SECURE ELECTRONIC EXAM

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF APPLIED MATHEMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

LÜTFÜ TARKAN ÖLÇÜOĞLU

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
CRYPTOGRAPHY

FEBRUARY 2016

Approval of the thesis:

SECURE ELECTRONIC EXAM

submitted by **LÜTFÜ TARKAN ÖLÇÜOĞLU** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Department of Cryptography, Middle East Technical University** by,

Prof. Dr. Bülent Karasözen
Director, Graduate School of **Applied Mathematics** _____

Prof. Dr. Ferruh Özbudak
Head of Department, **Cryptography** _____

Assoc. Prof. Dr. Ali Doğanaksoy
Supervisor, **Mathematics, METU** _____

Assist. Prof. Dr. Sedat Akleylek
Co-supervisor, **Department of Computer Engineering, OMU** _____

Examining Committee Members:

Prof. Dr. Ferruh Özbudak
Department of Mathematics, METU _____

Assoc. Prof. Dr. Ali Doğanaksoy
Department of Mathematics, METU _____

Prof. Dr. Ersan Akyıldız
Department of Mathematics, METU _____

Assist. Prof. Dr. Çetin Ürtiş
Department of Mathematics, TOBB ETU _____

Assist. Prof. Dr. Oğuz Yayla
Department of Mathematics, Hacettepe University _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: LÜTFÜ TARKAN ÖLÇÜOĞLU

Signature :

ABSTRACT

SECURE ELECTRONIC EXAM

Ölçüoğlu, Lütfü Tarkan

Ph.D., Department of Cryptography

Supervisor : Assoc. Prof. Dr. Ali Doğanaksoy

Co-Supervisor : Assist. Prof. Dr. Sedat Akleylek

February 2016, 61 pages

With the recent developments in information technology, computers and mobile devices have an important role in daily life. The field of education are also affected by these developments, as a result electronic learning (e-learning) subject becomes popular. By the developments in e-learning area, students are now able to reach their documents using computers instead of books. This situation provides teachers to perform exams in digital environments. Information security concepts in an exam should be satisfied in order to get reliable evaluation in education. Some of the candidates can use illegal ways to reach exam questions before exam or change their scores during or after exam. To prevent these, cryptography plays an essential role in electronic exam. Therefore, a secure electronic exam is one of the most vital and difficult part in e-learning security.

In this thesis, we first give a model satisfying authenticity, anonymity, secrecy and long term confidentiality on sensitive data. The model is using ID-Based cryptosystems for authentication, sanitizable signature schemes for data editing and verifiable secret sharing schemes for long term confidentiality issues. A novel secure electronic exam approach is proposed by combining sanitizable signature and ID-Based cryptography. Then, security analysis of the proposed model is discussed.

Keywords : ID-Based cryptography, sanitizable signatures, electronic learning, electronic exam, long term confidentiality

ÖZ

GÜVENLİ ELEKTRONİK SINAV

Ölçüoğlu, Lütfü Tarkan
Doktora, Kriptografi Bölümü
Tez Yöneticisi : Doç. Dr. Ali Doğanaksoy
Ortak Tez Yöneticisi : Yrd. Doç.Dr. Sedat Akleylek

Şubat 2016, 61 sayfa

Son zamanlarda bilgi teknolojisindeki gelişmeler ile, bilgisayarlar ve mobil cihazlar günlük hayatta önemli bir rol almıştır. Eğitim alanı da bu gelişmelerden etkilenmiş, sonuç olarak elektronik öğrenme (e-öğrenme) konuları popüler olmuştur. E-öğrenme alanındaki gelişmeler ile, öğrenciler kitaplar yerine bilgisayarlar kullanarak dokümanlara erişim olanağına sahip olmuşlardır. Bu durum, öğretmenlerin sınavları elektronik ortamda yapmasını sağlamıştır. Eğitimdeki güvenilir ölçme için bilgi güvenliği konseptleri sağlanmalıdır. Bazı adaylar, sınav sorularına sınavdan önce erişmek veya sonuçları sınav esnasında veya sonrasında değiştirmek gibi yasadışı yolları kullanmaktadırlar. Bunları önlemek için, kriptografi elektronik sınavda büyük rol oynamaktadır. Dolayısıyla güvenli elektronik sınav, e-öğrenme güvenliğinin en önemli ve en zor kısmıdır.

Bu tezde, ilk olarak kimlik denetimi, anonimlik, gizlilik ve hassas verilerin uzun süreli gizliliğini sağlayan bir model vereceğiz. Bu model, kimlik denetimi için kimlik tabanlı kripto sistem, veri düzenlemesi için sterilize edilmiş imzalama şemaları ve uzun süreli gizlilik için doğrulanabilir gizlilik paylaşım şeması kullanmaktadır. Sterilize edilmiş imzalama ve kimlik tabanlı kripto sistem ile birleştirilmiş özgün bir güvenli elektronik sınav modeli önerilecektir. Modelin bütün parçaları analiz edilerek derinlemesine ele alınacaktır. Daha sonra, önerilen modelin güvenlik analizi tartışılacaktır.

Anahtar Kelimeler : Kimlik tabanlı kriptosistem, sterilize edilmiş imzalar, elektronik öğrenme, elektronik sınav, uzun süreli gizlilik

To my wife and family

ACKNOWLEDGMENTS

First, I would like to express my very great appreciation to my thesis supervisor Assoc. Prof. Dr. Ali Doğanaksoy and co-supervisor Assist. Prof. Dr. Sedat Akleylek for their patient guidance, enthusiastic encouragement and valuable advices during the development and preparation of this thesis. Their willingness to give their time and to share their experiences have brightened my path.

I would like to thank my committee members, Prof. Dr. Ferruh Özbudak, Prof. Dr. Ersan Akyıldız, Assist. Prof. Dr. Çetin Ürtiş and Assist. Prof. Dr. Oğuz Yayla for their valuable suggestions and comments.

I would also like to thank the members of Institute of Applied Mathematics, my classmates and my colleagues.

I am grateful to my father Metin Ölçüoğlu, my mother Zahire Ölçüoğlu and my sister Hülya Kabakcı for always supporting and trusting to me.

Finally, the last but not the least, I would like to thank my beloved wife Rukiye Ölçüoğlu. Her encouragement, support and patience under all conditions have lightened my path and made me to complete this dissertation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ADM	Admissible Modification
BDH	Bilinear Diffie-Hellman Assumption
DB	Database Server
ETS	Educational Testing Services
GMAC	Graduate Management Admission Council
GMAT	Graduate Management Admission Test
MOD	Modification Instruction
NIST	National Institute of Standards and Technology
PKCS	Public-Key Cryptography Standards
PKG	Private Key Generator
TOEFL	Test of English as a Foreign Language
VSS	Verifiable Secret Sharing

CHAPTER 1

INTRODUCTION

In this chapter, we give firstly an overview of electronic learning, electronic exam and related work, then we continue with our objective and finally thesis' outline is given.

1.1 Overview

Electronic learning is one of the most popular topics of today. Universities and colleges are transferred their documents into the digitalized environment. Furthermore, these universities have started to allow their students to attend online courses. According to [10], in 2012, 71.7% of higher education institutions suggest online courses in USA and over 6 million students took at least one online course in 2011. These interest in the attendance of online courses enabled universities and institutes to hold their exams in electronic environment.

Electronic exam *(e-exam)* is one of the challenging problem in electronic learning, since it needs some cryptographic requirements. The informal definition of electronic exam can be done like the computer based version of paper based examination. However, there are some open problems in view of information security concepts to be solved for a secure electronic exam. To hold a secure electronic exam, one should provide authenticity, anonymity, secrecy of the questions and their relating answers and robustness in all stages of it.

1.2 Objective and Motivation

The traditional exams (paper-based exams) need long time from beginning to end. There are a lot of students attending national based examinations. It takes a lot of time to prepare and evaluate broad participation exams. To reduce the loss of time in those stages in an exam, electronic examination is needed. There are related work in the literature for electronic exam, but the security of it is not common. In this thesis, our objective is to design a model where users authenticate with their digital certificates. In the model, users can provide data to system, then the authority processes the provided data for the users and serves these data as an application. The model satisfies long

term confidentiality in the storage of the data. With both model and electronic exam application, we try to find the answers to the following questions:

- user authentication, key management, secrecy of the sensitive data in electronic exam,
- allowed modification of the data with the preservation of the original signature,
- long term confidentiality of the sensitive data.

To answer these questions, we propose a model which uses ID-Based Cryptosystem for authentication and key management, AES for the secrecy of the sensitive data. It is a hybrid system which combines symmetric key cryptosystems and asymmetric key cryptosystems. For preserving the original signature on modified data, we prefer to use sanitizable signature schemes. To achieve long term confidentiality of the sensitive data, we use verifiable secret sharing scheme. Next, we apply an electronic examination scheme based on the idea of the proposed model.

1.3 Contribution

To the best of our knowledge, there are many related work about electronic exam but the most prominent ones are by Castella-Roca et al. [18] and Huszti et al. [27]. In [18], Castella-Roca et al. suggested that electronic examination is divided into: exam setup, starting, holding and submission, grading of exam, getting the results and revisions. The security requirements of the stages are identified as authenticity, anonymity, correction, secrecy, detection of the copy and output. In their proposal, they assumed that the examination is held in a supervised environment with the roles of student, teacher and manager. The authenticity was provided by asymmetric key solution schemes. The anonymity of the both teachers and students named maximum impartiality was defined as privacy. The correction was provided by the integration of the exam questions and no alteration on any answers after submission. The secrecy of the exam questions which were prepared by the teachers, was satisfied with manager's public key encryption. In the proposed protocol, the manager is responsible for all the stages of the examinations based on trusted third party scheme.

The other prominent work was done by Huszti et al. [27] They proposed a scheme which has no trusted third party based on *n-servers*. The main idea of the proposed scheme is achieving the anonymity of the students and teachers' identity which were provided by the timed-release service. They suggested four roles: registrar, students, teachers and exam authority. The exam scheme was defined as the stages of registration, exam and grading. The security requirements for the proposed scheme were authenticity, anonymity, secrecy, robustness, correctness and output. As mentioned in [18], the authenticity was provided by asymmetric key solution. In the proposed scheme, Huszti et al. used $ElGamal$ public key cryptosystem in authentication. The main idea of the proposed scheme was the confidentiality of the student's identity such that they assumed that students might try to threaten or bribe teachers to obtain better

result. This identity secrecy was achieved by timed-released service. Questions and their related answers were prepared by a committee and both of them were encrypted by Mixnet's public key. The exam was hold by only registered students. The grading of the exam results were done by the teachers and the given grades were matched to real identity of the students by timed-release service at the determined time which can be thought as double sided blind review.

There are other electronic exam work which are for commercial solutions and describing lack of security of the scheme. The most popular and world wide spread electronic exam is *TOEFL* (Test of English as a Foreign Language) by *ETS* (Educational Testing Services). Until now, the total participant of the *TOEFL* is about 30 million [4]. The *TOEFL* score is accepted by over 9.000 colleges and universities all around the world [4]. There is no detailed work on the security of the *TOEFL* examination. They use a state-of-the-art encryption scheme for exam materials and all exam materials are downloaded the test center's computer with encryption. The decryption is done by the authority and the *TOEFL* software can detect whether any disruption or change done in the exam material [7]. The *TOEFL* is not the only electronic exam done around the world. The other popular one is *GMAT* (Graduate Management Admission Test) organized by *GMAC* (Graduate Management Admission Council). More than 250.000 people take the test per year, and the score of *GMAT* is accepted by over 1.500 universities located in 110 countries [5]. The *GMAT* uses biometric solution, palm vein technology, for the security of the examination [6].

There are some examples of electronic exams in Turkey over the last decade. The most common ones are done by Ministry of National Education and Measuring, Selection and Placement Center *(ÖSYM)*. Ministry of National Education organizes *Motor Vehicle Candidate Driver Exams* once a week in three places. 840 students participate this exam every week [9]. The examination is hold in a web-based platform by the secure network and all the exam rooms are monitored by the cameras [2]. *ÖSYM* is the other institute organizing electronic exams. The first e-exam hold by *ÖSYM* in 2014. The examination is about foreign language which holds in three cities of Turkey per month with 700 students [8].

1.4 Thesis Outline

In Chapter 2, we give some information about electronic learning and cryptographic primitives used throughout the thesis.

In Chapter 3, we propose a secure information sharing model with user authentication, anonymity, robustness and long term confidentiality on sensitive data. The protocols in the model are defined and the security analysis of the model is explained.

In Chapter 4, we present *secure electronic exam model* with respect to proposed model. The algorithms and cryptographic requirements for the electronic exam are introduced and the security analysis of the electronic exam application is given.

In Chapter 5, we give the conclusion of the thesis and future work.

CHAPTER 2

PRELIMINARY

In this chapter, we present the required definitions and cryptographic primitives that are used in this work. The chapter mainly consists of two parts: Electronic Learning and Cryptographic Techniques. In the first part, some basic definitions about electronic learning and electronic exam are given. The next part is about the cryptographic techniques that are used in the model and electronic exam application. Some detailed part of cryptographic techniques are given in the following chapters.

2.1 Electronic Learning Preliminaries

In this section, we give some brief information about electronic learning and electronic exam.

2.1.1 History

Electronic learning is one of the most popular topics of today's technology. The early definition of electronic learning based on 1980s. The word $e-learning$ was first defined in Los Angeles during the seminar of CBT system in 1999. This is also the first definition done in professional environment. Due to the improvement in internet technology, lots of institutes including universities have started to transfer their course materials into the digital environment. Not only universities but also other institutes like military, business and training sectors have transferred their materials into digital environment for quick access and secure storage on them. Therefore, the definition of e-learning cannot be thought only for schools or other education institutes since it addresses to other sectors. In [32], Nicholson defined e-learning as a software based online learning in a wide range through business to training sector and higher education to military.

Since e-learning in every institutes became widespread, this development in this brunch needed the examinations in digital environment. The next section will be about electronic examination.

2.1.2 Electronic Exam

Electronic exam is simply described as a digital version of paper based examination. For a secure electronic exam, the problems in Section 1.2 should be solved. In [12], electronic was exam defined as the management of examination through the internet or local network with the less workload on examination, gradings and review. Like [12], Bieniecki et al [14] defined electronic exam as a software which carries out the exam materials to the computer.

A secure electronic exam should consist of registration, question preparation, exam preparation, evaluation and archiving processes where all users have digital certificates. The security of the electronic exam relies on the security of the sensitive data of the examination storing in encrypted form.

2.2 Cryptographic Preliminaries

In this section, we give cryptographic techniques used throughout the thesis.

2.2.1 ID-Based Cryptosystems

The ID-Based cryptosystem is a public key cryptosystem firstly developed by Adi Shamir in 1984 [43]. Unlike the other public key cryptosystems, ID-Based cryptosystem uses an identity of any user which defines him uniquely instead of digital certificates. This identity could be e-mail, telephone number, passport number, national identity number, etc. In ID-Based cryptosystems, private keys are delivered by trusted third party.

Shamir's proposed scheme is not practical on encryption. It determines RSA like signature scheme. For encryption, the first practical solution is proposed by Boneh and Franklin in 2001 [15]. The security of encryption relies on the hardness of Diffie-Hellman problem with random oracle model. The computations are done by Weil Pairings on elliptic curves. The other efficient methods are proposed by Patterson in 2002 [34] and Sakai and Kasahara in 2003 [40]. The proposed models use bilinear pairings on elliptic curves.

In our model, we use Boneh and Franklin's proposed model. The proposed cryptosystem consists of *setup, extract, encrypt, decrypt* and *correctness* phases.

Setup:

Let $k \in \mathbb{Z}^+$ be a security parameter, e be an admissible bilinear map [1] and $\mathcal{G}$ be some

[1] Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two groups of order q, then the map $e : G_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ is bilinear if the followings are satisfied:
- $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}$. (Bilinear property)
- $e(P, P) \neq 1$, i.e., e does not send all pairs in $\mathbb{G}_1 \times \mathbb{G}_1$ to the identity of $\mathbb{G}_2$. (Non-degeneracy property)

BDH[2] parameter generator[3],

- Select a random generator $P \in \mathbb{G}_1$.
- Take a random $s \in \mathbb{Z}_q^*$ and $P_{Pub} = sP$.
- Select cryptographic hash functions H_1 and H_2 such that $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1^*$ and $H_2 : \mathbb{G}_2 \rightarrow \{0,1\}^n$ for some *n*.

Let $\mathcal{M}$ be a message space such that $\mathcal{M} = \{0,1\}^n$ and let $\mathcal{C}$ be ciphertext space such that $\mathcal{C} = \mathbb{G}_1 \times \{0,1\}^n$. The parameters are:$\langle q, \mathbb{G}_1, \mathbb{G}_2, e, n, P, P_{Pub}, H_1, H_2 \rangle$. The *master-key* is $s \in \mathbb{Z}_q^*$.

Extract:

For any user's identity *ID* $\in \{0,1\}^n$,

- Calculate $Q_{ID} = H_1(ID) \in \mathbb{G}_1^*$,
- Set $d_{ID} = sQ_{ID}$ where d_{ID} is private key and *s* is the master key.

Encrypt:

Let $M \in \mathcal{M}$ be a message to be encrypted under the public key ID,

- Calculate $Q_{ID} = H_1(ID) \in \mathbb{G}_1^*$,
- Select random $r \in \mathbb{Z}_q^*$,
- The ciphertext $\mathcal{C} = \langle rP, M \oplus H_2(g_{ID}^r) \rangle$ where $g_{ID}^r = e(Q_{ID}, P_{Pub}) \in \mathbb{G}_2^*$.

Decrypt:

The ciphertext $\mathcal{C} = \langle U, V \rangle$ encrypted with public key *ID* is decrypted using the private key $d_{ID} \in \mathbb{G}_1^*$ as follows:

- Calculate $V \oplus H_2(e(d_{ID}, U)) = M$.

The encryption and decryption can be verified as follows:

- There exist an efficient algorithm for computing $e(P,Q)$ for $P, Q \in \mathbb{G}_1$ (Computability property)

[2] BDH: Bilinear Diffie-Hellman Assumption is a variant of computational Diffie-Hellman assumption. It is secure against active and passive attacks since the problem is intractable [28]

[3] A randomized algorithm $\mathcal{G}$ is a BDH parameter generator if [15]

- $k \in \mathbb{Z}^+$ is a security parameter of $\mathcal{G}$,
- $\mathcal{G}$ is a polynomial time algorithm in *k*,
- the prime number *q* is generated by $\mathcal{G}$ where $\mathbb{G}_1 and \mathbb{G}_2$ are two groups of order *q* and *e* is an admissible bilinear map such that *e* : $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$

Correctness:

$$e(d_{ID}, U) = e(sQ_{ID}, rP) = e(Q_{ID}, P)^{sr} = e(Q_{ID}, P_{Pub})^r = g_{ID}^r$$

The signature scheme for ID-Based encryption consists of two phases: sign and verify. We combine both Boneh and Franklin's model [15] with Hess' [26] model for signature issues. In [26], Hess offered four phases for identity based signature scheme: *setup*, *extract*, *sign* and *verify*. We use Boneh and Franklin's setup and extract phases for identity based signature scheme.

Sign:

Let $M \in \mathcal{M}$ be a message to be signed. The user selects an arbitrary $P_1 \in \mathbb{G}_1^*$, takes a random integer $k \in \mathbb{Z}_q^*$ and computes

- $r = e(P_1, P)^k$
- $v = H_1(M, r)$
- $u = vsQ_{ID} + kP_1$

The signature *sign* = $(u, v) \in (\mathbb{G}_1^*, \mathbb{Z}_q^*)$

Verify:

The verification of the signature is done by checking the equality:

$$r \stackrel{?}{=} e(u, P)e(H_1(ID), -P_{Pub})^v.$$

The signature is accepted if and only if $v = H_1(M, r)$. The above equation holds in this way:

$$r = e(P_1, P)^k = e(kP_1, P) = e(u - vsQ_{ID}, P) = e(u, P)e(-vsQ_{ID}, P) = e(u, P)e(H_1(ID), -sP)^v = e(u, P)e(H_1(ID), -P_{Pub})^v$$

The signature scheme is based on ElGamal signature algorithm. Therefore, there are some other variants of it. Koyuncu et al. [29] showed different variants of ID-Based ElGamal signature schemes and insecure variants of above equation. There are also different variant of ID-Based signature except from ElGamal signature scheme. Choon et al. [19] showed ID-Based Signature from Gap Diffie-Hellman Groups.

2.2.2 Sanitizable Signatures

Sanitizable signatures allows the signer to determine a designate part of the signature to modify in the future. In other words, a signature can be modified from it's designated part by a sanitizer without losing the validation. The first definition of the sanitizable signature was introduced by Ateniese et al [11] in 2005. They defined the sanitizable signature as a modification of a designated part of the signature without any interaction with the first signer. The signature scheme has the properties *unforgeability, immutability, privacy, accountability* and *transparency*.

In our proposed model, we use sanitizable signature in Section 3.3 for minor corrections in the data. Mainly, we adhere to the security conditions of Brzuska et al. [17].

In [17], *sanitizable signature scheme* was defined as a scheme including seven algorithms: key generation of signer and sanitizer, signing, sanitizing, verification, proof and judge.

Key Generation:

There exist key generation algorithms for the signer and sanitizer. Private and public key pairs based on the security parameter n are generated by these algorithms:

$$(pk_{sig}, sk_{sig}) \leftarrow KGen_{sig}(1^n)$$
$$(pk_{san}, sk_{san}) \leftarrow KGen_{san}(1^n)$$

Signing:

The inputs of $Sign$ algorithm are given below.

- Message: $m \in \{0,1\}^*$,
- Security parameter: n,
- Secret key of the signer: sk_{sig},
- Public key of the sanitizer: pk_{san},
- Description $ADM \in \mathbb{N} \times 2^{\mathbb{N}}$ of admissible block where ADM contains the number of l blocks in m.

The output of $Sign$ algorithm is a signature σ (or $\perp$, indicating an error):

$$\sigma \leftarrow Sign(m, sk_{sig}, pk_{san}, ADM)$$

In the proposed protocol, we use ID-Based signature scheme as $Sign$ algorithm.

Sanitizing:

The inputs of $Sanit$ algorithm are given below.

- Message: $m \in \{0,1\}^*$,
- Signature σ,
- Public key of the signer: pk_{sig},
- Secret key of the sanitizer: sk_{san},
- Modification instruction which modifies the message m: $MOD \subseteq \mathbb{N} \times \{0,1\}^l$

The outputs of $Sanit$ algorithm are a modified message m' and a new signature σ' (or probably $\perp$ in the existing of an error).

$$(m', \sigma') \leftarrow Sanit(m, MOD, \sigma, pk_{sig}, sk_{san})$$

Verification:

The inputs of Verify algorithm are given below.

- Message: $m \in \{0,1\}^*$,
- Signature σ,
- Public key of the signer: pk_{sig},
- Secret key of the sanitizer: sk_{san}.

The output of Verify algorithm is a bit $d \in \{true, false\}$ which verifies the correctness of the message m and the signature σ with respect to public keys pk_{sig} and pk_{san}.

$$d \leftarrow \text{Verify}(m, \sigma, pk_{sig}, pk_{san})$$

Proof:

The inputs of Proof algorithm are given below.

- Secret key of the signer: sk_{sig},
- Message: $m \in \{0,1\}^*$,
- Signature σ,
- Polynomially many message-signature pairs: $(m_i, \sigma_i)_{i=1,2,\ldots,q}$
- Public key of the sanitizer: pk_{san}.

The output of Proof algorithm is a string $\pi \in \{0,1\}^*$:

$$\pi \leftarrow \text{Proof}(sk_{sig}, m, \sigma, (m_1, \sigma_1), ..., (m_q, \sigma_q), pk_{san})$$

Judge:

The inputs of $Judge$ algorithm are given below.

- Message: $m \in \{0,1\}^*$,
- Signature σ,
- Public key of the sanitizer: pk_{san}.
- Public key of the signer: pk_{sig},
- Proof: π.

The output of $Judge$ algorithm is a a decision $d \in \{Sig, San\}$ which shows the creation of the message and signature pair has done by the signer or sanitizer.

$$d \leftarrow Judge(m, \sigma, pk_{sig}, pk_{san}, \pi)$$

The correctness of the sanitizable signature as follows:

Signing Correctness:

The signing correctness is accepted by the Verify function:

$$\text{Verify}(m, \sigma, pk_{sig}, pk_{san}) = true.$$

for any $n \in \mathbb{N}$ is a security parameter, $(sk_{sig}, pk_{sig}) \leftarrow KGen_{sig}(1^n)$ and (pk_{san}, sk_{san}) $\leftarrow KGen_{san}(1^n)$ are any pairs, $m \in \{0,1\}$ is any message, $ADM \in \mathbb{N} \times 2^{\mathbb{N}}$ and any $\sigma \leftarrow Sign(m, sk_{sig}, pk_{san}, ADM)$.

Sanitizing Correctness:

The sanitizing correctness is accepted by the Verify function:

$$\text{Verify}(m', \sigma', pk_{sig}, pk_{san}) = true$$

for any $n \in \mathbb{N}$ is a security parameter, $(sk_{sig}, pk_{sig}) \leftarrow KGen_{sig}(1^n)$ and (pk_{san}, sk_{san}) $\leftarrow KGen_{san}(1^n)$ are any pairs, $m \in \{0,1\}$ is any message, σ with $Verify(m, \sigma, pk_{sig}, pk_{san}) = true$, any $MOD \subseteq \mathbb{N} \times \{0,1\}^l$ matching ADM from σ, and any pair $(m', \sigma') \leftarrow Sanit(m, MOD, \sigma, pk_{sig}, sk_{san})$.

Proof Correctness:

The sanitizing correctness is accepted by the $Judge$ function:

$$Judge(m', \sigma', pk_{sig}, pk_{san}, \pi) = San$$

for any $n \in \mathbb{N}$ is any security parameter, $(sk_{sig}, pk_{sig}) \leftarrow KGen_{sig}(1^n)$ and (pk_{san}, sk_{san}) $\leftarrow KGen_{san}(1^n)$ are any pairs, $m \in \{0,1\}$ is any message, σ is any signature, any MOD matching ADM from σ, and any $(m', \sigma') \leftarrow Sanit(m, MOD, \sigma, pk_{sig}, sk_{san})$ with $Verify(m', \sigma', pk_{sig}, pk_{san}) = true$, and any $m_1, ..., m_q$ and $ADM_1, ..., ADM_q$ with $\sigma_i \leftarrow Sign(m_i, sk_{sig}, pk_{san}, ADM_i)$ and $(m, \sigma) = (m_i, \sigma_i)$ for some i, any $\pi \leftarrow Proof(sk_{sig}, m', \sigma', (m_1, \sigma_1), ..., (m_q, \sigma_q), pk_{san})$.

A sanitizable signature should satisfy several security requirements such that $unforgeability$, $immutability$, $privacy$, $transparency$ and $accountability$. The informal definitions of these properties are presented below:

Unforgeability:

Nobody can produce valid signature without the knowledge of private keys sk_{san}, sk_{sig}. By this definition, nobody can compute (m', σ') such that Verify$(m', \sigma', pk_{sig}, pk_{san}) = true$.

Immutability:

The sanitizer can only modify the ADM, he cannot modify any part of the signature differs from ADM.

Privacy:

No one can restore the message from the sanitized parts. In other words, original message is unrecoverable from the sanitized part.

Transparency:

Only the signer and the sanitizer can distinguish whether the signatures are from them or not.

Accountability:

In case of any dispute, neither sanitizer nor signer are responsible for the message from others such that the signer is able to prove that the message is sanitized by the sanitizer.

2.2.3 Long Term Confidentiality

The secrecy of the sensitive data is the most important component in every systems. In this thesis, we use Feldman's secret sharing schemes [23] to assure long term confidentiality.

Feldman's verifiable secret sharing (VSS) based on gathering the secret from the proper participants although there are some inconsistent participants. First definition of veri-

fiable secret sharing was made in 1985 by Chor et.al [20]. The idea in [20] was getting secret from the participants which were simultaneous-broadcast networks. However, this proposed scheme was used in many systems but Feldman introduced the most common use of verifiable secret sharing. It was based on the combination of Shamir's secret sharing scheme and homomorphic encryption [4].

Feldman's VSS runs as follows: Let p and q are primes such that $q \mid p-1$, $g \in \mathbb{Z}_p$ of order q, $d(x)$ is polynomial over $\mathbb{Z}_p$ with coefficients $p_1, p_2, ..., p_k$ selected by the dealer. Then the values $g^{p_1}, g^{p_2}, ..., g^{p_k}$ are broadcasted by the dealer and the value $s_i = p(i)$ ***(mod p)*** is transmitted secretly to each participants $P_i, i = 1, .., n$. The verification of each participant's share is done by the equation:

$$g^{s_i} \stackrel{?}{=} (g^{p_0})(g^{p_1})^i (g^{p_2})^{i^2} ... (g^{p_k})^{i^k} \textit{(mod p)} \tag{2.1}$$

The equation holds when each participant's share is proper and the complete dealing of share is satisfied with the all participant' proper share.

[4] The early definition of homomorphic encryption was introduced by Rivest et.al [38]. Craig Gentry [24] described fully homomorphic scheme which supports both addition and multiplication on ciphertext. A cryptosystem is additive homomorphic such that $E(p_1, p_2) = E(p_1) + E(p_2)$ where E is encryption function and p_1 and p_2 are plaintexts. A cryptosystem is multiplicative homomorphic such that $E(p_1, p_2) = E(p_1).E(p_2)$ where E is encryption function and p_1 and p_2 are plaintexts.

CHAPTER 3

MODEL OVERVIEW

In this chapter, we propose a secure information sharing model where both users and authority communicate in secure channel. With this model, all users will have digital signatures and the authority will treat as a trusted third party. All processes in the model will be done by electronic signatures. The secrecy of sensitive data is crucial part of the model. We will propose an information storage model with long term confidentiality.

The proposed model can be considered as an infrastructure for electronic exam. Not only electronic exam, but also the other systems where users and authority interact, can be derived from this model. For example, electronic banking systems, electronic voting systems, online health service systems including sensitive health information of people, electronic registration systems, etc.

The model has four protocols:

- User Identification Protocol
- Data Providing Protocol
- Data Process Protocol
- Data Storage/Retrieving Protocol

There are two roles in the model: *authority* and *users*. The roles of the users varied in three other roles: *data provider*, *data user* and *data editor* (Figure 3.1). Besides the protocols above, there is an application phase which is prepared by the authority. All data processes are done in application phase and at the end of it, the users and the authority interact with the product of this phase. The authority is responsible for all protocols and phases in the model, users are only responsible for their authorized protocols and phases.

For any user, *User Identification Protocol* determines the role and public-private key pairs of the user. If the user has *Data Provider* role, user can supply data to system with *Data Providing Protocol*. If the user has *Data Editor* role, the user can do some processes on the data with *Data Process Protocol*. If the user has *Data User* role, the user interacts with the authority in *Application Phase* by the data prepared for him.

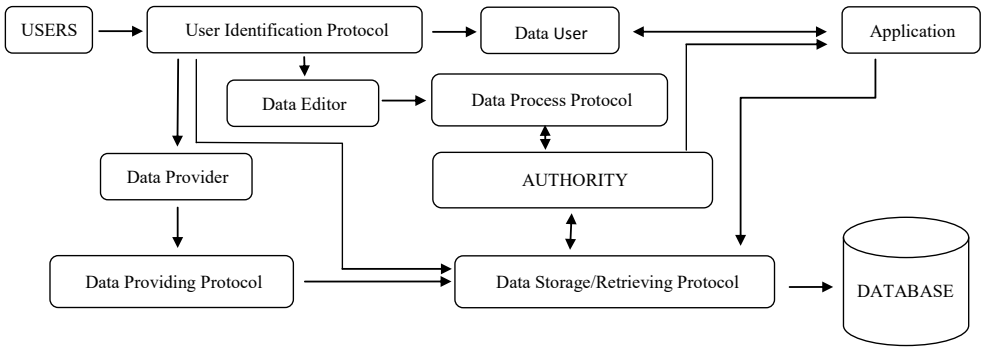


Figure 3.1: Model Overview

All the data used in the model is stored in the database by *Data Storage/Retrieving Protocol*.

In this chapter; user registration to system, private and public key generation for users, transmission of personal information of any user to authority, and system authentication will be explained in User Identification Protocol. The next section will emphasize the data providing protocol by data providers. In the third part, data process protocol will be explained. The data storage and retrieving part will be explained in fourth part. The last part will cover the application of the data and user interaction between authority.

3.1 User Identification Protocol

In this section; system registration, generation of public and private key pairs, and authentication will be explained. At the end of this protocol, any user related to system will be able to do his work in an authenticated environment.

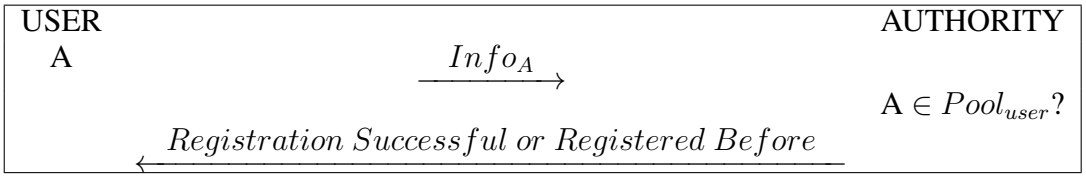
3.1.1 System Registration

In any model, the authority wants to identify his users in the system. The first step of this identification is done by system registration. In our proposed model, some personal information about any user (name, surname, username, password, e-mail, telephone number, passport number, national identity number, etc.) are needed for registration. The first registration to the system is done by registration offices supplied by the authority. By this registration method, one can gather biometric information of any user. This is another option which can be integrated to the id-card systems. In our proposed scheme, we do not want any biometric information of the user since it increases the cost and there are obstacles in law to get the biometric information of anyone. After registered in registration office, users can interact with authority in a secure channel which is assumed by us. For storage issues, we do this by the model's

Data Storage/Retrieving Protocol (see Section3.4).

Any user ***A***, with personal information ***Info*** can register system like this:
$Info_A$:= {name, surname, username, password[1], e-mail, telephone number, passport number, national identity number}

Figure 3.2: System Registration Protocol



In our model, the authentication is done between users and authority. As we know, human based authentication relies on three items [41]:

- Something you know: The most frequently usage of authentication system. Passwords are the example of used in here. The usage of *something you know* is simple, however it should be forgotten by the users.
- Something you have: This prevents forgetting something you know, but this should be with you all the time. As an example, smartcards or mobile phones are something you have.
- Something you are: This provides you not to forget something and not to lose something. As an example, fingerprints are something you are. It is more expensive than the others. There should be special readers (fingerprint reader, retinal scan, etc.) for authentication.

The proposed model basically relies on password and digital signature authentication. If someone has an biometric information for authentication, the model is compatible with it. So, in our proposed model; something you know: *passwords*, something you have: *digital signatures*, something you are: *fingerprints or retina* (if authority decides to use).

3.1.2 Public and Private Key Generation

Any registered user should have digital signature for signing every processes in the model. In this part, asymmetric key encryption solutions are used. The most commons are RSA cryptosystem, elliptic curve digital signature algorithms, Paillier cryptosystems, and ID-Based digital signature methods.

We use ID-Based encryption scheme (see Section2.2.1) for digital signature issues. The authority will treat as Private Key Generator (PKG). Therefore, the authority should decide his public and private key pair as follows [15]: The authority

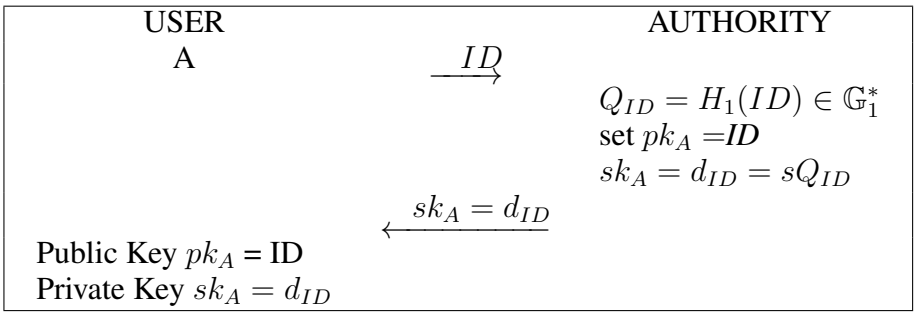
[1] The hash of password will be stored in the database. Salting is used for passwords. Our proposed model is using PKCS#5 for salting. [1]

- chooses two groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of order *q* depending on the security parameter *k* $\in \mathbb{Z}^+$.
- determines an admissible bilinear map *e*, such that $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ and chooses a random generator $P \in \mathbb{G}_1$.
- chooses a random $s \in \mathbb{Z}_q^*$ and sets his public key $pk_{Aut} = P_{Pub} = sP$, and private key (master key) $sk_{Aut} = s \in \mathbb{Z}_q^*$.
- chooses two cryptographic hash functions H_1 and H_2 such that,
  $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1^*$ and
  $H_2 : \mathbb{G}_2 \rightarrow \{0,1\}^n$ for some *n*,
  then the message space $\mathcal{M}$ and the ciphertext space $\mathcal{C}$ are:
  $\mathcal{M} = \{0,1\}^n$ and $\mathcal{C} = \mathbb{G}_1 \times \{0,1\}^n$ so that
  the public parameters are: $\langle q, \mathbb{G}_1, \mathbb{G}_2, e, n, P, P_{Pub}, H_1, H_2 \rangle$ and private parameter is $s \in \mathbb{Z}_q^*$.

Any user in the system should contact with authority to get his private key. In the system, public key is uniquely identifying user's identity such that national identity number, passport number, electronic mail address, telephone number, etc. can be used. For given identity *ID*$\in \{0,1\}^*$ of the user *A*, the system generates private key like that:

- The authority computes $Q_{ID} = H_1(ID) \in \mathbb{G}_1^*$, then the public key for user *A*: pk_A=*ID*.
- The authority sets the private key of user *A* $sk_A = d_{ID} = sQ_{ID}$.

Figure 3.3: Key Generation Protocol



At the end of this phase, every user will have public and private key pairs. Here, we assumed that the communication channel between users and authority is secure.

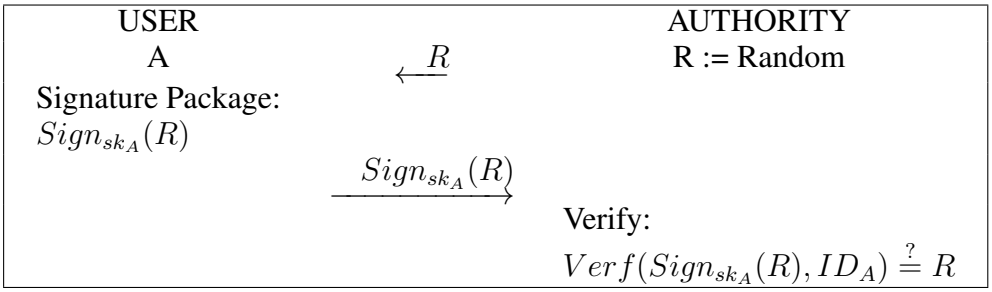
3.1.3 System Authentication

In this section, we show that how authentication done by ID-Based cryptosystem. System authentication is done by both users and authority's public and private key pairs. Authority generates a random *R* and sends to the user. User should sign this *R* with his

private key and sends to authority. If the signature is valid then the user authenticate to system.

Authentication done in two phases: *Sign* and *Verify*.

Figure 3.4: System Authentication Protocol



The signature function:

$$Sign_{sk_A}(R)\colon \{0,1\}^n \to \mathbb{G}_1^* \times \mathbb{Z}_q^*$$
$$R \mapsto Sign_{sk_A}(R) = (u, v)$$

where $u = vsk_A + kP_1, v = H_1(R, r), r = e(P_1, P)^k$ (see Section2.2.1)
The verification function:

$$Verf((u, v), ID_A)\colon (\mathbb{G}_1^* \times \mathbb{Z}_q^*) \times \{0,1\}^n \to \{0,1\}$$
$$((u, v), ID_A) \mapsto Verf((u, v), ID_A) = \{0,1\}$$

At the end of this protocol, authority decides whether the user should authenticate the system or not upon the result of *Verf* function.

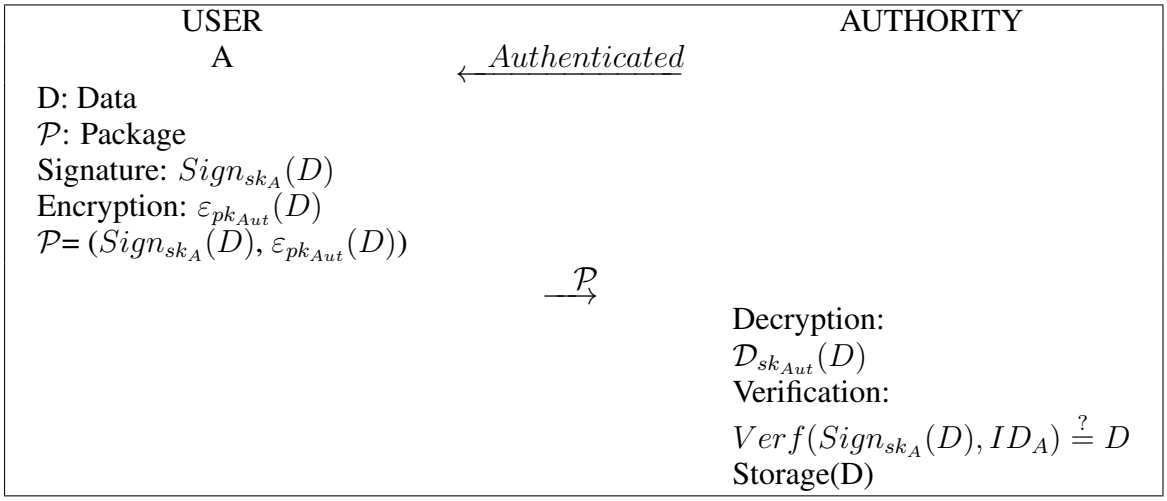
3.2 Data Providing Protocol

The model needs some data from the users. Not only all users, but also some users who have data providing role provides data to system. The user role definition is done by the authority. In this section, we explain how any required data will be provided to system by authenticated users.

First, users with data providing role should authenticate the system by user identification protocol. Then, users prepare required data, sign it with their private keys and encrypt this data with authority's public key. At the end, the data package consists of encrypted form of the data and the signature, after that this package is sent to the authority. The authority first decrypts the data with his private key and checks the signature. If the verification is done, then the data will send to database and store at there by data storage protocol.

Any authenticated user with data providing role should send the package like that:

Figure 3.5: Data Providing Protocol



In the protocol (Fig. 3.5), the encryption function is defined as:

$$\begin{aligned} \varepsilon_{pk_{Aut}}(D)\colon \{0,1\}^n &\to \mathbb{G}_1 \times \{0,1\}^n \\ D &\mapsto \varepsilon_{pk_{Aut}}(D) = (u, v) \end{aligned}$$

where $u = rP$ ($r \in \mathbb{Z}_q^*$ random) and $v = D \oplus H_2(g_{ID}^r)$, $g_{ID}^r = e(Q_{ID}, P_{Pub})$

the decryption function is defined as:

$$\begin{aligned} \mathcal{D}_{pk_{Aut}}(u, v)\colon \mathbb{G}_1 \times \{0,1\}^n &\to \{0,1\}^n \\ (u, v) &\mapsto \mathcal{D}_{pk_{Aut}}(u, v) = D \end{aligned}$$

where $D = v \oplus H_2(e(D_{ID}, u))$

Here, the authority uses his private key to recover the data *D*. After decryption, authority uses *Verf* function for verification of the data. The authority stores the data via Data Storage/Retriving Protocol (see Section 3.4) after decryption and verification phases.

3.3 Data Process Protocol

The provided data should be in some standard format in order to make some processes on it. For that reason, authority should check the data before the storage of it. In this section, we explain how the provided data revised by the authority.

In our proposed protocol, data process should be done by the authority or the users who have data process role (data editor) supplied by the authority. Every data in the model should be in a standard format. Sometimes, it is not possible to get the standardization on the required data. The data editor should check the correctness of the data. Some part of the data should be misspelled or there should be some errors on it. The data editor corrects that kind of non-standard things on the data. After that correction,

he signs the data with his private key, arrange the reference of the data and store to database with data storage protocol. For example, in electronic exam, a question maker prepares his question with some misspelling or sometimes it is possible to forget to write right answer in choices. Instead of refusing whole question, if the question maker determines the choices as admissible modification, data editor can change that part and accept the question. In our model, sanitizable signature schemes (Section 2.2.2) is used after any editing in the data, so that every data provider attaches a description of the admissible modifications *ADM* for the data. Here, the data editor will act as *sanitizer*.

Any user with data process role should edit the data like that:

Figure 3.6: Data Process Protocol

AUTHORITY		DATA EDITOR
D: Data Signature: $Sign_{sk_A}(D)$ ADM: Admissible modifications $\mathcal{P}$: Package $\mathcal{P}$= ($Sign_{sk_A}(D)$, D, ADM)		
	$\xrightarrow{\mathcal{P}}$	
		Modification: MOD Sanitization: $Sanit(D, MOD, Sign_{sk_A}(D),$ $pk_A, sk_{san}) \rightarrow (D', Sign'_{sk_A})$
	$\xleftarrow{(D', Sign'_{sk_A})}$	
$Verify(D', Sign'_{sk_A}, pk_A, pk_{san})$ Storage(D)		

In the above protocol, the data D', will be the data to be stored and the signature $Sign'_{sk_A}$ will be the new signature.

3.4 Data Storage/Retrieving Protocol

In this section, we explain how any data in the model will be stored and gathered. The secrecy of the system relies on the secrecy of the data used in it. Therefore, the sensitive data of the model should be kept in a secure way and it should provide a long term confidentiality. To achieve the security requirements, we use verifiable secret sharing scheme for long term confidentiality. We published parts of this section in [31].

The data confirmed by both the owner and authority (or data editor) where it is kept in authority's database. In our proposed model, the data storage protocol includes *n* databases. The databases no need to be located in the same place, they can be settled in different areas. In this protocol, the data is sent to other databases by the authority who behaves as *dealer*. By the previous protocol, every data has main and reference part. The main part of any data consisting of the sensitive part of the data and the

reference part of the data consists of the address of the data in the database. Since the reference part of the data is just addresses of it, there is no necessity for encryption on it. In our model, the reference part of any data is kept without any encryption in the main database of the authority. Unlike the reference part of data, the main part of the data is kept in the encrypted form and separated into the *n* pieces which will be kept in *n-databases*. The system mainly uses $(n, k)-$threshold scheme where $k - 1 < n/2$.

The AES [21] is used for the encryption of the sensitive part of the data. To provide long term confidentiality, verifiable secret sharing scheme is used. We use Feldman's VSS scheme [23]. This scheme allows us to provide prevention of misbehaving dealers' inconsistent dealings. In order to provide long term confidentiality, the shares should periodically be renewed by the authority. *Randomized secret* is proposed by Ostrovsky and Yung [33] such that this randomized secret is verified by all dealers and updated by a polynomial. For renewal phase, this technique can be applied but we prefer to use Feldman's VSS for stable sharing and right dealing of the secret. We preferred to use the same methodology used in [25] for periodic update of the share in our model.

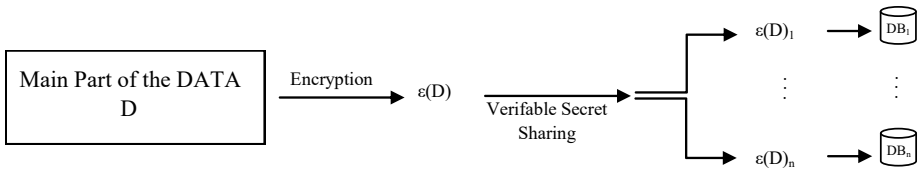


Figure 3.7: Storage to Database

Any data in the model is split into *n-pieces* with secret sharing scheme by the authority. The authority first encrypts the data with AES and sends the to related databases. (Fig. 3.7) Each piece of the data should be verified by the related database in order to be stored. After that verification, for providing long term confidentiality the stored data should be renewed in some periods. We show these parts in details as follows:

3.4.1 Verification of the Shared Secret

In this part, we show that how any piece of the data in the related database is verified. The verification of the secret is done by the authority as follows: Let *p* and *q* be primes such that $q|p - 1$. Let $g \in \mathbb{Z}_p$ of order q. Then the authority applies following steps in order to transmit the secret and verification. Authority

1. chooses the polynomial *p(x)* over $\mathbb{Z}_p$ whose coefficients are $p_0, p_1, ..., p_k$ defined and explained in Section 2.2.3,
2. determines the data to be transmitted,
3. broadcasts the values $g^{p_0}, g^{p_1}, ..., g^{p_k}$
4. transmits the value of $\mathcal{E}(D)_i = p(i)$ *(mod q)* to each database servers DB_i.

Each database servers DB_is check the equation (3.1) whether the share is proper or not.

$$g^{s_i} \stackrel{?}{=} (g^{p_0})(g^{p_1})^i(g^{p_2})^{i^2}...(g^{p_k})^{i^k} \textit{(mod p)}. \tag{3.1}$$

The sharing of the secret is completed with the verification of the above equation by all database servers.

3.4.2 Renewal of the Secret Share

The secret should be updated periodically in order to provide long term confidentiality. The challenging section in this part is the unstable share updates in the share renewal section by the adversary. In [25] and [16] proposed verifiable secret sharing schemes to solve that kind of challenges. In order to detect the wrong dealt shares by the database servers, we preferred to use the same method used in [25]. Each database servers DB_i's renew the secret share as follows:

1. The polynomial $\delta_i(z) = \delta_{i1}z^1 + \delta_{i2}z^2 + ... + \delta_{ik}z^k$ such that k is random numbers $\{\delta_{im}\}_m$ from $\mathbb{Z}_q$ where $m \in \{1, ..., k\}$ is defined by each DB_is.
2. DB_i calculates $\epsilon_{im} = g^{\delta_{im}}$ *(mod p)* where $m \in \{1, ..., k\}$.
3. DB_i calculates $u_{ij} = \delta_i(j)$ *(mod q)* where $j \in \{1, ..., n\}$ and $e_{ij} = \mathcal{E}_j(u_{ij})$, $\forall i \neq j$
4. The message $M_i^{(t)} = (i, t, \epsilon_{im}, e_{ij})$ where $j \in \{1, ..., k\} - \{i\}$ and the signature $\sigma_i(M_i^{(t)})$ are produced and broadcasted by DB_i.
5. DB_i decrypts the e_{ij} and verifies share correction by the equation (3.1) with using
   $g^{u_{ji}} \stackrel{?}{=} (\epsilon_{j1})^i(\epsilon_{j2})^{i^2}...(\epsilon_{jk})^{i^k}$ *(mod p)*
6. The above equation holds with the other participants' correct messages. Therefore, the messages came from the other participants are accepted by DB_i and the verification is completed.
7. DB_i's own share updated by
   $s_i^{(t)} \leftarrow s_i^{(t-1)} + (u_{1i} + u_{2i} + ... + u_{ni})$ *(mod q)*
   and the other variables are erased.

There should be some irregularities during renewal part. In case of such irregularity in the verification part, the misbehaving participant should be detected by the dealer. Each server should agree on the misbehaving participant. A random is sent to misbehaving participant by the dealer. This random is signed and encrypted by the participant and resent to the authority. The verification is done when the random is verified by the misbehaving participant. If the misbehaving participant can not verify the random, the renewal period updated by the dealer with the equation (3.2)

$$s_i^{(t)} \leftarrow s_i^{(t-1)} + \sum_j u_{ji} \tag{3.2}$$

where $j \neq d$ *(mod q)*.

3.4.3 Retrieving the Data

In this part, we show that how needed data gathered from the related database servers. Any data in the model is separated into reference and main part. The reference part of the gathered data is decided by the authority. With these references, the authority determines the IDs of each data. Then the authority retrieves the required data as follows:

1. Let the reference part of the data be $RD = \{ID_1, ID_2, ..., ID_n\}$.
2. $DATA\ D = \{\mathcal{E}(D_1), \mathcal{E}(D_2), ..., \mathcal{E}(D_n)\}$ is the related encrypted data in the database servers.
3. $\mathcal{E}(D_i) = s_r = p^t(r)$ is the data retrieved from the databases such that $r \in \mathcal{B}$ where $\mathcal{B}$ is the group of servers with wrong shares.
4. The k-degree random polynomial δ_i over $\mathbb{Z}_q$ where $\delta_i(r) = 0$ is chosen by each database servers $P_i \in \mathcal{D} = \mathcal{A} - \mathcal{B}$ and each of them calculates $\delta_{i0} = -\sum_j \delta_{ij} r^j$ *(mod q)*, $j \in \{1, ..., k\}$ where $\mathcal{A}$ is the group of servers.
5. Each P_i broadcasts $\mathcal{E}_j(\delta_i(j))$, $i, j \in \mathcal{D}$.
6. The new share $s'_r = s_r + \sum_j \delta_j(i)$ is created by each P_i's and sent to P_r with $\mathcal{E}_r(s'_i)$
7. P_r decrypts the share, s_r is recovered by polynomial interpolation.
8. Finally, the authority uses the encryption key and decrypt the $s_r = \mathcal{E}(D)$ and gets $\overline{\mathcal{D}}(\mathcal{E}(D)) = D$ as required.

3.5 Application Phase

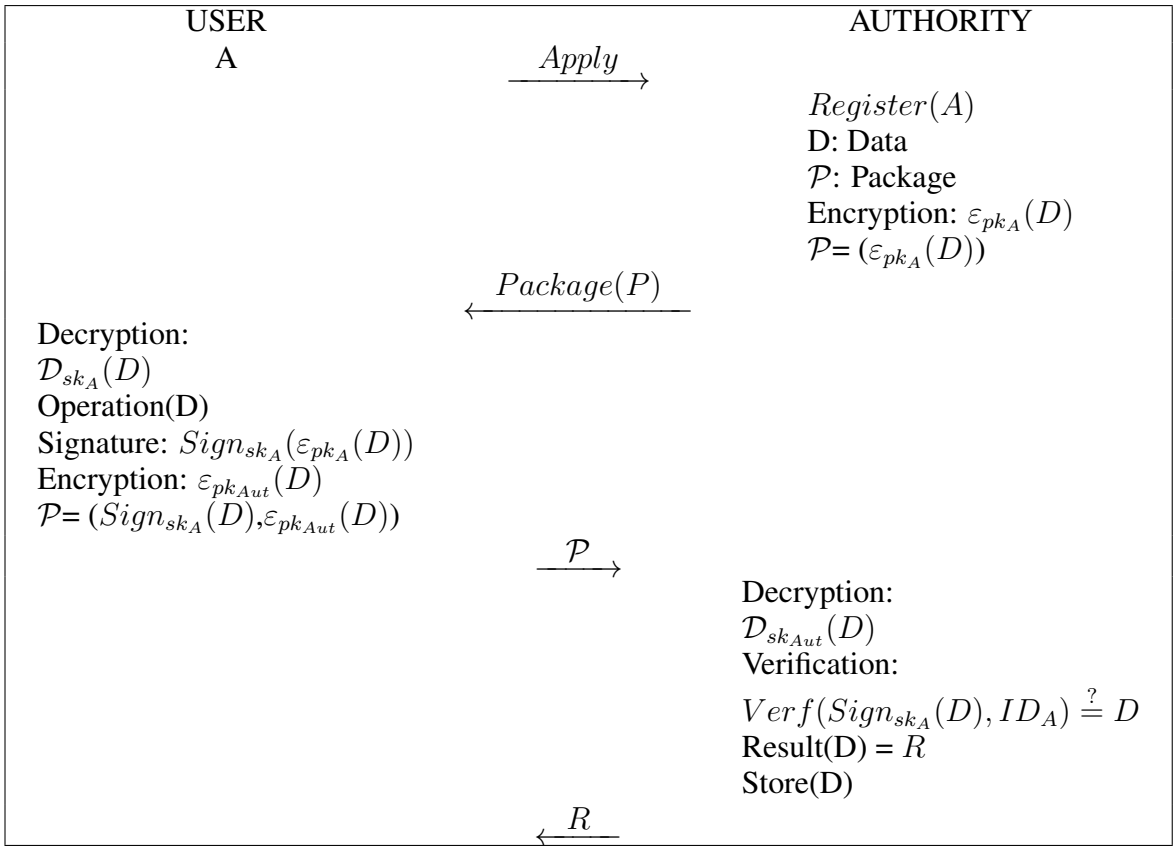
In this section we explain how the users and the authority are interacting themselves. The user needs some application from the model. For example, if the model is applied

to banking transaction, in this part user does some transactions in his account and sends the directions to authority to complete transactions. On the other hand, if the model is applied to health service system, this part can be thought of the process of the medical record of the patients. If we think about the electronic examination, this part of the model is thought of the exam part.

Here, the users should apply to authority in order to enroll application of the model. The authority prepares packages to each users. Users decrypt the packages with their private keys. They can do some processes on the package in the phase. After finishing the processes on the package, they sign it with their private key and encrypt with authority's public key. The authority gets the package, decrypts with his private key and verifies it. Then authority does some operations on the package relating to the application and send the results of the operation to the users. At the end, all the processes done in the application phase are stored in the databases with data storage and retrieving protocol.

The protocol of the application explained in Table 3.8.

Figure 3.8: Application Phase Protcol



In the protocol, the $Operation$ function is defined:

$$Operation(D)\colon \{0,1\}^n \to \{0,1\}^n$$
$$D \mapsto Operation(D)$$

the $Result$ function is defined:

$$Result(D)\colon \{0,1\}^n \to \{0,1\}^n$$
$$D \mapsto Result(D)$$

such that, the output is the value of results done in $Operation$ function. At the end, all the data used in application part are archived by $Store$ function with Data Storage/Retrieving protocol.

3.6 Security Analysis

In this section we give some security analysis of the proposed model with respect to authentication of the users, privacy of the sensitive and the presence of malicious servers and malicious users. Here, we assume that the lines between authority to database and authority to users are secure. We published a preliminary version of security analysis of the model in [31] as an application of the model as electronic examination.

Theorem 3.1. *The proposed system possesses authenticity between users and authority.*

Proof. Every users in the system have public and private key pairs. The proposed system uses ID-based cryptosystem for authentication. Any user A signs a random R which is sent by the authority. The signature of the random R is $Sign_{sk_A}(R)$. The authority verifies the signature of the random with the function $Verf$. The authentication is provided if and only if $Verf(Sign_{sk_A}(R), ID_A) \stackrel{?}{=} R$ holds. □

Theorem 3.2. *The proposed model provides long term secrecy on the sensitive data.*

Proof. The sensitive data D of the model is divided into n-pieces by secret sharing scheme. Each piece of the D is encrypted with AES and sent to related database. Nobody else gets the decrypted form of the piece of D without knowing the key k used in AES.

For long term confidentiality, the proposed model uses verifiable secret sharing scheme. Each piece of the data is renewed periodically as mentioned section 3.4.2. This provides long term secrecy on the sensitive data. □

Theorem 3.3. *Any data D in the servers can be reconstructed from the honest servers if there are at most $t-1$ malicious database servers and the system preserves its security.*

Proof. In section 3.4 it is assumed that (t, n)-threshold scheme with $t - 1 < n/2$. There are $t = n/2$ trusted servers, if there are at most $t-1$ malicious database servers. Therefore, the data D can be gathered from $t = n/2$ trusted servers with secret sharing scheme. □

Theorem 3.4. *In the renewal of the secret share, the data storage protocol has the database servers' authenticity.*

Proof. In the above phase, the message M should be signed by each database server DB_i with his private key. Each server verifies the signature of the message M and the equation of Feldman's VSS is satisfied. The authenticity of the database servers is supplied by this verification. □

Theorem 3.5. *If any malicious sanitizer tries to modify the message blocks which are not determined by the signer as modifiable then this will produce a wrong signature.*

Proof. Any malicious sanitizer $\mathcal{M}$ gets the signature σ_i for the data D from the signer with the description of ADM_i from the signer with his keys pk_{sanM}. Let (D', σ', pk'_{san}) be the valid pair, then with the usage of them same key $pk'_{san} = pk_{sanM}$, m' will be different in some blocks of D in at least one non-determined modifiable blocks. [17] □

CHAPTER 4

SECURE ELECTRONIC EXAM MODEL

In this chapter we introduce our *Secure Electronic Exam Model* in details. We provide the required materials for the model in Section 3. *The Secure Electronic Exam Model* is also derived from the model introduced in Section 3.

Here, the model is defined firstly, then all components of model and protocols are introduced. Finally, we summarize our work and conclude the chapter.

4.1 Introduction

As mentioned in Section 1 there is not so much source for the security of electronic exam in the literature. Previous studies mentioned about the application of the electronic exam so that the security of the electronic exam remained in the backwards. To hold a secure electronic exam, a lot of cryptographic problems should be solved. In this chapter, we applied the model introduced in Section 3 to solve the information security related problems and hold a secure electronic exam.(Figure 4.1) The implementation of the proposed model is performed on Microsoft Visual Studio 2010 and SQL Server Management Studio 2008 used as database server. For cryptographic issues we modified Microsoft Cryptography Library and The Bouncy Castle Crypto API [3].

The model in Section 3 is related to secure electronic examination as follows:

- Registration $\leftrightarrow$ User Identification Protocol
- Login $\leftrightarrow$ User Identification Protocol
- Key management system $\leftrightarrow$ User Identification Protocol
- Authentication to system $\leftrightarrow$ User Identification Protocol
- Question entry $\leftrightarrow$ Data Providing Protocol
- Supervising of questions $\leftrightarrow$ Data Process Protocol
- Candidate registration to exam $\leftrightarrow$ User Identification Protocol

- Exam preparation $\leftrightarrow$ Application Phase Protocol & Data Storage/Retrieving Protocol
- Exam and Results $\leftrightarrow$ Application Phase Protocol
- Archive $\leftrightarrow$ Data Storage/Retrieving Protocol

In registration part, users introduce themselves to system. Login part is almost same with the other applications. Users login to system with their usernames and passwords. Any user is able to see his authorized parts. Key management system produces asymmetric keys. In this application, we used RSA for asymmetric key issues. In authentication to system part, system authentication via RSA key and signing are done. In question entry part, exam question are sent to question pool. The questions in the pool are revised by supervisors in supervising of questions parts. The candidates enrolling the exam should register in candidate registration to exam part. Exam authority uses tags of the questions in order to prepare exam in exam preparation part. Exam is hold by the exam authority with the attendance of candidates and proctors. Results and archive process are done by the exam authority after exam.

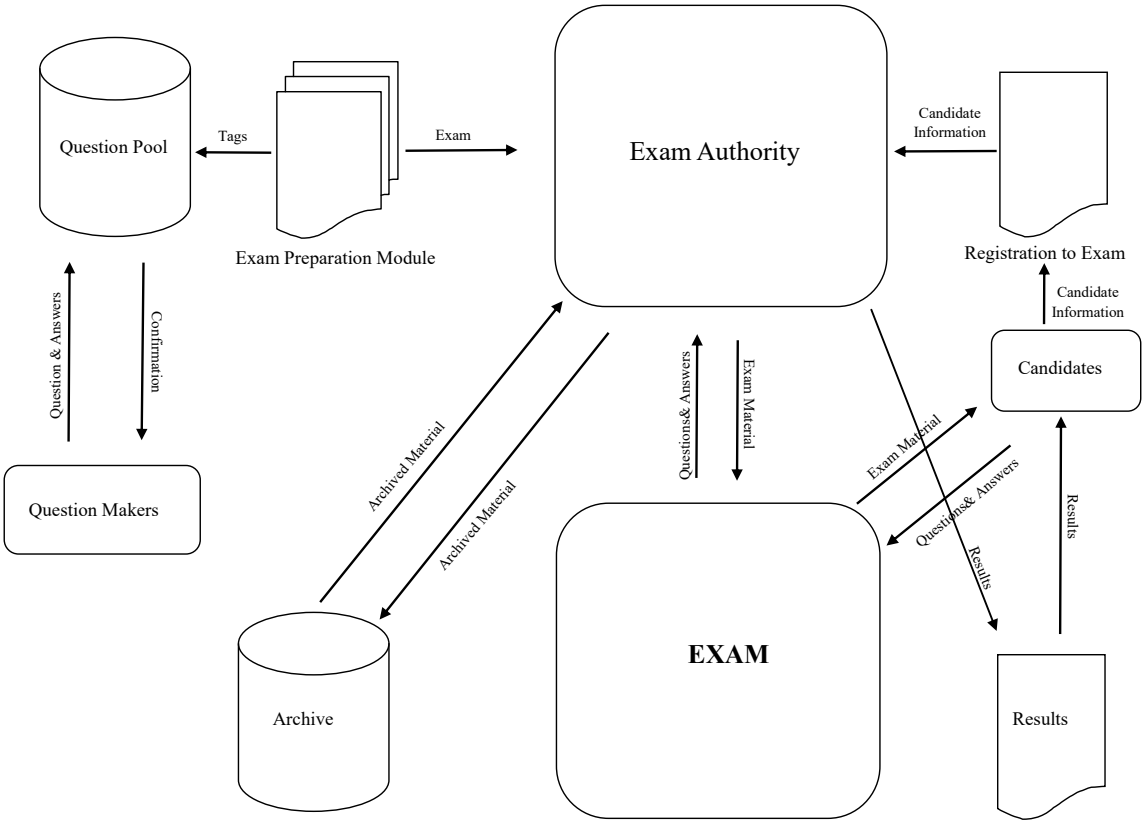


Figure 4.1: Proposed Model.

We define the roles for the model:

- Exam Authority

- Candidates
- Question makers
- Question supervisors
- Proctors

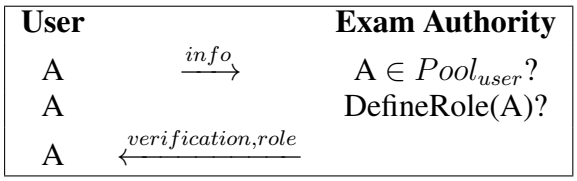
Exam authority is responsible for every processes of the exam. Candidates are only responsible for enrolling exam and seeing their results in the exam. Question maker can write and edit questions. Question supervisors are responsible for checking for the questions in the pool and rewrite them if required. Proctors are only responsible for the processes in the exam places. (The system is flexible for other user roles.)

4.2 Registration

In this section, the registration to system in electronic exam is explained. Every user in electronic exam should register to system in order to do all processes in the system. Some information about the user is needed for this part.

In our proposed model, we assume that, the first registration of any user is done in the centers supplied by the authority. In the center, the authorization of the user is defined. The default authorization is given to all users. The protocol for registration is given below.

Figure 4.2: Registration Protocol



In registration part, name, surname, username, password, and user e-mail are needed. (Figure B.1) The exam authority checks whether the user registered before or not. The role of the user is determined by the authority with the procedure *DefineRole()*. With the role definition, every user will have different privileges in the exam. The password is being hashed and the hash of it stored in the database. For password salting, our system uses $PKCS\#5$ [1]. The usage of $PKCS\#5$ provides different hashes in the database even two passwords are the same. Let, for any user A, we define A's info like that:

- name: A_{name}
- surname: $A_{surname}$
- username: $A_{username}$

- password: A_{pwd}
- e-mail: A_{email}

The algorithm in registration as follows:

Algorithm 1 User Registration

Input: User info: A_{name}, $A_{surname}$, $A_{username}$, A_{pwd}, A_{email}
Output: Verification message M

1: **if** A $\in Pool_{user}$ = true **then**
2: M = User registered before
3: **else** Store(User info)
4: M = Registration is successful
5: **end if**
6: **return** M

The procedure $Store()$ in Algorithm 1 stores user's info in the following way:

$$Store(InfoA) = \begin{cases} \text{Info A} = A_{pwd} & \text{write database with PKCS\#5 Standards} \\ \text{else} & \text{write database with no change} \end{cases}$$

The PKCS#5 stands for *Password-Based Cryptography Standard* was published by RSA Laboratories in October 5, 2006 [1]. The main idea of PKCS#5 is based on the *salt*[1] and *iteration count* [2] of password-based encryption.

The password encryption protocol in our proposed scheme (Algorithm 2) was done by using the library: *Bouncy Castle* [3]. The KDF *(Key Derivation Function)* is generated by the procedure *Pkcs5S2ParametersGenerator()*. The salt is randomly chosen and the number of iteration is 1000. The hash of the password derived by applying GenerateDerivedMacParamaters() procedure to KDF. A *MAC (Message Authentication Code)* derived from the password, salt, and the hash are generated by this procedure. With these parameters, the procedure *Store()* applied and the user registration is completed.

4.3 Login

The login into the system is explained in this section. After registrations, users should login to system in order to achieve processes in the electronic exam. The login system is almost the same with the other systems. Users verify their usernames and passwords. System checks for authentication and the authorized parts are enabled for the users. The protocol is done by the verification of both user's name and password:

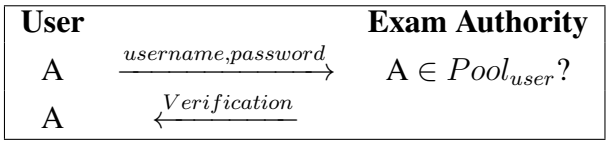
[1] The salt was firstly described in 1979 by [39]. The main idea of using salt is to expand the set of passwords. In [39] a 12-bit random number was generated and added to password as a salt. When the user later logs on the system, 12-bit salt extracted from the password and append to typed one for authentication. This salt mechanism make it harder to prepare a password table for attacks. (Dictionary Attack). Technically, salting mechanism done by a *key derivation function* KDF by applying both *password* P and *salt* S to get a *derived key* DK in [1] such that

Algorithm 2 Salting and Iteration

Input: pwd: A_{pwd}, pwd_{byte}
Output: h: password hash, s: salt

1: pwd_{byte} = ConvertToByte(pwd)
2: iteration:= 1000, r:= random
3: s := r
4: KDF := Pkcs5S2ParametersGenerator()
5: Initialize KDF: KDF := Init(pwd_{byte}, s, iteration)
6: h := GenerateDerivedMacParamaters(KDF,pwd_{byte})
7: **return** h, s

Figure 4.3: Login Protocol



In this protocol, the password of the user is used for the hash which was created in the registration part. If there is no such user then the system refuses for login. If the user registered before, system checks for password authentication. The password authentication process is the same process in the registration part. User's password, hash and salt are used for authentication. In registration part, user's salt are kept in the database. System regenerates the hash with respect to user's salt and checks for password authentication. (Algorithm 3)

Algorithm 3 System Login

Input: username: $A_{username}$, pwd: A_{pwd} , usersalt: A_{salt}
Output: Verification message M

1: pwd_{byte} = ConvertToByte(pwd)
2: iteration:= 1000, r:= random, s:= salt, uh:= userhash
3: s := A_{salt}
4: KDF := Pkcs5S2ParametersGenerator()
5: Initialize KDF: KDF := Init(pwd_{byte}, s, iteration)
6: h := GenerateDerivedMacParamaters(KDF,pwd_{byte})
7: **if** Check(uh,h)=true **then**
8: M:= true
9: **else**
10: M:= false
11: **end if**
12: **return** M

Login process is not enough for doing processes in the system. Users should authenticate themselves to system with their digital signatures. System can produce public-

DK = KDF(P,S)

[2] The purpose of *iteration count* is to increase the difficulty attacks on passwords. By iteration count, the cost of password produced key increased so that the cost of exhaustive search. In [1] at least 1000 iterations are needed.

private key pairs for this process. If any user has some other public-private key pairs, the system is also compatible with them. The next session will explain key management in details.

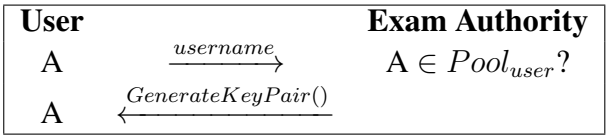
4.4 Key Management System

In this section, public-private key management system is introduced. The main idea in this section is related to the fundamentals of public key cryptosystems. In the system, *2048-bit* RSA [37] key pairs are generated for authentication and digital signature issues.

4.4.1 Key Generation in the System

Every user should have public-private key pairs in our system. These key pairs allow users authenticate to the system and sign question and other issues. If a user has another public-private key pair, the system is compatible with them. The protocol for this part:

Figure 4.4: Key Generation Protocol



Users without any key pairs, generate their public and private key pair with the following algorithm:

Algorithm 4 GenerateKeyPair()

Input: username: $A_{username}$
Output: $Public$ & $Private$ Keys

1: length := 2048
2: KeyAlgorithm := RSA
3: r := RSAKeyPairGenerator()
4: Initialize r: r := Init(random, length)
5: $Public$:= r.PublicKey
6: $Private$:= r.PrivateKey
7: **return** $Public$, $Private$

Here, we store all public keys in the database to use them in authentication issues. The private key of the user is downloaded to user's computer in PEM [3] format.

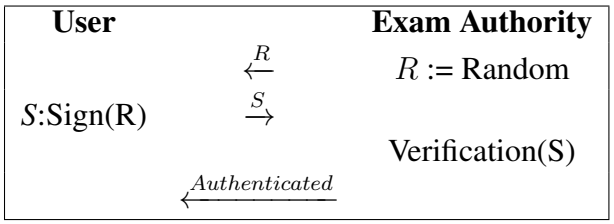
[3] PEM stands for Privacy-enhanced Electronic Mail is a standard format for SSL tools and encoded in Base64. [30]

4.5 Authentication to System

Every user should authenticate the system with their public and private key pairs. In the system we used *RSA Key Authentication Scheme* for authentication whose standards from *NIST* (National Institute of Standards and Technology) by *BouncyCastle.*

For authentication, system generates a random text *R* and sends to the user. User signs the R with his private key and sends the authority. Authority checks the signature and if the verification is done then the user authenticates the system.

Figure 4.5: Authentication Protocol



In the authentication protocol, *Sign(R)* is done in client side developed by desktop application for the protection of private key in internet based program. The algorithm *Sign(R)* is defined as:

Algorithm 5 Signing The Data

Input: Data: D, Private Key: sk_A
Output: Signature: S

1: Read(sk_A)
2: S := GetSignature(D, sk_A)
3: **return** S

The verification is done in the server side by the authority. The algorithm *Verification(S)* is defined as:

Algorithm 6 Verification Of The Signature

Input: Random Text: R, Signature: S, Public Key: pk_A
Output: Message: M

1: signer := SignerUtilities.GetSigner(SHA384WithRSAEncryption)
2: Initialize signer: signer := Init(pk_A)
3: BlockUpdate signer: signer := BlockUpdate(R)
4: Verify the signature: signer $\leftrightarrow S$
5: **if** Verify the signature = true **then**
6: M:= true
7: **else**
8: M:= false
9: **end if**
10: **return** M

We used $SHA-384$ [4] as signature algorithm. The system is also compatible with other signature algorithms. We applied other algorithms (RIPEMD-160[5], SHA, MD-5 [6], MD-4, MD-2) in Bouncy Castle library and get some results. The results are about the generation of the keys, signing and verifying the messages about 100 times in an Intel(R) Core(TM) i7-2600 CPU 3.40 GHz 8.00GB RAM computer.

Table 4.1: 192 bit Elliptic Curve Key Pairs

Algorithm	Key Generation (ms)	Sign Message (ms)	Verify Message(ms)
RIPEMD-160	38,79	38,43	56,46
SHA-1	38,49	38,05	56,08
SHA-224	38,38	37,95	55,97
SHA-256	39,09	38,69	56,80
SHA-384	38,49	38,12	56,14
SHA-512	38,57	38,02	56,18

Table 4.2: 239 bit Elliptic Curve Key Pairs

Algorithm	Key Generation (ms)	Sign Message (ms)	Verify Message(ms)
RIPEMD-160	63,92	63,67	93,62
SHA-1	63,66	62,80	92,47
SHA-224	63,01	62,60	91,52
SHA-256	63,83	63,28	93,01
SHA-384	63,75	63,11	93,14
SHA-512	62,83	62,61	91,78

Table 4.3: 256 bit Elliptic Curve Key Pairs

Algorithm	Key Generation (ms)	Sign Message (ms)	Verify Message(ms)
RIPEMD-160	74,17	73,90	108,71
SHA-1	74,76	74,54	108,96
SHA-224	74,26	73,96	108,74
SHA-256	74,06	73,31	107,86
SHA-384	74,95	74,44	109,35
SHA-512	74,43	73,71	108,55

According to the results in Table 4.1, Table 4.2,Table 4.3,Table 4.4 and Table 4.5 as the key size getting larger, the durations are getting larger. Elliptic Curves are faster than RSA with respect to key generation but RSA is faster than Elliptic Curves with respect to signing message and verifying the message. In our model, we use signing and verifying message more than generation so that we use RSA algorithm in our key management issues.

[4] SHA stands for Secure Hash Algorithm published by NIST (National Institute of Standards and Technology) [35]

[5] RIPEMD-160 stands for RACE Integrity Primitives Evaluation Message Digest is a 160-bit hash function developed by Hans Dobbertin, Antoon Bosselaers, and Bart Preneel in 1996. It basically designed in the principles of MD4. [22]

[6] MD-5 is a message digest algorithm takes as input a message of arbitrary length and produces as output a 128-bit developed by Ron Rivest in 1992. [36]

Table 4.4: 1024 bit RSA Key Pairs

Algorithm	Key Generation (ms)	Sign Message (ms)	Verify Message(ms)
MD-2	278,72	8,25	<1
MD-4	317,47	8,20	<1
MD-5	312,39	8,23	<1
RIPEMD-128	279,97	8,26	<1
RIPEMD-160	269,31	8,23	<1
RIPEMD-256	299,96	8,31	<1
SHA-1	266,12	8,17	<1
SHA-224	315,40	8,28	<1
SHA-256	316,61	8,20	<1
SHA-384	291,77	8,21	<1
SHA-512	305,62	8,20	<1

Table 4.5: 2048 bit RSA Key Pairs

Algorithm	Key Generation (ms)	Sign Message (ms)	Verify Message(ms)
MD-2	4620,36	52,89	2,01
MD-4	3899,69	52,88	1,95
MD-5	4249,55	52,64	1,93
RIPEMD-128	4034,55	53,02	1,99
RIPEMD-160	4044,38	53,15	2,01
RIPEMD-256	3849,54	52,85	1,98
SHA-1	4090,01	53,10	1,97
SHA-224	4079,16	54,32	2,01
SHA-256	3635,12	53,60	1,98
SHA-384	3754,43	52,87	1,96
SHA-512	3390,91	52,48	1,95

4.6 Question Entry

In this section we explain how questions are send to system by the question makers. The questions are the sensitive part of the electronic exam. The security of the electronic exam relies on the security of the question. Qualified users who have question entry role can supply questions to the system.

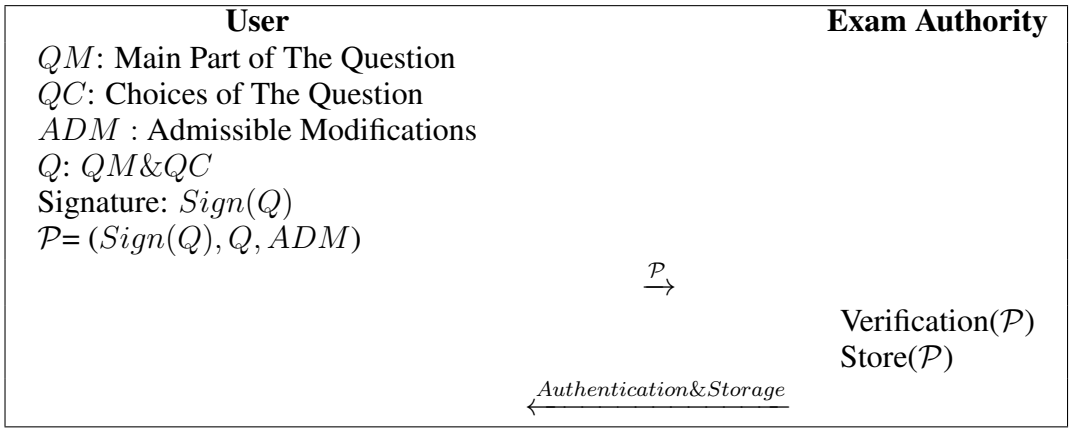
Question entry is only done by the users with the question entry role. Users can write questions by using question entry module (Fig. B.7). In our model, a question should have main part, choices and tags. The tags of the questions are the references about the questions. No one can get the questions from the tags. The tags of any question are given below.

- Subject,
- Category,
- Subcategory,

- Hardness.

These can be expanded if needed. Once a question is written, users should determine the tags. After the completion of writing questions, question maker downloads the question into his machine and sign it with his private key. The signing issue is the same process in Section (4.5). The signed question is send to authority and authority verifies and stores it. For sanitizable signature, question maker should determine the admissible modifications (ADM). Basically, for electronic exam ADMs are the choices of the questions. The protocol for signing a question as follows:

Figure 4.6: Question Signing Protocol



In the above protocol, the $Sign(Q)$ and $Verification$ functions are the same in section 4.5. In the storage part AES is used for encryption. The algorithm of $Store$ as follows:

Algorithm 7 Storage Of Package

Input: Package: $\mathcal{P}$
Output: Message: M

1: QM:Question Part, QC: Choices, R:Right Answer, h: Hash of Question, k: Key, ID_A: Identity of the user A, ADM: Admissible Modifications
2: Encryption(QM) := AES.SimpleEncrypt(QM,k)
3: Insert(Encryption(QM), C, R, h, ID_A),ADM)
4: **if** Insertion **then**
5: M:= successful
6: **else**
7: M:= not successful
8: **end if**
9: **return** M

The main part of the question is encrypted. The other parts are no need to be encrypted since someone cannot get the question from the tags. The user ID and the hash of the questions are also stored in the database with time stamp. In the case of any accusation, the hash and the time stamp can be used for non-repudiation of the question.

4.7 Supervising of Question

In this section we explain the supervising of the questions in the pool. For a reliable examination, the questions should be proper. Sometimes question makers make some errors on questions or misspell some parts of the questions. Due to protect that kind of irregularities on the questions, all the questions should be supervised by the supervisors supplied by the authority. Here we assume that, question supervising is done at the centers supplied by the authority.

The encrypted form of the questions come to question supervisor's screen. The encryption is done with AES but in the model we split into AES key into n pieces with Shamir's Secret Sharing Scheme [42]. Each pieces of the key are given to the members of a committee by the authority. At least k members come together in order to decrypt the questions. We prefer to use (n,k)-threshold scheme as (5,3)-threshold scheme which can be extended if needed.

After the decryption, question supervisor can edit the question or approve it. In our model, a question should be approved by the t different question supervisors. If a question is edited any time, it should be waited to be supervised t times by other supervisors in order to be used in the exam. Once the question supervisor edits a question the original signature of question is stored in the database. In the proposed model we prefer t as 3 and this can be extended if needed. The new signature of the question and time stamp are replaced with the previous one. The protocol for creating new signature and time stamp is the same with the protocol in Section 4.6. If the supervisor only approves the question then there is no need to sign it again.

The protocol for question entry as follows:

Figure 4.7: Supervising of Question

Question Supervisor		**Exam Authority**
		QM: Main Part Of The Question
		QC: Choices Of The Question
		Q: $QM\&QC$
		Shamir's Secret Sharing: SSS
		$\mathcal{P}$:Package := $SSS(Decryption(Q))$
	$\xleftarrow{\mathcal{P}}$	
Edit($\mathcal{P}$)		
	$\xrightarrow{\mathcal{P},Sign(Q)}$	
		Verify($Sign(Q)$)
		Store($SSS(Encryption(Q))$

The $Store()$ algorithm is the same with the algorithm in Section 4.6. We use (n, k)-threshold scheme in SSS with Lagrange Interpolation [7] to split and get the AES key.

[7] Lagrange polynomial $P(x)$ of degree $\leq$ (n-1) which passes through n points $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$ $P(x) = \sum_{j=1}^{n} P_j(x)$ where $P_j(x) = y_j \prod_{k=1, k\neq j}^{n} \frac{x-x_k}{x_j-x_k}$ published by Joseph Louis Lagrange in 1795. [44]

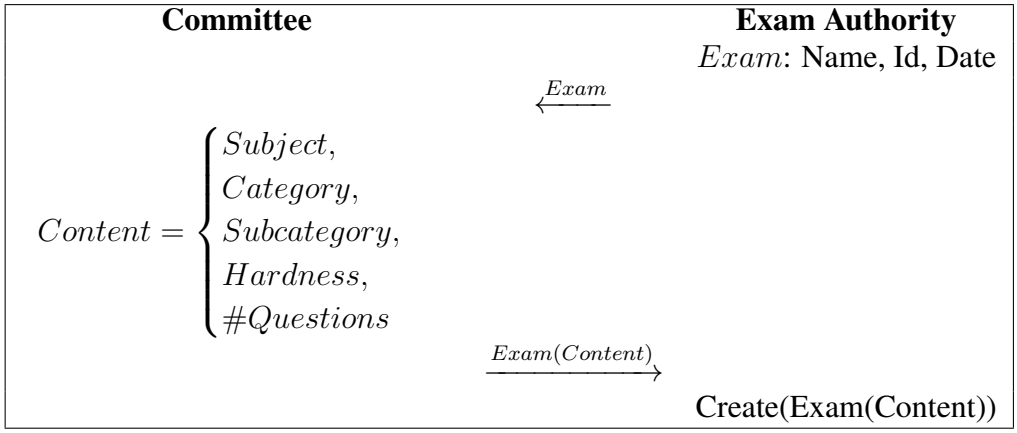
4.8 Exam Preparation

In this section we explain how an electronic exam is prepared by the authority. For any exam, specified number of questions are needed. This can be done directly chosen by a committee or automatically with the tags of questions. In our model, we use the tags of questions for preparing examinations.

The authority determines the name, ID and the date for the exam. For questions, a committee determines the subjects, category, subcategory and hardness of the questions. With these references, the system selects the questions from the database. The hash and time stamps of all the questions are created due to non-repudiation of the exam. Here, we assume that there is no duplication of the questions in the pool.

The exam preparation protocol as follows:

Figure 4.8: Exam Preparation Protocol



In the above protocol the function $Create(Exam)$ works as follows:

Algorithm 8 Exam Creation

Input: Exam Content = $\{Subject, Category, Subcategory, Hardness, \#Questions\}$
Output: Exam

1: **while** $\#Question > 0$ **do**
2: select SecureRandom($Question_{ID}$) from Database where
3: $\{Subject, Category, Subcategory, Hardness\} \in \{ExamContent\}$
4: set $\#Question := \#Question - 1$
5: insert $Question_{ID}$ to $Exam$
6: insert $Hash(Question_{ID})$ to $Exam$
7: insert $AnswerKey(Question_{ID})$ to $Exam$
8: **end while**

In the above algorithm, the questions and their related hashes and answer keys are selected randomly from the database with respect to exam content. We use $SecureRandom$[8]

[8] BouncyCastle uses SP 800-90A and X9.31 standards for generating random numbers. [13]

function in BouncyCastle library to produce random number so that all the questions are selected randomly from the database.

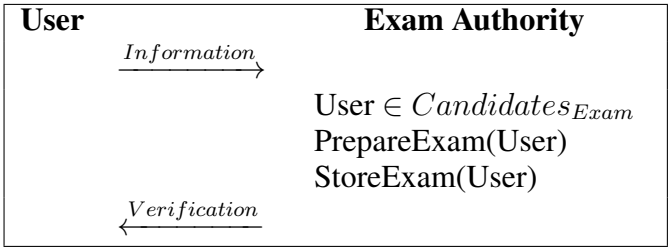
4.9 Candidate Registration

In this section candidate registration to the exam is explained. Any user has to register examination in order to enroll that exam. In the registration part, some of the information can be needed. In our model, we do not want any extra information excluded from the user registration part. But some information related to examination can be needed.

The authority first checks whether the user registered before or not. For any user who registered to examination, related exam materials will be prepared. The authority selects the question IDs from the prepared exam and makes it ready for the user.

The protocol for the candidate registration as follows:

Figure 4.9: Candidate Registration Protocol



In the above protocol, exam materials (questions, hashes, etc.) for the user are prepared by the functions $PrepareExam()$ and $StoreExam()$.

4.10 Exam and Results

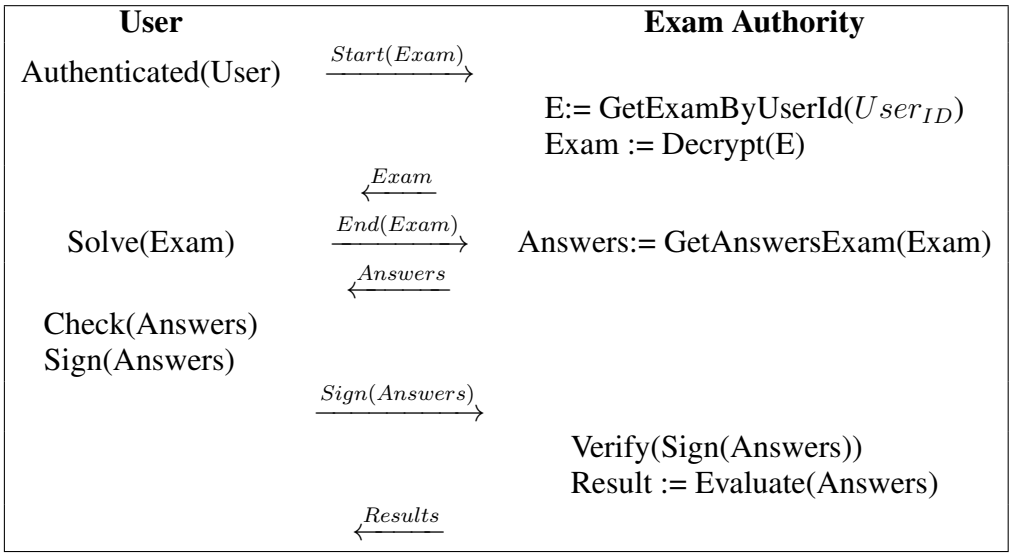
In this section, we explain how an exam holds and is evaluated in an electronic examination. For any examination, exam materials, candidates and authority come together. In our model, the exam holds in the centers supplied by the authority with the proctors. We assume that all the computers are supplied by the authority and only connect with the authority.

In the exam day, candidates and proctors come the specified place in scheduled time. They should authenticate system (Section4.5) in order to enroll examination. Each candidates' exam questions come to their specified computer in encrypted form. Exam can be started by the proctors or manually by the candidates. When the exam starts, the questions decrypted and can only be seen by the candidates. During the exam, candidates select the answers and these answers are stored to database by the authority. The exam can be ended by the candidate at any time. When the exam ends, candidates are not able to change their answers and the answers are shown the candidates in order

to sign and evaluation. Candidates check their answers and download them into their computer. They sign the answers in another program which not web-based program. (Section4.5) The signature of the answers are checked by the authority and evaluated if the verification is done. The evaluation of the exam is done by the matching of the answers of the candidate and his related answer key. We assume that all these processes are done in a secure line [9] between candidates computer and authority.

The protocol for exam and results as follows:

Figure 4.10: Exam And Results Protocol



In the above protocol, authority selects and prepares exam for any user by the function $GetExamByUserId$. For any exam $GetAnswersExam$ functions gives the answers of the selected exam.

Algorithm 9 GetExamByUserId

Input: ID := User ID, eID = Exam ID
Output: Exam

1: select Exam from Database where $ExamID = eID$
2: Exam := Shuffle(Exam,ID)

In Algorithm 9 the procedure $Shuffle$ just shuffles the order of questions for the user. The function $Evaluate$ evaluates the results of the user with respect to his answers.

4.11 Archive

In this section, we explain how all the things are archived in the system. All the processes in the electronic examination should be stored in order to get the history of the

[9] SSL will be used.

exam in the future. The archiving system simply based on the storage of the data in the system. All the processes in an exam; registered users, selected questions and related answers, answer key for the exam, users' selected questions and their answers,the hashes and time stamps are stored in the database.

In our model we use the following functions for archiving:

Algorithm 10 Archiving of Registered User for Exam

Input: User ID, Username, Name, Surname, Exam ID, Exam Name
Output: User ID, Username, Name, Surname, Exam ID, Exam Name, Hash, Time Stamp

1: ToBeHash := Concatenate(User ID Username, Name, Surname, Exam ID, Exam Name)
2: Hash := ToBeHash
3: Insert(User ID, Username, Name, Surname, Exam ID, Exam Name, Hash, Time Stamp)

Algorithm 11 Archiving of Selected Questions and Their Answers

Input: Question Order, Exam ID, Exam Name, Exam Date, Question ID, Question, Choices
Output: Question Order, Exam ID, Exam Name, Exam Date, Question ID, Question, Choices, Hash, Time Stamp

1: ToBeHash := Concatenate(Question Order, Exam ID, Exam Name, Exam Date, Question ID, Question, Choices)
2: Hash := ToBeHash
3: Insert(Question Order, Exam ID, Exam Name, Exam Date, Question ID, Question, Choices, Hash, Time Stamp)

Algorithm 12 Archiving of Answer Key

Input: Exam ID, Question Order, Question ID, Right Answer
Output: Exam ID, Question Order, Question ID, Right Answer, Hash, Time Stamp

1: ToBeHash := Concatenate(Exam ID, Question Order, Question ID, Right Answer)
2: Hash := ToBeHash
3: Insert(Exam ID, Question Order, Question ID, Right Answer, Hash, Time Stamp)

Algorithm 13 Archiving of User Selected Questions and Their Answers

Input: User ID, User Question Order, Question Order, Exam ID, Exam Name, Exam Date, Question ID, Question, Choices
Output: User ID, User Question Order, Question Order, Exam ID, Exam Name, Exam Date, Question ID, Question, Choices, Hash, Time Stamp

1: ToBeHash := Concatenate(User ID, User Question Order, Question Order, Exam ID, Exam Name, Exam Date, Question ID, Question, Choices)
2: Hash := ToBeHash
3: Insert(User ID, User Question Order, Question Order, Exam ID, Exam Name, Exam Date, Question ID, Question, Choices, Hash, Time Stamp)

With the algorithms (10, 11, 12, 13) all the processes in the examination are stored in the database. In any accusation, the authority should use hashes and time stamps in order to solve.

To summarize our system, there are four roles: authority, question makers, question supervisors, candidates, and proctors. All these roles are defined by registration protocol. The users login the system with their passwords. Authentication is done by authentication protocol with public and private key pairs. If a user has question maker role, the user can provide questions to system and sign them with his public private key pairs. All the questions in the system are supervised by the users with question supervisor role. The exam is prepared by the authority by using the tags of the questions. Users with candidate role take exam in a supervised environment with proctors. Exam is evaluated by the authority and the results are sent to candidates. All the processes in the model are archived by the authority. (Fig 4.11)

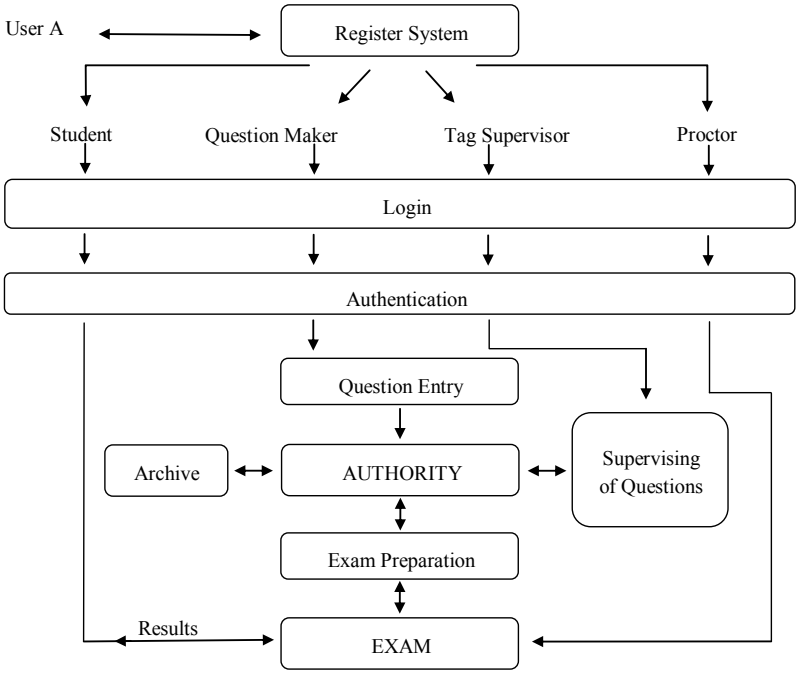


Figure 4.11: Electronic Exam Model.

4.12 Security Analysis

In this section, we provide security analysis of the model in order to show that our model is secure. We assume that the line between users and authority is secure and every user in the system does not share their private keys. In this section, we emphasis

on the robustness of the system with respect to authentication, privacy of the sensitive data and anonymity of the users.

Theorem 4.1. *The electronic exam system provides authenticity in order to achieve progresses in the stages.*

Proof. After *login* a user uses his private key to sign the randomize *R* came from authority in Section 4.5. The authority checks $Sign(R) = R^{sk_{user}}$ by using user's public key. Since the authority knows both pk_{user} and R then computes $R^{sk_{user} \cdot pk_{user}} \stackrel{?}{=} R$. If the equation holds, then the user authenticates the system and eligible for the functions in the system. □

Theorem 4.2. *The electronic exam system provides secrecy on sensitive data of the exam.*

Proof. The sensitive data for the electronic exam is the questions and their related answers. The questions are supplied by the question makers. Question maker prepares his question and sign the question with his private key. The authority verifies the signature of the question by using the public key of the question maker. If the signature of the question is verified, the authority stores the question by using AES with the key k. In the model $(5, 3) - threshold$ scheme used for splitting the key k. Since any question in the database is encrypted with k, nobody else get the decrypted form of question without knowing the key. □

Theorem 4.3. *The anonymity of the question makers are provided.*

Proof. In Section 4.7, all questions in the pool are supervised by the user determined by the authority. For any question, question supervisor edits the question if needed. For any question, let $ID_{question}$ be the ID number of the question. For any question supervisor, $ID_{question}$ is visible, so that the anonymity of the question makers is provided. □

CHAPTER 5

CONCLUSION

The development of technology affects every stage of our life especially the education. Universities, colleges and other institutes have started to hold their lessons in electronic environment. Furthermore, these companies have started to hold their examination in electronic environment. Electronic exam is one the most challenging topic in this area. In this thesis, we investigated a secure electronic exam scheme which satisfies authenticity, anonymity, robustness and secrecy of the sensitive data. First, we gave information about electronic learning, electronic exam and cryptographic requirements used throughout the thesis. Then we proposed a model which has protocols supplying long term confidentiality of the sensitive data and data sharing of users and authority with digital signatures. Then with this proposed model, we made an electronic exam application satisfying authenticity, anonymity, robustness and secrecy of the sensitive data.

In electronic examination, the sensitive data is the questions and their relating answers. All the users in any electronic exam have to authenticate themselves to the system. Exam materials should be kept in the secret and only students who take the exam should be able to decrypt them. Students should not be able to change their scores after examination and all relating data for the examination must be kept in secret and only authorised users should be able to access them. Regarding to these points in Section 3, we proposed a new model covering long term confidentiality on sensitive data, using ID-Based cryptosystems for authentication, sanitizable signature scheme for editing the data if needed and asymmetric key solutions for secrecy. The proposed model not only applied on electronic exam but also can be applied on other system like banking systems, health-care systems and so on. To abide by the proposed model we developed an electronic exam application which satisfies the anonymity and authenticity of the users, immutability of the scores and other sensitive data.

To construct a secure electronic exam, we proposed a model in this thesis. As a future work, one can apply the proposed model for other needs like banking system, personal health record systems and so on. The other work should be the security analysis of electronic exam in mobile devices without trusted third party schemes.

REFERENCES

[1] *PKCS #5 v2.1: Password-Based Cryptography Standard*, RSA Laboratories, 2006.

[2] *Motorlu Taşıt Sürücü Adayları Sınavı e-Sınav Uygulama Kılavuzu (Motor Vehicle Driver Candidate Examination e-Exam Application Guide)*, Turkish Republic Ministry of National Education, 2012.

[3] The Legion of the Bouncy Castle, 2015, `http://www.bouncycastle.org/`.

[4] About the Toefl Pbt Test, 2016, `https://www.ets.org/toefl/pbt/about`.

[5] Gmac Statistics, 2016, `http://www.gmac.com/why-gmac/gmac-interactive-map/gmac-statistics-video.aspx`.

[6] Gmat Exam Security, 2016, `http://www.gmac.com/gmat/the-gmat-advantage/gmat-exam-security.aspx`.

[7] How ETS Protects the Integrity of the Toefl Test, 2016, `https://www.ets.org/toefl/institutions/about/security`.

[8] Measuring, Selection and Placement Center, 2016, `http://www.osym.gov.tr/`.

[9] Ministry of National Education, 2016, `http://www.meb.gov.tr/`.

[10] I. E. Allen and J. Seaman, *Changing Course: Ten Years of Tracking Online Education in the United States.*, ERIC, 2013.

[11] G. Ateniese, D. H. Chou, B. de Medeiros, and G. Tsudik, Sanitizable Signatures, pp. 159–177, 2005.

[12] C. Ayo, A. A. Ayodele, I. Akinyemi, and U. Ekong, The Prospects of E-examination Implementation in Nigeria, Turkish Online Journal of Distance Education, 22, pp. 125–134, 2007, ISSN 1302-6488.

[13] E. B. Barker and J. M. Kelsey, *Recommendation For Random Number Generation Using Deterministic Random Bit Generators (revised)*, US Department of Commerce, Technology Administration, National Institute of Standards and Technology, Computer Security Division, Information Technology Laboratory, 2007.

[14] W. Bieniecki, J. Stańdo, and S. Stoliński, Information Technologies in a Process of Examination in Poland, Inf. Syst. Manag. VII, p. 29, 2010.

[15] D. Boneh and M. Franklin, Identity-Based Encryption from the Weil Pairing, SIAM J. of Computing, 32(3), pp. 586–615, 2003.

[16] J. Braun, J. Buchmann, C. Mullan, and A. Wiesmaier, Long Term Confidentiality: a Survey, Designs, Codes and Cryptography, 71(3), pp. 459–478, June 2014, ISSN 0925-1022.

[17] C. Brzuska, M. Fischlin, T. Freudenreich, A. Lehmann, M. Page, J. Schelbert, D. Schröder, and F. Volk, Security of Sanitizable Signatures Revisited, pp. 317–336, 2009.

[18] J. Castella-Roca, J. Herrera-Joancomarti, and A. Dorca-Josa, A Secure E-exam Management System, pp. 8 pp.–, April 2006.

[19] J. C. Choon and J. H. Cheon, An Identity-based Signature From Gap Diffie-Hellman Groups.

[20] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch, Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults, pp. 383–395, 1985.

[21] J. Daemen and V. Rijmen, *The Design of Rijndael*, Springer-Verlag New York, Inc., 2002.

[22] H. Dobbertin, A. Bosselaers, and B. Preneel, RIPEMD-160: A Strengthened Version of Ripemd, 1039, pp. 71–82, 1996.

[23] P. Feldman, *A Practical Scheme for Non-interactive Verifiable Secret Sharing*, SFCS '87, IEEE Computer Society, Washington, DC, USA, 1987.

[24] C. Gentry, Fully Homomorphic Encryption Using Ideal Lattices, pp. 169–178, 2009.

[25] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung, *Proactive Secret Sharing Or: How to Cope With Perpetual Leakage*, 1995.

[26] F. Hess, Efficient Identity Based Signature Schemes Based on Pairings, pp. 310–324, 2002.

[27] A. Huszti and A. Petho, A Secure Electronic Exam System, Publicationes Mathematicae Debrecen, 77(3), pp. 299–312, 2010.

[28] M. Kim and K. Kim, A new identification scheme based on the bilinear diffie-hellman problem., 2384, pp. 362–378, 2002.

[29] H. Koyuncu, K. Kaya, and A. A. Selcuk, An Analysis of the Generalized ID-Based ElGamal Signatures, Proceedings of ISCTURKEY 2008, pp. 106–110, 2008.

[30] J. Linn, C. Barker, J. Bidzos, M. Bishop, D. Cohen, C. Fox, M. Gasser, R. Housley, S. Kent, J. Laws, S. Lipner, D. Nessett, M. Padlipsky, M. Smid, S. Walker, and S. Wilbur, Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures", RFC 1421, 1993.

[31] L. T. Ölçüoğlu and S. Akleylek, Data Storage of Electronic Exams, Proceedings of ISCTURKEY 2015, 2015.

[32] P. Nicholson, *A History of E-Learning*, Springer Netherlands, 2007.

[33] R. Ostrovsky and M. Yung, *How to Withstand Mobile Virus Attacks (Extended Abstract)*, PODC '91, ACM, New York, NY, USA, 1991.

[34] K. G. Paterson, ID-based Signatures from Pairings on Elliptic Curves, 2002.

[35] PUB-FIPS, Secure Hash Standard (SHS), 2012.

[36] R. Rivest, The MD5 Message-Digest Algorithm, 1992.

[37] R. Rivest, A. Shamir, and L. Adleman, A Method for Obtaining Digital Signature and Public-Key Cryptosystems, Communication of the ACM, 21(2), pp. 120–126, 1978.

[38] R. L. Rivest, L. Adleman, and M. L. Dertouzos, On Data Banks and Privacy Homomorphisms, Foundations of secure computation, pp. 169–177, 1978.

[39] M. Robert and T. Ken, Password Securtiy: A Case History, Communication of the ACM, 22(11), pp. 594–597, 1979.

[40] R. Sakai and M. Kasahara, ID based Cryptosystems with Pairing on Elliptic Curve, IACR Cryptology ePrint Archive, pp. 54–54, 2003.

[41] F. B. Scheider, Something You Know, Have, or Are, 2005, `https://www.cs.cornell.edu/courses/cs513/2005fa/nnlauthpeople.html`.

[42] A. Shamir, How to Share a Secret, Commun. ACM, 22(11), pp. 612–613, November 1979, ISSN 0001-0782.

[43] A. Shamir, Identity-Based Cryptosystems and Signature Schemes, 1984.

[44] E. W. Weisstein, Lagrange Interpolating Polynomial, 2004.

APPENDIX A

Definition of Functions and Procedures

Pkcs5S2ParametersGenerator(): Generates key derivation function for salting with respect to PKCS#5 parameters.
GenerateDerivedMacParamaters(): Generates MAC (Message Authentication Code) derived from the password.
ConvertToByte(string s): Converts the input *s* into the bytes.
RSAKeyPairGenerator(): Generates null public and private keys using RSA algorithm.
SignerUtilities.GetSigner(Signature Algorithm): Creates an object of *Signature Algorithm*
AES.SimpleEncrypt(Q,k): Encrypt the object Q with AES algorithm with the key k.
AnswerKey($Question_{ID}$): Selects the answer key for the question with ID.
PrepareExam(User:) Exam questions and their related hashes are selected for the user.
StoreExam(User): Exam materials for the user are stored in the database.
GetExamByUserId($User_{ID}$): Prepares exam materials for the user with ID = $User_{ID}$
GetAnswersExam(Exam): Selects the answers for the specified exam.
Evaluate(Answers): Evaluate the answers with the exam key.

APPENDIX B

Sample Screen Shots of Secure Electronic Exam Application

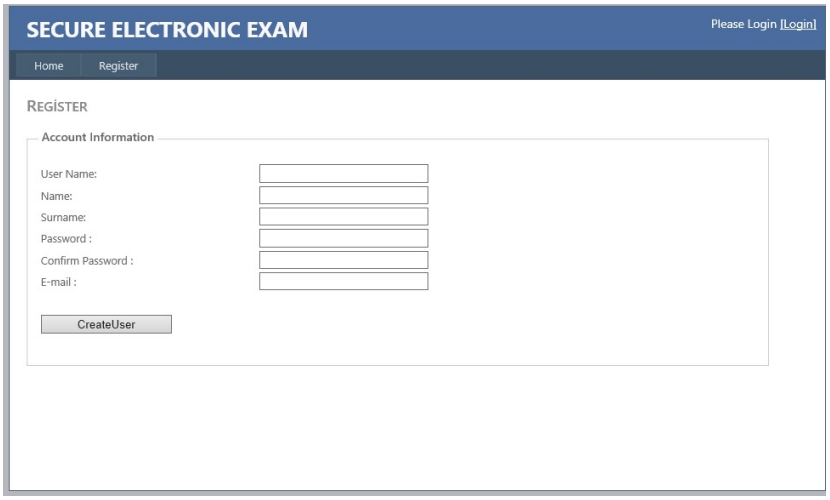


Figure B.1: Register Screen

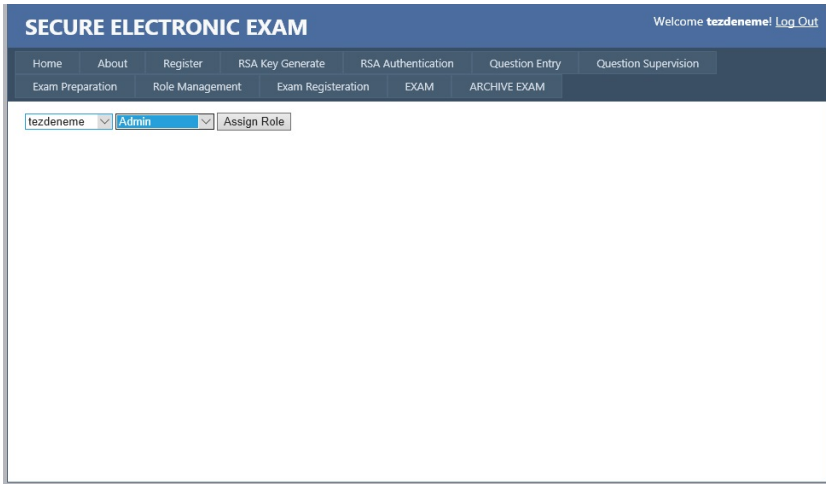


Figure B.2: Role Management Screen

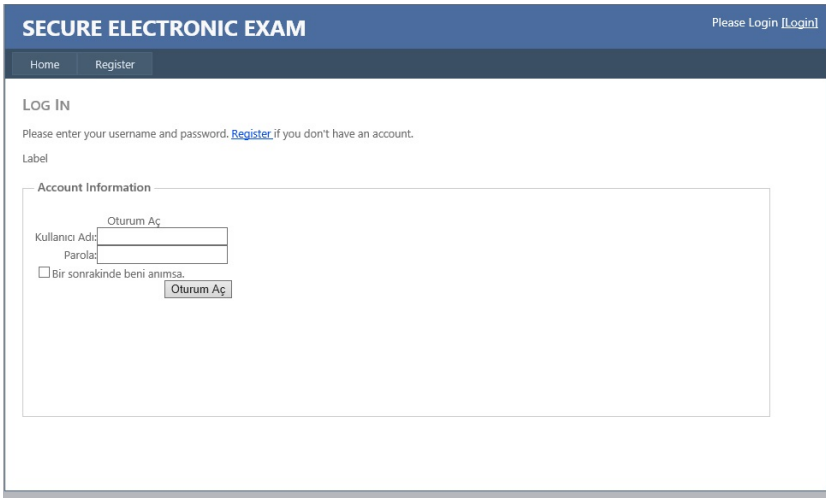


Figure B.3: Login Screen

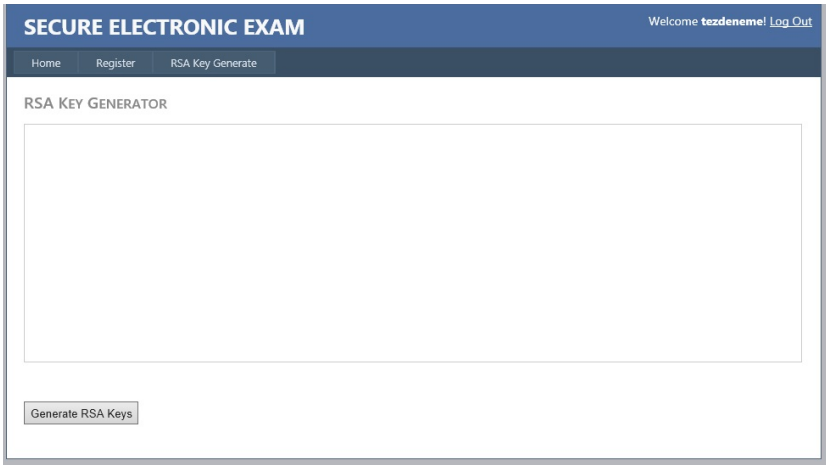


Figure B.4: Key Generation Screen

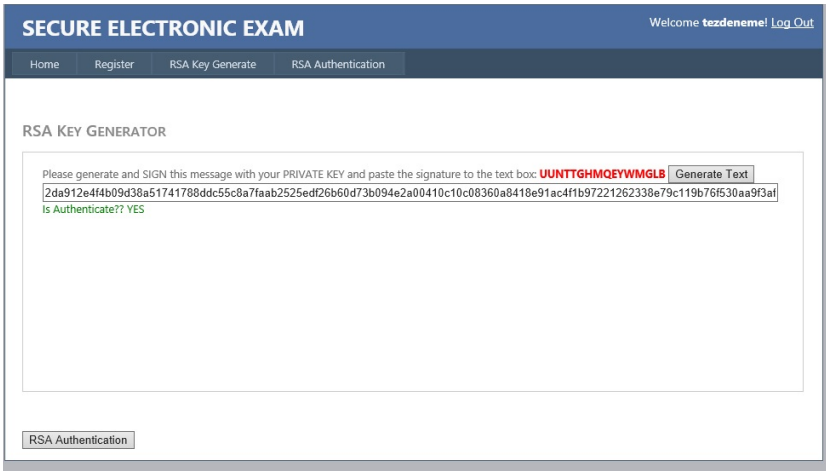


Figure B.5: Authentication Screen

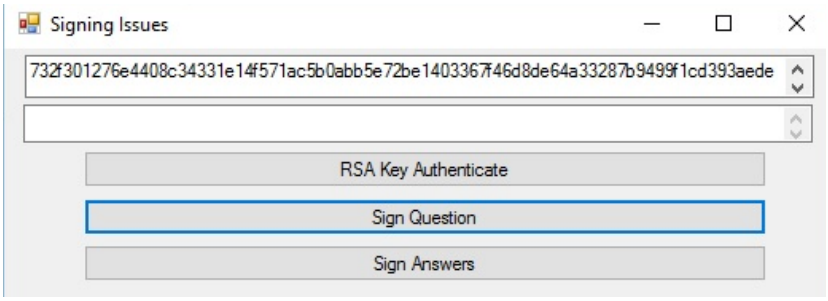


Figure B.6: Signing Question Screen

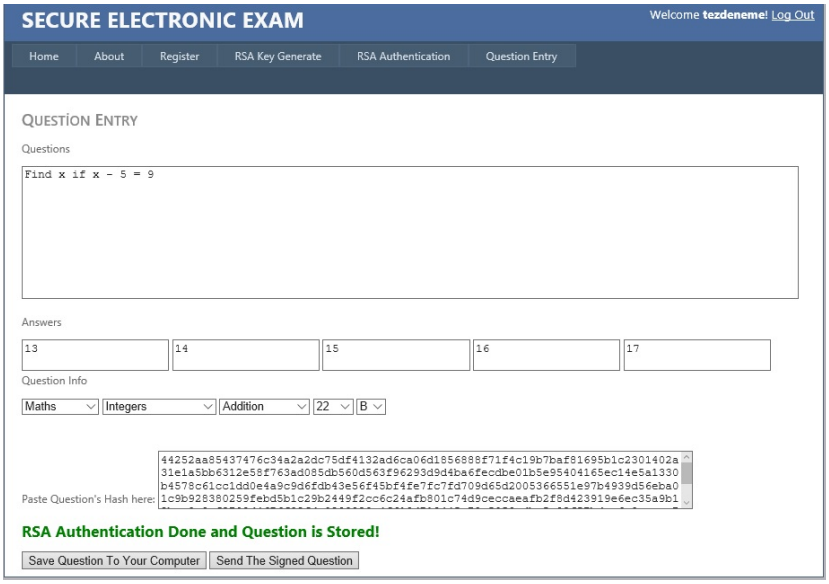


Figure B.7: Question Entry Screen

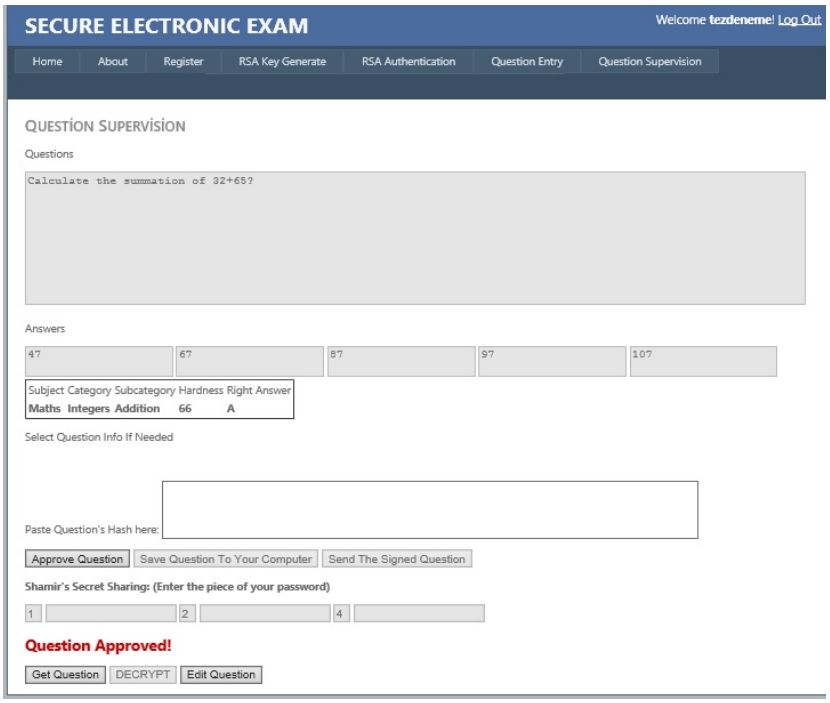


Figure B.8: Supervising of Question Screen

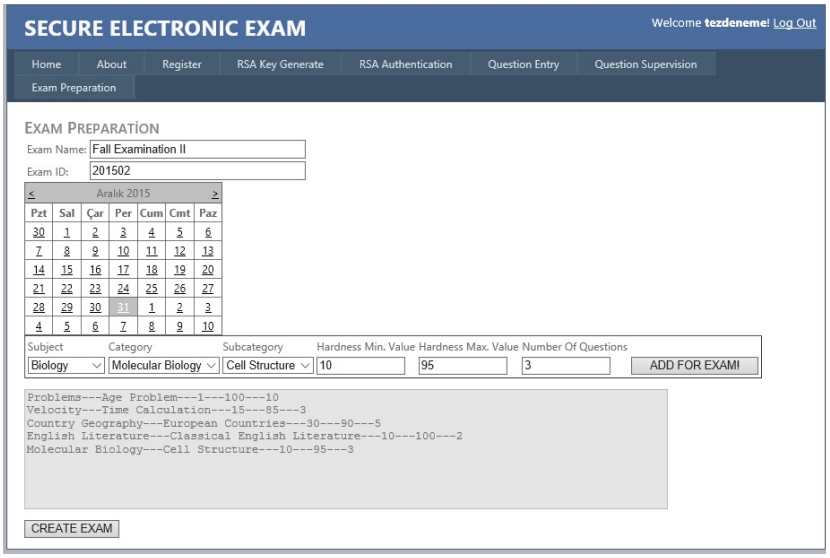


Figure B.9: Exam Preparation Screen

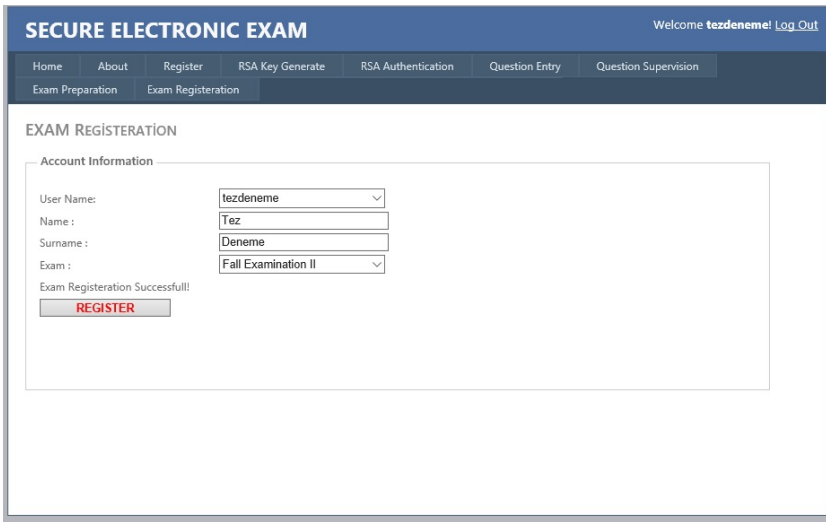


Figure B.10: Exam Registration Screen

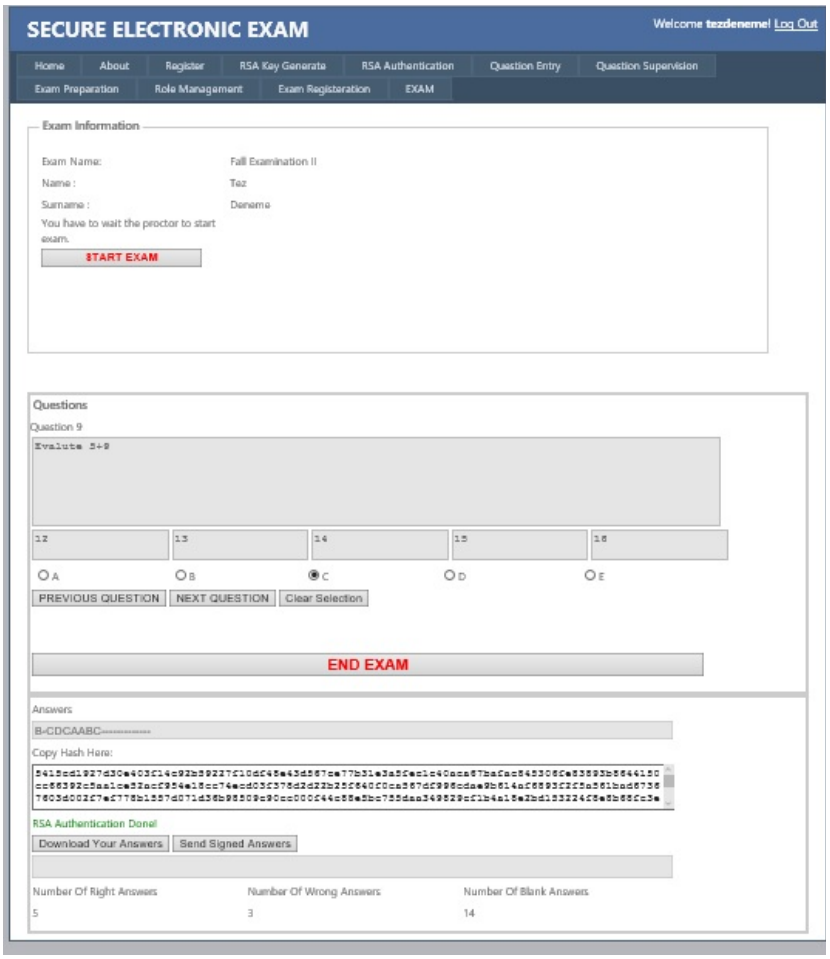


Figure B.11: Exam and Results Screen

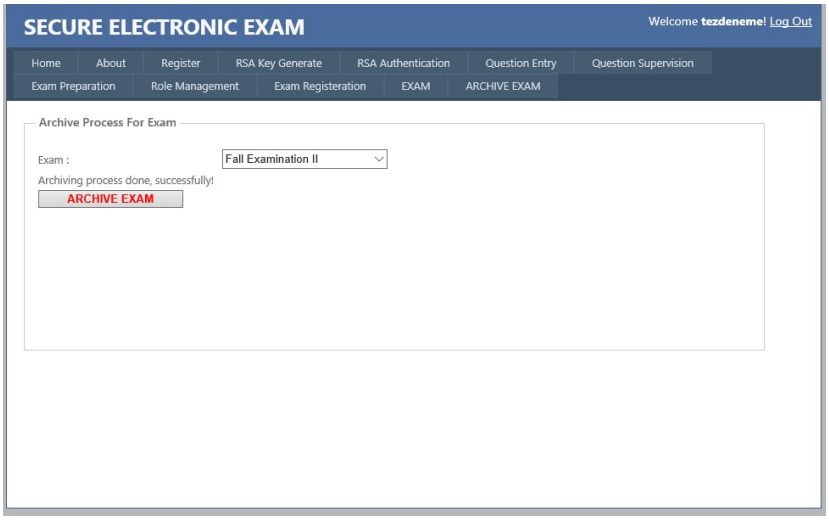


Figure B.12: Archiving Screen

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Ölçüoğlu, Lütfü Tarkan
Nationality: Turkish
Date and Place of Birth: Sep 24th 1980, Ankara
Marital Status: Married

EDUCATION

Degree	Institution	Year of Graduation
M.S.	Department of Cryptography, METU	2009
B.S.	Department of Mathematics, Hacettepe University	2003
High School	İncirli High School	1998

PROFESSIONAL EXPERIENCE

Year	Place	Enrollment
2005- Present	ÖSYM	ÖSYM Specialist

PUBLICATIONS

International Conference Publications

L.T. Ölçüoğlu and S.Akleylek, "Data Storage of Electronic Exams" Proceedings of ISCTURKEY2015, 2015.