Enhanced TTP Architecture in Securing Web Service

Srinath K S[#1], **Prabhakar M**[#2]
Dept. of CSE,
[#]*Sambhram Institute of Technology,*
Bangalore-560097, India
[1]solansrinath@gmail.com
[2]laxmi.prabakar@gmail.com

Abstract-**Web Services use various security standards and protocols. The conventional frameworks applying XML Encryption use DOM-based XML processing. The DOM API is tree-based and therefore the whole document must be parsed before data can be encrypted or decrypted. XML encryption specifies how XML elements should be encrypted to provide message confidentiality. In this paper, a SAX based streaming for XML processing that reduces the memory consumption and execution time for parsing is used. DES is used for generating cipher and Base 64 encoding is used to encode the cipher in character format. A Trusted Third Party authority is also proposed to provide authentication using X.509 certificates. The system has been tested for different attack scenarios and the results found to be satisfactory.**

Keywords-**Web Services, Document Object Model, Simple API for XML, Data Encryption Standard, Trusted Thirty Party Authority**

I. Introduction

Web services help in improving the efficiency and flexibility of web communication. E-business provides service via internet and these services are described in XML format. The e-business applications attract attackers that can manipulate the back end of an application where all the personal data are stored. These data often include everything from credit card numbers to medical information. The firewalls and/or Intrusion Detection Systems (IDS) are used to protect date from attackers, but companies are unaware that their assets are exposed even through firewalls and IDS.

Authentication, authorization and auditing do not provide adequate security for the threat protection, information leakage and malicious activity. Most attacks on traditional web-based applications exploit security holes by sending requests that bypass authentication. This results in websites being brought down. Well known Web Application security attacks include: Structured Query Language **(**SQL) injection attacks, Uniform Resource Locator (URL) parameter tampering, URL directory traversals that attempt to bypass publicized URLs by accessing resources directly, and submission of URL strings that try to overwhelm the application by sending corrupt parameters (or queries) for unexpectedly large data sets. In this paper, the different attacks on web service are identified, and use Extensible Markup Language (XML) encryption technique to solve the attacking problem [1],[4],[8].

II. Problem Statement

In the web services, applications that communicate with external applications interchange sensitive data. The data may be read by unauthorized users by applying various attacking techniques such as data tampering, network eavesdropping, man-in middle and many more attacks. The consumer and provider of web services are not able to trust each other for sensitive transactions. This affects the authentication, authorization, integrity, non-repudiation, and confidentiality of data which reduces the quality of service. In this article, we address web security mechanisms such as stream encryption and decryption; and the Trusted Third Party (TTP) is used to validate the sender and the corresponding receiver for defeating the aforementioned attacks**.**

III. Design and Implementation

The following subsections give the detailed design patterns, issues in web service security and TTP system architecture.

A. Design Issues in Web Service Security

Designing the web service security presents interesting set of challenges. Web Service Security mechanisms such as authentication, authorization, data integrity and confidentiality play an important role in providing basic levels of Web Service security. The above mentioned attacking techniques reduce the level of security. Various mechanisms were explored to overcome such attacks, but these mechanisms are weak and could not provide high efficiency. Thus, a stream based encryption/decryption algorithm and trusted third party authentication needs to be developed to thwart the various attacks.

XML file is needed to be encrypted and decrypted using an appropriate encryption algorithm. Data Encryption Standard (DES) is a widely-used method of data encryption using a private (secret) key that is difficult to break. Both the sender and the receiver

© 2014 IJAIR. ALL RIGHTS RESERVED

must know and use the same private key. This scheme uses 56-bit key and is hard to break using brute-force or statistical analysis attacks. Further, XML data needs to be parsed before it can be further processed. The XML data may be in the form of Web Service Description Language (WSDL) or Simple Object Access Protocol (SOAP) message [9].

B. TTP System Architecture

The TTP web service architecture is based on Service Oriented Architecture (SOA) [6],[2]. The different entities involved are: Provider, consumer and registry. The WSDL files are uploaded by providers to the registry and the same being downloaded by the consumers. The SOAP based communication is carried out between the consumers and providers. Hence all these entities needed to be completely authenticated before any communication takes place. In this paper, the authentication mechanism is delegated to the TTP.

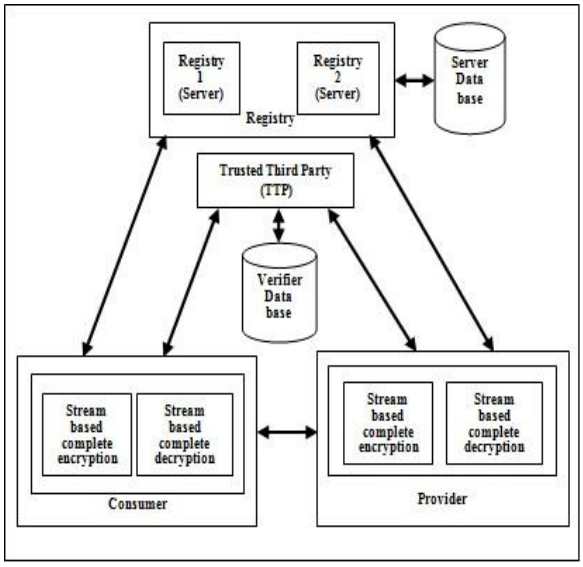


Fig 3.1: TTP Web security architecture

Fig 3.1 gives the architecture of proposed web service security. It includes four main entities: namely, Consumer, Provider, Registry and TTP. The provider develops service and there descriptions are written in WSDL file. This service is published in registry1 using WSDL file. Before sending the WSDL file, it is encrypted to secure the information. The registry1 stores the received WSDL file in registry database. The consumer looks for the appropriate service in registry2 and downloads the encrypted WSDL file from its database. The consumer decrypts the downloaded WSDL file to extract information about the service. The below steps are followed before establishing the communication between consumer and provider.

a) First consumer sends his ID to TTP and requests the certificate

b) TTP checks for authorization of consumer, and sends the requested certificate

c) After receiving the certificate, consumer sends it to provider

d) Now, provider verifies the certificate for authenticity

Similarly, provider authenticity is also validated.

1) *Provider*

A service provider describes its services using WSDL. This definition is published in registry of services. The registry may use Universal Description Discovery and Integration (UDDI) or other forms of directories. This entity has two modules: namely, stream based encryption and decryption. The WSDL file is encrypted before it is sent to registry to achieve integrity. The encrypted SOAP request from consumer is also decrypted at provider.

2)*Consumer*

A service consumer searches the registry to locate a service and determine how to communicate with it. The service consumer uses the WSDL to send a request to the service provider. Here, encrypted communication is used to avoid any data tampering. Further, service consumer sends encrypted SOAP requests to service provider directly. This entity also contains decryption module which decrypts incoming WSDL or SOAP response.

3)*Registry*

The UDDI registry is intended to eventually serve as a means of discovery of Web services described using WSDL. It is an entity that accepts and stores encrypted WSDL file from service providers and provides encrypted WSDL files to interested service consumers.

4)*Trusted Third Party (TTP)*

TTP is an entity which facilitates interactions between two parties trusting the third party. The Third Party reviews all critical transaction communications between the parties, based on the ease of creating fraudulent digital content. In TTP module, the relying parties use this trust to secure their own interactions. This entity has two modules, viz., *signer* module that issues digital identity certificate to both the parties and *verifier* module that validates the certificates.

© 2014 IJAIR. ALL RIGHTS RESERVED

IV. Implementation of Stream based Encryption and Decryption

The following few sections elaborate the important modules of our Web security architecture: stream based encryption and decryption.

A. Stream based Encryption

A stream based encryption technique encrypts stream of XML data to produce the corresponding cipher using DES algorithm. The key is a 56-bit representation of the character sequence specified by the user. Fig 4.1 shows the internal design of encryption module [3].

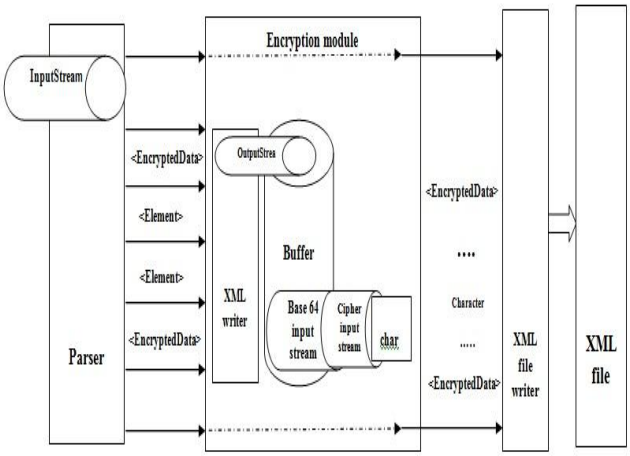


Fig 4.1: Internal design of the encryption module

A stream parser reads the input file and reports each parsing event as it occurs. These events are forwarded to encryption module. If the character sequence contains non-parsable data, it is encoded with Base-64 encoder and then encapsulated. This process is repeated until the end of the document is found. Subsequently, the final process of the internal writer module is initialized [10],[11],[5].

The output stream encrypts the incoming events using DES algorithm and Base-64 encoder encodes the incoming encrypted characters. All the output tags are written into the specified output file. The obtained cipher is embedded between <EncryptedData> and </EncryptedData> tags. Finally, the modified cipher is written to a XML file.

B. Stream based Decryption

A stream Decryption is a method of decrypting cipher to produce original text in which DES algorithm and key are applied to each event in a data stream. Fig 4.2, shows the internal design of decryption module. A stream parser reads the input, the encrypted XML file and reports the occurrence of the parsing events. The reported stream events are compared with tag name <EncryptedData>. If it is found, these events are forwarded to decryption module. The decryption module performs Base-64 decoding and then decrypts the parsed XML tags. It is assumed that the key is reliably communicated in private channel. The resultant XML tags are written into an output XML file [5],[10].

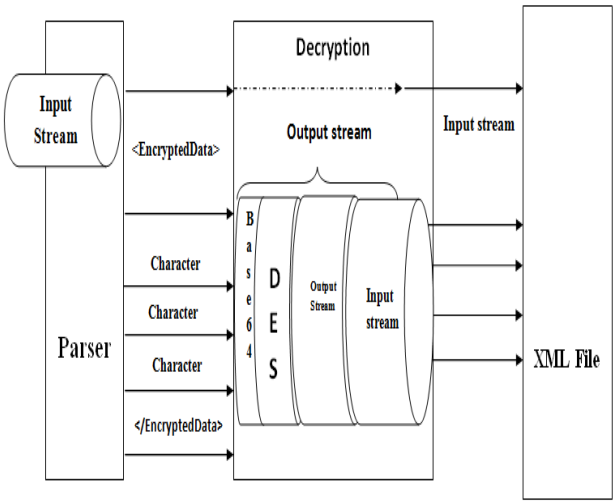


Fig 4.2: Internal design of the decryption module

C. Implementing SAX Parser

The simple API for XML (SAX) is capable to read XML data, but writing is not supported. A SAX parser reads the XML source and for every detected XML element (e.g. tags attributes, namespaces, etc.), it creates an event, so that a stream of events is generated while it is parsing.

D. TTP Signature generation and verification

The TTP performs validation and certification of consumer and provider. The transaction between consumer and provider occurs only after proper authentication. TTP ensures this authentication and enables a signature generation mechanism for certification and corresponding signature verification for conforming authentication. It reviews all critical transaction communications between them and any attempt for creating fraudulent digital content is reported, thereby securing the interactions. TTP contains two modules namely, *signature generator* and *signature verifier*.

The flowchart in Fig 4.3 shows the working of signature generator module. It reads the client ID and compare with the content of verifiers data base to check whether it is registered. If the client ID is registered, the signer module generates signature and public key in the following manner. Initially the signer module generates the pair of public key and

© 2014 IJAIR. ALL RIGHTS RESERVED

private key using Digital Signature Algorithm (DSA). It selects the private key to sign the client's data file. The data file is processed block-wise with the size of block equals to 1024 bytes. These bytes are signed using DSA and the signed data is written to a file (called as certificate). The public key is encoded using X.509 encoder. The encoded public key and signed file are sent to the client.

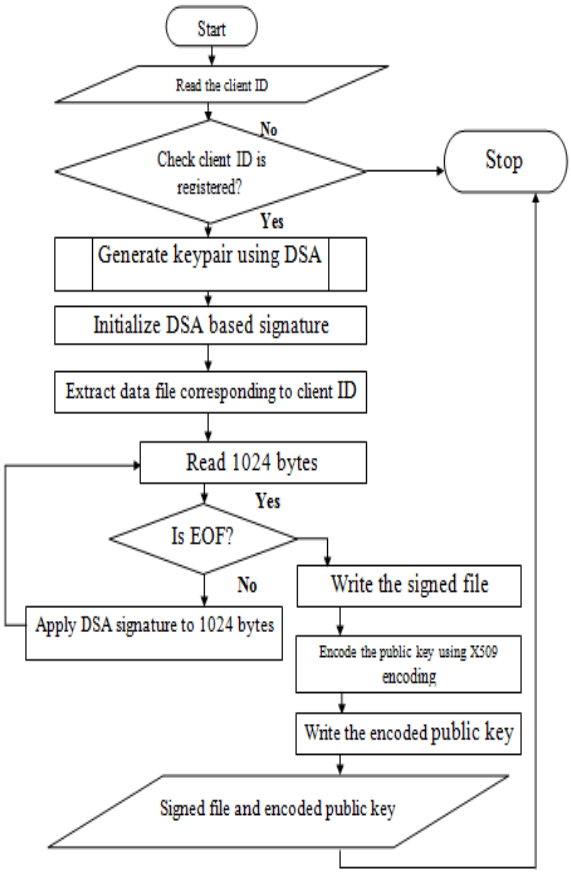


Fig 4.3: Flow chart for TTP Signer

Fig 4.4 gives the flow of signature verification module. It accepts client ID, signed file and encoded public key for authentication. The encoded public key is decoded using X.509 decoder and signature object is created using the decoded public key. Similar to the signature generation mechanism, blocks of 1024 bytes are extracted from the data file and signed using signature object. If the signing process is success the data file is valid, otherwise it is considered as invalid and the same is communicated to the sender [7].

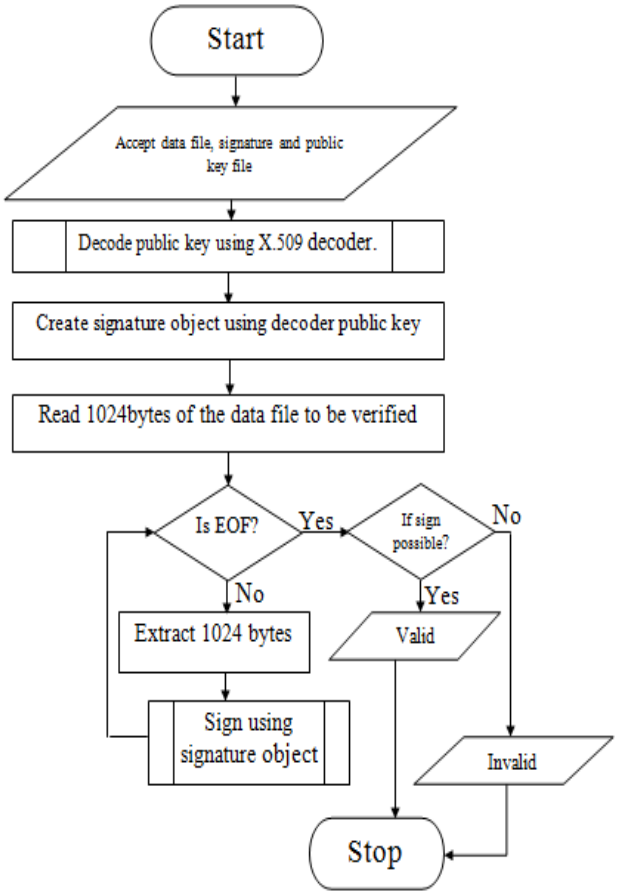


Fig 4.4: Flow chart for Signature and public key verification

E. Attack Analysis

During the process, client and server assume that they are communicating with each other. But sometimes, the communication can pass through the attacker, in which the encrypted tags are modified by attacker and forwarded. In attack analysis, the received file is matched with the original file. If any mismatch occurs, output is assigned the value 1, otherwise value 0. Table I shows the obtained value in the experiment and Fig 4.5 depicts the match and mismatch values of tags modified. It can be inferred that, irrespective of the number of tags modified mismatch is always reported.

F. Key Sensitivity

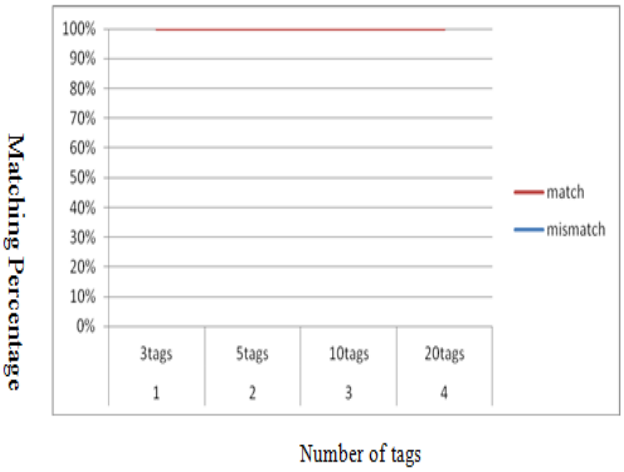
Key sensitivity is analyzed by changing the character of key used for encrypting and decrypting the XML file. After the modification of a key does not produce any output, it is recorded as zero; if produces wrong output, it shall be recorded as one; if produces output it is recorded as two. Table II shows the observed values, and Fig 4.6 depicts the graph of

© 2014 IJAIR. ALL RIGHTS RESERVED

same. It can be inferred that the mismatch accuracy is almost 100% for any number of characters (of the key) modified.

Table I: Attack Analysis

Sl. No	No. of Tags modified	Mismatch	Match
1	3	1	0
2	5	1	0
3	10	1	0
4	20	1	0



Sl. No	Fabrication	Response
1	1character(last)	1
2	2character	0
3	3character	0
4	5character	0

Table II: Key Sensitivity

Fig 4.5: match and mismatch values of tag modification

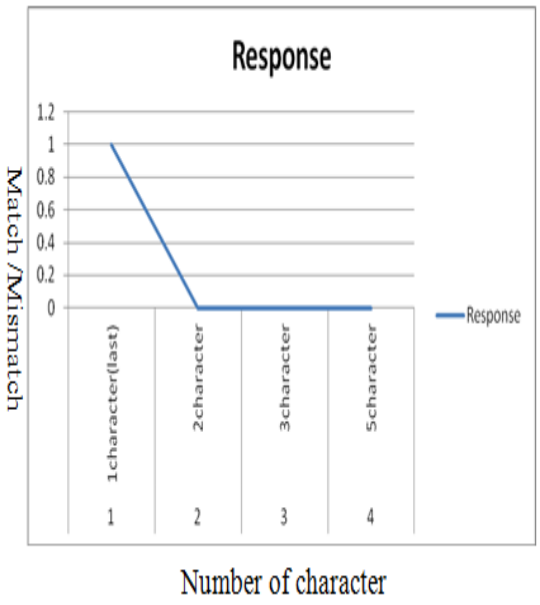


Fig 4.6: Key sensitivity

G. Time Profiling

The time taken for encryption and decryption is computed by taking the difference of start and end time of the module. This enables to check the speed of encryption and decryption and thus gives an account of efficiency of the algorithm. The results have been tabulated in the Table III, and Fig 4.7 graphically shows the time for encryption and decryption. It can be observed that the time taken for encryption and decryption does not vary much and hence complexity is evenly distributed in client and server. Also, the type of encryption will not affect the distribution of the complexity.

Table III: Time profiling for complete encryption

Sl. No	Size of XML file	Time taken for encryption	Time taken for decryption
1	50 Tags	446	437
2	100 Tags	467	464
3	150 Tags	485	496
4	200 Tags	504	522

© 2014 IJAIR. ALL RIGHTS RESERVED

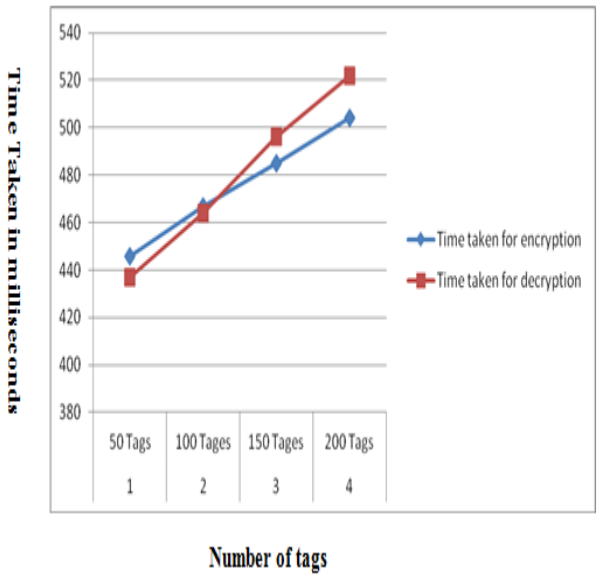


Fig 4.7: Time profiling for complete encryption and decryption

V. Conclusion

In this paper, streaming-based XML encryption and decryption has been designed and implemented. Experiments were conducted to demonstrate the efficiency of the system against different attacks and techniques. This study shows many possible attacks on Web service and the countermeasures to avoid these attacks. It also presents technique used in Web service security.

The results shows streaming-based XML encryption and decryption technique provides a better way of resisting XML attacks. It also includes access control mechanism for certifying and validating the users. The analysis reveals the use of stream based XML encryption and decryption is faster and less memory consumption compare to DOM based parsing system.

Authors Information

Srinath K S B.E., M.Tech., MISTE., was born in Bangalore, India. He completed his Bachelor Degree in Engineering with specialization in Information Science and Masters Degree in Technology with specialization in Network and Internet Engineering from Visvesvaraya Technological University, Belgaum, Karanataka. Presently he is working as Lecturer, Department of Computer Science & Engineering, Sambhram Institute of Technology, Bangalore, India. His areas of interest are Web Service, Web Security, Network security & cryptography and Programming Microprocessors.

Prabakaran M, was born in 1970 at Salem, Tamilnadu, India. He received his Master of Computer Applications degree from Madurai Kamaraj University, Madurai, Tamilnadu, India in 2003, and Master of Philosophy degree in 2004 from Periyar University, Tamilnadu, India. He was awarded with Master of Engineering degree in 2007 from Anna University, Chennai, India. Now, he is pursuing his Ph.D., under Anna University, Chennai, under the guidance of Dr. Mahadevan, Principal, Annai College of Engineering, Tanjaore, Tamilnadu, India. He has published his research articles in various Indian and Foreign Journals. His area of research is Mobile Adhoc Networks. He is much interestingly carrying out his research in Vehicular Adhoc Networks.

He is life member of ISTE. Presently he is working as Assistant Professor in Dept. of Computer Science & Engineering, Sambhram Institute of Technology, Bangalore, India.

References

[1] Brad Hill, "A Taxonomy of Attacks against XML Digital Signatures and Encryption", iSEC Partners, 2007.

[2] Bhavani Thuraisingham, "Secure Semantic Service-Oriented", Auerbach Publications, 2011.

[3] Benjamin Sanno, "Streaming-Based XML Encryption and Decryption", Bochum, October 14, 2010.

[4] Esmiralda Moradian and Anne Hakansson, "Possible attacks on XML Web Services", IJCSNS International Journal of Computer Science and Network Security, Vol. 6, No. 1B, January 2006, pp. 154-170.

[5] Keiko Hashizume and Eduardo B. Fernandez, "Symmetric Encryption and XML Encryption Patterns", Proceedings of the 16th Conference on Pattern Languages of Programs, Chicago, August 28th – 30th, 2009, Vol. 13, No.2, pp 122-124.

© 2014 IJAIR. ALL RIGHTS RESERVED

[6] James McGovern, Sameer Tyagi, Michael Stevens and Sunil Matthew, "Java Web Services Architecture", Morgan Kaufmann Publishers, 2003.

[7] John T. Kohl, B. Clifford Neuman, and Theodore Y. T'so, "The Evolution of the Kerberos Authentication System", Distributed Open System IEEE Computer Society Press, 1994, pp. 78-94.

[8] Meiko Jensen, Nils Gruschka and Ralph Herkenhoener, "A survey of attacks on web services", Computer Science - Research and Development (CSRD), Vol. 24, No. 4, Nov 12, 2009, pp 185-197.

[9] Michael Schrefl, Katharina Grun and Jurgen Dorn, "SemCrypt – Ensuring Privacy of Electronic Documents Through Semantic-Based Encrypted Query Processing". ICDEW '05 Proceedings of the 21st International Conference on Data Engineering Workshops, April 2005, pp 1191.

[10] Rupesh Kumar, Mario Munoz Organero, Rajat Agrawal, "XML Secure Documents for a Secure e-Commerce Architecture", Global Journal of Enterprise Information System, Vol. 2, No. 1 , 2010, pp. 35-45.

[11] Taflan I. Gundem and Mustafa F. Celikel, "STRUCTURE ENCRYPTION IN XML", Information Technology and Control, Vol. 38, No. 1, 2009, pp 72 – 80.

© 2014 IJAIR. ALL RIGHTS RESERVED