A Proposed SOAP Model Against Wrapping Attacks and Insecure Conversation

Rajni Mohana [1] Deepak Dahiya [2]

[1] Faculty/CSE/Senior Lecturer
Jaypee University of Information Technology
Solan ,PIN-173 234,(H.P.), India

[2] Faculty/CSE/Professor
Jaypee University of Information Technology
Solan ,PIN-173 234,(H.P.), India

Abstract

The web services in SOA are under the heterogeneous ownership domains, there should be a uniform means to offer, discover and interact with each other. Ensuring interoperatability among the web service which is under various ownership domains is the most important challenge. One of the major interoperatablilty issue is protecting the SOAP message from rewriting attacks and insecure conversation as the contents of a SOAP message protected by an XML Signature as specified in WS-Security can be altered without invalidating the signature. The paper presents a proposed SOAP model avoids rewriting attacks and ensures secure conversation. The model highlighted three possible recommendations namely, using shared key for encrypting timestamp in the message body for generating corresponding signature; Secondly, using value referencing both for signature validation and message processing; and finally encrypting the whole SOAP body instead of sending an open SOAP Message in the network to prevent unauthorized access. The paper at the end concludes that the proposed model not only successfully detects rewriting attacks and establishes secure conversation but it also has less overhead in terms of performance metric time which is an important issue in security.

Keywords: *Ws-secure conversation, wrapping attacks, SOAP message, rewriting attacks, WS-Security.*

1. Introduction

The communication among these web services is established using SOAP messages which are based on XML. XML [1] is the most relevant means to provide interoperatablity among various entities. When in network a XML file can be prone to hacking and unauthorized access, thus affecting the data integrity and confidentiality which are supposed to be important issues of communication. Data integrity and confidentiality can be breached by rewriting attack [2].

In any business to business (B2B) application a messages may carry vital business information, their integrity and confidentiality needs to be preserved by ensuring secure conversation in a group of reliable and authenticated web services. This is another challenge that a SOAP message transmission faces in web service applications among enterprises. Let's consider the generalized SOAP structure as shown in fig 1. It can be seen that the signed data object is referenced using a reference Uniform Resource Identifier (URI) within the XML Signature element, which is a child of the XML signature element. Thus the signed object is inside the XML Signature Element. The signed data object contains the XML Signature Element, which contains its signature, within it. Therefore the signed data object is the parent of its signature element [3]. This indirect referencing does not give any information regarding the actual location of the signed object.

Processing of the given SOAP message consists of two independent steps [4]:

i. The recipient first searches for the referenced element and then computes the digest value over this element and compares it to the value given in the digest value. Later he verifies for the signature value.

ii. Next step is to execute the function defined in the SOAP body.

Therefore, the signed object can easily be relocated and the signature value still remains valid. This situation leads to wrapping attacks. Wrapping attacks are also called as rewriting attacks and both the terms will be used in th paper synonymously. Further, if the SOAP request passes through intermediaries en route to the destination web service, and if an authorized person has modified the content pretending someone else as the constructor, then there is no way to find out the end-to-end integrity

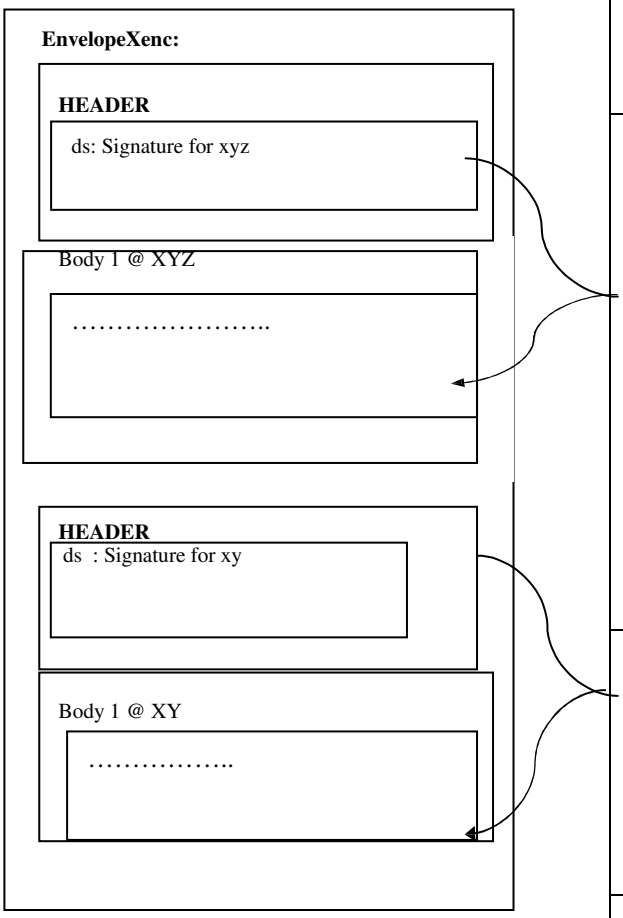

Fig 1. SOAP Structure

2. Related Study

Many researchers [4,5,6,7,8,9] have provided various SOAP model to avoid rewriting attacks. Table 1 provides the description and the tradeoff of the work of the various researchers.

3. Proposed Model

As evident from the related study and subsequent analysis of the work of past researchers have made us conclude that many researchers have worked towards securing SOAP message. This led to the exploration of the tradeoffs vis- à-vis the various solutions given by them.

Researchers	Work description	Tradeoff
[4] Sebastian Gajek, Meiko Jensen, Lijun Liao, and J¨orgSchwenk	Recommended Fast*XPath*	Fast*XPath* is not flexible as it limits the abilities of defining a signature reference [4]
[5]AzzedineBenameur	Recommended using depth information, parent information and using *Id* attribute to uniquely identifying the parent.	Sending these information over the network is also not same, as it can also be tampered by malicious attacker [9]
[6]Mohammad AshiqurRahaman, Andreas Schaad	Recommended maintaining information and sending it with the SOAP message • Number of child elements of the root (Envelope). • Number of header elements. • Number of references for signing element. •Predecessor, Successor, and sibling relationship of the signed object.	It may not comply with the schema of the W S * standards and might even violate further processing of XML messages [7].
[7] Yong Liu, Haixia Zhao, Yaowei Li	Suggests signing the SOAP account , if the intermediaries have their own SOAP Account then there will be a nested signature	The process will be slow as signature generation is a slow process.
[8]smriti kumar sinha, AzzedineBenameur	Suggest using Context-Sensitive Signature (CSS)	The trade off is that the context is to be generated and stored in the reference element of the signature in header section before signing the message [8].

Table 1: A summary of the SOAP model proposed by various researchers

We propose the SOAP model with three recommendations and ensure by implementing a case study that it will prevent rewriting attacks and have end to end integrity.

The proposed model has following three recommendations

- Using timestamp in the message body and generate corresponding signature, the signature and Timestamp needs to be encrypted by a shared key.
- Using path referencing for signature validation and message processing both, instead of using two different ways of referencing for both.
- Encrypting the signature and the timestamp with a shared key and put the encrypted value in the SOAP header. Later encrypt the whole SOAP body instead of sending an open SOAP Message in the network to prevent unauthorized access.

should be protected from n-house and out-of-the-house (foreign) intruders. Out-of-the-house intruders, which are not the part of actual conversation, will not be able to decrypt the message as they will not have the shared key according to the model. They can at most cause total destruction of the message and make it unreadable for the entities in conversation. This attack can easily be traced down as the receiver will not be able to decrypt the message with the shared key.

An in-house intruder can be dangerous as it possesses the right key and can decrypt the message and modify it and send it further pretending that it is not the constructor [10, 11].

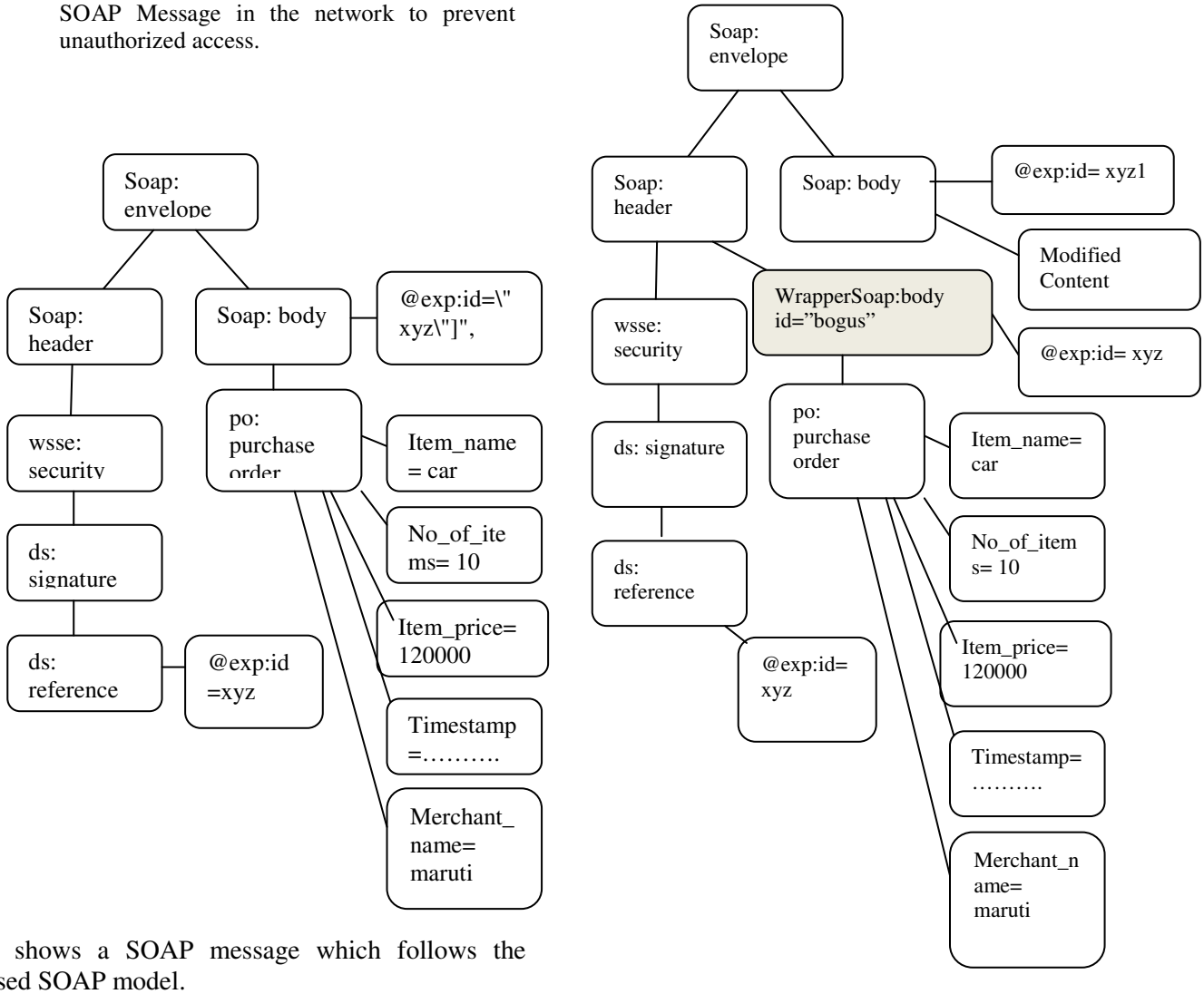

Fig 2 shows a SOAP message which follows the proposed SOAP model.

4. Working of the Model

To ensure data integrity and confidentiality of a message it

As the SOAP model uses value referencing, it will process the given SOAP message with the following two independent steps:

i. The recipient first searches for the referenced element *SOAP-ENV:Envelope/SOAP-ENV:Header[@exp:id=\"xyz\"]"* and then computes the digest value over this element and compares it to the value given in the digest value. Later he verifies for the signature value.

ii. Next step is to execute the function defined in the SOAP body using the same Value referencing *SOAP-ENV:Envelope/SOAP-ENV:Header[@exp:id=\"xyz\"]"*.

There may be two kinds of attacks which can be performed by in-house intruders:

a. Tampering the message created by different set of entity in communication

b. Re-locating the message or header under different header.

Tampering the message created by different set of entity in communication

If an authorized person tampers the message then the signature will not match at the first step. If the person is authorized and he has the key of the signature generated then after modifying , he will have to generate the signature again with his own key, thus he cannot pretend that someone else is the producer of the message and he is not.

Re-locating the message or header under different header.

If an authorized person changes the message without invalidating the signature and pretends that someone else is the creator of the message, the model is able to locate the rewriting attack. This is explained as follows. Let's consider that the SOAP message shown in fig 2 is tampered by an authorized person as shown in fig 3. In fig 3 as the content is tampered, but it will be detected as the signature will not be validated as per the algorithm.

The recipient when it searches the actual SOAP body according to the path *SOAP-ENV:Envelope/SOAP-ENV:Header[@exp:id=\"xyz\"]"* , it will not be able to find the actual SOAP body in the path as it has different id *XYZ1*. Thus the signature will not validate. Instead of directly jumping to Id, it performs path referencing.

5. Results and Observations

The performance of the SOAP model has to be analyzed as any signature and encryption operation on XML message requires considerable XML processing time which is directly proportionate to the size of the message [11].

The proposed SOAP model will have an overhead in terms of

- Time required for encrypting the SOAP body
- Time required for decrypting the SOAP body.

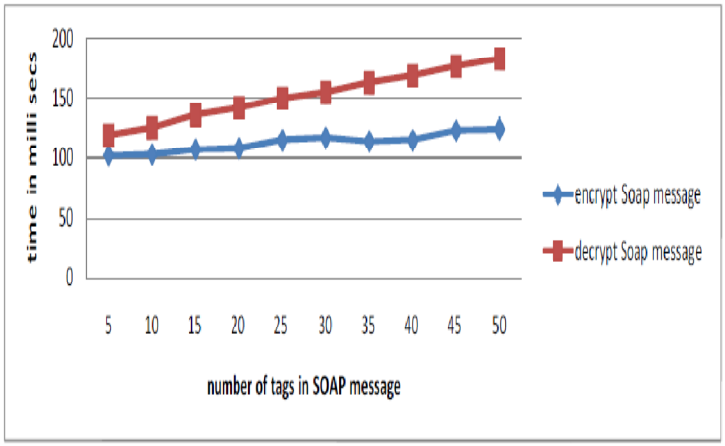

Fig 4: encryption and decryption of the SOAP message with respect to the number of tags in the SOAP message

The performance metrics of the model was also evaluated in terms of time complexity. The graph clearly shows that the increase in y-axis is too less in comparison to the corresponding increment in the y-axis. The percentage of growth was .04%. The red line in the graph shows time taken to decrypt the message with key G_s. It was observed that the time required to decrypt a message of 5 tags was 119 milliseconds and as the SOAP message size grew to 50 tags the time take to decrypt the message was 183 milliseconds.

In this case also it was seen that the overhead of decrypting doesn't subsequently increase with the number of tags in the SOAP message. The percentage of growth in the time with respect to the number of tags was 0.14%.

Complexity Analysis: SOAP Message Encryption

The analysis of algorithm complexity for the SOAP message encryption rests on the assumption that each single instruction is completed in one unit of time. Creating a SOAP message involves examination of the elements namely, *header, hashValue, Body, Item-details (N-Number Of Item).*

Creating any element in SOAP message requires execution of the following three instructions

a) Function to create the root elemnt

QName rootElement=new QName(NameSpace,Prefix);

b) Function to create a text node

QName tagName=new QName("tagName");

c) Function to append the text node into root

rootElement.addChildElement(tagName);

From the code it was seen that the total number of instruction used to create a SOAP message with N element is
4*3 + (1+3) + 3*N=3*N+15

This implies the need for *O(3*N+15) =O(N)* instructions for creation of SOAP message.

Thus the overhead to create the SOAP message is O (N). It can be inferred that the overhead is significantly less.

Complexity Analysis: SOAP signature generation and its encryption
Generation and Encryption of a SOAP message requires the following steps namely

1) Generation of public and private key by means of key generation algorithm like DSA
2) Generation of signature with a signature generation algorithm
3) Generation of key information
4) Generation of Signature Information and lastly
5) Appending the signature.

Following assumptions for the analysis

1) The key generation algorithm takes O(k(M)) instructions to generate the signature where M is independent of N.
2) The Signature algorithm depends upon the number of elements in the SOAP message and let it takes O(S(N)) instructions to generate the signature.
3) The information of key is not depending on size of the message and so let it takes O(I(M)) instruction to generate information.
4) The information of signature is not dependent upon size of message and let it takes O(SI(R)) to generate the signature information.
5) Appending takes O(1) instructions to append the signature.

Hence the algorithm of signature generation takes
O(max(k(M),S(N),I(M),SI(R)))

Thus it can be inferred that the time complexity of signature generation depends on the type of key generation algorithm for encryption and Signature generation algorithm chosen.

6. Conclusions

A SOAP message transmission in an enterprise web service application faces the challenges of rewriting attacks and secures conversation. The properties of the proposed SOAP model are simplicity and light transmission. The working of the model is based on three possible recommendations namely, using shared key for encrypting timestamp in the message body for generating corresponding signature; Secondly, using value referencing both for signature validation and message processing; and finally encrypting the whole SOAP body instead of sending an open SOAP Message in the network to prevent unauthorized access.

The paper also proves that the trade off the model is significantly less and in terms of O(N), where N is the number of tags of SOAP message.

References

[1] www.w3.org/TR/xpath/
[2] A. Ghourabi, T. Abbes and A. Bouhoula, "Experimental analysis of attacks against web services and countermeasures," In Proceedings of the 12th International Conference on Information Integration and Webbased Applications & Services (iiWAS '10). ACM, New York, NY, USA, 195-201, 2010.

[3] M. McIntosh and P. Austel,"*XML Signature Element Wrapping Attacks and Countermeasures*" in the proc. of the ACM workshop on Secure web services ,New York, NY, USA Pp: 20 – 27 , 2005.

[4] Sebastian Gajek, Meiko Jensen, Lijun Liao, and J¨orgSchwenk ,"*Analysis of Signature Wrapping Attacks andservices Security*," compsac, vol. 1, pp.23-30, 2007 31st Annual *Countermeasures*", in the proc of IEEE International Conference on Web services, ICWS, Los Angeles, CA Pp: 575 – 582, 2009

[5] Azzedine Benameur *,"XML Rewriting Attacks: Existing Solutions and Their Limitations*", Proceedings of the ACM workshop on Secure web services , New York, NY, USA, Pp. 53-60, 2008

[6] Mohammad Ashiqur Rahaman, Andreas Schaad, " *SOAP-based Secure Conversation and Collaboration*", in the proc. of IEEE International Conference on Web services, ICWS , Salt Lake City, UT , Pp 471 – 480, 2007.

[7]Yong Liu, Haixia Zhao, Yaowei Li," *Research On Secure Transmission Of Messages*", in the proc. of 12th International Conference on Computer Supported Cooperative Work in Design, Xi'an, China , Pp: 771 – 776, 2008.

[8]Smriti Kumar Sinha, AzzedineBenameur,"*A Formal Solution to Rewriting Attacks on SOAP Messages",* in the proc. of ACM workshop on Secure web services SWS '08 ACM New York, NY, USA Pp: 53 – 60 , 2008

[9] Nils Gruschka, Meiko Jensen, Florian Kohlar and Lijun Liao "*On Interoperability Failures in WS-Security: The XML Signature Wrapping Attack*" in Electronic Business Interoperability: Concepts, Opportunities and Challenges, IGI Global, Pp. 615-635, March, 2011.

[10] NavyaSidharth, JigangLiu "*Intrusion Resistant SOAP Messaging with IAPF*", in the Proceedings of the 2008 IEEE Asia-Pacific Services Computing Conference APSCC '08 Pp- 856-862 Washington, DC, USA

[11]NavyaSidharth, Jigang Liu, "*A Framework for Enhancing Web* International Computer Software and Applications Conference, 2007

Rajni Mohana is currently working as a senior lecturer in Department of CSE & IT at Jaypee University of Information Technology (JUIT), Waknaghat, India. Her current research areas are service oriented architecture , model driven architecture. She has an academic experience of 9 yrs and is also persuing her Ph.D from Jaypee University of Information Technology (JUIT), Waknaghat, India

Deepak Dahiya is currently working as Professor in the Department of CSE & IT at Jaypee University of Information Technology (JUIT), Waknaghat, India. He has M.S. and PhD degrees in Computer Science from BITS Pilani, India. His interdisciplinary research interests span over Software Engineering and IT Management**.** He is also a reviewer for various renowned journals from Elsevier, IET, Taylor Francis and Wiley. Deepak is a Visiting Researcher to RMIT University, Australia, Guest Faculty to Indian Institute of Management (IIM) Rohtak, India, Indian Institute of Management (IIM) Kozhikode, India and LNM Institute of Information Technology (LNMIIT) Jaipur, India. He is a senior member of IEEE and life member of Computer Society of India.