Chapter 3

A Solution on SOAP Message Integrity

SOAP messages with XML signatures under Web Service Security specification provide secure message exchange solutions in SOA based applications. Recent researches established that the solutions based on the specification have several limitations. XML rewriting attacks on SOAP messages exposed the vulnerability of SOAP messages and different solutions are proposed to counter the attacks. This chapter exposes few other limitations of Web service security in providing end-to-end integrity, specially part integrity and reuse issues, of multiple signed messages in a SOAP message in a document production workflow environment. This chapter also discusses the consequences of the limitation and establishes that it is not possible to address these issues at message level. It also proposes a solution in BPEL process level using a special protocol.

3.1 Introduction

Business processes today are modelled as workflows. A workflow is defined as a set of coordinated tasks. Recent trend in software industry is towards adopting Service Oriented Architecture (SOA) in applications. SOA applications are flexible, dynamically composable and inter operable. Web services provide a technology that can be used to implement SOA and are increasingly becoming the SOA implementation of choice. In a business workflow, tasks are implemented as Web services. Functionalities of each Web service is described in an interface, which is stored in a separate file. The interface is described using a language called Web Service Definition language (WSDL). Since the WSDL interface and the actual
implementation of the service are decoupled, the implementation can be replaced by another
implementation compliant to the published interface [24]. The Web services are co-ordinated
by another process, which is specified by Business Process Execution Language (BPEL). The
type of coordination done by the BPEL process is known as orchestration. The interface of
the BPEL process is also defined by another WSDL file. Therefore, a BPEL process is a
composite Web service, which in turn may be used in the composition of other composite
Web services [29]. Among Web Services, communications are made through Simple Object
Access Protocol (SOAP) messages. SOAP [12] is a W3C specification used for exchanging
messages among Web services. WSDL, BPEL, SOAP are all derivatives of XML languages.
Therefore, the foundation of Web service technology is on XML. An XML document is a
tree having a root element, which may have multiple elements as children, each child has
further child elements and so on. Each element may have multiple attributes. All elements
and attributes are user defined, no predefined elements and attributes. It's a specification to
define other markup languages. SOAP messaging framework consists of a message construct,
a processing model, an extensibility model and protocol binding to multiple underlying
transport protocols, like HTTP, FTP, SMTP etc. According to message construct the root
element of a SOAP message is envelope, which has an optional header and a mandatory body
element. The header element contains the meta-data only, under multiple header blocks, no
application payload. The body has the actual message, as XML payload. SOAP specifies a
distributed processing model which allows multiple intermediaries to process the message
and thus insert multiple new messages during the itinerary of the original message from the
original sender to the final receiver. This facilitates the use of SOAP framework in a
workflow environment. Security is an important requirement for any practical business
workflow. To provide message level security during storage and communication among
multiple Web services, OASIS published a standard specification on Web Service Security:
SOAP Message Security 1.1(WSS) [30], [31]. WSS uses XML signature [22] as a security
mechanism to address proof of origin and content integrity issues of XML messages. It also
allows to sign not only the whole XML payload of a SOAP message but also a part of it. The
goal of WSS is to ensure end-to-end message level security. When a message that traverses
multiple intermediaries, within and between business entities e.g. companies, divisions and
business units, it is secure over its full route through and between those business entities. This
includes not only messages that are initiated within the entity but also those messages that
originate outside the entity, whether they are Web Services or the more traditional messages
[30].
XML security is the basis for Web Services Security (WSS). In one direction, for access control, we have XACML specification. Research works on XACML in different environments are going on. For example, XACML as a security pattern and integration scheme, suitable for workflows in ambient intelligence environment is discussed in [32]. In the other direction, for message level security, works are going on towards secure SOAP messages and WSS provides the required security. WSS specification is flexible, but unfortunately, SOAP messages compliant to this specification are still vulnerable. A SOAP message is prone to attacks that can lead to several consequences, such as, unauthorized access, disclosure of information, identity theft etc. In the literature, the attacks are basically on-the-fly modifications of SOAP messages and are referred to as XML rewriting attacks [33]. Providing a solution to XML rewriting attack on SOAP messages is a vital challenge of Web Service Security. The attacks are framed on the weaknesses of XML signature. This opens up the flush door to security attacks and makes secure SOAP message exchange vulnerable [26]. One of the main security research focuses is XML rewriting attack. Different solutions for XML rewriting attacks are proposed in the literature to address this problem [13], [34], [35]. Recently, the first formal and almost complete solution to the XML rewriting attack was proposed [26]. The study motivated us to explore further limitations of WSS specification specially in a document production workflow in an office environment.

Apart from weaknesses of XML signature in SOAP messages, which led to XML rewriting attacks, it is found from the investigation that there are few more limitations of current WSS standard on SOAP messages, hitherto not discussed in Web Services Security literature but equally important, specially in the cases of workflow implementations as orchestrated or choreographed composite Web services. In this chapter, these limitations are brought to focus. Current version of WSS fails to address these issues. These were first discussed in [36] as issues of Multi-Part Multi-Signature Document (MPMSD) in the context of Document Production Workflow (DPW) in an office and a solution were proposed in [37]. These issues are found to be still relevant and important in SOAP security. In WSS standard, XML signatures and XML encryption [22] with inclusive and exclusive canonicalization ensure proof of origin, content integrity and confidentiality issues of each individual part, that is, each signed message, in a SOAP envelope during exchange and storage. But it fails to ensure the collective integrity issue of all the parts as a whole, which means all the signed messages in a SOAP envelope signed by the original sender and the intermediaries during the workflow. It also fails to address the part reuse issue [37]. WSS standard fails to ensure end-
to-end security in SOAP message level on these issues. As a consequence several attacks may
be possible, similar to the vulnerabilities described in the context of MPMSD [37]. It is also
shown in this chapter that it is not possible to ensure resolution of these issues in the SOAP
envelope directly. It has to be ensured in the BPEL process level through a protocol.

3.2 MPMSD

Document production in an office is based on a request-reaction-response paradigm. When a
message containing a request is received in an office, the office reacts to the request. The
reactions are recorded in the form of comments on the document and finally a response
message is dispatched. We can term the process as Document Production Workflow (DPW).
The resultant document of a DPW is termed as a Multi-Part Multi-Signature Document
(MPMSD). Therefore, a MPMSD is a case of the DPW. The first part of a MPMSD is the
request message and the last part is the response message and the intermediate parts are the
messages containing the comments of other reviewers, which means, the reactions. Each part
of a MPMSD is signed by the corresponding reviewer. The first reviewer is also termed as
the originator. MPMSD is a generic framework. Parts belonging to different cases may also
form a MPMSD. Moreover, multiple versions of a document may also be defined as a
MPMSD. A reviewer creates a part of a MPMSD in the context of a set of existing
documents constituted of rules, precedents and other support documents, which in turn may
be MPMSDs. Rules, precedents and support documents constitute a reference space of an
office. Here, in this chapter it has discussed only it from the work object or case point of view
in a document production workflow [38].

A Multi-Part Multi-Signature Document (MPMSD), \( D_w \), produced in a DPW \( W \), is an n-
tuple, \( n \geq 1 \), such that \( D_w = (d_1, d_2, d_3, \ldots, d_n) \). Each part \( d_i \) in turn is defined as a 3-
tuple \((m_i, \sigma_i, s_i)\), where \( m_i \) is the comment of the reviewer \( s_i \), and \( \sigma_i \) is the signature of \( s_i \).
MPMSD is basically an abstract representation of a SOAP envelope, containing multiple
signed messages.
3.2.1 A Scenario

To understand the salient features of a DPW in an office let us consider the following scenario of a DPW. An employee, A submits an application, mA regarding her travel plans for approval to the manager, B of the department. B verifies the travel plans in the context of previous cases of employees from the team already in travel, type of leave to be granted for A during travel, standing rules, etc. and adds her comment, mB and forwards it to the finance officer, C. C also examines the case by verifying the budget allocation status under the head of account for travel, TA/DA rules in such cases and adds her comment, mC on the amount that may be granted and forwards it to the director, D. D also justifies the previous comments, approves the travel plans and adds the note of approval, may be in the form of office order, mD. A copy of the whole multipart document or only the office order mD may finally go back to the originator A and the original multi-part document is stored in a folder. The flow of the document is recorded in log-books. This is a case of the travel plan workflow. It is shown in figure 3.2.
3.2.2 Security Issues

For every individual part of a MPMSD, the standard security issues are proof of origin, content integrity, confidentiality, non-repudiation of signing etc. These are not the issues of discussion in this chapter. It is assumed that WSS standard addresses these issues for every message in a SOAP envelope efficiently and correctly as claimed. Apart from the security issues of each individual part of a MPMSD, the following special security issues related to a MPMSD as a whole are to be addressed additionally. These issues are the main points of discussion in this chapter.

1) Part Integrity of a MPMSD: Apart from the content integrity of each part, we also need the total part integrity of the whole document. The content integrity of all parts individually does not necessarily imply the integrity of the whole document. All parts must remain in order in which they were added to the document. Removal of some parts and reordering of parts should not be allowed.

2) Reuse of Parts: Reuse of parts should not be allowed. Suppose, the ordered list of reviewers of a MPMSD is (A, B, C, D) as in the example given in figure 3.2. If D does not like what C has written on the document mC then D may co-operate with B to have B mark the document directly to D, bypassing C. B can do this by using the document mA||mB passed to it by A and reusing it.

3.3 Orchestration and Choreography

Often services are offered by different companies. A Web service composition consists of multiple invocations of other Web services in a specific order. A composition takes the form either of an orchestration or of choreography.

Orchestration describes how Web services interact with each other at the message level, including the business logic and the execution order of the interactions from the viewpoint of the partner controlling of the workflow execution. Orchestration deals with implementation management (i.e. process execution). Orchestration is a private process, controlled by one
party, and defines steps of an executable workflow. Orchestration is about what happens behind interfaces (i.e. process execution) [69].

Choreography describes the interaction between business partners in terms of the sequence of messages that are exchanged in a "peer-to-peer" fashion. In choreography, there is no central control of workflow execution. Choreography is more about what happens between interfaces. It can involve static and dynamically negotiated protocols. In this sense, choreography is a public, abstract process where conversations are composed by equals who define sequences of observable messages [69].

3.4 DPW using SOAP Messages

Document Production Workflow (DPW) is a group work. Therefore, a distributed protocol without any central authority, like the SMTP, POP3 of standard e-mail service, may prop up as a natural choice. In such a distributed approach, the digital document flows from mailbox to mailbox and the reviewers add their comments to the document. This mimics the flow of paper documents in manual offices. We know, protocol binding of SOAP framework allows mail protocols like SMTP, POP3 etc. Without any loss of generality, let us consider that the DPW is implemented using SOAP and WSS with protocol binding to either or both the standard mail protocols for message transmission. A sends a secure SOAP envelope containing a message on original travel plan m_A digitally signed by A, using XML signature to B and so on to C. If A and C collude, C may drop the comment of B and forward the case to D with her strong comment m_C. In a rigid workflow, like in e-Governance, where the reviewers are fixed a priori, such case may be detected by inspecting the absence of B's comment or approval. But in an ad hoc or flexible workflow, like those in many industries, where additional reviewer may be inserted from case to case basis, such collusion may go undetected. For example in our scenario, if manager B thinks that in case of A, comment of another project leader P of a project, where he works in addition to normal duty, is required before approval and forwards the case to P. The comment of adhoc reviewer P can easily be deleted by the colluding parties. Even if physically not deleted, the next reviewer can be bypassed simply by manipulating role and mustunderstand attributes in the security header of the SOAP message. In certain cases reordering of the comments may change the semantic of the case, which may effect the subsequent decisions. Regarding the reuse of parts, as
discussed above, any colluding party can reuse the partial MPMSD available with him or her with another subsequent reviewer in the workflow. This inhibits in itself a new type of attack on SOAP envelope specially in a workflow implementations.

Every reviewer signs not only own comment but also all previous comments up to the original message in the envelope. This can be done by the following steps –

1) Compose the new message.
2) Extract all previous messages from the SOAP envelope.
3) Arrange them in the order of timestamps.
4) Append the new message to the ordered list.
5) Input the composite message to canonicalization method.
6) Generate the message digest.
7) Sign the message and put the signature and other info under signedinfo element in security header block.

Since XML signature allows non-contiguous blocks of an XML message to be signed, this may solve the part integrity issue, but the question is how to enforce the user to do so. Moreover, it does not solve the reuse issue. The boundary case that is, if the ultimate receiver D may be a part of collusion. WSS specification claims SOAP envelope security not only during transmission but also during storage to intermediate processing nodes en-route to the ultimate receiver [30]. But it has seen that it is not possible to address these issues in SOAP envelopes directly in the message level. The end-to-end security, envisaged by WSS specification, of the messages contained in a SOAP envelop from the originator of the message A to the final receiver D keeping the intermediaries signed comments intact fails. These issues are more challenging when the envelope routes across many business entities.

3.5 Solution

The main objective of this chapter is to bring into focus some new limitations of WSS on SOAP, specially in workflow environment. However, a solution with a special protocol
exists. Details of the protocol are discussed in [37], [38]. This protocol addresses the security issues of MPMSD mentioned above. The protocol is based on a central arbitration mechanism. It is basically a client/server computing paradigm, where the arbiter is the server. All the cases flowing from one client (reviewer) to another are routed through the arbiter. In case of dispute, the decision of the arbiter, based on the stored evidences, is final and binding. It is discussed in detail in [37] that to address these issues an in-line Trusted Third Party (TTP), called an arbiter, is mandatory. The arbiter serves as the trusted intermediate agent in between the current reviewer and the next. To make a protocol as general as possible, researchers attempt to avoid the use of an arbiter. It is established that to provide non-repudiation with time information, an in-line TTP is anyway necessary [39]. To provide non-repudiation of a digital signature, time of the signature is essential as the key used in the signature may become public at a later time. In this environment, documents are persistent and so non-repudiation of a digital signature is essential. So an in-line TTP will be required. Since an in-line TTP is required for non-repudiation on time of sending or receiving documents, we can en-thrust the TTP, the additional responsibility of an arbiter for production of MPMSD as well. Further, to prevent the reuse of parts, an in-line TTP will evidently be required as immediate detection of such reuse will be necessary to prevent the taking of wrong decisions. When B and D collude to bypass C, C will not be aware of this till much later if no in-line TTP is present [37]. In this protocol, a reviewer submits a comment (part) for a MPMSD to the arbiter and the arbiter adds the part to the MPMSD and present the MPMSD to the next reviewer. The next reviewer can only read the presented multi-part document but cannot modify or add comments to it directly. He can only submit the comment to the arbiter and arbiter adds the comment to the document with proper timestamps and forwards it to the next reviewer.

In a Web service implementation, the BPEL process co-ordinates other Web services. Therefore, it may also act as the arbiter. A reviewer submits the signed comment, which is a SOAP message in an envelope, and the BPEL process extracts the signed message from the envelope and appends it to the envelope containing previous signed messages (MPMSD) in order and then presents modified envelope to the next reviewer. Next reviewer can only read the signed messages, verify the signatures if required but cannot resubmit the presented envelope again. A new envelope containing only her new comment can only be submitted. Therefore, the protocol presented in [37] can be implemented in a BPEL orchestrated workflow which is discussed in the following.
3.6 Discussion

The originator \( A_i \) submits her signed message \( d_i \) to the BPEL process \( N \). Intermediate reviewer \( A_i \) receives a SOAP message \( D_{w}^{i-1} \) from \( N \). \( D_{w}^{i-1} \) contains all the previous signed messages \( d_1, d_2, d_3 \ldots d_{i-1} \), signed by previous reviewers \( A_1, A_2, A_3 \ldots A_{i-1} \). \( A_i \) reviews the SOAP message \( D_{w}^{i-1} \), submits her signed comment \( d_i = (m_i, \sigma_i, A_1) \) to \( N \) and marks \( A_{i+1} \) as the next reviewer. BPEL process \( N \) adds the comment \( m_i \) to the body section of \( D_{w}^{i-1} \) and \( \sigma_i \), \( A_i \) and other necessary security information in a proper SignedInfo element in the security header. \( N \) also marks \( A_{i+1} \) as the next reviewer by defining the role and mustunderstand attributes accordingly resulting \( D_{w}^{i} \). \( N \) then delivers the document \( D_{w}^{i} \) to \( A_{i+1} \), who in turn submits her comment \( d_{i+1} = (m_{i+1}, \sigma_{i+1}, A_{i+1}) \) to \( N \). Thus the protocol starts from the originator till the message is delivered to target receiver after being reviewed by intermediate reviewers. The final recipient may be the originator.

3.7 Assumptions

It is assumed that review processes at the reviewers’ ends are implemented as Web services and the orchestration is done by the BPEL process. The communication between the BPEL process and the constituent Web services are done by using encrypted SOAP messages, encrypted with the public keys of the recipients or by any other shared keys shared between the BPEL process and each of the reviewers. Since this is not the actual discussion of this chapter, it can be assumed that it as a secured channel. Moreover, it is also assumed that for non-repudiation issues there are necessary tokens for non-repudiation of sending as well as of delivery in place. These were discussed in detail in the original protocol in [37]. Hence the corresponding steps in the protocol are given below where \( n_i \) and \( t_i \) signify nonces and timestamps respectively.

Protocol Steps:

1) \( N \rightarrow A_i : \{N, d_{i-1}, n_i, t_i\} \)
2) \( A_i \rightarrow N : \{ A_i, m_i, \sigma_i, A_{i+1}, n_i, t_i \} \)

3) \( N \rightarrow A_{i+1} : \{ N, d_i, n_{i+1}, t_{i+1} \} \)

4) \( A_{i+1} \rightarrow N : \{ A_{i+1}, m_i, \sigma_i, A_{i+2}, n_{i+1}, t_{i+1} \} \)

### 3.8 Chapter Summary

This chapter discusses certain new limitations of the present version of Web service security standard in a workflow environment for handling document flow in a typical office. SOAP processing model does not discuss these issues. The main focus of this chapter is to show the limitations of WSS and also to show how these issues can be solved by a protocol. From the above discussion it is clear that it is not possible to address these issues within the scope of message level. A central arbitration mechanism is mandatory. In case of orchestrated Web service based workflow, BPEL process can take the responsibility of this arbitration along with coordination of constituent Web services. However, in case of a complex workflow across many business entities, where multiple BPEL processes are to be coordinated, different protocol may have to be used in addition.

The case of choreographed workflow implementations is beyond the scope of discussion of this chapter today but the study will be more interesting. The future research is in this direction. The implementation of the protocol in both orchestrated and choreographed workflows in Web service world is the direction of future research work.