Object Oriented Analysis And Design Of El Gamal Digital Signature Algorithm For Document Authentication In E-Learning

Ambalika Ghosh\(^1\), Sunil Karforma\(^2\)

\(^1\)Research Scholar, Department of Computer Science, Burdwan University, INDIA
\(^2\)Professor, Department of Computer Science, Burdwan University, INDIA

Abstract

Now-a-days E-Learning System is very popular. But in this COVID-19 pandemic situation, it has become necessity. Classes, examinations, even result generation have all been done through online. But it involves several types of threats and risk factors as it is based on electronic communication media. To implement authenticity, integrity and non-repudiation of the information, we must use suitable encryption technique. In this paper we have proposed an object oriented analysis and design of El Gamal digital signature algorithm to implement authenticity, integrity and non-repudiation of mark sheet which would be generated by the Admin for the student at the time of publishing the result of an examination. We also have analysed the performance using a standard object oriented metric such as Metric for Object Oriented Design (MOOD metric) to ensure the software quality as well as justifying the good software design of our proposed model.

Keywords: E-learning, Covid-19 pandemic, Authenticity, Integrity, Non-repudiation, Encryption, ElGamal, Object oriented model, MOOD metric

I. INTRODUCTION

E-Learning has become popular for comfort and easy way to the freedom of study from anywhere. And now with this pandemic when people have to stay at home, it has become more important than ever [1]. The COVID-19 pandemic has been severely damaging the whole world. This situation has produced changes in the teaching–learning environment in every education sectors and teachers and students also feels the impact of it. As a result of the pandemic, universities were compacted to performing their operations primarily online with students [2]. COVID-19 has challenged the universities around the world to adopt online learning. The e-learning framework [3], however, is the best solution to enable online examination; online admit card issue, online payment, online result etc. to promote educational interactions among students, teachers & administrators [4,5]. But they must aware
of the threats and risks that may be faced during any kind of the above online operations [6, 7]. A secure online operation is always must expected, especially in this situation [8, 9].

In this paper we have applied ElGamal digital signature algorithm in an Object oriented model for security [10, 11] of information which is needed during generation of result for any examination of a university given by a student and published by admin electronically.

In our proposed system each student is needed to be enter his/her roll no and registration no to view the mark sheet. Admin generates the result for the respective examination of the particular student. This roll number, marks or student’s name may be altered by hacker causing insecure system. To protect the unwanted alternation of roll number, marks or student’s name from hacker, result is encrypted and signed by Admin using ElGamal digital signature algorithm.

In section II, we give a short brief that how a student can authenticate the marksheet which is encrypted and signed by Admin using ElGamal digital signature algorithm. In section III we have outlined how Object Oriented Modeling can be used to implement ElGamal digital signature to provide security of information during generation of a result. In this paper; we designed & embodied the UML [12, 13, 14] based Object Oriented Model of E-Learning System using ElGamal digital signature algorithm. For an efficient design, we have used CLASS Diagram (Fig.1) and SEQUENCE Diagram (Fig.2) [15] to represent the required class representation for programming purpose in the E-Learning system [16, 17]. In section IV we analyzed the performance of the system basedon the object oriented metrics i.e. MOOD metric [18-20]. Finally we made conclusion with some future scope in section V.

II. USE OF ELGAMAL DIGITAL SIGNATURE FOR DOCUMENT AUTHENTICATION

ElGamal Digital Signature Algorithm which was first proposed in 1985 by Taher ElGamal [21], is a public-key cryptosystem. It uses asymmetric key encryption for communicating between two parties and encrypting the message. The security strength of ElGamal is based on the hardness of solving discrete logarithms in a cyclic group that is even if we know \(g^m\) and \(g^n\) it is extremely difficult to compute \(g^{mn}\).[22].

Here we are describing how Admin can use the ElGamal Digital Signature algorithm to send an encrypted and signed Result to Student. The initial assumption is that everyone knows the generator \(B\) and the modulus \(U\) [23]. In the initialization phase, Admin do the followings:

Step1: Select a secret key \(S_{Admin}\); this is Admin’s secret key

Step2: Compute \(P_{Admin}\) using \(S_{Admin}\); this is Admin’s static public key

\[
P_{Admin} = (B)^{S_{Admin}} \mod U
\]

Step3: Select an ephemeral secret key; \(E_{i}\)

Step4: Compute an ephemeral public key; \(E_{p}\)
When Admin decides to sign a Marksheet $M_S^i$, he/she:

Step1: Uses the most secure one way hash function SHA3(), to compute the hashed value of the Marksheet $M_S^i$ combined with ephemeral public key $E_P^i$, just assigned a value for the hash

$$SHA3(M_S^i||E_P^i) = h_1^i$$

Step2: Creates the ElGamal signature

$$Sig_i = (E_S^i + h_1^i*S_{Admin}) \mod (U-1)$$

Step3: Now sends $P_{Admin}, E_P^i, M_S^i$, and $Sig_i$ to Student

Step4: When Student receives $P_{Admin}, E_P^i, M_S^i$, and $Sig_i$, Student computes the new hashed value, $h_2^i$ for the received marksheet $M_S^i$ and received ephemeral key $E_P^i$ sent by Admin

$$SHA3(M_S^i||E_P^i) = h_2^i$$

Step5: Verifies that $(B)^{Sig_i \mod U} = E_P^i*(P_{Admin})^{h_2^i \mod U}$

The reason above equation works is shown in algebraic transformation of the left side of the equation into the right side, showing they are equal

$$(B)^{Sig_i \mod U} = (B)^{E_S^i + h_1^i*S_{Admin}} \mod (U-1) \mod U$$

$$= (B)^{E_S^i} * (B)^{h_1^i*S_{Admin}} \mod U$$

[Rearrangement based on $g^{m+n} = g^m*g^n$]

$$= EP_i*(B)^{h_1^i*S_{Admin}} \mod U$$

[Substitution based on $E_P^i = (B)^{E_S^i} \mod U$]

$$= EP_i*(B)^{(S_{Admin} + h_1^i)} \mod U$$

$$= EP_i*(B)^{(S_{Admin})*h_1^i} \mod U$$

$$= EP_i*(P_{Admin})^{h_1^i} \mod U$$

[Substitution based on $P_{Admin} = (B)^{S_{Admin}} \mod U$]

We have shown that, $(B)^{Sig_i \mod U} = E_P^i*(P_{Admin})^{h_1^i} \mod U$

If this equality is possible then Student may be confident that Admin has signed the Marksheet and publish the Result which has not been modified after the signing procedure. So we can say that the integrity and authenticity of the information that depicted within the result is maintained properly.

### III. PROPOSED OBJECT ORIENTED MODEL
To depict our proposed system using UML we only consider the Class Diagram and Sequence Diagram.

A. CLASS DIAGRAM

The given Figure-1 demonstrates the organization of class hierarchy showing how an Admin can generate an encrypted and signed result and send it to Student using ElGamal digital signature algorithm in object oriented approach.

Class hierarchy [24] of the proposed E-Learning system is shown in Fig.1.

The necessary class diagram for the use of ElGamal algorithm in E-Learning security is given below:

The required three types of classes are: Base_ElGamal, Admin and Student. The description of the individual classes are given below-

Class Base_ElGamal- Base_ElGamal is the name of the abstract base class which has no object. It is use for the purpose of inheritance. It consists of two publicly inherited classes- Admin and Student. It has the following data members and member functions-

- long int h1i and EPi describes the hash value and the ephemeral public key respectively.
- static long int P1_obj (long int n, long int p, long int s1) describes the function for generating public key. It needs three arguments; one random number n, one prime number p and one secret key s1.
- char * Readmsg() describes the function that reads a message and also returns the message to the calling function.
- long double sigi(long intEPi, long int h1i, long int p, long int s1) describes the function to generate a signature for input message by using four arguments: ephemeral public key EPi, hash value h1i, a prime number p and a secret key s1. Then the generated signature is returned to the calling function.
- friend long int SHA3(char *, long int) describes the hash function which is used to generate a digital signature.
- friend long int Gen_prime() describes a function that generates a prime number and also returns the prime number to the calling function.

Class Admin-This class is publicly derived from base class Base_ElGamal. It consists of private data members i.e. s1 (secret key), ESi(ephemeral secret key), sig1(message signature), msg1(message) that are only accessible to the member functions of Admin class and public data member pk1 (public key) is accessible to all the functions of the same class as well as the member functions of other class. Member functions of other class can access the public data members of Admin class by creating an object of Admin class.

This class has the following member functions-
- Void get_s1() describes the function to get the secret key
- Long int get_ESi()describes the function to get the ephemeral key and also returns to the calling function
Student send (int n, int p) describes the function that returns an object to the Student using two arguments; a random number n and a prime number p. It is used to send the result as message, signature and the public key. This function helps to create an object of the Student class and then it calls get_s1() and calculate the public key by calling P1_obj() and then using the get_ESi() the ephemeral key is read and calculate the hash value. Finally it sends the signed result to the student.

**Class Student**-This class is publicly derived from base class Base_ElGamal. There is no private data members or functions in this class. It has only public data members and functions. So the data members are accessible to all other classes as well as same class. It consists of public data members i.e. sig2 (for receiving sign), pk2 (for receiving public key) and msg2 (for receiving message). The implementation of the following member function of this class is described below-

Void verify_sign(int n, int p) describes the function to verify the signature of the result using two arguments; a random number n and a prime number p.

The Driver Program segment of main() is described below-

```
main()
{
    Admin A;                    // A is an object of Admin class
    Student S;                 // S is an object of Student class
    int n=Generate_random();    // Random number generator
    int p=Generate_prime();     // Prime number generator

    A.get_s1();                // A gets secret key by invoking this function
    S=A.send(n, p);            // Admin A send signed result to student by invoking
    this
    S.verify_sign(n, p);       // Student S verify the signature of Admin by invoking
    this
}
```

```
Class Base_ElGamal

Public:
+long int h1i
+long int EPi
+static long int P1_obj (long int, long int, long int);
+char * Readmsg();
+long double sigi(long int, long int, long int, long int);
+friend long int SHA3(char *, long int);
+friend long int Gen_prime();
```

---

Webology (ISSN: 1735-188X)
Volume 19, Number 2, 2022
Publicly Derived

<table>
<thead>
<tr>
<th>Class Admin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Private:</td>
</tr>
<tr>
<td>- long int s1;</td>
</tr>
<tr>
<td>- long int ESi;</td>
</tr>
<tr>
<td>- long double sig1;</td>
</tr>
<tr>
<td>- char * msg1;</td>
</tr>
<tr>
<td>Public:</td>
</tr>
<tr>
<td>+ long int pk1;</td>
</tr>
<tr>
<td>+ void get_s1();</td>
</tr>
<tr>
<td>+ long int get_ESi();</td>
</tr>
<tr>
<td>+ student send(int, int);</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Class Student</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public:</td>
</tr>
<tr>
<td>+ long double sig2;</td>
</tr>
<tr>
<td>+ long int pk2;</td>
</tr>
<tr>
<td>+ char * msg2;</td>
</tr>
<tr>
<td>+ void verify_sign(int, int);</td>
</tr>
</tbody>
</table>

Fig.1: Class Hierarchy Diagram of ElGamal Digital Signature for Document Authentication

B. SEQUENCE DIAGRAM

A sequence diagram shows interaction among objects as a two-dimensional chart [25]. Here, we only describe the steps that are needed to encrypt and signed the result using ElGamal digital signature by Admin and send it to Student who further can verifies the signature for authentication of the result in E-Learning system with the help of Sequence diagram in Fig.2.

1. Select secret key
2. Compute static public key
3. Select ephemeral secret key
4. Compute ephemeral public key
5. Compute hashed values
6. Create signature on result
7. Encrypt
8. Enter roll and registration number
9. Send encrypted result and signature
IV. OBJECT ORIENTED METRICS BASED ANALYSIS OF PROPOSED MODEL

There are various object oriented metrics but here we have discussed about only MOOD metric. Every aspect of object oriented paradigm is covered by MOOD metrics [26]. MOOD metric also measures the basic features of object oriented design such as encapsulation (MHF and AHF), inheritance (MIF and AIF) and message-passing (CF) [27]. Here we will calculate the following values [28] of object oriented metrics (as discussed above) with respect to our proposed class diagram to analyze the performance of document authentication in E-Learning system using Elgamal digital signature.

**MHF (Method Hiding Factor):** It calculates encapsulation that shows the sum of the invisibilities of all methods of all classes.

Equation for MHF=$\frac{\sum_{i=1}^{TC} M_{h}(Ci)}{\sum_{i=1}^{TC} M_{d}(Ci)}$ //TC means total number of class

Where $M_{d}(Ci)= M_{v}(Ci)+ M_{h}(Ci)$ where $M_{d}(Ci)= \text{methods defined in class C}$, $M_{v}(Ci)= \text{methods visible in class C}$ and $M_{h}(Ci)= \text{methods hidden in class C}$ [29]

<table>
<thead>
<tr>
<th>MHF Table</th>
<th>Classes of Proposed System</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Base_ElGamal</td>
</tr>
<tr>
<td>$M_{h}(Ci)$</td>
<td>0</td>
</tr>
<tr>
<td>$M_{v}(Ci)$</td>
<td>5</td>
</tr>
<tr>
<td>$M_{d}(Ci)$</td>
<td>5</td>
</tr>
<tr>
<td>MHF</td>
<td>$\frac{0}{9} = 0$</td>
</tr>
</tbody>
</table>

Table 1 shows MHF value is very low, so our proposed system is easy to understand and implement.

**AHF (Attribute Hiding Factor):** It also calculates encapsulation that shows the sum of the invisibilities of all attributes of all classes.

Equation for AHF=$\frac{\sum_{i=1}^{TC} A_{h}(Ci)}{\sum_{i=1}^{TC} A_{d}(Ci)}$

$A_{d}(Ci)= A_{v}(Ci)+ A_{h}(Ci)$, where $A_{d}(Ci)= \text{total attributes defined in class C}$, $A_{v}(Ci)= \text{Attributes visible in class C}$ and $A_{h}(Ci)= \text{attributes hidden in class C}$

<table>
<thead>
<tr>
<th>MHF Table</th>
<th>Classes of Proposed System</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Base_ElGamal</td>
</tr>
<tr>
<td>$A_{h}(Ci)$</td>
<td>0</td>
</tr>
<tr>
<td>$A_{v}(Ci)$</td>
<td>5</td>
</tr>
<tr>
<td>$A_{d}(Ci)$</td>
<td>5</td>
</tr>
<tr>
<td>MHF</td>
<td>$\frac{0}{9} = 0$</td>
</tr>
</tbody>
</table>

Table 2 shows AHF value is also very low, so our proposed system is easy to implement.
Table 2 shows AHF value is within 0 and 1, so our proposed system is ok.

**MIF (Method Inheritance Factor):** It is calculated by the ratio of the sum of the inherited methods to the total number of methods in all classes.

Equation for $\text{MIF} = \frac{\sum_i TCM_i(Ci)}{\sum_i TCM_a(Ci)}$

Where $M_d(Ci)=M_d(Ci)+M_i(Ci)$, $M_a(Ci)=number$ of methods available, $M_d(Ci)=number$ of methods defined and $M_i(Ci)=number$ of methods inherited.

Table 2 shows AHF value is within 0 and 1, so our proposed system is ok.

**AIF (Attribute Inheritance Factor):** It is calculated by the ratio of the sum of the inherited attributes to the total number of attributes in all classes.

Equation for $\text{AIF} = \frac{\sum_i TCA_i(Ci)}{\sum_i TCA_a(Ci)}$

Where $A_a(Ci)=A_a(Ci)+A_i(Ci)$, $A_d(Ci)=number$ of attributes available, $A_d(Ci)=number$ of attributes defined and $A_i(Ci)=number$ of attributes inherited.

Table 4 shows AIF value is 0.29, so our proposed system is ok. If the value of AIF is 0 that means there is no existence of any attribute which means lack of inheritance occurs.
**CF (Coupling Factor):** It is calculated by the ratio of the maximum possible number of couplings to the actual number of couplings not imputable to inheritance of the class.

Equation for \( CF = \frac{\sum_{i=1}^{TC} \sum_{j=1}^{TC} \text{is\_client}(C_i, C_j)}{(TC^2 - TC)} \)

Where \( \text{is\_client}(C_c, C_s) = \begin{cases} 1 & \text{if } C_c \Rightarrow C_c \land C_s \neq C_s \\ 0 & \text{otherwise} \end{cases} \)

<table>
<thead>
<tr>
<th>CF Table</th>
<th>Classes of Proposed System</th>
<th>Base_ElGamal</th>
<th>Admin</th>
<th>Student</th>
<th>( \sum )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base_ElGamal</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Admin</td>
<td></td>
<td>0</td>
<td>X</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Student</td>
<td></td>
<td>0</td>
<td>0</td>
<td>X</td>
<td>0</td>
</tr>
<tr>
<td>CF</td>
<td></td>
<td>( \frac{0}{9-3} = \frac{0}{6} = 0 )</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5 shows CF value is 0 which indicates that our proposed system is well designed.

**V. CONCLUSION**

In this paper, we have applied ElGamal digital signature for message encryption with signature creation and verification. ElGamal security lies on the difficulty of solving the discrete logarithm problem means ElGamal would be broken if and only if the discrete logarithm problem could be solved efficiently. Here ElGamal digital signature is wrapped in Object Oriented Model using UML to achieve reusability of the code design of the system. Hence it makes the proposed system trustworthy. Object oriented design and analysis is important to make the system more reliable, flexible and obviously secure. MOOD metric based analysis provides us a good quality software where high cohesion and low coupling [30, 31] is maintained properly. It also helps us to analyse the performance as well as quality of the software. The participants such as Admin, Student, Teacher of our proposed system can generate result, certificate etc. securely with proper authenticity [32]. So finally we can conclude that, by using ElGamal digital signature security is achieved, all OOPs benefits are utilized and software quality is perfectly assured in our proposed system. For further improvement of security the elliptic curve version of ElGamal algorithm may be applied in our E-Learning system.

**REFERENCES**

[1] https://englishlive.ef.com/blog/study-tips/why-is-online-learning-so-important-during-coronavirus/
[21] https://www.sciencedirect.com/topics/computer-science/el-gamal
[22] https://www.geeksforgeeks.org/elgamal-encryption-algorithm/