A Novel Encrypted Database Technique to Develop a Secure Application for an Academic Institution

[1]Syed S. Rizvi, [2]Aasia Riasat, [3]Mustafa A. Khan and [4]Khaled M. Elleithy

[1, 3, 4]Computer Science & Engineering Department, University of Bridgeport, Bridgeport, CT

[2]Computer Science Department, Institute of Business Management, Karachi, Pakistan

srizvi@bridgeport.edu[1], aasia.riasat@iobm.edu.pk[2], mustafak@bridgeport.edu[3] ,elleithy@bridgeport.edu[4]

Abstract— **This paper presents the implementation of a secure application for an academic institution that offers numerous services to both students and the faculty. The primary focus of this paper is to provide a technical implementation of a new architecture for encrypting the database. The scope of this paper mainly includes but is not limited to symmetric and public-key cryptography, authentication, key management, and digital signatures. The final results of this paper demonstrate that what security features one should implement in order to achieve a highly secured application. This paper presents the implementation of a stand alone system that can be implemented on any legacy systems, and still operates effectively. In other words, it is self sufficient in terms of the data that it stores.**

I. Introduction

Some of the major services that the intended application offers to both students and the faculty are as follows:

- The intended application is flexible in a sense that it gives ability to add/delete users, courses, students, and documents.
- Flexibility to change passwords. The secure application provides highly transparent environment to its users (i.e., the students and the faculty can use this application in a highly transparent manner). There should be minimal input from the user due to security features.
- One of the key features that the proposed application offers is the "forgotten passwords". In other words, the secure application makes sure that if a user forgets his/her password, they should not completely lose their documents.
- In addition, the proposed application ensures that an administrator should not be able to decrypt the documents.
- Finally we design and develop this secure application by assuming that the communication is not secure at all.

Some of the security measures that we consider during the design and development of the targeted secure applications are as follows: Log all accesses activities to the server and provide features in the secure application to search for unusual access patterns. If possible, put an upper limit on the number of document that a single user can access or we should have a warning mechanism in the application to ensure fairness. Our secure application should have a permission system to the document that determines if a user is permitted to access it. If the documents are read-only, add a software application called "Secure Viewer" that never stores the document to disk. A user should also have the capability to add a specialized crypto board on the server. This crypto card would be used to encrypt/decrypt files on the server.

One of the major objectives of the targeted secure application is to provide secure storage of the faculty documents as well as maintaining authorized access to the documents for the authorized users. In order to maintain this level of security, there is a need to design a strong and secured application that let the documents of the faculty being kept secret by implementing data Integrity and confidentiality as well as making the documents partially shared or available. Our design approach, therefore, implements a complete line of defensive authentication and authorization cryptographic standards to protect the data and to maintain its integrity while at the same time making it available for the authorized users. In particular, in order to design and implement such a secured application, the following are the minimum key security-elements that should be addressed by us in this paper: User authentication and Authorization [1, 2], ACL Management & Access Availability [3], Data encryption and decryption [4, 5], Data integrality, and Document Accountability [2]. Fig. 1 shows the implementation of the above five security components for both faculty as well as the student-users. Furthermore, the class diagram as shown in Fig. 2.

II. Components Of The Proposed Architecture

A. User Authentication and Authorization

The secure application is certainly required to employ a strong mechanism to authenticate the users. The most frequently used strategy is asking for a user name and

[1]Contact author: srizvi@bridgeport.edu,

T. Sobh et al. (eds.), *Novel Algorithms and Techniques in Telecommunications and Networking,*
DOI 10.1007/978-90-481-3662-9_50,

password to authenticate the user. Some key points that we should consider in the design of authentication mechanism are: transmitting the password in clear (i.e., we may use SSL to protect the user privacy and to safe the application by being played in the hand of some intruder after he capture the network traffic and thus get the password). Also, it is required that the secure application provides secure storage of the user names and passwords along with a method to manage them, including resetting or revoking the passwords or user accounts. Our another important concern during the preliminary design of secure application is whether to store the password in some hash format or storing it in the plain text format as the user entered. In order to keeps the user confidentiality intact and also letting the password to become non human understandable, hashing, therefore, becomes one of our design choices.

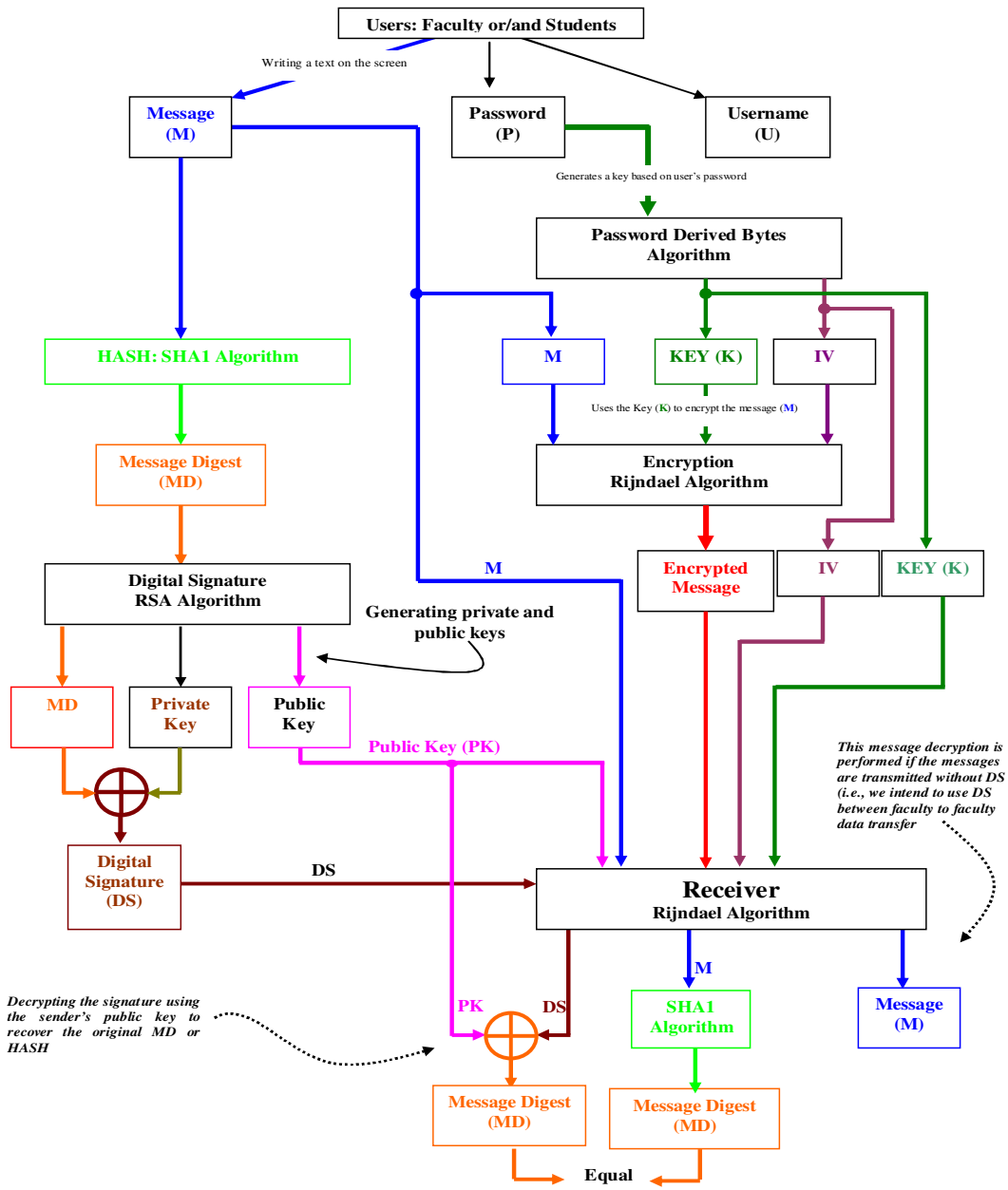

Fig. 1. Proposed Architecture for combining various security features for the intended application

B. ACL Management & Access Availability

One of the requirements of the secured application is making information always accessible to users who need it and who have sufficient permissions to access it. In order to achieve this task, the design of secure application should provide a robust mechanism to perform good management of document creation and access rights settings. Our secure application provides a number of features that, for example, allow owner to easily create and modify the documents, choose the encryption technique available in the secure application to store the document in encrypted format and most importantly setting the access control lists ACL. An owner can specify the objects and the accessibility domains associated.

C. Date Encryption and Decryption

The design of a secure application is not possible without the use of some encryption and decryption techniques. The secure application, therefore, should employ encryption and decryption technique for controlling the document integrity and accessibility. The advantages of symmetric key cryptography make our design choice rather straightforward. However, since both parties need the same key for effective communication to occur, key distribution becomes an issue. For our secure application, the encryption will takes place at the server where as the keys can be generated by the owner of the files entering some text. In addition, if file gets corrupted, the owner should be able to produce the same set of the keys if needed. The keys can be stored in encrypted format on the secured server, while just the server side application can access the file that contains the set of all keys that are used to encrypt the documents.

D. Data Integrity

Data integrity is one of the issues that we consider during the development phase of our secured application. The task is to make files secure by completely denying unauthorized access to the files while at the same time make sure that the files should be modified only by those (student or faculty) who are authorized to do so (If any) or can not be modified other than the owner of the document. We implement the

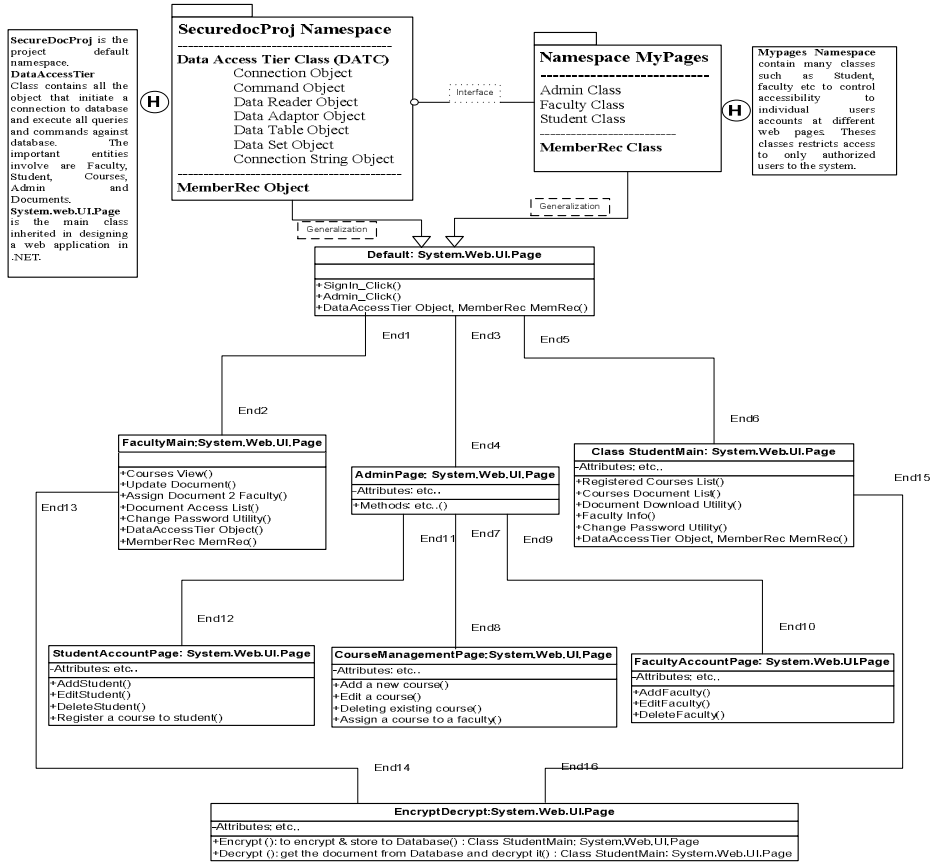

Figure 1. Class Diagram of a Secure Application for an Academic Institution

Fig. 2. Class diagram for the implemented project

concept of digital signatures that enable recipients to verify the integrity of an electronic document that is used. In our application, we ensure that the data integrity is maintained after implementing the digital signatures. One way of implementing this concept is the use of a one-way hash that creates a fixed-length hash value or message digest for a message of any length. With a hash attached to the original message, a user or owner can determine if the message was altered by re-computing the hash and comparing his or her answer to the attached hash.

III. Implementation Issues and Design Choices

In this section, we present our overall design structure for the targeted application. In addition, this section provides a comprehensive discussion on implantation issues and our design choices for implementing each security component we discussed above. The detailed flow diagram of the proposed project is shown in Fig. 3.

A. *Project Design Phase*

The Secure Document application is designed and developed to implement the security features that we have learned during this course of Network Security. This project was built using the .Net Framework and coded using C# as the base language. The main tools used in the project are: Visual Studio 2003 Development Environment, Asp .Net Framework, MS Access (Database) for storing projects entities and Documents, ADO.NET for database connectivity, Internet Information Server IIS 5.0 (web server), Secure Http (https). The database was designed in a way that it would suit the application flow and all the entities of the application. Next we describe each of them in some detail along with the over all database design. For the sake of simplicity, all the entities information are kept simple in database, although this information can be made comprehensive and complete in any real time implementation and as per the development requirements.

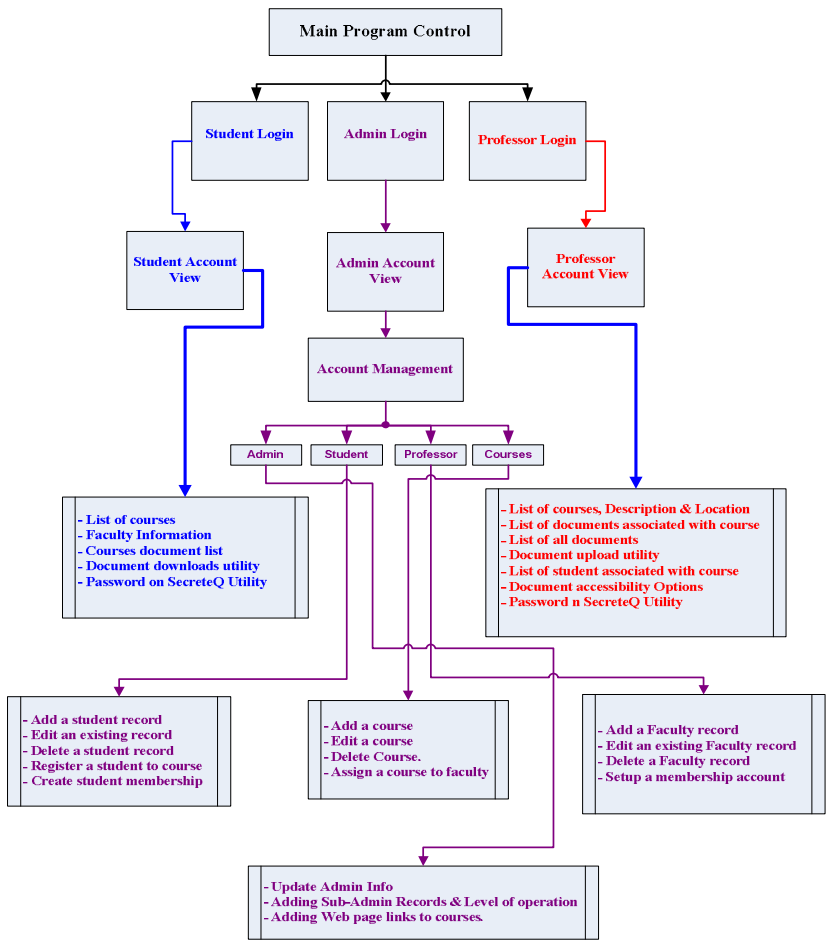

Fig. 3. Detailed Flow Diagram of Secure Application for an Academic Institution

B. *Proposed Security Design*

The proposed security design includes various security measures that are incorporated in the intended application.

1) Custom Base Class: In our project we have used ASP.NET Custom Base Class feature to secure access to all the project web pages, data and services available on them. For this purpose, we created custom base class called "My Pages" which is derived for System.Web.UI.Pages and consists of those classes that contain the code that put the security checks and take care of the process of authorization. All the web form's codes behind classes are derived from the Custom Base Class that provides the basic infrastructure for the web page's information access security. To implement this hierarchy, we implemented the .Net's most prominent feature: session management to maintain the user's identity at each step of the application. By using the custom based class implementation, we have avoided the URL spoofing in which a person who is not authorize to view the page contents or to access the resources offered by it can be able to access the page's contents

2) Dynamic Key Generation and Management: In order to prevent the unauthorized access to the keys that are used to secure the documents upon storage, the keys for encryption and decryption are chosen entirely at run time. With this approach, we avoided to store them at any place which consequently avoided any security threats. The system will be a bit slow in the response but will save us the cost of being insecure. The keys are generated based on the session objects information of the person which is being signed at the time of the document upload and encryption request.

C. *Basic Concepts Design and Flow Diagram*

The basic concept includes the users, custom, validation and calendar controls. Validating the user inputs throughout the pages include telephone number and date information. Updating the database based on the calendar when the user specify the date. The retrieved information from the database is displayed using data adapters, data sets, data grids and data list. The main tools used as a basic concept in .Net framework are: User Controls, Image Controls, Html File Control, Data List, Data Grid, Calendar Controls, Validation Controls, Regular Expressions, Data Readers, Data Adapters, and Data Sets. The data flow diagram is a high level representation of this project. It can be seen in Fig. 3 that the data flow from top to bottom where system administrator initiates and introduces students, courses, and faculty.

IV. Secure Document Application Implementation

In this section, we present a discussion on the technicalities we encountered during the development phase of this project. This includes implementation detail and interface choice. In this application, the flow of the application starts from the main (default) page where a person sign in and then based on its role or membership, he/she will be then directed to specific web pages and resources he can access. The main entities in this implementation include System Admin interface and Faculty and Student Interfaces as outlined below.

A. *System Admin and Faculty Interface*

The system admin interface contains the links to the pages where a system admin can perform course management, faculty & students accounts managements In addition, a system admin can assign courses to faculty and can register students to specific courses. The links at the system admin interface include faculty accounts, admin accounts, student accounts, courses management interfaces. System administrator manages student's accounts by adding, modifying, deleting student record. He/she can setup their login accounts and can register them to the desired courses

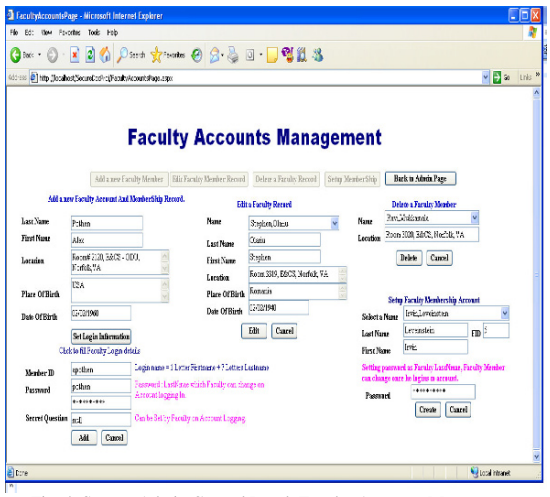

Fig. 4. System Admin Control Panel: Faculty Accounts Management

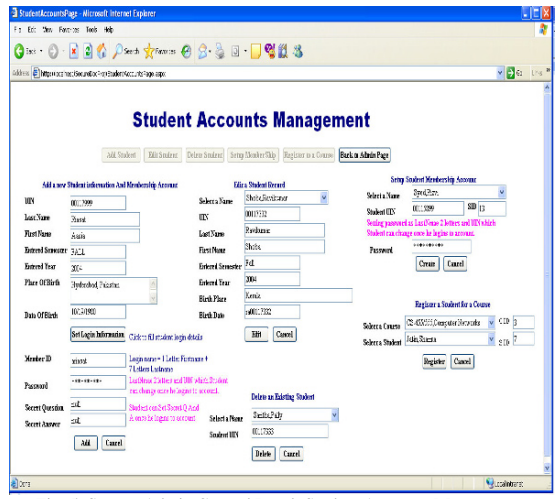

Fig. 5. System Admin Control Panel: Student Accounts Management

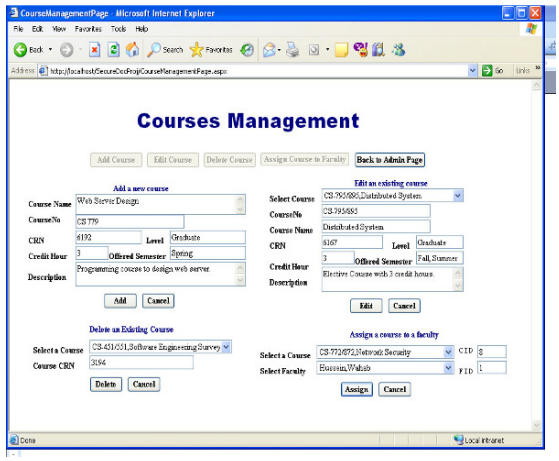

Fig. 6. System Admin Control Panel: Courses Management

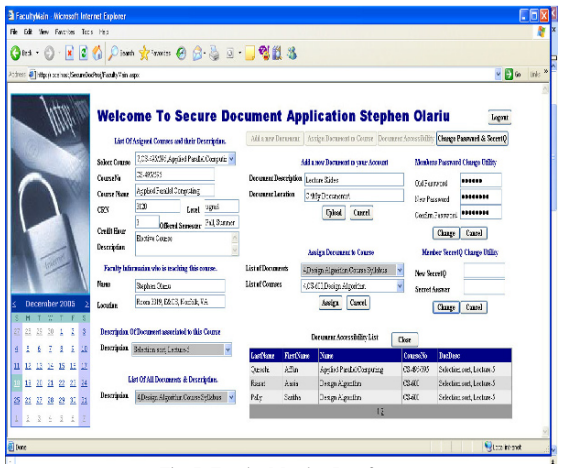

Fig. 7. Faculty Member Interface

offered by a certain semester. Figures 4, 5, and 6 show the different parts of the system administration.

When a faculty member logs in to the application, he/she is directed to a web page that provides the information and services that are only related to that faculty member. As one can see in Fig. 4, the faculty member has provided the information regarding the courses that are assigned to him and the documents (encrypted) that he has in his folder at the server. In addition when a course is selected, the page shows the documents that are related to that specific course. The list of students who have given the access to his (faculty member) documents are also shown here. The faculty member has given the option to change the accessibility permissions of the student by deleting the student record form the list for whom he doesn't want to allow the accessibility of the document. The documents are uploaded to the server in encrypted format and then stored into the data base as a BLOB. During the uploading and encryption, the secure Http Protocol is being used, so that the transfer of the documents takes place securely as shown in Fig. 7. In addition, Fig. 6 can be used by a system administrator to manage the courses for both faculty and students.

B. *Student Interface*

When a student logs into the secure document application, he has shown the list of his registered courses and their complete description including faculty information (see Fig. 5). He can choose any of the documents that he want to access and can click the download button. The download button extract the document that are stored in the database in the BLOB form and then decrypt it on to the-sever; finally the document is made available in the browser for the student. During the document transfer we again implemented secure Http protocol to securely transfer the document. The details are shown in Fig. 5. On the same page student can change his password or secret question and answer any time. Passwords and secret questions and answers are stored in the encrypted format in the database and hence.

V. Conclusion

In this paper, we presented a new design for providing comprehensive security for a secure application by combining many different security techniques using the .NET framework. The most prominent feature of the .NET is its full fleshed Cryptography-API that provides techniques of encryption and decryption while hiding all the technical details. This is one of the main reasons that we achieved the goal of completing this secure application. Secure HTTP communication provided by ASP.NET's API is also another most important and handy feature worth to mention here. Some of the tools used in the application include data access controls that avoid repetitive database programming, built in authentication features and security controls that enable automated management of user accounts and roles and simplified web deployment. The proposed project consists of different tools and techniques for building secure web applications with strong database accessibility and crypto graphic techniques. During the design phase, we learned and practiced many new techniques that we found very useful and interesting in the context of building a secure and powerful web application along with strong and real time database functionality.

References

[1] L. Moningi, "Authentication and Authorization in ASP.NET," September 09, 2003. Available at: http://www.c-sharpcorner.com/mrsharp.asp

[2] D. Watkins, "An Overview of Security in the .NET Framework," Project 42, Sebastian Lange, Microsoft Corporation. January 2002.

[3] J. Meier, A. Mackman, B. Wastell, P. Bansode, A. Wigley, K. Gopalan, "Security Practices: ASP.NET 2.0 Security Practices at a Glance," Microsoft Corporation, August 2005.

[4] A. Yao, "How to generate and exchange secrets," *Proceedings IEEE 27th Symposium on Foundations of Computer Science (FOCS)*, pp. 162–167, 1986.

[5] ISO/IEC 11770-2: 1996. "Key management - Part 2: Mechanisms using symmetric techniques," *International Organization for Standardization*, 1996.