now, signatures can be modified digitally. The application of digital signatures can be used to allow for document authenticity issues. The signature combines two methods namely Elgamal Algorithm and IDEA. The Elgamal algorithm is used to encrypt and decrypt signatures. The IDEA algorithm is used to strengthen signatures so that others don't modify them. The signature process begins with generating the public and private keys. The process of generating public keys (p,q,g,y) and private keys. With the signature of a binary document that is given by the sender to the recipient, the authenticity of the contents of the file, the identity of the sender, and the files that have been received by the recipient can be guaranteed. Affixing a digital signature is done by encrypting it with the sender's private key. In this way, checking the authenticity of documents and senders can be done.

Keywords: *Algorithm, Elgamal, Idea, Digital signature*

1. Introduction

As technology develops, the sending of a message also becomes less secure, so important data or documents will be secured by the user so that the data is not stolen or copied by other users[1]. Signature is a medium that is used as an identity for verification and legalization of information. Problems that often occur in digital signatures such as manipulating signatures, and scanning signatures without the knowledge of the owner of the sign. Then a signature is verified, whether the signature is really genuine (valid) or forged by another user[2]. The digital signature serves to ensure that the document is original and has never been modified. Digital signatures can be used in utilizing a cryptographic technique. Wet signatures are scanned and then embedded into a document. If a document is re-saved, the digital signature that has been created will be lost. This causes the document to become no longer original.

2. Theoretical basis

2.1 Definition of Cryptography

Cryptography is a technique for protecting information in communications, where information is changed with a certain key through encryption so that it becomes new information that cannot be understood by people who are not entitled to receive it, and that information can only be changed again by people who are entitled to receive it through decryption.

2.2 Definition of Digital Signature

Signatures begin to be modified digitally, digital signatures are user signatures that are verified to prove the authenticity of the signature, whether it is really valid. A signature can provide additional assurance for proof of origin, identity, status of an electronic document, transaction or message and can acknowledge the signatory's informed consent.

2.3 Definition of Elgamal Algorithm

Elgamal Algorithm decryption process consists of three processes namely the key formation process, the encryption process and the decryption process. The security of the Elgamal algorithm lies in the difficulty of calculating discrete logarithms on large prime modules,

so attempts to solve logarithmic problems are difficult to solve. The decryption process takes a long time due to the complexity of the complicated decryption process, twice the computation is required because the size of the ciphertext is larger than the plaintext.

2.4 Understanding the IDEA Algorithm

The IDEA algorithm is a block cipher (block cipher) that operates on 64-bit plaintext blocks. The key length is 128 bits. The same algorithm is used for the encryption and decryption processes. IDEA uses confusion and diffusion, in contrast to DES which uses permutation and As a substitute for confusion and diffusion.

2.5 ASCII

ASCII is used for information exchange and data communication. ASCII is a numeric code that represents a character. ASCII is used because computers only understand numbers. The ASCII function is to read characters in binary code where the numbers are 1 and 0. The series of 1 and 0 has meaning and is a code. For example, the sequence 01000001 (reading computer) has the meaning A.

2.6 ASCII Function

ASCII is used to represent numeric and alphabetic characters on the computer. The standard character encoding is to break all characters and define each character as a string of bits. ASCII can convert 1 bit to lowercase or uppercase. The binary value for the lower case "a" is 0110 0001. The binary value for the capital "A" is 0100 0001.

2.7 Flowchart Symbol

Basically, in designing flowcharts there are no absolute conditions that must be met. That's because flowcharts are made based on the idea of analyzing a problem in business. It's just that, you can design flowcharts when you already know the standard symbols that are commonly used in the process of making flowcharts. The flowchart symbols commonly used are the standard flowchart symbols issued by ANSI and ISO.

2.8 Unified Modeling Languange (UML)

UML (Unified Modeling Language) is a method in visual modeling that is used as a means of designing object-oriented systems. UML is very important for some people because UML functions as a bridge or interpreter bridge between system developers and users. Users can understand the system that will later be developed. UML is a visual language for modeling and communicating about a system using diagrams and supporting texts. UML only functions for modeling (Simatupang & Sianturi, 2019).

2.9 Visual Basic

Visual Basic is a programming language developed by Microsoft. Microsoft Visual Basic provides components that make it possible to create application programs that match the way things work on Windows displays. Some of the related interfaces in Microsoft Visual Basic are: Program menu section,Toolbars section, Toolbox parts, Section properties window.Forms section, Section of the project window. Section of the window lay-out.

3. Result and Discussion

3.1. General Overview

Cryptography is often used in securing document files, text messages, and images. Cryptography is very important in keeping data confidential from unauthorized persons so they cannot find out what the data contains. In this study the author will create an application to secure image files. The Elgamal Algorithm and the Idea Algorithm aim to secure signatures by combining the two methods, resulting in an algorithm that is not easily broken by its secrecy.

3.2. Problem Analysis

A digital signature is a signature that is often used in an all-digital era. The security of digital signatures is confidential which needs to be maintained for the authenticity of the signature, it can only be used by authorized parties. Therefore the authors recommend the Elgamal Algorithm and the Idea Algorithm to be able to provide security for digital signatures. The reason the author uses the Elgamal Algorithm and the Idea Algorithm, the Elgamal Algorithm and the Idea Algorithm are part of asymmetric cryptography. Formation of the key used is a prime number, by utilizing prime numbers and numbers binary that can resolve as well as secure cryptographic keys in digital signatures.

3.3. Research Methodology

In the Research Methodology used in the process of making the thesis are as follows :

1. Literature study on the Elgamal algorithm and the IDEA algorithm and several techniques that must be carried out to use these algorithms in classical cryptography through journals and sites on the internet[3].

2. Implementation of the Elgamal algorithm and the IDEA algorithm into the Visual Basic Net 2010 programming language.
3. Revise the design of the program that experienced an error (error)[4].

3.4 IDEA Encryption Algorithm

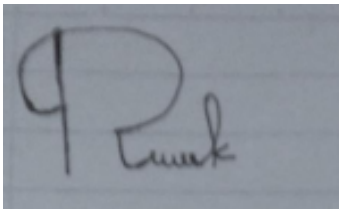

Fig 1. signature image

The signature image is changed to a binary application, so the binary results obtained are as follows:
0000010000000001 0000000001000000 0001000000000100 0000000100000000 0100000000010000 1000000000100000
Plainteks : 0100101101010101 0100111001000011 0100100100100011 1111001000111001 0110001111001010 1100001110100100

Round 1
01) L#1 = (X1*K1) mod (2^16+1) = 0000010000000001*0100101101010101 mod (2^16+1) = 0000001000011001
02) L#2 = (X2+K2) mod 2^16 = 0000000001000000 + 0100111001000011 mod 2^16 = 0100111000100011
03) L#3 = (X3+K3) mod 2^16 = 0001000000000100 + 0100100100100011 mod 2^16 = 0101100100100111
04) L#4 = (X4+K4) mod (2^16+1) = 0000000100000000 + 1111001000111001 mod (2^16+1) = 1111001100111001
05) L#5 = L#1 XOR L#3 = 0000001000011001 XOR 0101100100100111 = 0101101100111110
06) L#6 = L#2 XOR L#4 = 0100111000100011 XOR 1111001100111001 = 1011110100011010
07) L#7 = (L#5*K5) mod (2^16+1) = 0101101100111110*0110001111001010 mod (2^16+1) = 0001000001110101
08) L#8 = (L#6 + L#7) mod 2^16 = 1011110100011010 + 0001000001110101 mod 2^16 = 1010001101100001
09) L#9 = (L#8*K6) mod (2^16+1) = 1010001101100001*1100001110100100 mod (2^16+1) = 1000110011111011
10) L#10 = (L#7 + L#9) mod 2^16 = 0001000001110101 + 1000110011111011 mod 2^16 = 1001110010110001
11) L#11 = L#1 XOR L#9 = 0000001000011001 XOR 1000110011111011 = 1000111011100010
12) L#12 = L#3 XOR L#9 = 0101100100100111 XOR 1000110011111011 = 1001010111011100
13) L#13 = L#2 XOR L#10 = 0100111000100011 XOR 1001110010110001 = 1101001010010010
14) L#14 = L#4 XOR L#10 = 1111001100111001 XOR 1001110010110001 = 0110111110001000
For the following rounds:
X1 = L#11 = 1000111011100010
X2 = L#12 = 1001010111011100
X3 = L#13 = 1101001010010010
X4 = L#14 = 0110111110001000

Round 2
01) L#1 = (X1*K1) mod (2^16+1) = 1000111011100010*0100101101010101 mod (2^16+1) = 0111000001010111
02) L#2 = (X2+K2) mod 2^16 = 1001010111011100 + 0100111001000011 mod 2^16 = 1101100001111111
03) L#3 = (X3+K3) mod 2^16 = 1101001010010010 + 0100100100100011 mod 2^16 = 1011101110110001
04) L#4 = (X4+K4) mod (2^16+1) = 0110111110001000 + 1111001000111001 mod (2^16+1) = 1100001001110101
05) L#5 = L#1 XOR L#3 = 0111000001010111 XOR 1011101110110001 = 1100101111100110
06) L#6 = L#2 XOR L#4 = 1101100001111111 XOR 1100001001110101 = 0001101000001010
07) L#7 = (L#5*K5) mod (2^16+1) = 1100101111100110*0110001111001010 mod (2^16+1) = 0100000110011101
08) L#8 = (L#6+L#7) mod 2^16 = 0001101000001010 + 0100000110011101 mod 2^16 = 0101101110010001
09) L#9 = (L#8*K6) mod (2^16+1) = 0101101110010001*1100001110100100 mod (2^16+1) = 0100111010111011
10) L#10 = (L#7+L#9) mod 2^16 = 0100000110011101 + 0100111010111011 mod 2^16 = 0010111101101001
11) L#11 = L#1 XOR L#9 = 01110000 01010111 XOR 0100111010111011 = 0011111011101100
12) L#12 = L#3 XOR L#9 = 1011101110110001 XOR 0100111010111011 = 1111010100001010
13) L#13 = L#2 XOR L#10 = 1101100001111111 XOR 0010111101101001 = 1111011100010110
14) L#14 = L#4 XOR L#10 = 1100001001110101 XOR 0010111101101001 = 1110110100011100
For the following rounds:
X1 = L#11 = 0011111011101100
X2 = L#12 = 1111010100001010
X3 = L#13 = 1111011100010110
X4 = L#14 = 1110110100011100

Round 3
01) L#1 = (X1*K1) mod (2^16+1) = 0011111011101100*0100101101010101 mod (2^16+1) = 1100110101101101
02) L#2 = (X2+K2) mod 2^16 = 1111010100001010 + 0100111001000011 mod 2^16 = 1000010011001001
03) L#3 = (X3+K3) mod 2^16 = 1111011100010110 + 0100100100100011 mod 2^16 = 1000000110110101
04) L#4 = (X4+K4) mod (2^16+1) = 1110110100011100 + 1111001000111001 mod (2^16+1) = 0001000010101011
05) L#5 = L#1 XOR L#3 = 1100110101101101 XOR 1000000110110101 = 0100110011011000
06) L#6 = L#2 XOR L#4 = 1000010011001001 XOR 0001000010101011 = 1001010001100010
07) L#7 = (L#5*K5) mod (2^16+1) = 0100110011011000*0110001111001010 mod (2^16+1) = 0001100011101011

08) L#8 = (L#6+L#7) mod 2^16 = 1001010001100010 + 0100000110011101 mod 2^16 = 1101110111111111
09) L#9 = (L#8*K6) mod (2^16+1) = 1101110111111111*1100001110100100 mod (2^16+1) = 1001110100110101
10) L#10 = (L#7+L#9) mod 2^16 = 0001100011101011 + 0100111010111011 mod 2^16 = 0101000100101101
11) L#11 = L#1 XOR L#9 = 1100110101101101 XOR 1001110100110101 = 0101000001011000
12) L#12 = L#3 XOR L#9 = 1000000110110101 XOR 1001110100110101 = 0001110010000000
13) L#13 = L#2 XOR L#10 = 1000010011001001 XOR 0101000100101101 = 1101010111100100
14) L#14 = L#4 XOR L#10 = 0001000010101011 XOR 0101000100101101= 0100000110000110
For the following rounds:
X1 = L#1 = 0101000001011000
X2 = L#2 = 0001110010000000
X3 = L#3 = 1101010111100100
X4 = L#4 = 0100000110000110

Round 4
01) L#1 = (X1*K1) mod (2^16+1)= 0101000001011000*0100101101010101 (2^16+1) = 1010101101110111
02) L#2 = (X2+K2) mod 2^16 = 0001110010000000 + 0100 1110 0100 0011 mod 2^16 = 0010101011000011
03) L#3 = (X3+K3) mod 2^16 = 1101010111100100 + 0100100100100011 mod 2^16 = 1011110000110111
04) L#4 = (X4+K4) mod (2^16+1) = 0100000110000110 + 1111001000111001 mod (2^16+1) = 1111001110111111
05) L#5 = L#1 XOR L#3 = 1010101101110111 XOR 1011110000110111 = 0001011101000000
06) L#6 = L#2 XOR L#4 = 0010101011000011 XOR 1111001110111111 = 1101100101111100
07) L#7 = (L#5*K5) mod (2^16+1) = 0001011101000000*0110001111001010 mod (2^16+1) = 1010111100010111
08) L#8 = (L#6+L#7) mod 2^16 = 1101100101111100 + 1010111100010111 mod 2^16 = 0000100111100101
09) L#9 = (L#8*K6) mod (2^16+1) = 0000100111100101*1100001110100100 mod (2^16+1) = 1010000000000001
10) L#10 = (L#7+L#9) mod 2^16 = 1010111100010111 + 1010000000000001 mod 2^16 = 0001110101110001
11) L#11 = L#1 XOR L#9 = 1010101101110111 XOR 1010000000000001 = 0000101101110110
12) L#12 = L#3 XOR L#9 = 1011110000110111 XOR 1010000000000001 = 0001110000110110
13) L#13 = L#2 XOR L#10 = 0010101011000011 XOR 0001110101110001= 0011011110110010
14) L#14 = L#4 XOR L#10 = 1111001110111111 XOR 0001110101110001 = 1110010011001110
For the following rounds:
X1 = L#11 = 0000101101110110
X2 = L#12 = 0001110000110110
X3 = L#13= 0011011110110010
X4 = L#14 = 1110010011001110

Round 5
01) L#1 = (X1*K1) mod (2^16+1) = 0000101101110110*0100101101010101 mod (2^16+1) = 0000110100111101
02) L#2 = (X2+K2) mod 2^16 = 0001110000110110 + 0100111001000011 mod 2^16 = 0101010101110101
03) L#3 = (X3+K3) mod 2^16 = 0011011110110010 + 0100100100100011 mod 2^16 = 0111111001001001
04) L#4 = (X4+K4) mod (2^16+1) = 1110010011001110 + 1111001000111001 mod (2^16+1) = 0010111011110001
05) L#5 = L#1 XOR L#3 = 0000110100111101 XOR 0111111001001001 = 0111001101110100
06) L#6 = L#2 XOR L#4 = 0101010101110101 XOR 0010111011110001 = 0111101110000100
07) L#7 = (L#5*K5) mod (2^16+1) = 0111001101110100*0110001111001010 mod (2^16+1) = 0010100101000001
08) L#8 = (L#6+L#7) mod 2^16 = 0111101110000100 + 0010100101000001 mod 2^16 = 0100110000100101
09) L#9 = (L#8*K6) mod (2^16+1) = 0100110000100101*1100001110100100 mod (2^16+1) = 0001111011101011
10) L#10 = (L#7+L#9) mod 2^16 = 0010100101000001 + 0001111011101011 mod 2^16 = 0101000001011011
11) L#11 = L#1 XOR L#9 = 0000110100111101 XOR 0001111011101011 = 0001001111010110
12) L#12 = L#3 XOR L#9 = 0111111001001001 XOR 0001111011101011 = 0110000010100010
13) L#13 = L#2 XOR L#10 = 0101010101110101 XOR 0101000001011011 = 0000010100101110
14) L#14 = L#4 XOR L#10 = 0010111011110001 XOR 0101000001011011 = 0111111010101010
For the following rounds:
X1 = L#11 = 0001001111010110
X2 = L#12 = 0110000010100010
X3 = L#13= 0000010100101110
X4 = L#14 = 0111111010101010

Round 6
01) L#1 = (X1*K1) mod (2^16+1) = 0001001111010110*0100101101010101 mod (2^16+1) = 1100000100111001
02) L#2 = (X2+K2) mod 2^16 = 0110000010100010 + 0100111001000011 mod 2^16 = 0001111011110001
03) L#3 = (X3+K3) mod 2^16 = 0000010100101110 + 0100100100100011 mod 2^16 = 0100110010011101
04) L#4 = (X4+K4) mod (2^16+1) = 0111111010101010 + 1111001000111001 mod (2^16+1) = 1011001110001111
05) L#5 = L#1 XOR L#3 = 1100000100111001 XOR 0100110010011101 = 1000110110100100
06) L#6 = L#2 XOR L#4 = 0001111011110001 XOR 1011001110001111 = 1010110101111110
07) L#7 = (L#5*K5) mod (2^16+1) = 1000110110100100*0110001111001010 mod (2^16+1) = 0100010011011111
08) L#8 = (L#6+L#7) mod 2^16 = 1010110101111110 + 0100010011011111 mod 2^16 = 1111011110011111
09) L#9 = (L#8*K6) mod (2^16+1) = 1111011110011111*1100001110100100 mod (2^16+1) = 0000110011100011
10) L#10 = (L#7+L#9) mod 2^16 = 0100010011011111 + 0000110011100011 mod 2^16 = 1110000100000011
11) L#11 = L#1 XOR L#9 = 1100000100111001 XOR 0000110011100011 = 1100110111011010
12) L#12 = L#3 XOR L#9 = 0100110010011101 XOR 0000110011100011 = 0100000001111110
13) L#13 = L#2 XOR L#10 = 0001111011110001 XOR 1110000100000011 = 1111111111110010
14) L#14 = L#4 XOR L#10 = 1011001110001111 XOR 1110000100000011 = 0101001010001100

For the following rounds:
X1 = L#11 = 1100110111011010
X2 = L#12 = 0100000001111110
X3 = L#13 = 1111111111110010
X4 = L#14 = 0101001010001100

Round 7
01) L#1 = (X1*K1) mod (2^16+1) = 1100110111011010*0100101101010101 mod (2^16+1) = 0100110000011101
02) L#2 = (X2+K2) mod 2^16 = 0100000001111110 + 0100111001000011 mod 2^16 = 0010111010000011
03) L#3 = (X3+K3) mod 2^16 = 1111111111110010 + 0100100100100011 mod 2^16 = 1000100100101001
04) L#4 = (X4+K4) mod (2^16+1) = 0101001010001100 + 1111001000111001 mod (2^16+1) = 1001100110110011
05) L#5 = L#1 XOR L#3 = 0100110000011101 XOR 1000100100101001 = 1100010100110100
06) L#6 = L#2 XOR L#4 = 001011101000 0011 XOR 1001100110110011 = 1011011100110000
07) L#7 = (L#5*K5) mod (2^16+1) = 1100010100110100*0110001111001010 mod (2^16+1) = 0111011100000101
08) L#8 = (L#6+L#7) mod 2^16 = 1011011100110000 + 0111011100000101 mod 2^16 = 1101101110110101
09) L#9 = (L#8*K6) mod (2^16+1) = 1101101110110101*1100001110100100 mod (2^16+1) = 1111010011011111
10) L#10 = (L#7+L#9) mod 2^16 = 0111011100000101 + 1111010011011111mod 2^16 = 1011100000111001
11) L#11 = L#1 XOR L#9 = 0100110000011101 XOR 1111010011011111 = 1011100011000010
12) L#12 = L#3 XOR L#9 = 1000100100101001 XOR 1111010011011111 = 0111110111110110
13) L#13 = L#2 XOR L#10 = 001011101000 0011 XOR 1011100000111001 = 1001011010111010
14) L#14 = L#4 XOR L#10 = 1001100110110011 XOR 1011100000111001 = 0010000110001010
For the following rounds:
X1 = L#11 = 1011100011000010
X2 = L#12 = 0111110111110110
X3 = L#13 = 1001011010111010
X4 = L#14 = 0010000110001010

Round 8
01) L#1 = (X1*K1) mod (2^16+1) = 1011100011000010*0100101101010101 mod (2^16+1) = 0111010100001011
02) L#2 = (X2+K2) mod 2^16 = 0111110111110110 + 0100111001000011 mod 2^16 = 0000110001001101
03) L#3 = (X3+K3) mod 2^16 = 1001011010111010 + 0100 1001 0010 0011 mod 2^16 = 1101111110000101
04) L#4 = (X4+K4) mod (2^16+1) = 0010000110001010 + 1111001000111001 mod (2^16+1) = 1100101110110111
05) L#5 = L#1 XOR L#3 = 0111010100001011 XOR 1101111110000101 = 1010101010001110
06) L#6 = L#2 XOR L#4 = 0000110001001101 XOR 1100101110110111 = 1100011111111010
07) L#7 = (L#5*K5) mod (2^16+1) = 1010101010001110*0110001111001010 mod (2^16+1) = 1011111101000001
08) L#8 = (L#6+L#7) mod 2^16 = 1100011111111010 + 1011111101000001 mod 2^16 = 0000011101000111
09) L#9 = (L#8*K6) mod (2^16+1) = 0000011101000111*1100001110100100 mod (2^16+1) = 0011000011101001
10) L#10 = (L#7+L#9) mod 2^16 = 1011111101000001 + 0011000011101001mod 2^16 = 1001001101111001
11) L#11 = L#1 XOR L#9 = 0111010100001011 XOR 0011000011101001 = 0100010111100010
12) L#12 = L#3 XOR L#9 = 1000100100101001 XOR 0011000011101001 = 1011100111000000
13) L#13 = L#2 XOR L#10 = 0000110001001101 XOR 1001001101111001 = 1001111100110100
14) L#14 = L#4 XOR L#10 = 1100101110110111 XOR 1001001101111001 = 0101100011001110
For the following rounds:
X1 = L#11 = 0100010111100010
X2 = L#12 = 1011100111000000
X3 = L#13 = 1001111100110100
X4 = L#14 = 0101100011001110
The results of the IDEA Algorithm encryption will then be combined with the Elgamal Algorithm encryption.

3.5 Elgamal Encryption Algorithm

The Elgamal Algorithm encryption process will be obtained by combining IDEA Algorithm encryption results with Elgamal Algorithm calculations[5]. The following is the result of the IDEA Algorithm encryption: 01000101 = 162, 11100010 = 71, 10111001 = 157, 11000000 = 3, 10011111 = 149, 00110100 = 44, 01011000 = 26, 11001110 = 115. These results will be converted to the ASCII table, the results obtained are as follows:

Table 1: Elgamal Encryption

I	Character	Plainteks Mi	Plaintext mi (ASCII)
1	Oh	M1	162
2	G	M2	71
3	Ø	M3	157
4	EXT	M4	3
5	Ò	M5	149
6	,	M6	44
7	SUB	M7	26
8	S	M8	115

The next step, The process of determining random numbers ∈ {0,1, ... .257} then the ASCII value is entered into the m value blocks sequentially, so that it becomes :

Table 2: ASCII to m value

Mn	Mark	Key

m1	162	3
m2	71	5
m3	157	7
m4	3	9
m5	149	11
m6	44	13
m7	26	15
m8	115	17

Then calculate $y=g^x$ mod p dam $m1=bi.a1^{p-1-x}$ mod p. Let p,g,x be used to calculate y :
p = 257.
g = 17.
x = 11.
Then p,g,x is used to calculate y :
$y = g^x$ against p.
$y = 17^{11}$against 257.
y = 223.
The results of this algorithm are:
The public key is triple (223, 17, 257).
the private key is pair (11, 257).

Table 3: Elgamal Encryption Results

Encryption a **(a = g^k towards p)**	**Encryption b** **(b = y^k.m mod p)**
a1 = 17^7 against 257	b1 = 223^7.162 vs. 257
= 137	= 117
a2 = 17^9 against 257	b2 = 223^9.71 against 257
= 17	= 74
a3 = 17^{15} against 257	b3 = 223^{15}.157 vs. 257
= 136	= 118
a4 = 17^{27} against 257	b4 = 223^{27}.3 against 257
= 34	= 45
a5 = 17^{30} against 257	b5 = 223^{30}.149 vs. 257
= 249	= 54
a6 = 17^{32} against 257	b6 = 223^{32}.44 vs. 257
= 1	= 44
a1 = 17^7 against 257	b1 = 223^7.162 vs. 257
= 137	= 117
a2 = 17^9 against 257	b2 = 223^9.71 against 257
= 17	= 74
a7=17^{35} against 257	b7 = 223^{35} 26 against 257
= 30	= 185
a8 = 17^{40} against 257	b8 = 223^{40} 115 against 257
= 249	= 216

After getting the encryption values a and b, the calculation results are arranged in an alternating pattern:
a1,b1,a2,b2,a3,b3,a4,b4,a5,b5,a6,b6,a7,b7,a8,b8.
So as to form the shape of the cipher text:
137,117,15,74,136,118,34,45,249,54,1,44,30,185,249,216.
In character form it becomes:
ëuSIJêv"-¨6SOH,RS╣Ï

3.6 Elgamal description

Table 4: Convert Characters to ASCII

I	Character	Planteks MI	Plainteks mi ASCII
1	**e**	M1	137
2	**in**	M2	117
3	**AND**	M3	15
4	**J**	M4	74
5	**eh**	M5	136
6	**in**	M6	118
7	"	M7	34
8	**-**	M8	45
9	¨	M9	249
10	**6**	M10	54
11	**SOH**	M11	1
12	**,**	M12	44

13	**RS**	M13	30
14	╣	M14	185
15	¨	M15	249
16	**Ï**	M16	216

Decrypt the ciphertext of b by doing calculations, the following formula:
n = bi.ai^{p-1-x} against 257.
n1 = 117.137^{245} against 257
= 88.
n2 = 15.74^{245} against 257
= 91.
n3 = 116.118$^{257-1-11}$ against 257
= 234.
n4 = 34.45^{245} against 257
= 229.
n5 = 54.249^{245} against 257
= 230.
n6 = 44.1^{245} against 257
= 44.
n7 = 185.30$^{257-1-11}$against 257
= 26.
n8 = 216.249$^{257-1-11}$ against 257
= 149.
After getting the mn value, each m value resulting from decryption becomes an ASCII code converted back into characters[6]. With the following results:

Table 5: Convert ASCII to Character

Plaintex Code ASCII	Character
88	X
91	[
234	And
229	HE
230	m
44	,
26	SUB
149	Ò

3.7 Idea description

The results of the Elgamal Encryption will be described with the Idea Algorithm. 88 = 00001101
91 = 01101101
234 = 01010111
229 = 01010011
230 = 01100111
44 = 00110101
26 = 00001011
146 = 01001001

Round 1
01) L#1 = (X1*K1) mod (2^16+1) = 0000110101101101*0100101101010101 mod (2^16+1) = 0010111100100001
02) L#2 = (X2+K2) mod 2^16 = 0101011101010011 + 0100111001000011 mod 2^16 = 0011101010110001
03) L#3 = (X3+K3) mod 2^16 = 0110011100110101 + 0100100100100011 mod 2^16 = 0001111010001111
04) L#4 = (X4+K4) mod (2^16+1) = 0000101101001001 + 1111001000111001 mod (2^16+1) = 1111100111110101
05) L#5 = L#1 XOR L#3 = 0010111100100001 XOR 0001111010001111 = 0011000110101110
06) L#6 = L#2 XOR L#4 = 0011 1010 1011 0001 XOR 1111 1001 1111 0101 = 1100001101000100
07) L#7 = (L#5*K5) mod (2^16+1) = 0011000110101110*0110001111001010 mod (2^16+1) = 0001011001010001
08) L#8 = (L#6+L#7) mod 2^16 = 1100001101000100 + 0001011001010001 mod 2^16 = 1101010010110101
09) L#9 = (L#8*K6) mod (2^16+1) = 1101010010110101*1100001110100100 mod (2^16+1) = 0000001101100101
10) L#10 = (L#7+L#9) mod 2^16 = 0001011001010001 + 0000001101100101 mod 2^16 = 0001010010001101
11) L#11 = L#1 XOR L#9 = 0010111100100001 XOR 0000001101100101 = 0010110001000100
12) L#12 = L#3 XOR L#9 = 0001111010001111 XOR 0000001101100101 = 0001110111101010
13) L#13 = L#2 XOR L#10 = 0011101010110001 XOR 0001010010001101 = 0010111000111100
14) L#14 = L#4 XOR L#10 = 0000 1011 0100 1001 XOR 0001 0100 1000 1101 =0001111111000100
For the following rounds:
X1 = L#11 = 0010110001000100
X2 = L#12 = 0001110111101010
X3 = L#13 = 0010111000111100
X4 = L#14 = 0001111111000100

Round 2

01) L#1 = (X1*K1) mod (2^16+1) = 0010110001000100*0100101101010101 mod (2^16+1) = 0011011001100111
02) L#2 = (X2+K2) mod 2^16 = 0001110111101010 + 0100111001000011 mod 2^16 = 0101010001011001
03) L#3 = (X3+K3) mod 2^16 = 0010111000111100 + 0100100100100011 mod 2^16 = 011001001000001
04) L#4 = (X4+K4) mod (2^16+1) = 0001111111000100 + 1111001000111001 mod (2^16+1) = 1010010100001101
05) L#5 = L#1 XOR L#3 = 0011011001100111 XOR 0110000010000001 = 0101011011100110
06) L#6 = L#2 XOR L#4 = 0101010001011001 XOR 1010010100001101 = 1111000101010100
07) L#7 = (L#5*K5) mod (2^16+1) = 0101011011100110*0110001111001010 mod (2^16+1) = 1001101011000001
08) L#8 = (L#6+L#7) mod 2^16 = 1111000101010100 + 1001101011000001 mod 2^16 = 0001011110110101
09) L#9 = (L#8*K6) mod (2^16+1) = 0001011110110101*1100001110100100 mod (2^16+1) = 0100100001100111
10) L#10 = (L#7+L#9) mod 2^16 = 1001101011000001 + 0100100001100111 mod 2^16 = 1101011010010111
11) L#11 = L#1 XOR L#9 = 0011011001100111 XOR 0100100001100111 = 0111111000000000
12) L#12 = L#3 XOR L#9 = 0110010010000001 XOR 0100100001100111 = 0010110011100110
13) L#13 = L#2 XOR L#10 = 0101010001011001 XOR 1101011010010111 = 1000001011001110
14) L#14 = L#4 XOR L#10 = 1010010100001101 XOR 1101011010010111 = 0110001110011010
For the following rounds:
X1 = L#11 = 0111111000000000
X2 = L#12 = 0010110011100110
X3 = L#13 = 1000001011001110
X4 = L#14 = 0110001110011010

Round 3
01) L#1 = (X1*K1) mod (2^16+1)= 0111111000000000*0100101101010101 mod (2^16+1) = 0000001000011001
02) L#2 = (X2+K2) mod 2^16 = 0010110011100110 + 0100111001000011 mod 2^16 = 0110010100101001
03) L#3 = (X3+K3) mod 2^16 = 1000001011001110 + 0100100100100011 mod 2^16 = 1100101111101101
04) L#4 = (X4+K4) mod (2^16+1) = 0110001110011010 + 1111001000111001 mod (2^16+1) = 1010101001001111
05) L#5 = L#1 XOR L#3 = 0000001000011001 XOR 1100101111101101 = 1100100111110100
06) L#6 = L#2 XOR L#4 = 0110 0101 0010 1001 XOR 1010101001001111 = 1100111101100110
07) L#7 = (L#5*K5) mod (2^16+1) = 1100100111110100*0110001111001010 mod (2^16+1) = 0100001001010011
08) L#8 = (L#6+L#7) mod 2^16 = 1100111101100110 + 0100001001010011 mod 2^16 = 1010110010001101
09) L#9 = (L#8*K6) mod (2^16+1) = 1010110010001101*1100001110100100 mod (2^16+1) = 0011110001010001
10) L#10 = (L#7+L#9) mod 2^16 = 0100001001010011 + 0011110001010001 mod 2^16 = 0111111000101011
11) L#11 = L#1 XOR L#9 = 0000001000011001 XOR 0011110001010001 = 0011111001001000
12) L#12 = L#3 XOR L#9 = 0110010010000001 XOR 0011110001010001 = 0101100011010000
13) L#13 = L#2 XOR L#10 = 0110010100101001 OR 0111111000101011 = 0001101000000010
14) L#14 = L#4 XOR L#10 = 1010101001001111 XOR 0111111000101011 = 1101010001100100
For the following rounds:
X1 = L#11 = 0011111001001000
X2 = L#12 = 0101100011010000
X3 = L#13 = 0001101000000010
X4 = L#14 = 1101010001100100

Round 4
01) L#1 = (X1*K1) mod (2^16+1) = 0011111001001000*0100101101010101 mod (2^16+1) = 0110001101010111
02) L#2 = (X2+K2) mod 2^16= 0101100011010000 + 0100111001000011 mod 2^16 = 0011000110110011
03) L#3 = (X3+K3) mod 2^16 = 0001101000000010 + 0100100100100011 mod 2^16 = 0101011100100001
04) L#4 = (X4+K4) mod (2^16+1) = 1101010001100100 + 1111001000111001 mod (2^16+1) = 0101111001001101
05) L#5 = L#1 XOR L#3 = 0110001101010111 XOR 0101011100100001 = 0011010001110110
06) L#6 = L#2 XOR L#4 = 0011000110110011 XOR 0101111001001101 = 0110111111111110
07) L#7 = (L#5*K5) mod (2^16+1) = 0011010001110110*0110001111001010 mod (2^16+1) = 1100010101010011
08) L#8 = (L#6+L#7) mod 2^16 = 0110111111111110 + 1100010101010011 mod 2^16 = 1001100101010011
09) L#9 = (L#8*K6) mod (2^16+1) = 1001100101010011*1100001110100100 mod (2^16+1) = 0100101011000101
10) L#10 = (L#7+L#9) mod 2^16 = 1100010101010011 + 0100101011000101 mod 2^16 = 1010111110110111
11) L#11 = L#1 XOR L#9 = 0110001101010111 XOR 0100101011000101 = 0010100110010010
12) L#12 = L#3 XOR L#9 = 0101011100100001 XOR 0100101011000101 = 1111110111100100
13) L#13 = L#2 XOR L#10 = 0011000110110011 XOR 1010111110110111 = 1001111000000100
14) L#14 = L#4 XOR L#10 = 0101111001001101 XOR 1010111110110111 = 1111000111111010
For the following rounds:
X1 = L#11= 0010100110010010
X2 = L#12 = 1111110111100100
X3 = L#13 = 1001111000000100
X4 = L#14 = 1111000111111010

Round 5
01) L#1 = (X1*K1) mod (2^16+1) = 0010100110010010*0100101101010101 mod (2^16+1) = 0000010100100111
02) L#2 = (X2+K2) mod 2^16 = 1111110111100100 + 0100111001000011 mod 2^16 = 1000110001010111
03) L#3 = (X3+K3) mod 2^16 = 1001111000000100 + 0100100100100011 mod 2^16 = 1101000010100111
04) L#4 = (X4+K4) mod (2^16+1) = 1111000111111010 + 1111001000111001 mod (2^16+1) = 0111101111011111
05) L#5 = L#1 XOR L#3 = 0000010100100111 XOR 1101000010100111 = 1101010110000000
06) L#6 = L#2 XOR L#4 = 1000110001010111 XOR 0111101111011111 = 1111011110001000
07) L#7 = (L#5*K5) mod (2^16+1) = 1101010110000000*1111011110001000 mod (2^16+1) = 0001000110010111

08) L#8 = (L#6+L#7) mod 2^16 = 1111011110001000 + 0001000110010111mod 2^16 = 1110111011011111
09) L#9 = (L#8*K6) mod (2^16+1) = 1001100101010011*1100001110100100 mod (2^16+1) = 0100101011000101
10) L#10 = (L#7+L#9) mod 2^16 = 1110111011011111 + 0100101011000101 mod 2^16 = 1001001101110101
11) L#11 = L#1 XOR L#9 = 0000010100100111 XOR 0100101011000101 = 0100111111100010
12) L#12 = L#3 XOR L#9 = 1101000010100111 XOR 0100101011000101 = 1001101001100010
13) L#13 = L#2 XOR L#10 = 1000110001010111 XOR 1001001101110101 = 0001111100100010
14) L#14 = L#4 XOR L#10 = 0111101111011111 XOR 1001001101110101 = 1110100010101010
For the following rounds:
X1 = L#11= 0100111111100010
X2 = L#12 = 1001101001100010
X3 = L#13 = 0001111100100010
X4 = L#14 = 1110100010101010

Round 6
01) L#1 = (X1*K1) mod (2^16+1) = 0100111111100010*0100101101010101 mod (2^16+1) = 1100000100010001
02) L#2 = (X2+K2) mod 2^16 = 1001101001100010+ 0100111001000011 mod 2^16 = 0010001100010001
03) L#3 = (X3+K3) mod 2^16 = 0001111100100010 + 0100100100100011 mod 2^16 = 0101000110010001
04) L#4 = (X4+K4) mod (2^16+1) = 1110100010101010 + 1111001000111001 mod (2^16+1) = 0110011010001111
05) L#5 = L#1 XOR L#3 = 1100000100010001 XOR 0101000110010001 = 1001000010000000
06) L#6 = L#2 XOR L#4 = 0010001100010001 XOR 0110011010001111 = 0100010110011110
07) L#7 = (L#5*K5) mod (2^16+1) = 1001000010000000*1111011110001000 mod (2^16+1) = 1010101000001001
08) L#8 = (L#6+L#7) mod 2^16 = 0100010110011110 + 0001000110010111mod 2^16 = 0011010011000111
09) L#9 = (L#8*K6) mod (2^16+1) = 1001100101010011*1100001110100100 mod (2^16+1) = 0100101011000101
10) L#10 = (L#7+L#9) mod 2^16 = 0011010011000111 + 0100101011000101 mod 2^16 = 0111111001100001
11) L#11 = L#1 XOR L#9 = 1100000100010001 XOR 0100101011000101 = 1000101111010100
12) L#12 = L#3 XOR L#9 = 0101000110010001 XOR 0100101011000101 = 0001101101010100
13) L#13 = L#2 XOR L#10 = 0010001100010001 XOR 0111111001100001 = 0101110101110000
14) L#14 = L#4 XOR L#10 = 0110011010001111 XOR 0111111001100001 = 0001100011101110
For the following rounds:
X1 = L#11= 1000101111010100
X2 = L#12 = 0001101101010100
X3 = L#13 = 0101110101110000
X4 = L#14 = 0001100011101110

Round 7
01) L#1 = (X1*K1) mod (2^16+1) = 1000101111010100*0100101101010101 mod (2^16+1) = 0110110001111001
02) L#2 = (X2+K2) mod 2^16 = 0001101101010100+ 0100111001000011 mod 2^16 = 0101001010110111
03) L#3 = (X3+K3) mod 2^16 = 0101110101110000 + 0100100100100011 mod 2^16 = 0011001011001011
04) L#4 = (X4+K4) mod (2^16+1) = 0001100011101110 + 1111001000111001 mod (2^16+1) = 1110011011001001
05) L#5 = L#1 XOR L#3 = 0110110001111001 XOR 0011001011001011 = 0101111010110010
06) L#6 = L#2 XOR L#4 = 0101001010110111 XOR 1110011011001001 = 1011010001111110
07) L#7 = (L#5*K5) mod (2^16+1) = 0101111010110010*1111011110001000 mod (2^16+1) = 0100111101001011
08) L#8 = (L#6+L#7) mod 2^16 = 1011010001111110 + 0100111101001011mod 2^16 = 1111100010001011
09) L#9 = (L#8*K6) mod (2^16+1) = 1111100010001011*1100001110100100 mod (2^16+1) = 1010001101100101
10) L#10 = (L#7+L#9) mod 2^16 = 0100111101001011 + 1010001101100101 mod 2^16 = 1110110110011111
11) L#11 = L#1 XOR L#9 = 0110110001111001 XOR 1010001101100101 = 1100111100011100
12) L#12 = L#3 XOR L#9 = 0011001011001011 XOR 1010001101100101 = 1001000110101110
13) L#13 = L#2 XOR L#10 = 0101001010110111 XOR 1110110110011111 = 1011111100101000
14) L#14 = L#4 XOR L#10 = 1110011011001001 XOR 1110110110011111 = 0000101101010110
For the following rounds:
X1 = L#11= 1100111100011100
X2 = L#12 = 1001000110101110
X3 = L#13 = 1011111100101000
X4 = L#14 = 0000101101010110

Round 8
01) L#1 = (X1*K1) mod (2^16+1) = 1100111100011100*0100101101010101 mod (2^16+1) = 1110101011110111
02) L#2 = (X2+K2) mod 2^16 = 1001000110101110 + 0100111001000011 mod 2^16 = 1101111111101101
03) L#3 = (X3+K3) mod 2^16 = 1011111100101000 + 0100100100100011 mod 2^16 = 1111011011101011
04) L#4 = (X4+K4) mod (2^16+1) = 0000101101010110 + 1111001000111001 mod (2^16+1) = 1111100011100001
05) L#5 = L#1 XOR L#3 = 1110101011110111 XOR 1111011011101011 = 1111100011100001
06) L#6 = L#2 XOR L#4 = 1101111111101101 XOR 1111100011100001 = 0010011100011100
07) L#7 = (L#5*K5) mod (2^16+1) = 1111100011100001*1111011110001000 mod (2^16+1) = 0001111001101101
08) L#8 = (L#6+L#7) mod 2^16 = 0010011100011100 + 0001111001101101 mod 2^16 = 0011101011110111
09) L#9 = (L#8*K6) mod (2^16+1) = 0011101011110111*1100001110100100 mod (2^16+1) = 0000010101101001
10) L#10 = (L#7+L#9) mod 2^16 = 0001111001101101 + 0000010101101001 mod 2^16 = 0001100010110011
11) L#11 = L#1 XOR L#9 = 1110101011110111 XOR 0000010101101001 = 1110111110011110
12) L#12 = L#3 XOR L#9 = 1111011011101011 XOR 0000010101101001 = 1111001110000010
13) L#13 = L#2 XOR L#10 = 1101111111101101 XOR 0001100010110011 = 1100011101011110
14) L#14 = L#4 XOR L#10 = 1111 1000 1110 0001 XOR 1100 0111 0101 1110 = 0011111110111111

Output Transformation
01) Y1 = (X1*K1) mod (2^16+1) = 1110111110011110*0100101101010101 mod (2^16+1) = 1011110010000011
02) Y2 = (X2+K2) mod 2^16 = 1111001110000010 + 0100111001000011 mod 2^16 = 1000001000100001
03) Y3 = (X3+K3) mod 2^16 = 1100011101011110 + 0100100100100011 mod 2^16 = 1010111011111101
04) Y4 = (X4*K4) mod (2^16+1) = 0011111110111111*1111001000111001 mod (2^16+1) = 1101001100001001
Result of Decryption :
Y1 = 1011110010000011
Y2 = 1000001000100001
Y3 = 1010111011111101
Y4 = 1101001100001001

4. Conclusion

The signature is an important identity for the user. With the Elgamal and Idea algorithm methods[7], we can find out whether the signature belongs to the user. Signatures using the elgamal and idea algorithm methods can maintain the authenticity of the signature. The signature confidentiality cannot be modified by other users, so the signature is more secure.

5. Suggestion

The suggestions that the author can convey are as follows:
1. The system is designed only to be able to find out whether the signature is a user's signature or another user's signature using the elgamal and idea algorithm methods.
2. A system created to maintain signatures using the elgamal and idea algorithm methods.

Acknowledgement

Praise be to Allah SWT who has given health, enjoyment so that I can work on this journal. Thanks to my beloved parents and sister who always support and give input.

References

[1] Arifin, J., & Naf'an, M. Z. (2017). Verifikasi Tanda Tangan Asli Atau Palsu Berdasarkan Sifat Keacakan (Entropi). Jurnal Infotel, 9(1), 130. https://doi.org/10.20895/infotel.v9i1.136.
[2] Isnaini, H. F., Karyati, K., & Matematika, J. P. (2017). Penerapan Skema Tanda Tangan Schnorr pada Pembuatan Tanda Tangan Digital Implementation of Schnorr Signature Scheme in The Form of Digital Signature. 12(1), 57–64.
[3] Komputer, D. I., Matematika, F., Ilmu, D. A. N., & Alam, P. (2019). Pengamanan internet of things (iot) untuk tanda tangan digital menggunakan algoritme elgamal signature scheme selfi qisthina.
[4] Noor, M. U. (2021). Tanda Tangan Digital: Otoritas pada Arsip Elektronik. JIPI (Jurnal Ilmu Perpustakaan Dan Informasi), 6(1), 17. https://doi.org/10.30829/jipi.v6i1.8165
[5] Rahman, F., & Alamsyah, D. N. (2016). Dengan Metode Idea (International Data Encryption Algorithm). 281–289.
[6] Simatupang, J., & Sianturi, S. (2019). 1 , 2 1,2. 3(2).
[7] Zulian Vicky Hernando, & Purwantoro, P. (2022). Implementasi Tanda Tangan Digital (Digital Signature) Menggunakan Algoritme Elgamal Pada Dokumen Di Implementation of Digital Signature Using the Elgamal Algorithm on Documents At the Flight Education and Training Center (Bp3) Curug Web-Based. September, 386–393.