# IMPROVING OF ONLINE BANKING SECURITY

Zin Zin Maw, Khin Than Mya
University of Computer Studies, Yangon
zin2maw@gmail.com

## *ABSTRACT*

*XML (Extensible Markup Language) is a general-purpose specification for creating custom markup languages. XML Encryption provides end-to-end security for applications that require secure exchange of structured data. XML-based data transfer systems are useful in nowadays. In recent years, a technology emerged that made use of XML metadata itself to encrypt XML elements directly into a different XML document. This paper explores the use of XML encryption with AES in securing web data and user authentication check with Cookie authentication that requires a certain level of privacy using the example of an online banking system.*

*Keywords*: XML Encryption, online banking, encryption, AES, RSA, Cookie

## 1. INTRODUCTION

In its primitive days, limiting unauthorized access to high security data on the web was an unreliable process. Banking systems in those days had to rely on existing security technologies provided by the network level implementation and content encryption. A technology that supports the flexible nature of XML metadata was lacking at the moment.

XML data to by written and read by ordinary web servers and clients using web technologies [7].If well implemented, the same methods and techniques can be employed on different systems that make heavy use of secure data to transport necessary data. Such practices offer a fairly reliable level of security while retaining the usefulness of XML.

The system of this paper accompanies is developed as a web-based banking system that exchange the data between the server and clients in the form of XML document. In the process of encrypting the documents, several concepts of

XML encryption are demonstrated and implementation methods such as AES(128), RSA and cookies-based security are highlighted.

## 2. RELATED WORK

Thales introduced WebPIN uses in online banking security. WebPIN had three primary functions which are user authentication based on PIN verification from 3DES, message authenticity based on the use of 3DES ANSI MACS are sent from the client to web server and data privacy based on the use 3DES ECB or CBC encryption of message sent to received from web server [8]. In the proposed system, user authentication is base on password encryption with RSA algorithm and cookie authentication. Data integrity is based on XML encryption with AES 128.

## 3. BACKGROUNG THEORY

This paper is based on XML encryption for data integrity with advanced encryption standard(AES) also known as Rijndael and RSA algorithm.

### 3.1 XML Encryption

XML Encryption, also known as XML-Enc, is a specification, governed by a W3C recommendation, that defines how to encrypt the contents of an XML element.

Although XML Encryption can be used to encrypt any kind of data, it is nonetheless known as "XML Encryption" because an XML element (either an EncryptedData or EncryptedKey element) contains or refers to the cipher text, keying information, and algorithms.

Both XML Signature and XML Encryption use the KeyInfo element, which appears as the child of a SignedInfo, EncryptedData, or EncryptedKey element and provides information to a recipient about what keying material to use in validating a signature or decrypting encrypted data. The KeyInfo element is optional: it can be attached in the message, or be delivered through a secure channel [4].

## 3.2 Advanced Encryption Standard (AES)

Advanced Encryption Standard (AES) is an encryption standard adopted by the U.S. government. The standard comprises three block ciphers, AES-128, AES-192 and AES-256, adopted from a larger collection originally published as Rijndael.

The Rijndael cipher was developed by two Belgian cryptographers, Joan Daemen and Vincent Rijmen, and submitted by them to the AES selection process [1].

The AES cipher is specified as a number of repetitions of transformation rounds that convert the input plain-text into the final output of cipher-text. Each round consists of several processing steps, including one that depends on the encryption key. A set of reverse rounds are applied to transform cipher-text back into the original plain-text using the same encryption key.

### 3.2.1 AES Description

* KeyExpansion using Rijndael's key schedule
* Initial Round
1. AddRoundKey
* Rounds
1. SubBytes—a non-linear substitution step where each byte is replaced with another according to a lookup table.
2. ShiftRows—a transposition step where each row of the state is shifted cyclically a certain number of steps.
3. MixColumns—a mixing operation which operates on the columns of the state, combining the four bytes in each column
4. AddRoundKey—each byte of the state is combined with the round key; each round key is derived from the cipher key using a key schedule.

* Final Round (no MixColumns)
1. SubBytes
2. ShiftRows
3. AddRoundKey

## 3.3 RSA

In cryptography, RSA is an algorithm for public-key cryptography. RSA is widely used in electronic commerce protocols, and is believed to be secure given sufficiently long keys and the use of up-to-date implementations[5].

### 3.3.1 RSA Algorithm

```
RSA Key generation
{
    Select two primes p and q such that p!=q;
    n = pxq
    Ø(n )=  (p-1)x(q-1)
    Select e such that 1<e< Ø(n) and e is
    compromise to Ø(n)
     d = e^-1 mod Ø(n)   //d is inverse of e
                          modulo Ø(n)
    Public Key  = (e,n)
    Private Key = d
    Return Public Key and Private Key
}
RSA Encryption(P,e,n)
{
    C = Fast_Exponentiation(P,e,n)
   return C
}
RSA Deceryption(C,d,n)
{
    P = Fast_Exponentiation(C,d,n)
    return P
}
```

## 3.4 Java Architecture for XML Binding

Java Architecture for XML Binding (JAXB) allows Java developers to map Java classes to XML representations. JAXB provides two main features: the ability to marshal Java objects into XML and the inverse, i.e. to unmarshal XML back

into Java objects. In other words, JAXB allows storing and retrieving data in memory in any XML format, without the need to implement a specific set of XML loading and saving routines for the program's class structure [9].

JAXB is particularly useful when the specification is complex and changing. In such a case, regularly changing the XML Schema definitions to keep them synchronized with the Java definitions can be time consuming and error prone [6].

## 4. Overview of the System Design

Online banking system is widely used in today technology, systems that control the security of the online banking system is required. The sequence diagram of the proposed system is shown in Figure 1. In the proposed system, user needs to register at online banking sever. When each user (client) requests to register at the banking server, the server set cookies on customer browser. Then, the client sends banking information in XML format which is encrypted with AES-128 algorithm. Banking server decrypts the received encrypted XML data and stores it in its database. And then it acknowledges the client about the completion of registration.

When logging into the banking server, the client needs to send username and encrypted password. The password is encrypted with RSA algorithm. The server compares the decrypted password with the password stored in its database. It also authenticates cookies. If the verification is valid, the client can request to see banking information. Server gives banking information to user in an encrypted XML file format. The client has to decrypt it first and can make online banking transaction after that. When the client requests to log out, server sets a new cookie for the next login.

The banking system can be divided into two parts. All the banking operations and encryption tasks are carried out on the server. The client and server communicate by encrypting data. The server executes tasks such as creating users, depositing money and withdrawing money from the bank accounts.

The Encryption process can be divided into two parts. The first part is the process of encrypting the cipher information in the XML file. The cipher information itself is visible in the XML file. So encrypting this information is necessary. This is done by using AES 128 algorithm. The resultant key is used the encrypted the cipher info

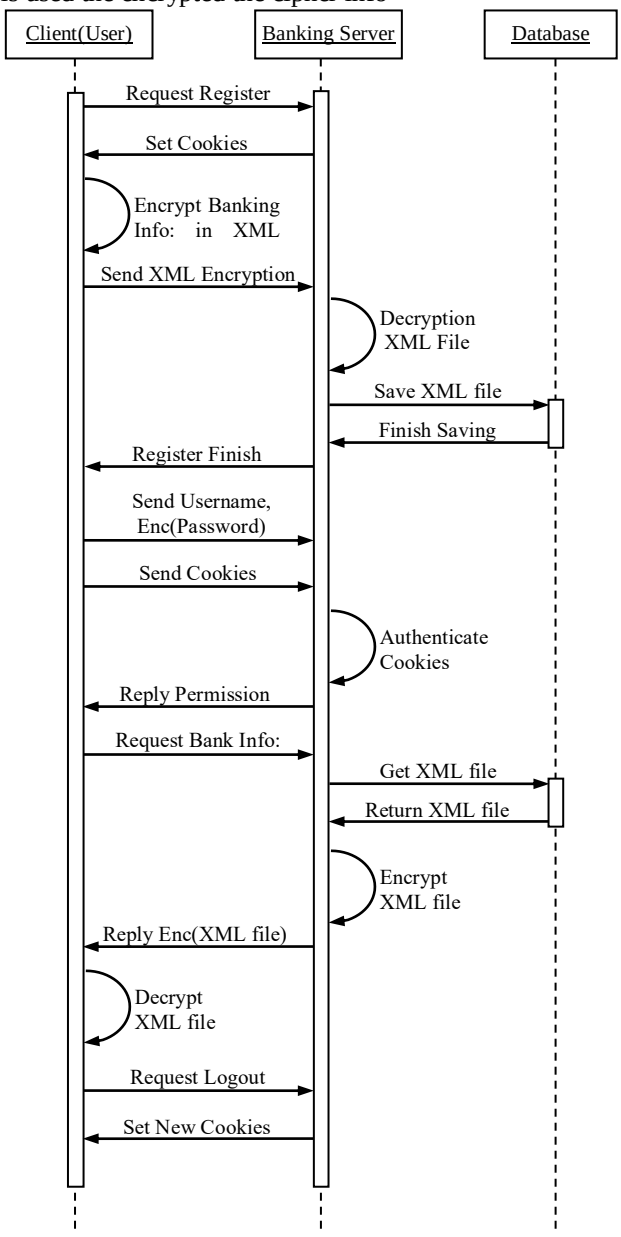


Figure 1. Sequence diagram for Proposed System

in the XML file. Encryption in the XML uses AES algorithm to encrypt the actual XML content. After

generating the key and encrypting the XML content, this information is encrypted so that only the receiver who has this key of the key will be able to get the information.

## 4.1 Step By Step Process Of The System

There are nine stages involved in online banking security system. The first three step is for user authentication and the left for sending encrypted XML files between the server and clients.

- Key Generation

Public and private keys are generated using RSA algorithms. Java implementations of RSA outlined in Theoretical Background section are used to generate necessary keys. The keys are kept for transmission to clients using cookies mechanism and for XML Encryption.

- Password Authentication

When user is send encrypted password with its own private key. Server decrypts password with public key and authenticates this password from web browser with password from server database. Figure 2 is an illustration of password authentication.

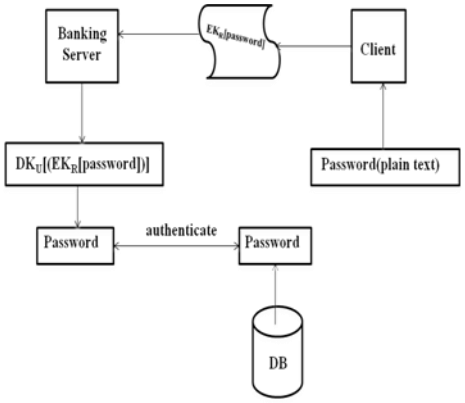


Figure 2. Password Authentication

- Cookie Authentication

Server sets cookie on user browser when user registers to banking server. Cookie store security information, AES 128 key. Cookie is a good place for saving security information. Cookie set after first log in cookie check on second login. Immediately after second login cookie updated. On third login verify using updated cookie.

- Marshalling

All the data structures used in the banking system that has to be transported has to be converted into XML before any encryption can take place. In Figure 3, the end format is XML and Java objects will be directly mapped into corresponding XML elements using JAXB, marshalling.

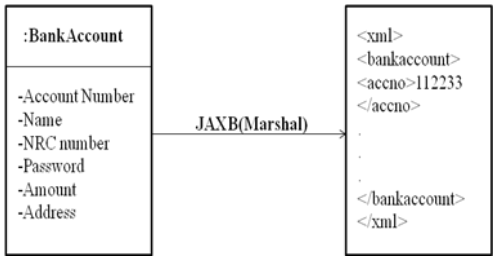


Figure 3. Marshal to XML file

- Encryption

These stage and the following stage are sending encrypted XML file from server to client. Java Cryptography Extensions implementations of Advanced Encryption Standards are used for encrypting XML contents. The resultant XML data is reformatted in a suitable XML encryption structure. The new format contains the keys and hashes for decrypting and validating XML data. Figure 4 show XML Encryption process.


Figure 4. XML Encryption Process

.

- Transport

The http cookie carries the public keys that can be used to decrypt the XML security data. The cookie is embedded in http header. The http body contains the XML document itself. Session enabled serves and uninterrupted http request are used to guarantee only the necessary portion of data is downloaded and uploaded.

- Decryption

Cookie in http header is extracted and the public key is read so that XML keys and hash data can be decrypted. Decryption of XML file is reversed of Encryption. XML contents perform necessary validations. Appropriate transformations put the XML document into its original structure.

- Validation

The security part of encrypted XML file contains information that can be used to check the data integrity of XML content after decryption. Java or JavaScript libraries are used to perform the necessary checks.

- Unmarshalling

The last stage is to convert the plain XML document into the form that can be represented in Java and JavaScript objects. The resultant objects are ready to be consumed by program entities.

## 5. Experimental Result

This system is implemented using Java programming language. This system was tested on the computer with Intel [R] Pentium(R) Dual CPU, 2.16GHz, 1.95GB of RAM. The time to do "Marshalling" "Encryption", "Decryption" and Unmarshalling" processes is shown in Table 1 and Table 2. In this tables describe the testing times result of 5 different XML file size.

Table 1. Time for Marshalling and Encryption process

| XML file size | Marshalling time | Encryption time |
|---|---|---|
| 5KB | 200ms | 500ms |
| 10KB | 250ms | 800ms |
| 15KB | 310ms | 1200ms |
| 20KB | 370ms | 1500ms |
| 25KB | 420ms | 1900ms |

Table 2. Time for Decryption and Unmarshalling process

| XML file size | Decryption time | Unmarshalling time |
|---|---|---|
| 5KB | 350ms | 180ms |
| 10KB | 600ms | 220ms |
| 15KB | 1000ms | 300ms |
| 20KB | 1200ms | 350ms |
| 25KB | 1600ms | 400ms |

The following Figure 5 and Figure 6 is draw for the values in Table 1and 2.

Time duration for marshalling and encryption in Table 1 as shown in the Figure 5.

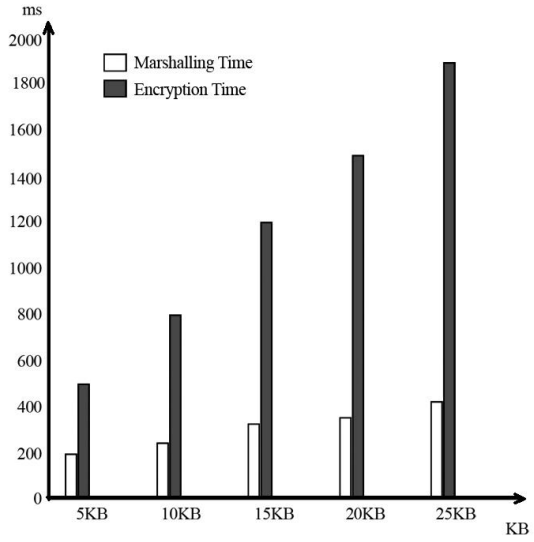


Figure 5. Time duration of Marshalling and Encryption

Time duration for decryption and unmarshalling in Table 2 as shown in the Figure 6.

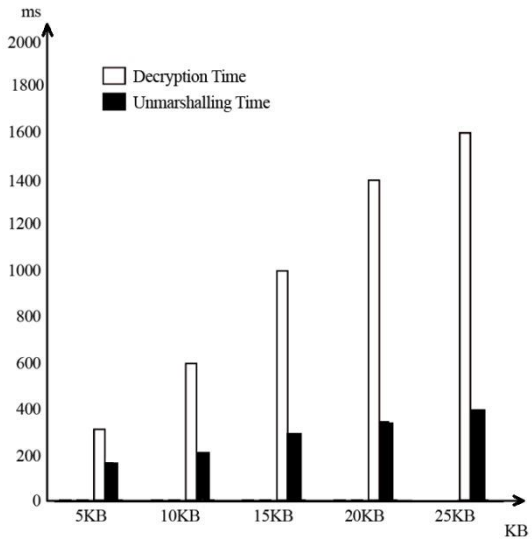


Figure 6. Time duration of Decryption and Unmarshalling

## 6. Conclusion

This system incorporates XML encryption, encryption algorithms such as AES 128 and RSA, http cookies and client-server web architecture. A simple banking system is developed to see these different technologies co-operating together. The resultant system makes it obvious that encryption at the XML level is possible and a cookie mechanism can be used to further strengthen the security provided by XML digital signature.

## References

[1] J.Daemen, V.Rijmen, *The Design of Rijndale: AES- The Advanced Encryption Standard*, Springar Verlag 2002, ISBN 3-540-42580-2

[2] J. Garms, D.Somerfield ,"Professional Java Security" http://www.amazon.com/exec/obidos/ASIN/18610

[3] Jim Hu, "XMl Encryption specs approved", http://news.cnet.com/2100-1023-976701.htm

[4] T. Imamura, B. Dillaway, Ed Simon,"XML Encryption Syntax and Processing", http://www.w3.org/TR/2002/PR-xmlenc-core/20021003

[5] P.Riikonen , "RSA Algorithm", http://iki.fi/priikone/docs/rsa.pdf

[6] E.Rusty Harold, "Java Network Programming" http://www.amazon.com/exec/obidos/ASIN

[7]Ed Simon, P.Madsen, C. Adams, "AnIntroduction to XML Digital Signatures", http://www.xml.com/pub/a/2001/08/08/xmldsig.html, 2001

[8] T.Thales, "Securing Internet Home Banking", http://www.thales-esecurity.com/WebPIN Internet Banking WP.pdf

[9] W3C Members, "XML Signature Syntax and Processing", http://www.w3.org/TR/xmldsig-core/, 2008