A FRAMEWORK FOR AUDITABLE, PAPER-EQUIVALENT ELECTRONIC FORMS USING DATA LOGGING, SECURE ACCESS CONTROLS, AND ELECTRONIC CERTIFICATES

By

René D. Badía-Reyes

A thesis submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE
in
COMPUTER ENGINEERING

University of Puerto Rico
Mayagüez Campus
2006

Approved by:

Manuel Rodríguez Martínez, Ph.D. Member, Graduate Committee	Date
Pedro Rivera Vega, Ph.D. Member, Graduate Committee	Date
Bienvenido Vélez, Ph.D. President, Graduate Committee	Date
Isidoro Couvertier, Ph.D. Chairperson of the Department	Date

ABSTRACT

The Paper Equivalent Forms framework provides equivalent or superior authenticity, non-repudiation, access control, and integrity than that of paper-based records. Data input by the user and authorship information is logged internally and also verified by a third-party by providing trusted digital "approvals" certifying that a particular change has been made. These measures provide the means to verify the authenticity of the information stored within the forms and whether the data has been tampered with or not. A series of analysis on the system's security, performance, and load capabilities were performed. This research attempts to prove that the benefits obtained by the additional security and reliability measures outweigh the impact they have on the system's performance. Furthermore, time-consuming areas in the communication protocol were identified and an improved version of the framework was developed, in which the performance of the system is greatly improved.

RESUMEN

El sistema de Paper Equivalent Forms provee autenticidad, no-renegación, controles de acceso, e integridad equivalentes o superiores al de los récords almacenados en papel. Los datos sometidos por el usuario e informacion de autoría asociada, son registrados internamente y verificados por una tercera entidad que provee pruebas digitales confiables certificando que un cambio en particular fue realizado. Estos medios nos permiten verificar la autenticidad de la informacion almacenada en los formularios y si los datos han sido alterados. Se realizó un análisis de seguridad, rendimiento y capacidades de carga del sistema. Esta investigación intenta probar que los beneficios obtenidos por las medidas de seguridad y confiabilidad compensan por el impacto que las mismas tienen en el rendimiento del sistema. Las areas del protocolo de comunicacion que consumían más tiempo fueron identificadas y una versión mejorada del sistema fue desarrollada, en la cual el rendimiento del sistema es mejorado grandemente.

Dedication...

ACKNOWLEDGMENTS

Acknowledgments...

TABLE OF CONTENTS

ABSTRACT ii

RESUMEN iii

ACKNOWLEDGMENTS vi

TABLE OF CONTENTS vii

LIST OF TABLES ix

LIST OF FIGURES x

LIST OF SYMBOLS AND ABBREVIATIONS xi

1 Introduction 1

1.1 Motivation . . . 1

1.2 Purpose . . . 1

2 Theoretical Background 3

2.1 Auditing . . . 3

2.2 Part 11 . . . 3

2.3 Cryptography and Certificates . . . 5

2.4 Biometrics . . . 6

2.5 Relational Databases . . . 7

2.6 Web Services . . . 10

3 Literature Review 11

3.1 Auditing . . . 11

3.2 Biometrics . . . 12

3.3 Relational Databases . . . 14

3.4 Web Services . . . 15

3.5 Commercial Software Solutions . . . 16

4 Paper Equivalent Forms Framework 17

4.1 General System Architecture . . . 17

4.2 Central Data Server . . . 18

4.2.1 Application Database (CDS' Database) . . . 18
4.2.2 CDS Web Services . . . 18
4.3 Clients . . . 20
4.3.1 Capable Clients . . . 20
4.3.2 Limited Clients . . . 21
4.3.3 PEF Forms Manager . . . 21
4.3.4 Reports Administrator . . . 23
4.3.5 Audit Trail Analyzer . . . 25
4.4 Certification Authority . . . 25
4.4.1 Records and Certificates Database (CA's Database) . . . 26
4.4.2 CA Web Services . . . 27
4.4.3 Encryption Algorithms . . . 27
4.4.4 Record Audit and Possible Attacks . . . 27
4.5 Transaction Processing Protocol . . . 28
4.5.1 Protocol 1 . . . 28
4.5.2 Protocol 2 . . . 28

5 Experimental Analysis **29**
5.1 Materials and Equipment . . . 29
5.2 Experimental Scenarios . . . 29
5.3 Performance and Load Analysis . . . 29
5.4 Security Analysis . . . 30

6 Conclusions and Future Work **33**

APPENDICES **36**

1 Relational Schema **37**

2 Detailed Processing Times **39**

3 Full Experimental Results **40**

LIST OF TABLES

1 Protocol 1 - CDS *addRecord* Processing Times 29
2 Protocol 1 - CDS *addRecord* Processing Percentage (Graph) 29
3 Protocol 2 - CDS *addRecord* Processing Times 30
4 Protocol 2 - CDS *addRecord* Processing Percentage (Graph) 30
5 Protocol 1 - CA *genCertificate* Processing Times 30
6 Protocol 1 - CA *genCertificate* Processing Percentage (Graph) 31
7 Protocol 2 - CA *genCertificate* Processing Times 31
8 Protocol 2 - CA *genCertificate* Processing Percentage (Graph) 31
9 Protocol 1 - Additional Measures Overhead 31
10 Protocol 2 - Additional Measures Overhead 31
11 Comparison of Protocol 1 and Protocol 2 Total Times 32

LIST OF FIGURES

1 System Architecture 17
2 Entity-Relationship Diagram for the Application Database 19
3 Simplified Data Storage Schema 20
4 Limited Client Application Prototype 21
5 PEF Forms Manager 22
6 Template Design 23
7 Form Data Input 24
8 Change Confirmation 24
9 Audit Trail 24
10 Audit Form Passed 25
11 E-R Diagram for the CA Database 26
12 Alternate E-R Diagram for the CA Database 26

LIST OF SYMBOLS AND ABBREVIATIONS

ADB - Application Database
BIR - Biometrics Identification Resource
CA - Certification Authority
CDS - Central Data Server
DB - Database
DBMS - Database Management System
DRM - Digital Rights Management
DES - Data Encryption Standard
EDI - Electronic Data Interchange
ER - Entity-Relationship
FDA - Food and Drug Administration
FM - Forms Manager
HTTP - Hyper-Text Transfer Protocol
PEF - Paper Equivalent Forms
RDBMS - Relational DBMS
RSA - Rivest-Shamir-Adleman Encryption Algorithm
SAML - Security Assertion Markup Language
SOAP - Simple Object Access Protocol
SQL - Structured Query Language
SSL - Secure Sockets Layer
TCP - Transmission Control Protocol
TLS - Transport Layer Security
UDDI - Universal Description, Discovery, and Integration
W3C - World Wide Web Consortium
WS - Web Service(s)
WSDL - Web Services Description Language
XML - eXtended Markup Language

1 Introduction

1.1 Motivation

When someone writes a check, an inspector collects data, or any other type of form is filled out on paper, it is hard proof that the information provided and any changes made to it were indeed provided by that person. This is achieved by corroborating signatures, initials, and other similar means of authentication. For data input into an electronic computer form it is hard to trust its authenticity if there are no means to confirm who entered the information or even if it was altered afterwards. This is the case despite the fact that such information is generally kept in relatively "secure" places (like restricted access servers and databases).

In an entity like a pharmaceutical company, a government agency, or even a hospital, paper-based forms are a required part of most of its processes. Even when using computer-based data, most of it is meant as an alternate storage, input by a data entry or secretary; paper forms are still expected to be filled out first. This type of system is not very practical given what current technology has to offer in the form of large data storage capacity, computer graphical user interfaces, web services, mobile computers, and biometrics devices. This is particularly important in environments where government and security agencies need to perform audits. In such entities it is also important that someone from the inside (like an administrator) that has the necessary access permissions to alter that information does not tamper with it. Paper records pose no problems for these agencies (from their point of view), but in order to make the transition to electronic data, some security, confidentiality and trustworthiness requirements must be met.

1.2 Purpose

The result of this research is to provide a framework for electronic forms that is as trustworthy as physical paper and even provides additional security procedures and data logging (audit trail-keeping) mechanisms. In particular, the framework provides equivalent or superior authenticity, non-repudiation, access control, and integrity than that of paper-based records. In our proposed framework, data input by the user and related information (such as the date, time, author, and value) will be logged internally and also verified by a third-party by providing trusted

digital “approvals” (certificates) certifying that a particular change has been made. These measures will provide the means to verify the authenticity of the information stored within the forms and whether the data has been tampered with or not.

This research also defines the interaction or communication protocol between the server(s) and its client(s) and its requirements. Likewise, the interaction or communication protocol between the proposed central company or organizational server and the certification authority, as well as the structure of the proposed certificates is also defined and characterized.

A series of analysis on the system’s security, performance, and load capabilities were performed. The impact of the proposed security and reliability measures on the overall performance of the system is quantified and analyzed. In this analysis, the proposed system, with each measure added sequentially or separately, is compared with a typical base system with the same functionality. This research attempts to prove that the benefits obtained by the additional security and reliability measures outweigh the impact they have on the system’s performance.

A processing time analysis was made on the most time-critical and frequently used process, record addition. The important areas of the communication protocol were classified and the most time-consuming area was identified. Taking these results into consideration, an improved version of the framework was developed, in which the performance of the system is greatly improved, further justifying the use of the added measures.

2 Theoretical Background

This section provides a theoretical background on the topics related to this research.

2.1 Auditing

An audit is an examination of a system, process, organization, or product, among others things, that is performed to verify that it operates according to certain rules (standards, practices, or regulations). It may also evaluate the controls that are in place to determine if they conform with such rules. Such audits are common in accounting and the pharmaceutical industry.

In [16], the author presents different uses of audit trails and their applications in different areas of work and study. Audit trails are a log of records, transactions or communications that are related to a single person, account or entity. They generally present a chronological account of the events or changes that occurred or were performed by someone. It can then be used to recreate with high certainty the state of the related entity at a certain moment in time. The author in [16] states that audit trails, whether computer-based or manually produced, play a significant part in detecting and preventing fraud within systems.

In 2002, several bogus accounting practices on several companies were discovered. A long list of companies with questionable reporting was found. This challenges caused a loss of credibility in auditing, and worsened an already unsteady stock market and the economy in general. Several accounting firms were blamed since they are the ones who are expected to provide independent certification of financial reports. A study [1] conducted on 2002 in Australia examined the use in the courtroom of audit trail data from law enforcement agencies to corroborate evidence. Several criteria were identified for the significance of audit information. Of those, the ones relevant to this research are: proof of user activity, technical security for audit trails, audit trail content, recording of all activity, and positive identification of users.

2.2 Part 11

As presented in [6] by the Food and Drug Administration (FDA), the 21 CFR Part 11 regulations state the "criteria under which the agency considers electronic records, electronic signatures, and handwritten signatures executed to electronic records to be trustworthy, reliable, and

generally equivalent to paper records and handwritten signatures executed on paper". It allows for electronic records that meet the specified requirements to be used in lieu of paper records. The FDA may inspect the computer systems, controls, and documentation maintained under Part 11.

When closed systems (those in which system access is controlled by people who are responsible for the content of the electronic records) are used, procedures and controls designed to ensure the authenticity, integrity, and, when appropriate, the confidentiality of electronics records shall be employed. The signer cannot be able to readily repudiate the signed record as not genuine. For open systems (those in which system access is not controlled by people who are responsible for the content of electronic records) the same procedures and controls as those for closed systems are in effect. However, additional measures must be taken, such as document encryption and use of appropriate digital signature standards, to ensure record authenticity, integrity, and confidentiality.

Under Part 11, signed electronic records must contain the following information associated with the signing:

- The printed name of the signer
- The date and time when the signature was executed
- The meaning or purpose associated with the signature

A constraint on electronic signatures is imposed which states that each signature shall be unique to one individual and shall not be reused by, or reassigned to, anyone else. For identification codes or passwords several controls must be employed:

- The uniqueness of each combined identification code and password must be maintained.
- The identification code and password should be periodically checked, recalled, or revised.
- Lost, stolen, missing, or otherwise potentially compromised token, cards, and other devices that bear or generate identification code or password information must be de-authorized and temporary or permanent replacement must be issued.

An independent contracting firm named Labcompliance, has summarized [15] the primary requirements of the FDA Part 11 regulation as follows:

- Use of validated equipment and computer systems

- Secure retention of records for instant analysis reconstruction
- User-independent, computer-generated, time-stamped audit trails
- System and data security, data integrity, and confidentiality through limited authorized system access
- Use of secure electronic signatures for closed and open systems, and digital signatures for open systems

2.3 Cryptography and Certificates

Cryptography is the practice of using linguistic and mathematical techniques for securing information, particularly in communications. It has a variety of applications including, for example, encryption, authentication, digital signatures, electronic voting and digital cash. In [23] the authors study the problem of intranet security within organizations and Electronic Data Interchange (EDI) business solutions. But the basic needs that they identified can be applied to any secure system. These needs are:

- *Privacy* - provided by the ability to encrypt messages across an insecure network.
- *Authentication* - which verifies the identities of the agents involved in a transaction or process
- *Integrity* - ensuring that files or messages have not been altered in transit
- *Non-repudiation* - which prevents agents from denying a certain action
- *Access Control* - determining who is given access to a system and what resources they can access

Encryption is used in order to protect communications channels from eavesdroppers. It uses an algorithm and a key (one of which, or both, are kept secret from outsiders) to transform data. Modern reliable encryption is based on known algorithms that are mathematically proven to be efficient. The strength of such algorithms relies on the secrecy of the key. There are two main forms of encryption: *secret-key or symmetric encryption* and *public-key encryption.*

Symmetric encryption is based on a key shared between two parties. The same key both encrypts and decrypts messages. The Data Encryption Standard (DES) [26] is the symmetric algorithm traditionally used, but has since been replaced by more powerful ones. In [24], the author

describes the origin and definition of public-key cryptography. Public-key cryptography evolved from an attempt to attack two of the most difficult problems associated with symmetric encryption: key distribution, and the need for *digital signatures*, the electronic equivalent of signatures used in paper documents. Public-key algorithms rely on one key for encryption and a different but related key for decryption. They have the following important characteristic: it is computationally infeasible to determine the decryption key given only knowledge of the cryptographic algorithm and the encryption key. Also, some algorithms like RSA [14](one of the most widely-used and also one of the first), also exhibit the following characteristic: either of the two related keys can be used for encryption, with the other used for decryption. The two keys used for public-key encryption are known as the *public key* and the *private key*. The private key is always kept private, but it is referred to as such, rather than a "secret key" to avoid confusion with symmetric encryption. The public key is generally placed in a public register or freely distributed, while the companion key is kept private and unavailable to outsiders.

Electronic certificates are like people's ID cards and are used to verify identities. A trusted certification authority (TCA) manages and distributes these certificates. By time-stamping the signatures, so that they expire after some given period of time, it renders their abuse more difficult. As stated in [24], X.509 [12] is an important certificates standard because its certificate and authentication protocols are used in a variety of contexts such as: e-mails, internet protocol and transport protocol security, and secure electronic transactions. It is based on the use of public-key cryptography and digital signatures. The certificate includes (among other things): an identifier of the subject or user, an identifier of the issuer of the certificate, an identifier of the signature algorithm used, the public key of the subject, and the signature, which contains a hash code of all the other fields and the signature algorithm identifier. Our proposed system will use a similar type of certificate, but smaller in scope and more tailored for our expected use (certifying records additions).

2.4 Biometrics

Biometrics refers to the science and technology for measuring and analyzing human physiological or behavioral characteristics such as fingerprints, eye retinas and irises, voice patterns,

facial patterns, and hand measurements, usually for authentication purposes. The digital counterpart of such measures is usually known as a Biometric Identification Resource (BIR). Biometrics have been studied and used for some time now for the preserving of privacy and the security of sensitive information as well as in other areas, such as criminal investigation.

Fingerprints are probably the most widely used biometrics. Some research is being made for ways to improve the performance and precision of fingerprinting. Each biometrics has its strengths and weaknesses and the choice depends on the application. There is no all-in-one solution that can effectively meet all the requirements of all applications. The article in [22] describes other forms of biometrics, besides fingerprints, that have proven useful as well. A method that has been used for at least 15 years is hand recognition. It has several advantages over fingerprinting: identification data takes about 9 bytes compared to the various hundreds of bytes or kilobytes needed to store the detailed ridge data of fingerprints. However, hand geometry data may not be as unique as fingerprints; yet its false-acceptance and false-rejection rates of less than 0.2% are acceptable for a lot of access control methods.

Another recognition system, which closely approximates that used by us everyday, is face recognition. There are systems in development which identify relationships in facial geometry. A set of feature vectors are derived by a neural network by performing certain mathematical transformations on video images of faces. These vectors can then be compared to stored records to identify and individual.

Finally, eye recognition systems have been used to verify identity for years. Retina-scan identification products filter the infrared spectrum off the beam of a flashlight bulb and measures light intensity at various points on the retina. A camera records the infrared light as it comes back and converts its analog signal into a digital byte code.

2.5 Relational Databases

The concept of relational databases was first introduced by Codd [4] in 1970 with his *relational data model*. The term *relation* is used in its accepted mathematical sense, it is a set that satisfies certain properties of a combination of other sets. Given sets S_1, S_2, ..., S_n, R is a relation on these n sets if it is a set of n-tuples each of which has its first element from S_1, its second from

S_2, an so on. More concisely, R is a subset of the Cartesian product $S_1 \times S_2 \times \cdots \times S_n$. S_j is known as the j^{th} domain of R. An array is frequently used to represent relations. An array which represents an n-ary relation R has the following properties:

- Each row represents an n-tuple of R.
- The ordering of rows is immaterial.
- All rows are distinct.
- The ordering of columns is significant and corresponds to the ordering of the domains on which R is defined.
- The significance of each column is partially conveyed by labeling it with the name of the corresponding domain.

As envisioned by Codd, "As time progresses, each n-ary relation may be subject to insertion of additional n-tuples, deletion of existing ones, and alterations of the elements of any of its existing n-tuples". Of course, modern database management systems (DBMS) provide a much larger variety of operations on the data as the querying language evolved and DBMS systems incorporate more features.

The entity-relationship (ER) model as defined by Chen [2] and after its many modifications, allows us to describe the data involved in a real-world application in terms of objects (or *entities*) and their relationships and is widely used to develop initial database designs. These designs can then be translated to the schemas used by DBMS systems. An ER diagram presents all the entities (objects or actors) of the system along with the relations among them. The boxes represent entities, ovals represent their attributes, double ovals are composite attributes, and diamonds represent the relations among entities. All entities and some of the relations translate into or have equivalent tables in the database. The author in [19], identifies the first three steps in the database design process as:

1. *Requirements Analysis* - Identifying what data is to be stored in the database, what applications will be built on top of it, and what operations are more frequent and subject to performance requirements.

2. *Conceptual Database Design* - Developing a high-level description of the data to be stored, and the constraints that need to be held over this data. This step generally involves the creation of an ER Model as a semantic data model.

3. *Logical Database Design* - Involves the selection of a DBMS system to implement the database and the conversion of the conceptual database design into a database schema in the data model of the relational DBMS or a *relational database schema*.

Part of our proposed research is to define and implement a database schema for storing the electronic forms and records which allows for flexibility in the content and format of the forms and their templates.

Recent studies have focused on improving the security within DBMS systems and database-oriented applications. The authors of [11] study database security from a cryptographic point of view. They propose integrating modern cryptography technology into relation database management systems (RDBMSs) to solve some major security problems present in them. An example of a database application is that of web stores. When we look at the purchase activity of a customer and trace the data flow, there are two main security issues than need to be addressed:

1. *Secure Data Transmission* - When a customer submits his/her confidential information through a web browser, the information should remain confidential on its way to the server, and the DB server

2. *Secure Data Storage and Access* - When the confidential data arrive at the DB server, the data should be stored in such a way that only people with proper authorization can access them.

A credit card number is well protected on its way to the a web server via a Secure Sockets Layer (SSL) [7] connection. However, once the data arrive at the DB, it is not stored or processed in a sufficiently secure way. Only recently, many major database companies are now adopting the loose-coupled approach of adding optional security support to their products. User management in RDBMS typically includes user account creation, maintenance, and user authentication. A database administrator (DBA) is responsible for creating and managing user accounts. This account information is stored in system catalog tables. However, the problem with this process is that the

DBA can impersonate any other user by changing the system catalogs and he/she can do things on a user's behalf without being authorized or detected by the user or the system. The major security mechanism deployed in RDBMSs is access control. It it based on the concept that if a user has the corresponding privileges, then he/she can access a particular database object. A DBA, however, has all the system privileges, which gives him/her the capability to do the most damage to the system.

2.6 Web Services

A *Web Service* (WS) is a collection of protocols and standards designed to ease the exchange of data between applications or systems over intranets and the Internet. Web services provide interoperability through the use of various standards and technologies. eXtended Markup Language (XML) is used to provide information about the data in a document to users of varying platforms. The Simple Object Access Protocol (SOAP) [29] is used for cross-platform interapplication communication. The Web Services Description Language (WSDL) [3] is used to describe online services. Finally, the Universal Description, Discovery, and Integration (UDDI) [27] protocol is used to find available Web services on the Internet or corporate networks.

Because Web services are a fairly new technology, it is not yet widely used and accepted and some users and companies are reluctant to work with Web services because they do not have much experience with the specifications and the security problems mentioned above. However, because Web services offer a great variety of new possibilities and are showing great promise, Web services security measures are expected to be widely adopted in the near future.

3 Literature Review

This section presents some important publications and different work related to this research and its topics.

3.1 Auditing

The use of audit trails is very important for the information technology security evaluation model of the Common Criteria (CC) [5]. Security auditing (which is provided in CC Part 2, Chapter 3) involves the recognition, recording, storage, and analysis of information related to security-relevant activities. Audit trails are used to determine which of such activities took place and who performed them.

Audit trails were even considered and included in the "Help America Vote Act of 2002" by the U.S. Congress [25], which allowed for electronic voting systems. The section on audit capacity in the act states that "the voting system shall produce a record with an audit capacity for such system". The act also called for the system to support manual audit capacity and produce a permanent paper record. The voter should be able to change the ballot or correct any error before the permanent record is produced. This permanent paper records could then be used for any recount conducted. However, the vendors of such systems protected their products under restrictive trade-secret agreements with the counties that purchased them. This caused the integrity of the audit trail generated by the computer to be unverifiable. People are not provided with any way of validating whether the equipment that recorded the ballots is operating properly or that the printed ballots accurately represent their intended votes.

The work in [20] describes a method which allows secure audit logs to be used for computer forensics. The method works as follows. Let's assume that U is an un-trusted machine; in other words, it is not physically secure or tamper-resistant enough to prevent it from being taken over by an attacker. This machines needs to build and maintain a file of audit log entries of some processes, measurements, events, or tasks. With a minimal amount of interaction with a trusted machine, T, the method prevents an attacker who gains control of U at a time t to read log entries made before time t, and if such entries are altered or deleted by him/her, it will be detected when U next interacts with T. According to the authors, no cryptographic method can be used to protect

the audit log entries written after an attacker has gained control of U or to prevent the deletion of log entries. The only thing the cryptographic protocols can do is guarantee the detection of such changes, assuming U eventually manages to communicate with T. The proposed method assumes that U initially shares a secret key with T. The security of the log file is thus provided by the following actions:

1. The log's authentication key is hashed, using a one-way hash function, immediately after a log entry is written. This new value overrides the old one.
2. Each log entry's encryption key is derived, using a one-way process, from that entry's authentication key.
3. Each log entry contains an element in a hash chain that serves to authenticate the values of all previous log entries.
4. Each log entry contains its own permission mark that defines which log entries can be accessed by partially trusted users.

As mentioned before, the authors present a way to use this method as an aid in forensic analysis. The discussion assumes that audit log entries detect an intrusion. If an attacker can gain control of U without triggering an alarm condition and associated audit-log entry, then this system cannot help. There are two types of suspicious entries: valid entries that indicate an intrusion, and invalid entries that indicate that the audit log has been tampered with. Since the attacker has only two options: leave the incriminating log entries in the log or delete them and ensure the deletion will be noticed, one of these two suspicious entry types will indicate that a break-in has occurred. If there is an invalid entry, one can assume that all entries after the last valid one are suspect and that all entries before the first valid one are genuine.

3.2 Biometrics

The work in [10] presents a method for improving the precision of fingerprint verification systems. According to [13], the fingerprint is unique and invariant with aging, thus it can be used for user authentication by comparing two fingerprints. In order to make fingerprint identifications, a fingerprint examiner relies on the details of ridge structures of the fingerprint. The structural features, known as minutiae, are composed of the points where ridges end or bifurcate. Each

minutia is described by the position in the coordinate system, the direction it flows, and its type (ridge ending or bifurcation). A fingerprint verification system consists of two phases: enrollment and verification. At first, the fingerprint image of an enrollee is acquired and preprocessed. Then, the minutiae are extracted from the image and stored as an enrolled template. In the verification phase, the fingerprint is read from a claimer, the similarity between the enrolled minutiae and the input minutiae is estimated. However, it is possible to detect false minutiae and miss true ones. The authors of [10] use multiple fingerprint images on the enrollment phase to discard the false minutiae and compensate for the missed minutiae, thus improving the reliability of each fingerprint image. After performing experiments using FVC 2002 databases [8] by using this method, a reduction of the equal error rate of 1.38% was achieved, when compared to using a single impression. It also achieved a false match rate (FMR) 100 of 6.15%, compared to 40% for single impressions.

In [28], a method is described for binding a cryptographic key with the biometric template of a user stored in a database in such a way that the key cannot be revealed without a successful biometric authentication. The authors developed this method to be incorporated into Digital Rights Management (DRM) systems. In such systems, a user must first be authenticated in order to have access to the digital content. When using a generic cryptographic system the user authentication is possession based; that is, possession of the key is sufficient evidence to establish user authenticity. However, because cryptographic keys are long and random, they are difficult to memorize. They are stored somewhere (like a computer file or smart card) and released based on an alternate authentication mechanism, such as passwords. Most passwords can be easily guessed because they are simple or broken by dictionary attacks. Password-based authentication systems perform accurately as intended by their designers since they do not involve any complex pattern recognition. Biometric signals and their representations of a person, on the other hand, vary dramatically depending on the acquisition method and environment, and the user's interaction with the acquisition device. Some of the common reasons for biometric signal/representation variations are: inconsistent presentation, irreproducible presentation, and imperfect signal/representation acquisition. Because of these complex variations, determining whether two presentations of a biometric identifier are the same typically involves complex pattern recognition and decision making.

The basic idea proposed in [28] is that the biometric component performs user authentica-

tion, while a generic cryptographic system can handle the other components of containment, such as secure communication. If a legitimate user wishes to access certain digital content, he/she offers a biometric sample to the system; if the sample matches, the cryptographic key is then released. They key can then be used to decrypt the content and thus, the user now has access to the content. This method is known as *biometric-based key release*. Instead of storing the cryptographic key in the user's record, it can be hidden within the user's biometric template itself. When there is a successful biometric match, the correct cryptographic key is extracted from the biometric database template and released into the system. However, it is important that the cryptographic key be monolithically (uniformly) bound with the biometric template in such a way that it cannot be revealed without a successful biometric authentication. This last method is known as *biometric key generation* or *binding*.

3.3 Relational Databases

Current RDBMSs provide little or no data encryption. Data are usually stored in tables in the same form they are loaded, mostly in their plain text form. The owner of the table, or anyone with the appropriate privileges, can read or alter the contents of the table. Recently, DB vendors have started to support encryption by means of a PL/SQL package to encrypt/decrypt data. However, this type of packages suffers from the same drawback as any other database object: the DBA can replace it with a version that contains trapdoors, through which he/she can get hold of any confidential information. Therefore, these packages cannot support truly secure database encryption. As long as the DBA is allowed to control security without any restriction, the whole system becomes vulnerable and can be compromised. Traditionally, a data dictionary stores all of the information that is used to manage the objects in a database. It consists of many catalogs and views. These are maintained by the DB server and update through the execution of system commands. However, a DBA can still make changes in a catalog table if he/she wants to do so.

The authors of [11] propose the use of a *security catalog* to remedy the above problems. A security catalog is like a traditional system catalog but with two security properties: it can never be updated manually by anyone, and its access is controlled by a strict authentication and authorization policy. Some columns in a table could store security-related information and are

called *security columns or fields*. Each security field has a corresponding security flag field that specifies how the value of the field can be accessed. If the table that stores users, for example, had its password field made a secure one, only a user himself/herself would be able to change his/her password. A *security dictionary* consists of all the security catalogs and provides a secure and reliable repository where information can be safely stored. In a similar manner, specific table fields can be made to support encryption. These security and encryption measures can be incorporated into existing RDBMS by simply extending some relevant SQL statements. By using a security catalog, no one (including the DBA) will be able to manipulate other users' confidential information or impersonate other people without being detected and caught.

Sesay, et. al., [21] propose a three-layered model to support data encryption within databases. The first layer is the user interface layer which contains two blocks: one for low level (L1) users and one for high level (L2) users. The database objects are classified into public (classified or unclassified) and private objects. All users have access rights to their own personal *private* data and to unclassified public data, whilst those in L2 have access rights to both *unclassified* and *classified* public data. All users posses a unique key, K_P, that they use when accessing their encrypted private data. The second layer is the DB management layer which contains two blocks: the mandatory access control (MAC) system, and another that includes a tamper-free controller (KC) linked with a trusted subject (TS). The KC is in charge of generating and storing two sets of encryption keys, a K_P for each user's private data and K_j for classified data. It also encrypts sensitive data before being stored in the DB and decrypts data in response to users queries that satisfy the security requirements. The TS is in charge of managing subjects and objects and their privileges. Finally, the bottom layer contains the database. The DB system stores unclassified data in the clear while classified and private data are stored in encrypted form. This database encryption scheme is proved to be efficient by providing maximum security to the database whilst minimizing the added time cost for encryption and decryption.

3.4 Web Services

The author of [9] gives a survey of the common security problems related to Web services and the most recent efforts to deal with them. Web services raise a new security concern by

opening up networks by letting outside users access databases, applications, and internal users. Basic Web services transactions are unencrypted and unsecured. Current web browsers support SSL and Transport Layer Security (TLS), but these protocols do not scale well to complex, high-volume transactions, which are typical of Web services. The reason for this is due to the need of SSL and TLS systems to decrypt data every time it arrives at a new Web server and then encrypt the data for transmission to the next server. The original SOAP version by the World Wide Web Consortium (W3C) provided no security.

The Security Assertion Markup Language (SAML) [17] defines security-related schemas for structuring documents. It defines schemas for the structure of documents that include information related to user identity and access or authorization rights. SAML functions as a framework for exchanging authentication, attribute, and authorization assertions (proofs of identity) across multiple participants over the Internet using protocols such as HTTP and SOAP. SAML can also indicate the authentication method that must be used with a particular message, such as a password, Kerberos authentication ticket, hardware token, or X.509 digital certificate. It can work either via a centralized certificate authority or directly between users.

The Web Services Security (WS-Sec) protocol [18] has been developed as a way for Web services to work with several different security models via SOAP extensions. WS-Sec lets security data to be attached to the headers of SOAP messages. It lets companies send messages with digital signatures that tell recipients whether documents have been altered during transmission and whether the documents are actually from the supposed sender.

The main problem with Web services security is that the XML is transferred over HTTP, allowing traffic to pass through firewalls via TCP port 80. This results in unblocked communications between networks whose firewalls block all ports except the ones that Web protocols (hence, HTTP) use. This problem, of course, can be solved by upgrading firewalls to recognize, examine, and filter XML and SOAP traffic.

3.5 Commercial Software Solutions

...

4 Paper Equivalent Forms Framework

This section describes the Paper Equivalent Forms (PEF) framework and all its parts.

4.1 General System Architecture

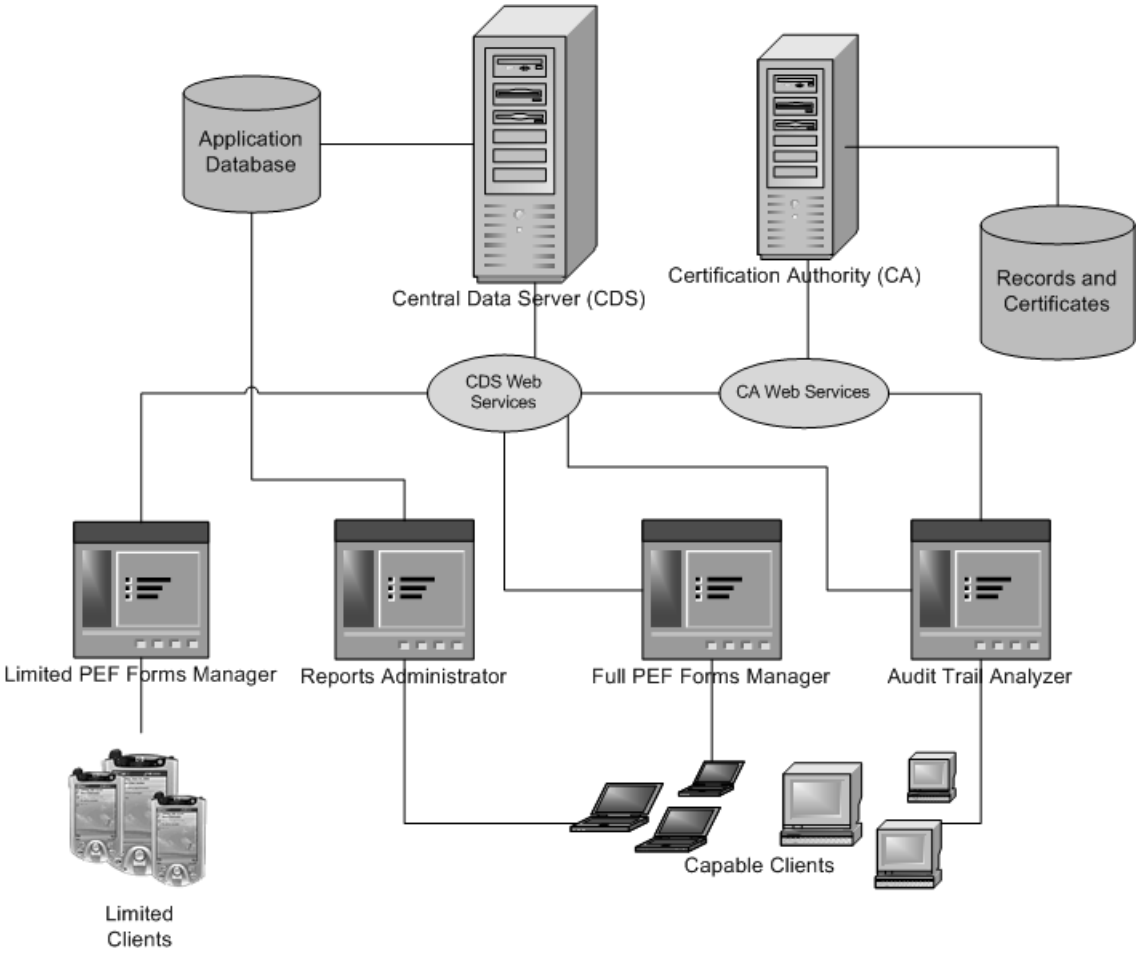

Figure 1. System Architecture

The Paper Equivalent Forms (PEF) system keeps track of data as it is filled out in electronic forms and any changes made to it, when the changes were made, and by whom. Figure 1 shows the general architecture of the proposed system. When changing the value of a particular field in a form the user must provide proof of his/her identity by some means of authentication (passwords or BIR's). A certificate acknowledging the change will then requested from a third-party Certification Authority (CA), so the form can be later audited. This provides a means of

comparing reports, verifying changes, and/or finding out about unwanted or dishonest alterations. The Reports Administrator shown in Figure 1 is beyond the scope of this research and should be the work of future or even parallel research.

4.2 Central Data Server

The Central Data Server (CDS) provides centralized storage and data replication as well as coordinates the different requests by the clients. Whenever a client must access or update a form or template, it must make a request to the Central Data Server which must look through the database and carry out the appropriate action based on the corresponding client's request. It may need to retrieve the data from the database or update it. The CDS uses XML Web Services for handling the various clients' requests.

4.2.1 Application Database (CDS' Database)

The Application Database (ADB) contains all the forms, templates, and records data. The Entity-Relationship (E-R) Diagram used for the ADB is shown in Figure 2. This diagram presents all the entities (objects or actors) of the system along with the relations among them. The boxes represent entities, ovals represent their attributes, double ovals are composite attributes, and diamonds represent the relations among entities. All entities and some of the relations translate into or have equivalent tables in the database.

**Add info about the relational schema here... or maybe in an appendix?

Relational Schema

Every form is linked to its corresponding template and the fields it contains are extracted from the template. A form contains rows that represent inputs taken at different times. Each row contains fields (headers and columns) that contain the actual data and define its type. A field may be altered by the user, in which case a new record will be created for that field containing information about the change made.

4.2.2 CDS Web Services

The CDS uses XML Web Services for handling the various clients' requests. The CDS Web Services provide the following methods: " saveUser - Create a new PEF user in the database authenticated by a password or BIR.

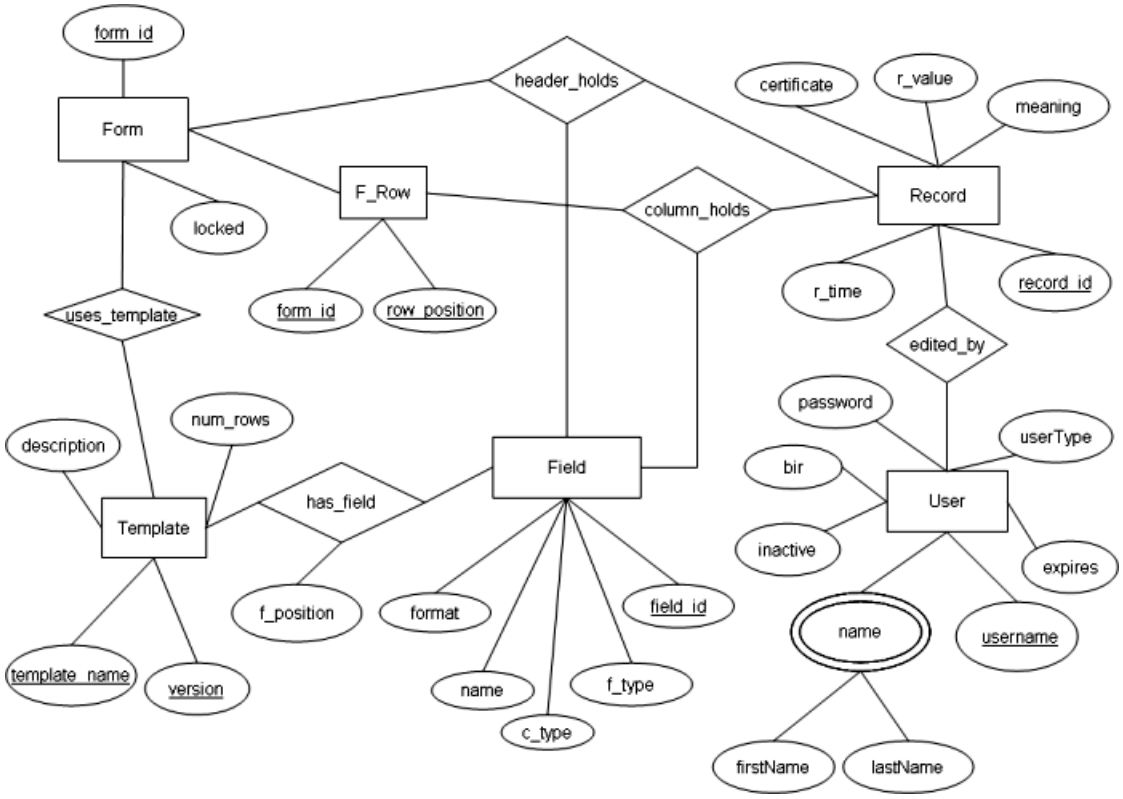

Figure 2. Entity-Relationship Diagram for the Application Database

1. checkUser - Check the validity of a user and verify the given password or Biometric Identification Resource (BIR) (fingerprint, retinal scan, etc.).
2. getUsers - Return a list of all the users in the ADB.
3. saveTemplate - Save a template and its fields in the ADB.
4. getTemplatesInfo - Returns a list with the XML representation of all the templates available in the ADB matching a user query containing each one's name, description, and version.
5. fetchTemplate - Retrieve a template from the ADB based on the template's name, version, and description.
6. createForm - Create a new form in the ADB.
7. getFormsInfo - Returns a list with the XML representation of all the forms available in the ADB matching a user query containing each one's id, name, and its template name and version.

8. fetchForm - Retrieve a form from the ADB based on the form's id and its template name and version.
9. addRecord - Store a new record in the ADB when a field in a form is changed.
10. auditForm - Request an audit to be performed on a particular form.

4.3 Clients

Clients are categorized by their data storage and processing capabilities. Depending on their resources, data will be stored using a database, in XML files, or both. They communicate with the server using XML and Web Services.

4.3.1 Capable Clients

A Capable Client is one with extended resources. These may include workstation-type computers and some portable computers. In such clients data storage capabilities are greater and either a local database or XML files can be used. The capable client's application should provide some additional features over the limited version like the ability to have multiple forms open at the same time, extended server connectivity, and more customizable preferences, among other things.

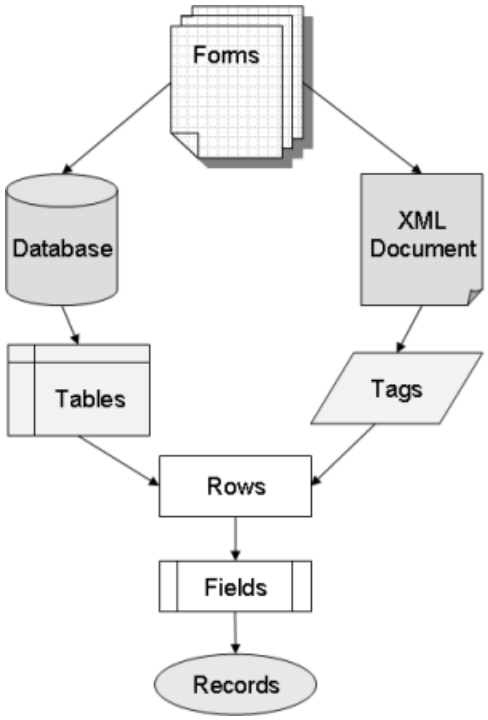

Figure 3. Simplified Data Storage Schema

It can be appreciated from Figure 3 that by using XML documents we can have an almost uniform representation of the forms and their data which simplifies the translation from one to the other. Consequentially, the PEF Forms Manager (see Section 4.3.3) provides the user with the ability of exporting forms (along with their data) into XML documents.

4.3.2 Limited Clients

A Limited Client is one with limited resources and capabilities. These may include mobile devices like Pocket PCs or portable computers. In such clients, the client application capabilities are limited and XML documents (stored as local files) can be used for storing forms and templates data. The mobile application has various limitations and presents new problems that must be taken into consideration for its development.

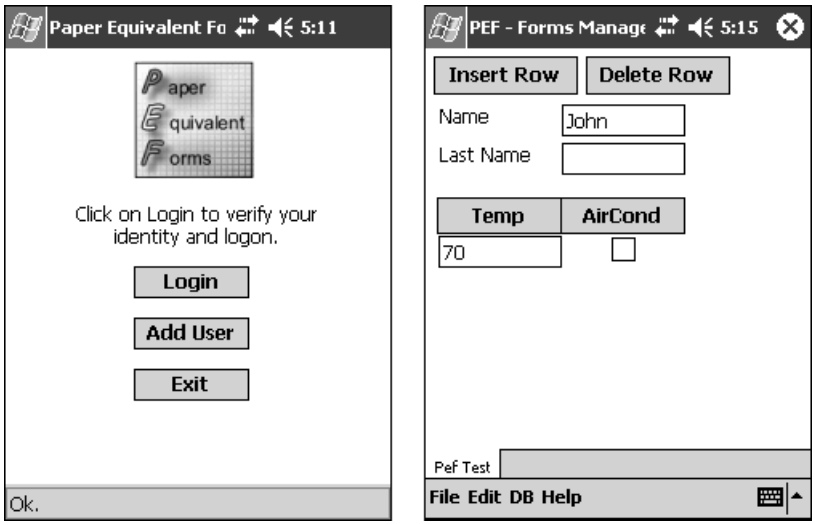

Figure 4. Limited Client Application Prototype

Figure 4 shows an early version of the PEF Forms Manager (see the next section) in development for limited clients. In particular, the screen shot shown is from a Pocket PC.

** Explain that the limited clients are left for future work.

4.3.3 PEF Forms Manager

The PEF Forms Manager (FM) (shown in Figure 5) is the client application used to manage the forms and templates and for data entry. The Full and Limited PEF FM is the client application used on Capable Clients and Limited Clients, respectively. When the PEF FM is

started, the user must login to the program by some form of authentication. Currently, Capable Clients use either passwords or fingerprint scans (via an external fingerprint scanner). For Limited Clients, the user will have to provide his or her fingerprint via the scanner provided by the Pocket PC. After this, the user can begin using the forms manager to open, edit and create templates and forms.

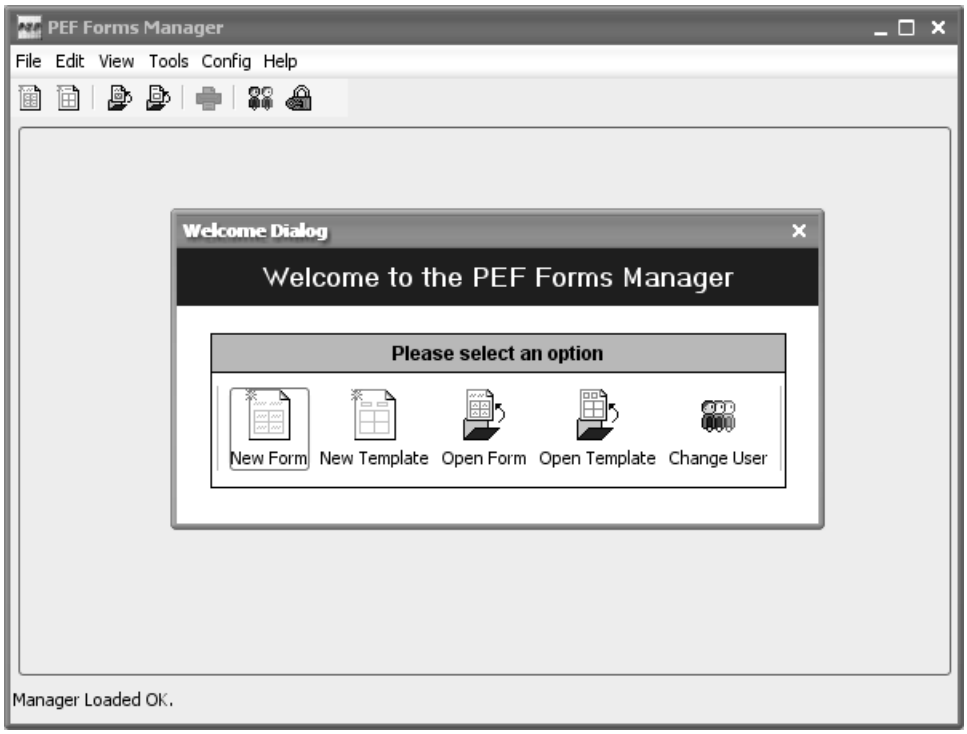

Figure 5. PEF Forms Manager

A user first creates a form template by using a template designer window (Figure 6) that defines the format and layout for a particular type of document. Once the template is created, multiple form instances can be generated to hold the actual information (Figure 7). Forms, as well as templates, can then be edited and manipulated.

When a user changes the value of one the fields in a form, the program will ask the user to confirm the change or cancel the change (see Figure 8). If the user confirms it, at this moment the record will be sent to the Certification Authority (CA) which will sign the record with a private key to produce a certificate. It then attaches this certificate to the record, stores a copy of both the record and its certificate and then sends it back to the client application so it can be stored in

the server's database.

The date, time, author, and value of each record is stored. A new record is added after each change to the field. Information about changes to all form fields is stored in logs called "audit trails" (Figure 9). They generally present a chronological account of the events or changes that occurred or were performed by someone. It can then be used to recreate with high certainty the state of the related entity at a certain moment in time and all changes it went through.

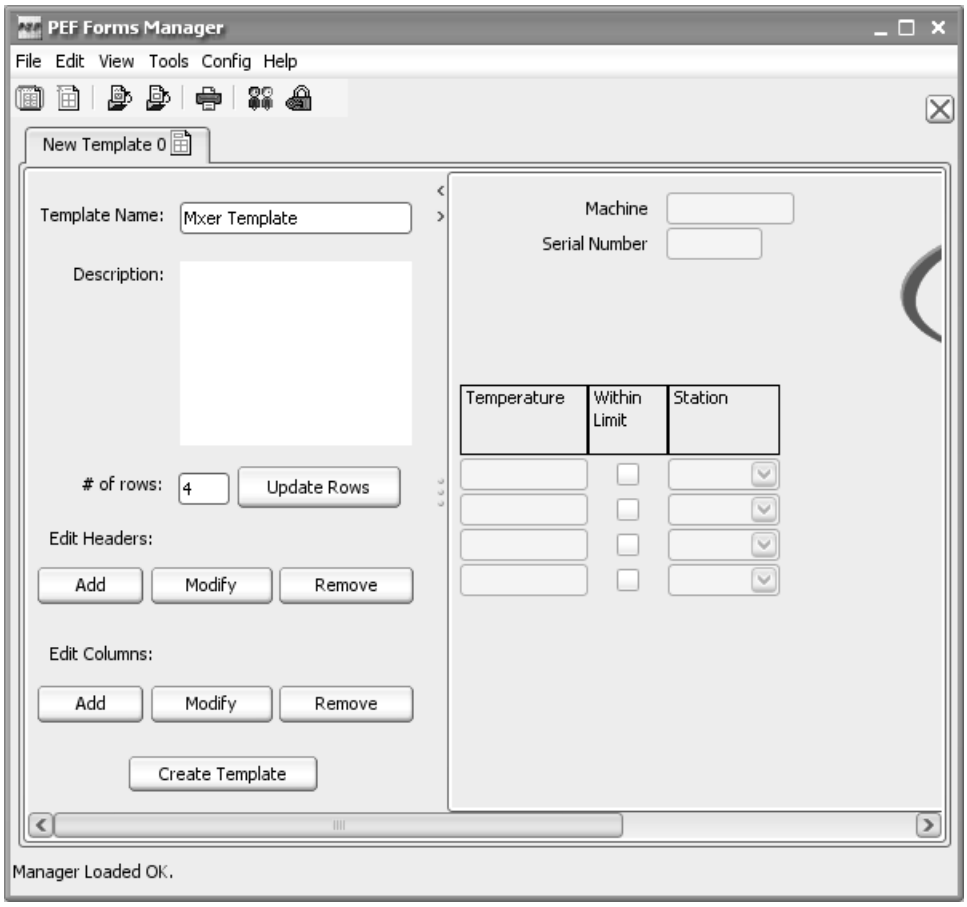

Figure 6. Template Design

4.3.4 Reports Administrator

The Reports Administrator is an application with which users (managers, in particular) can make queries into the database and get the results in a presentable, user-friendly manner by means of reports, graphics, and tables, among others. This application can be a web application at the server side or stand-alone client application.

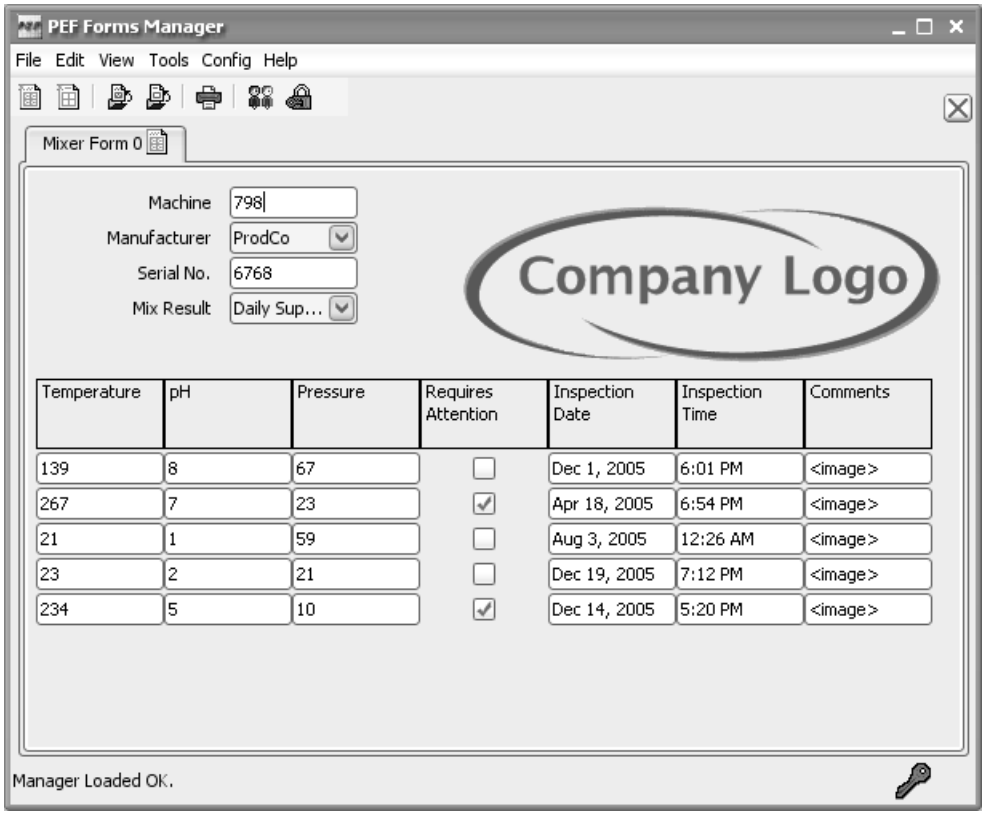

Figure 7. Form Data Input

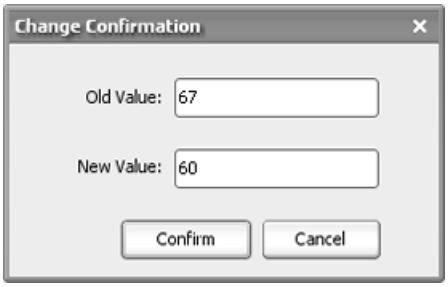

Figure 8. Change Confirmation

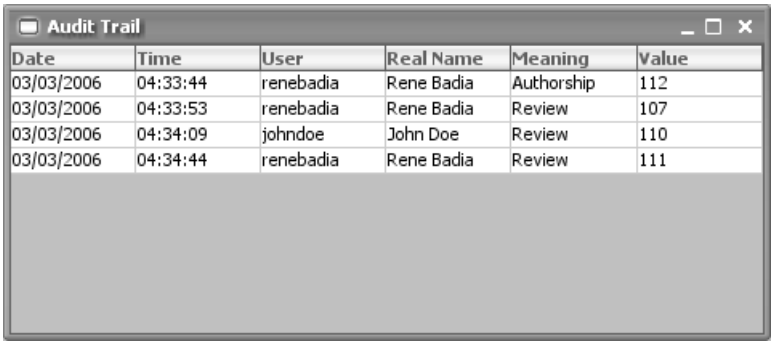

Figure 9. Audit Trail

4.3.5 Audit Trail Analyzer

The Audit Trail Analyzer application can be used to audit the data on the ADB, i.e. compare it against the data stored in the CA. In particular, it tests that the records stored in the ADB match with their corresponding records on the CA's DB. It also tests that the records values in the ADB match those stored in the corresponding certificates.

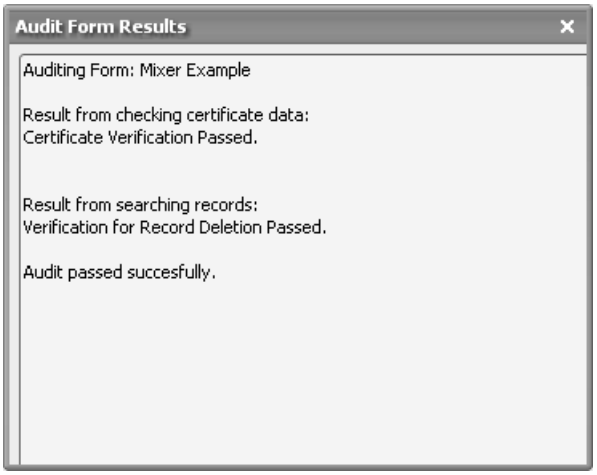

Figure 10. Audit Form Passed

** Bad Audit Form analysis output example

Currently, the Audit Trail Analyzer is embedded into the PEF Forms Manager. This application could also be a web application at the server side or an stand-alone client application.

4.4 Certification Authority

The Certification Authority (CA) is a trusted server or third-party that enables the functionality of auditing the data on the ADB by signing each record and storing its own copy of it. When a user confirms a record, it is sent to the CA, which will then sign the record with a private key to produce a certificate. It then attaches this certificate to the record, stores a copy of both the record and its certificate in its own DB and then sends it back to the CDS so it can be stored in the ADB.

4.4.1 Records and Certificates Database (CA's Database)

The Records and Certificates Database or CA's Database (CADB) contains all the records that have been added to the forms using the PEF FM and their corresponding certificates.

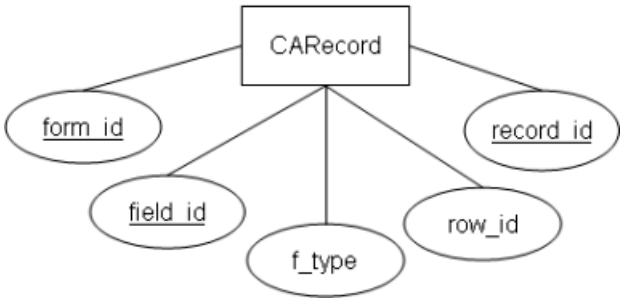

Figure 11. E-R Diagram for the CA Database

The Entity-Relationship (E-R) Diagram used for the CADB is shown in Figure 11.

** Explain the diagram **

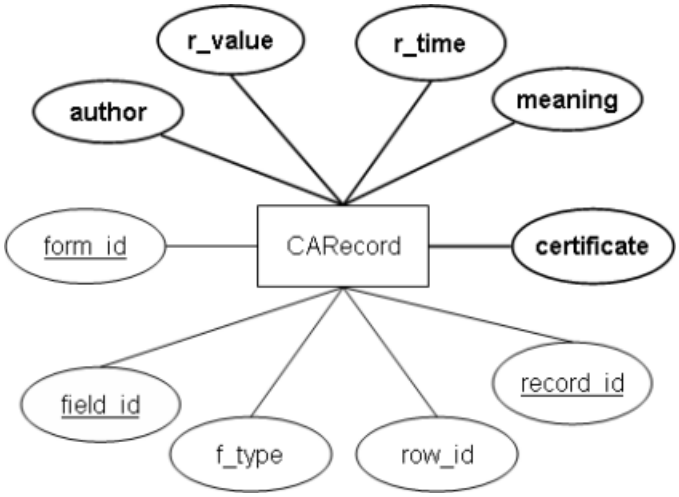

Figure 12. Alternate E-R Diagram for the CA Database

An possible alternative for the CADB is to replicate all records and certificates. The E-R Diagram for this alternate CADB is shown in Figure 12.

** Explain the alternative **

4.4.2 CA Web Services

The CA uses XML Web Services for handling the CDS requests for storing records. It uses cryptographic algorithms to produce a certificate based on the record and stores it on its CADB. The CA Web Services provide the following methods:

1. getCertificate - Creates a certificate for the record based on the record's properties by signing the data with the CA's private key. Then stores the record and the certificate on the CADB. When this method is called a transaction is started to account for any errors that may happen in the process and while the CDS saves its own version of the record. This method returns a number used to reference the transaction that was started and which must be provided when calling the next method.
2. finishCAtrans - Finishes the transaction that was started when the previous method was called and commits the changes (makes the record update permanent) if the CDS operations were successful or rolls back any changes otherwise.

4.4.3 Encryption Algorithms

Encryption algorithm used by the CA.

** Encryption formula

$$certificate \;=\; E_{C_R}(record\ id \mid record\ value \mid time) \tag{1}$$

** Decryption formula

$$recordid \mid record\ value \mid time \;=\; D_{C_U}(certificate) \tag{2}$$

4.4.4 Record Audit and Possible Attacks

** Audit Process

** change value directly

** delete the record

** change the certificate

4.5 Transaction Processing Protocol

Protocol used between the user and the PEF server, and between the PEF server and the CA.

4.5.1 Protocol 1

** Exchange Diagram for addRecord

** Two-face commit diagram

** Crash-recovery diagram

4.5.2 Protocol 2

** Exchange Diagram for addRecord

5 Experimental Analysis

5.1 Materials and Equipment

** TODO - Be more specific, get actual system specs

For the completion of experiments, the following materials and equipment were used:

- One or more worstation-type PCs for the full clients' development and carrying out the tests.
- One or two server-type computers to serve as the CDS and/or CA.
- A biometrics device; preferably, one for capturing fingerprint scans.

5.2 Experimental Scenarios

** Diagram with single-user scenario

** Diagram with 5-users scenario

5.3 Performance and Load Analysis

A series of tests will be carried out to analyze the performance of the system in terms of throughput (number of records and/or certificates added per second) and system load capacity (how many users can the system effectively attend without becoming unstable or unusable).

Area	**Total Time (min)**	**Avg. Time (ms)**
Axis	18.179	31.615
Network	20.141	35.027
Database	0.711	1.236
Certificates	12.191	21.201
Two-Phase Commit	15.690	27.287

Table 1. Protocol 1 - CDS *addRecord* Processing Times

Area	**Percentage**
Axis	38.080%
Network	42.190%
Database	1.489%
Certificates	25.537%
Two-Phase Commit	32.867%

Table 2. Protocol 1 - CDS *addRecord* Processing Percentage (Graph)

Area	Total Time (min)	Avg. Time (ms)
Axis	18.179	31.615
Network	20.141	35.027
Database	0.711	1.236
Certificates	12.191	21.201
Two-Phase Commit	15.690	27.287

Table 3. Protocol 2 - CDS *addRecord* Processing Times

Area	Percentage
Axis	38.080%
Network	42.190%
Database	1.489%
Certificates	25.537%
Two-Phase Commit	32.867%

Table 4. Protocol 2 - CDS *addRecord* Processing Percentage (Graph)

Area	Total Time (min)	Avg. Time (ms)
Axis	18.179	31.615
Network	20.141	35.027
Database	0.711	1.236
Processing	12.191	21.201
Encryption	15.690	27.287

Table 5. Protocol 1 - CA *genCertificate* Processing Times

5.4 Security Analysis

The robustness of the system's security and access controls will be analyzed by identifying common attacks and possible attacks on the new system and how the system would identify, prevent and/or protect against them. These attacks can be either internal or external and can be directed against user authentication, user authorization, and data integrity.

Area	Percentage
Axis	38.080%
Network	42.190%
Database	1.489%
Processing	25.537%
Encryption	32.867%

Table 6. Protocol 1 - CA *genCertificate* Processing Percentage (Graph)

Area	Total Time (min)	Avg. Time (ms)
Axis	18.179	31.615
Network	20.141	35.027
Database	0.711	1.236
Processing	12.191	21.201
Encryption	15.690	27.287

Table 7. Protocol 2 - CA *genCertificate* Processing Times

Area	Percentage
Axis	38.080%
Network	42.190%
Database	1.489%
Processing	25.537%
Encryption	32.867%

Table 8. Protocol 2 - CA *genCertificate* Processing Percentage (Graph)

Measure	Total Time (ms)	Overhead
Certification	24000	25%
Two-Phase Commit	12000	15%
Crash-Recovery	24000	25%
Certification & Two-Phase Commit	36000	40%
Certification & Crash-Recovery	48000	50%
Two-Phase Commit & Crash-Recovery	36000	40%

Table 9. Protocol 1 - Additional Measures Overhead

Measure	Total Time (ms)	Overhead
Certification	24000	25%
Two-Phase Commit	12000	15%
Crash-Recovery	24000	25%
Certification & Two-Phase Commit	36000	40%
Certification & Crash-Recovery	48000	50%
Two-Phase Commit & Crash-Recovery	36000	40%

Table 10. Protocol 2 - Additional Measures Overhead

Area	**Protocol 1 (ms)**	**Protocol 2 (ms)**	**Pct. Difference**
Axis	38.080	40.0	+1.1%
Network	42.190	35.0	+0.5%
Database	150	40.0	-100%
Certificates	25.537	25.0	-0.01%
Two-Phase Commit	32.867	34.0	+5%
Total	300	180	-150%

Table 11. Comparison of Protocol 1 and Protocol 2 Total Times

6 Conclusions and Future Work

...

** Future Work:

* Limited Clients Application.

* Reports Generator Web Application.

* Batch-mode record processing and certificate generation.

* Change records storage from a single records table to a records table per form or a records table per template.

* Scalable servers, Peer-to-Peer Clients/Servers

* Fault-Tolerance

* Cryptographic Strength and Performance Analysis (RSA key size, encryption algorithm, etc.).

REFERENCES

[1] Allinson, C., *Audit Trails in Evidence - A Queensland Case Study*, J. Info. Law Tech., March 2002; www2.warwick.ac.uk/fac/soc/law/elj/jilt/2002_1/allinson/.

[2] Chen, P.P., *The Entity-Relationship Model - Toward a Unified View of Data*, ACM Transactions on Database Systems (TODS), vol. 1, no. 1, pp. 9-36, March 1976.

[3] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S., *Web Services Description Language (WSDL) Version 1.1*, W3C Note, March 2001; www.w3.org/TR/wsdl.html.

[4] Codd, E.F., *A Relational Model of Data for Large Shared Data Banks*, Communications of the ACM, vol. 13, no. 6, pp. 377-387, June 1970.

[5] Common Criteria Implementation Board, *Common Criteria for Information Security Evaluation*, Version 2.1, 1999; csrc.nist.gov/cc.

[6] Food and Drug Administration, U.S. Department of Health And Human Services, *21 CFR, Part 11, Electronic Records; Electronic Signatures*, Federal Register, vol. 62, no. 54, pp. 13430-13466, March 1997.

[7] Freier, A., Karlton, P., and Kocher, P., *The SSL Protocol Version 3.0*, Internet-Draft, November 1996; wp.netscape.com/eng/ssl3/draft302.txt.

[8] FVC 2002 web site; bias.csr.unibo.it/fvc2002.

[9] Geer, D., *Taking Steps to Secure Web Services*, Computer, vol. 36, no. 10, pp.14-16, October 2003.

[10] Gil, Y., Ahn, D., Pan, S., and Chung, Y., *Access Control System with High Level Security Using Fingerprints*, Proceedings of the 32nd Applied Imagery Pattern Recognition Workshop 2003 (AIPR'03), pp. 238-243, October 2003.

[11] He, J., and, Wang, M., *Cryptography and Relational Database Management Systems*, 2001 International Symposium on Database Engineering & Applications, pp. 273-284, July 2001.

[12] Housley, R., Ford, W., Polk, W., and Solo, D., *Internet X.509 Public Key Infrastructure - Certificate and CRL Profile*, RFC 2459, January 1999; www.ietf.org/rfc/rfc2459.txt.

[13] Jain, L.C., Halici, U., Hayashi, I., Lee, S.B., and Tsutsui, S., *Intelligent Biometric Techniques in Fingerprint and Face Recognition*, CRC Press LLC, 1999

[14] Jonsson, J., and Kaliski, B., RSA Laboratories, *Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1*, RFC 3447, February 2003; www.ietf.org/rfc/rfc3447.txt.

[15] Labcompliance, *Electronic Records and Signatures - FDAs 21 CFR Part 11*, October 2003; www.labcompliance.com/e-signatures/overview.htm.

[16] Mercuri, R., *On Auditing Audit Trails*, Communications Of The ACM, vol. 46, no. 1, pp. 17-20, January 2003.

[17] Oasis Security Services Techincal Committee, *SAML V2.0 OASIS Standard Specification Set*, March 2005; www.oasis-open.org/committees/security.

[18] Oasis Web Services Security Technical Committee, *Web Services Security (WS-Sec) Protocol*, January 2004; www.oasis-open.org/committees/wss.

[19] Ramakrishnan, R., and Gehrke, J., Database Management Systems, 3rd. Edition, International Edition, McGraw-Hill, ch. 2, pp. 25-27, 2003.

[20] Schneier, B., and Kelsey, J., *Secure Audit Logs to Support Computer Forensics*, ACM Transactions on Information and System Security, vol. 2, no. 2, pp. 159-176, May 1999.

[21] Sesay, S., Yang, Z., Chen, J., and Xu, D., *A Secure Database Encryption Scheme*, Second IEEE Consumer Communications and Networking Conference 2005 (CCNC 2005), pp. 49-53, January 2005.

[22] Sims, D., *Biometric Recognition: Our Hands, Eyes, and Faces Give Us Away*, IEEE Computer Graphics and Applications, vol. 14, no. 5, pp. 14-15, September 1994.

[23] Sousa, J.P., and Mendona, J.M., *Intranet Security: An Increasing Concern in Industrial Environments*, Proceedings of the IEEE International Symposium on Industrial Electronics 1997 (ISIE'97), vol. 1, pp. 35-38, July 1997.

[24] Stallings, W., Cryptography and Network Security - Principles and Practices, 3rd. Edition, Prentice-Hall, ch. 9, pp. 259-262, and ch. 14, pp. 419-422, 2003.

[25] U.S. Congress, Help America Vote Act of 2002, Conference Version of Draft Bill, October 2002; www.acm.org/usacm/Legislation/ElectionReformConference.pdf.

[26] U.S. Department of Commerce - National Bureau of Standards, *Data Encryption Standard* (DES), FIPS PUB 46, January 1977.

[27] UDDI Spec Technical Committee, *UDDI Version 3.0.2*, October 2004; uddi.org/pubs/uddi_v3.htm.

[28] Uludag, U., Pankanti, S., Prabhakar, S., and Jain, A.K., *Biometric Cryptosystems: Issues and Challenges*, Proceedings of the IEEE, vol. 92, no. 6, pp. 948-960, June 2004.

[29] XML Protocol Working Group, *Simple Object Access Protocol (SOAP) Version 1.2*, W3C Recommendation, June 2003; www.w3.org/TR/soap/.

APPENDICES

1 Relational Schema

** Update this section with new SQL scripts

```
Forms (
    form_id : int, //auto-increment
    locked : int, // 0 if from not locked, 1 if it is
    template_name : varchar(50), //from Templates
    version : varchar(50) //from Templates
);

Templates (
    template_name : varchar(50),
    version : varchar(50),
    description : varchar(150),
    num_rows : int
);

Fields (
    field_id : int, // auto-increment
    name : varchar(20),
    format : varchar(25),
    c_type : varchar(25),
    f_type : varchar(25),
    template_name : varchar(50), //from Templates
    version : varchar(50), //from Templates
    f_position : int
);

F_Rows (
    form_id : int, //from Forms
    row_position : int
);

Records (
    form_id : int, //from Forms
    field_id : int, //from Fields
    record_id : int,
    row_position : int, //from Rows, -1 if a Header field
    username : varchar(10), //from Users
    r_value : varchar(200),
```

```
    r_time : datetime,
    certificate : binary (64),
    meaning : varchar(25)
);

Users (
    username : varchar(10),
    firstname : varchar(25),
    lastname : varchar(25),
    usertype : varchar(40),
    passhash : int,
    bir : binary(1024)
);

CARecords (
    form_id : int,
    field_id : int,
    f_type : varchar(25),
    record_id : int,
    row_id : int
);
```

2 Detailed Processing Times

** Explain in more detail the processing and access time division...

3 Full Experimental Results

** Include the results obtained in each repetition