A Message Security in SOA Using XML Encryption with using ODD-EVEN Data Methodology

Ankita Gandhi
Student of M.Tech, COE Department, SITE Nathdwara, Rajasthan, India

Asst. Prof. Ajay Dhabaria
Assistant Professor, COE Department, SITE Nathdwara, Rajasthan, India

Abstract

XML is the language of business transaction & used as a specific standard format to exchange any documents or messages. XML encryption which is the one most popular technology to handle complex requirement for securing any XML file. In this paper, we represent the implementation of message security in SOA (Service Oriented Architecture). For that we use XML Encryption with Odd –Even data Techniques compliance with W3C's working draft for XML encryption. Here we have to take two security factor confidentiality & data Integrity. Confidentiality is maintained by applying XML Encryption & Data integrity is maintained by XML Signature. SOA is largely based on the concept of services. In terms of services - communication between consumer and provider. A provider is a system that implements a service so that other systems can call it. A consumer is a system that calls a service (uses a provided service). so here we have to create a web service & also measure the message security.

Keywords: - XML security, XML encryption, symmetric key, asymmetric key, SOA, Web Service;

1. Introduction

XML encryption standard was established by the W3C as formal version of W3C recommendations in Dec 2002.W3C XML Encryption Syntax and Processing specifies the process for encrypting the data and represent it in to XML format. XML encryption uses any security algorithm to encrypt & decrypt the data. In this paper use AES algorithm for encryption & decryption. XML encryption can handle ASCII and non-ASCII data. This paper is Designed to enable user to perform encryption on various binary format files; e.g. jpeg, doc, txt, etc. which is consider as a Message. Implementation of XML encryption with take a odd-even data methodology for securing message in web services. The implementation complies with W3C XML Encryption Syntax and Processing Standard [1] for XML Encryption Format and data structure. Web service is used for securing purpose. Web service decryption language (WSDL) is used for create a web service. The encrypted message is first stored in web service and then decrypts it through web service [6].

2. Message Security Analysis in SOA using XML Encryption

As a message security, the confidentiality and privacy of the sensitive information must be protected during transfer. But there are some time not transfers information correctly. Whenever we apply any encryption algorithm for secure message it is somewhat easily decrypt. So, here we are improved the security level by level in SOA.[5] Our encrypted message stored in web service so, only web service through our encrypted message will decrypt.

So, now below we have described our Architecture for Securing Message in SOA using XML encryption with odd- even data methodology. Using this architecture we are easily prove that whatever message is stored in web service, it is easily secure. Here we are show the architecture for understanding of the related work. We are going in step by step in related work to prove the desire output.

3. Architecture for securing Message in SOA using XML Encryption

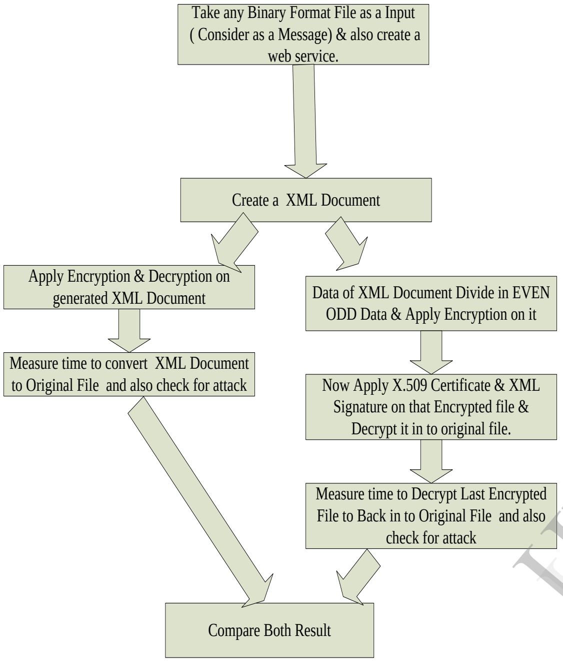

Take any Binary Format File as a Input (Consider as a Message) & also create a web service.
Create a XML Document
Apply Encryption & Decryption on generated XML Document
Data of XML Document Divide in EVEN ODD Data & Apply Encryption on it
Measure time to convert XML Document to Original File and also check for attack
Now Apply X.509 Certificate & XML Signature on that Encrypted file & Decrypt it in to original file.
Measure time to Decrypt Last Encrypted File to Back in to Original File and also check for attack
Compare Both Result

Fig 3.1. Architecture for securing message

Here we are represent the architecture for improve the message security in SOA. In above figure we have to mention the flow of the implementation.

4. Related Work

STEP:1 The first operation of XML encryption process is to read the binary data from the source document (e.g. jpeg file), followed by Base64 encoding operation on the binary data. Which is shown in below fig.4.1 and The output of the Base64 encoding is used to generate ASCII based XML syntax document as an input for XML Encryption processes. The output for it given in fig.4.3

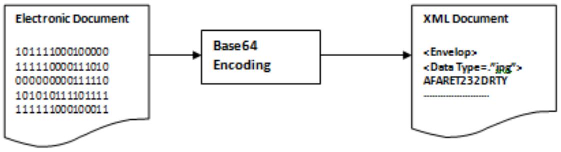
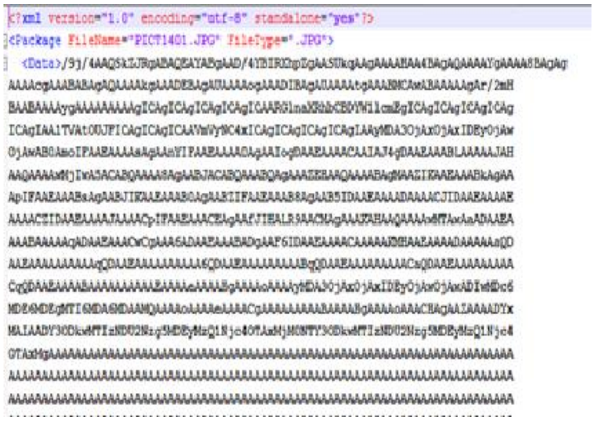


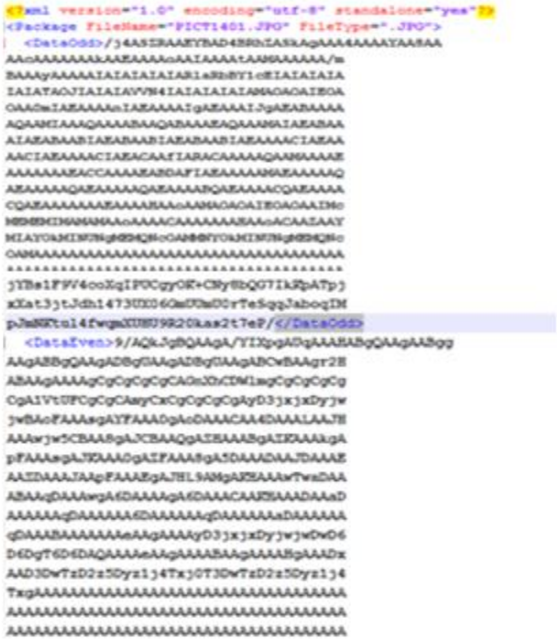
Fig 4.1. Base64 Encoding



```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<Package FileName="PICT1401.JPG" FileType=".JPG">
  <Data>/9j/4AAQSkZJRgABAQEAYABgAAD/4YBIRXhpZgAASUkqAAgAAAABAA4BAgAQAAAAYgAAAA8BAgAg
AAAAcgAAABABAgAQAAAAkgAAADEBAgAUAAAAogAAADIBAgAUAAAAtgAAABMCAwABAAAAAgAr/2mH
BAABAAAAygAAAAAAAAAgICAgICAgICAgICAgICAARGlnaXRhbCBDYW1lcmEgICAgICAgICAgICAg
ICAgIAA1TVAtOUJPICAgICAgICAAVmVyMC4xICAgICAgICAgICAgIAAyMDA3OjAxOjAxIDEyOjAw
OjAwABGAmoIFAAEAAAAsAgAAnYIFAAEAAAA0AgAAIogDAAEAAAACAAIAJ4gDAAEAAABLAAAAAJAH
AAQAAAAwMjIwA5ACABQAAAA8AgAABJACABQAAABQAgAAAZEHAAQAAAABAgMAAAGIKAAEAAABkAgAA
ApIFAAEAAABsAgAABJIKAAEAAAB0AgAABIIFAAEAAAB8AgAAB5IDAAEAAAADAAAACJIDAAEAAAAE
AAAACZIDAAEAAAAJAAAACpIFAAEAAACEAgAAfJIHALR9AACMAgAAAZAHAAQAAAAwMTAwAaADAAEA
AAABAAAAAqADAAEAAACwCpAAA6ADAAEAAAABADgAAF6IDAAEAAAACAAAAAIDMHAAEAAAADAAAAAaQD
AAEAAAAAAAAAAqQDAAEAAAAAAAAA6QDAAEAAAAAAAABgQDAAEAAAAAAAAACaQDAAEAAAAAAAAA
CgQDAAEAAAABAAAAAAAAAAEAAAAeAAAABgAAAAoAAAAyMDA3OjAxOjAxIDEyOjAwOjAwADIwMDc6
MDE6MDEgMTI6MDA6MDAAMQAAAAoAAAAeAAAACgAAAAAAAABAAAABgAAAAoAAACHAgAAIAAAADYx
MAIAADY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MDEyMzQ1Njc4
GTAxMgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Fig 4.2. Plaintext XML Document

STEP: 2 Applying Odd-Even Data Methodology on generated XML Documents. So, we got below output. Means here we are trying to divide data in even & odd data form. So, we improve the security level.



```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<Package FileName="PICT1401.JPG" FileType=".JPG">
  <DataOdd>/j4ASIRAAEYBAD4BRhIASkAgAAA4AAAAYAABAA
AAoAAAAAAAkAAEAAAAoAAIAAAAtAAMAAAAAAA/m
BAAAyAAAAAIAIAIAIAIAR1aRbBY1cEIAIAIAIA
IAIATAOJIAIAIAVVN4IAIAIAIAIAMAOAOAIEOA
OAAOmIAEAAAAnIAEAAAAIgAEAAAIJgAEABAAAA
AQAAMIAAAQAAAABAAQABAAAEAQAAAMAIAEABAA
AIAEABAABIAEABAABIAEABAABIAEAAAACIAEAA
AACIAEAAAACIAEACAAfIABACAAAAAQAAMAAAAE
AAAAAAAEACCAAAAEABDAFIAEAAAAAMAEAAAAAQ
AEAAAAAQAEAAAAAQAEAAAABQAEAAAAACQAEAAAA
CQAEAAAAAAAEAAAABAAoAAMAOAOAIEOAOAAIMo
MEMEMIMAMAMAAoAAAACAAAAAAAABAAoACAAIAAY
MIAYOAMIMTNgMEMQMoOAMMMYOAMIMTNgMEMQMo
OAMAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
jYBa1FPV4ooXqIPUCgyOK+CNy8bQG7IkRpATpj
xXat3jtJdh1473UX06QmUUmU0rTeSqgJaboqIM
pJmNKtol4fwgmXUMU9R20kas2t7eP/</DataOdd>
  <DataEven>9/AQkJgBQAAgA/YIXpgADgAAABABgQAAgAABgg
AAgABBgQAAgADBgUAAgADBgUAAgABCwBAAgr2H
ABAAgAAAAgCgCgCgCgCAGmJDvCDMlmgCgCgCgCg
CgA1VtUPCgCgCAmyCxCgCgCgCgAyD3jxjxDyjw
jwBAoFAAAagAYFAAAOgAoDAAACAA4DAAALAAJH
AAAwjw5CBAABgAJCBAAQgAIBAAABgAIKAAAkgA
pFAAAagAJKAAAOgAIFAAABgA5DAAADAAJDAAAE
AAIDAAAJAAgFAAAEgAJHL9AMgAZHAAAwTwaDAA
ABAAgDAAAwgA6DAAAAgA6DAAACAAEHAAADAAaD
AAAAAAgDAAAAAA6DAAAAAAgDAAAAAAaDAAAAAA
gDAAABAAAAAAAeAAgAAAAyD3jxjxDyjwjwDwD6
D6DgT6D6DAQAAAAeAAgAAAABAAgAAAABgAAADx
AAD3DwTzD2z5Dyz1j4TxjOT3DwTzD2z5Dyz1j4
TxgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Fig 4.3. Divide data in ODD-EVEN data

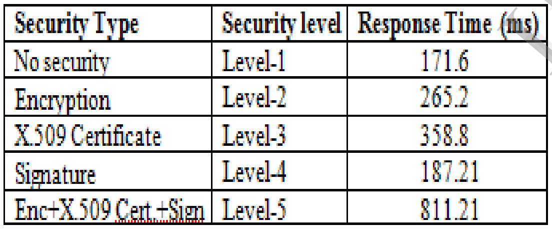
STEP:3 Apply XML Encryption on Odd-Even data using AES Algorithm. So, we are apply here security on even –odd data. So, somewhat security level is increase.

STEP:4 we want to apply for improve the message security level by level in which we have to implement

1) X.509 Certificate: Which is created through write a below code:
→ use makecert.exe (Certificate Creation Tool). For that we have to write the below code:
Makecert –r -pc – n "CN=XML_ENC_TEST_CERT" –b 01/01/2005 -e 01/01/2020 -sky exchange - ss my

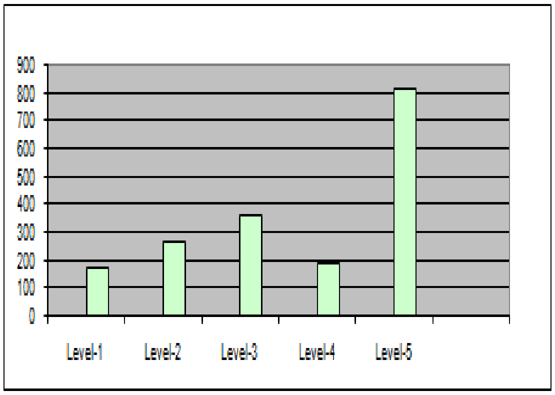
2) XML Signature: XML Signature is used for provide the data integrity. In asp.net we can used the EnvelopedSignatureTransform ()[3] to create a enveloped Signature & add it in to created reference object. And after that created signature is attached with the final encrypted document.

STEP:5 generate graph for level by level security for secure message. By measure the decryption time. It is shows that here we have improve security level. Below table shows the output of response time for decrypt the message.



Security Type	Security level	Response Time (ms)
No security	Level-1	171.6
Encryption	Level-2	265.2
X.509 Certificate	Level-3	358.8
Signature	Level-4	187.21
Enc+X.509 Cert.+Sign	Level-5	811.21

Table 1 Response Time Vs. Security Level



Graph 1 Improve Security Level

STEP: 6 finally we have to compare both result, in which first is apply encryption on data. & Check for attack. And second, apply encryption on even data, odd data & also provide more security through XML Signature & X.509 Certificate. In which 2nd result is improve compare with 1st result.

5. Conclusion

We have described the architecture of Message Security in SOA using encryption through a combination usage of symmetric and asymmetric algorithms to create an encrypted XML that complies with W3C standard. We have also shown that the integrity of the binary document is preserved after subjecting to encryption and decryption process. Therefore, this is a successful practical implementation of the XML encryption on any binary documents which is here considered as a Message. And also message is secure in created web service against the Attack. This is proving bye using the soapUI tool.

6. Future Work

The potential future work will be able to extend this XML security solution to support multiple files encryption and defining multiple recipients of the encrypted XML file. And also improve for more attackers.

7. Reference

[1] Takeshi Eastlake, Blair Dillaway, Ed Simon, "XML Encryption and Processing" W3C, Recommendation 10 December 2002.URL=http://www.w3.org/TR/xmlenc-core/
[2] Koji Miyauchi, "XML Signature / Encryption – The Basic of Web Services Security" NEC Journal of Advanced Technology, Vol 2.
[3] Merlin Hughes, Takeshi Imamura, Hiroshi Maruyama, "Decryption Transform for XML Signature", W3C Recommendation 10 December 2002. URL=http://www.w3.org/TR/xmlenc-decrypt/
[4] Philippe Camacho, "An Introduction to XML Signature and XML Encryption with XMLSec", URL=http://www.dcc.uchile.cl/~pcamacho/tutorial/web/xmlsec/xmlsec.html

[5] T. Erl. “Service-Oriented Architecture: Concepts, Technology, and Design”. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2005.
[6] M. O'Neill. “Web Services Security”. McGraw-Hill, Inc., New York, NY, USA, 2003.