

On Achieving Trustworthy SOA-Based Web Services

Zheng Jianwu Liu Mingsheng Liu Hui

Department of Information Engineering,
Shijiazhuang Railway Institute, Hebei 050043, China
{zhengjw, liums, liuhui}@sjzri.edu.cn

Abstract

This work is inspired by the intend to construct SOA-Based E-Government. We first emphasize the importance of taking measures for solving security problems facing Web Services, we then have an in-depth look at SOA-Based Web Services, including its architecture, underlying technologies, transmission model, and SOAP message. By leveraging the understanding of securing Web Services, we conclude that the essentials of achieving trustworthy SOA-Based Web Services are to accomplish the task of attestation and authorization of ultimate service endpoints, and to establish trustworthy service channel between them.

Furthermore, the XML-Based cryptographic techniques and contemporary strategy for achieving trustworthy SOA-Based Web Services are introduced. Specifically, WS-Security is detailed in terms of accomplishing essential tasks of achieving trustworthy SOA-Based Web Services.

Keywords: SOA, Web Services, Trustworthy, Security Token, WS-Security, XML

1 Introduction

Web Service is one of the hottest topics nowadays, it is modular software applications built to run over the Internet, basing on open standards for definition, discovery, and interoperability, by enterprises for rapidly responding to changing market, enhancing competitive power, and pursuing business profit. Because Service-Oriented Architecture (SOA) is capable of simplifying the application development and integration by providing an architectural framework for building, integrating and deploying, using the same set of standards that enable basic Web Services, SOA is suggested and recommended as underlying architecture for establishing self descriptive, reusable, and loosely coupled Web Services.

As enterprises today increasingly depend on Information Technology (IT) to help drive business innovation and provide competitive differentiation, many are turning to SOA-Based Web services for creating flexible and agile IT infrastructures necessary to be able to be responsive to changing business demands. According to Forrester Research, by the end of 2005, 89% of large enterprises, 61% of medium-sized enterprises and 40% of small to midsized businesses were expected to be using SOA.

With the advent of SOA, Web Service, and especially their wide acceptance and adoption, great progress has been achieved, however, it is not the time to celebrate. It is a long way to the ultimate success, i.e. achieving secure and trusted environment for business over the Internet. Following reasons always account for the further and necessary tasks, which should be accomplished for preventing the IT infrastructures from attacks or malicious services.

- The exchange of business information over the Internet – an untrusted public network – gives rise to considerable security concerns.
- It is nearly impossible for the enterprises to consistently implement the best security practices and enforce security policies across the extended enterprises nowadays, due to the limits of current technologies. (The enterprises should struggle to achieve the best, instead of being pessimistic.)
- The SOA-Based Web Services make it even easier for the enterprises to expose business processes and critical resources to the hostile outside environment, potentially increasing security risks.

These factors are widely believed to be the major factors inhibiting the successful implementation of SOA and the deployment of Web Services, and therefore it is urgent for the industry to develop and implement effective security measures for achieving secure SOA-Based

Web Services, and for business success as the ultimate goal.

2 Anatomy of SOA-Based Web Services

2.1 SOA-Based Architecture and its Underlying Technologies

Fig. 1 illustrates the architecture of SOA-Based Web Service, consisting of three different entities, i.e. service repository, service requestor, and service provider, and three distinct operations, i.e. creating, provisioning, and using Web Services respectively. Three main technologies are developed specifically for implementing SOA-Based Web Services, i.e. WSDL, UDDI and SOAP.

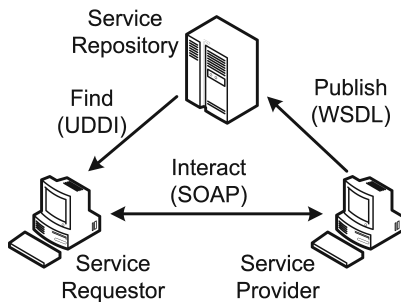


Figure 1. Architecture of SOA-Based Web Service

WSDL : The interface of a Web Service is usually described using the Web Services Description Language(WSDL) [1]. A WSDL document describes a Web Service as “a set of endpoints operating on messages containing either document-oriented or procedure-oriented information”. It abstractly describes messages and operations, i.e. what messages are the input or output of what operation, then specifies what concrete endpoints offer the described operations and over which network protocols these concrete endpoints can be reached.

UDDI : The Universal Description Discovery and Integration (UDDI) [2] offers a discovery mechanisms to locate the needed Web Services. The common example is that a Web Service publishes its existence and interfaces (defined with WSDL) to a UDDI server, whereas potential users can discover the service using the UDDI registry.

SOAP : Simple Object Access Protocol (SOAP) [3] describes how XML messages [4] can be exchanged

in service-oriented architectures. The basic building block of SOAP is a stateless and one-way message exchange, which can be easily extended to request–response and request–multiple-response message exchanges.

2.2 Transmission Model of SOA-Based Web Services

We can conclude the transmission model of SOA-Based Web Services as depicted in Fig. 2. It is evident that service interaction in SOA-Based Web Services incorporates two types of implementing blocks, i.e. application-oriented block and connection-oriented block. The former block guarantees the flexibility and freedom for the enterprise to be responsive to changing market and business, and the later block ensure the enterprise to leverage IT infrastructures already created and implemented, namely Web Services can be delivered with existing infrastructures.

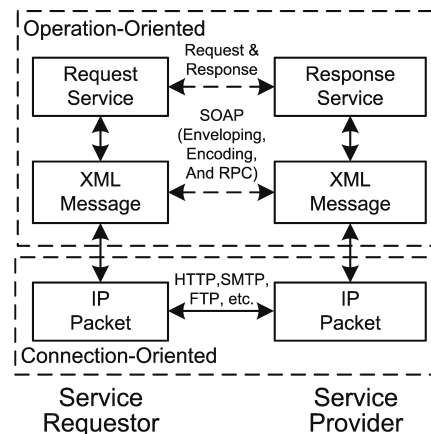


Figure 2. Transmission Model of SOA-Based Web Services

Obviously, these two types of implementing blocks can therefore achieve end to end security and point to point security respectively, with related strategies being implemented (e.g. SSL and IPSec for the connection-oriented block). As far as how to secure SOA-Based Web Services is concerned, it is the most important to take effective measures for securing element-wise service interaction, which is taken place in application-oriented block.

2.3 Anatomy of SOAP Message

In a SOA-Based Web Service, as shown in Fig. 1 and Fig. 2, SOAP messages are exchanged through multi hops (ultimate service endpoints and intermediaries). A

SOAP message is an envelope that has child elements of an optional header and a mandatory body containing core service information. The basic form of a SOAP message is identified as an XML 1.0 document, as illustrated with the XML document below, with `Envelope` element qualified with the namespace `http://www.w3.org/2003/05/soap-envelope`.

```
<?xml version='1.0' ?>
<env:Envelope xmlns:env=
  'http://www.w3.org/2003/05/soap-envelope' ?>
<env:Header>
...
</env:Header>
<env:Body>
...
</env:Body>
</env:Envelope>
```

Two key parts of a SOAP message are message header and message body, which are described as follows.

1. **SOAP Message Header:** which is optional. It contains zero or more namespace-qualified elements, `<wsse:Security>`, which will be discussed in Section 4.2.1. Unlike the SOAP message body, which may not be modified, the message header gives all intermediaries the right of accessing (read and write) to its child elements.

Each header element, `<wsse:Security>`, will be processed by at most one service node, however, others may examine header elements not targeted at them. A service node can add header elements for subsequent service nodes before forwarding the SOAP message, however, it should delete any header elements targeted at it.

2. **SOAP Message Body:** which is classified as SOAP request and SOAP response according to the functional characteristics of the message.

SOAP Request: which contains zero or more child elements. If multiple child elements are present, they represent a single unit of work, multiple units of work, or some combination of work and data. Message elements are analogous to paper documents that have an understood XML-Based structure.

SOAP Response: which may contain a document, an RPC response, or a SOAP fault. SOAP fault is a special case of SOAP response, and is the only body child element that is defined in SOAP specification [3]. A SOAP fault is generated in response to errors or to carry other status information.

3 Achieving Trustworthy SOA-Based Web Services

Section 2 gives us the minimum of needed requisites for understanding the security problems facing the Web Services, and this section will detail the needed security services for solving related security problems, and conclude the essentials of achieving trustworthy Web Services, which is valuable for developing effective techniques and strategies for securing the Web Services in practice.

3.1 Needed Security Services

It is required that both the service requestor and the service provider are legitimate communication parties, privacy & integrity of service information exchanged between them (through multi-hops and across the Internet) should be guaranteed, and effective measures should be taken to frustrate possible attacks originated by the adversaries. Needed security services are classified as follows.

- Identity legitimacy of two ultimate service endpoints should be assured.
- Confidentiality & integrity of service messages exchanged should be guaranteed.
- Policy for managing authorization and access control should be implemented.
- Furthermore, it is highly possible that IT infrastructures may be attacked by the adversaries, especially in nowadays-networked environment. Following two factors always account for the successful attacks.

1. Vulnerabilities of the IT infrastructures,
2. Tricks and malicious intend (money theft, identity theft, etc.) of the adversaries.

Therefore, the power to fighting attacks, for example replay attack, denial-of-service, phishing attack and so on, is also required for securing SOA-Based Web Services.

3.2 Essentials of Securing SOA-Based Web Services

According to the “Trustworthy” notion presented in references [5][6], we can conclude that following tasks should be accomplished for successfully securing IT infrastructures, namely to implement effective measures for providing the needed security services mentioned

above, or for solving security problems facing nowadays networked information systems (e.g. SOA-Based Web Services).

1. Constructing trustworthy networked endpoints via proper mechanisms for identity authentication and authorization, which assures identity legitimacy of intending communication parties and proper right allocation.
2. Establishing trustworthy communication channel via effective security strategies (protocols), which guarantees the trustworthy transmission, i.e., data privacy & integrity, and power to fighting attacks.

In the same way, the essential tasks of securing SOA-Based Web Services are to complete source authentication and/or peer identity authentication, and to establish trustworthy service channel between service request application and service response application as shown in the transmission model depicted in Fig. 2.

Fig. 3 depicted the scenario of anticipated trustworthy service interaction in SOA-Based Web Services.

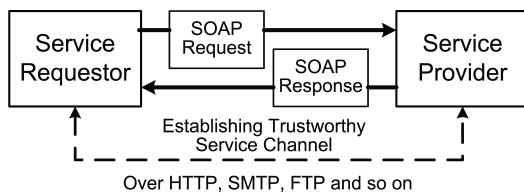


Figure 3. Anticipated Trustworthy Service Interaction

4 Contemporary Techniques and Strategy for Securing SOA-Based Web Services

This section aims at developing effective strategies for securing SOA-Based Web Services, precisely, it will introduce the contemporary techniques and strategies targeting at achieving trustworthy SOA-Based Web Services.

4.1 Underlying XML-Based Cryptographic Techniques

There are various XML-based security specifications emerging in the industry, which are especially developed for manipulating XML-Based messages or documents (SOAP message is a qualified XML document). The key specifications or techniques underlying XML-based security framework are (not limited to) as follows.

4.1.1 XML Encryption

The XML Encryption [7] is defined by W3C (World Wide Web Consortium) and addresses the issue of data confidentiality. Encrypted data and other necessary information are embedded in a specifically defined structure, `EncryptedData`, according to the specification. `CipherData` is the mandatorily required child element, possibly encoded in base64, or a reference to the encrypted object. Elements `EncryptionMethod`, `KeyInfo`, and `EncryptionProperties` are all optional in `EncryptedData`. Child elements of `EncryptedData` are concisely explained as follows.

- `CipherData`, which contains the resulting encrypted data or a reference to the encrypted object.
- `EncryptionMethod`, which specifies the encryption algorithm (and the related key size).
- `KeyInfo`, which provides the information needed by the recipient to decrypt the cipher data. The receiving entity is assumed to know how to perform the decryption, if this child element is omitted.
- `EncryptionProperties`, which holds additional information related to the encryption.

The following code fragment is an instance of `EncryptedData`. The first line indicates that the “element value” is being replaced, and identifies this encryption for ease of being referred. The identifier of this encryption will be referred for notifying the recipient (decryption entity) the key being utilized for the encryption (detailed in Section 4.2.3), and therefore `KeyInfo` element is omitted. `EncryptionMethod` specifies the encryption algorithm is triple-DES with Cipher Block Chaining confidentiality mode, and the encrypted data is represented with `CipherValue` element.

```
<xenc:EncryptedData
Type='WebLink/xmlenc#Element'
wsu:Id='encl1'>
  <xenc:EncryptionMethod Algorithm=
'WebLink/xmlenc#tripledes-cbc' />
  <xenc:CipherData>
    <xenc:CipherValue>
d2FpbmdvbGRfE0lm4byV0...
    </xenc:CipherValue>
  </xenc:CipherData>
</xenc:EncryptedData>
```

Note: “<http://www.w3.org/2001/04>” is replaced by “WebLink” for neat illustration.

4.1.2 XML Signature

The XML Signature [8] is the specification initiated and defined by W3C and IETF (Internet Engineering Task Force), which provides element-wise data integrity and authentication.

A signature is contained in the structure `Signature`, which is made up of two required child elements, `SignedInfo` and `SignatureValue`, and two optional child elements, `KeyInfo` and `Object`. Child elements of an XML Signature are explained as follows.

- `SignedInfo`, which specifies the algorithms for producing signature, including the algorithm for calculating message digest and algorithm for signing the data, lists the references (one or more) to the data being signed, and records the value of the message digest calculated by applying specified digest algorithm to the referred data.
- `SignatureValue`, which represents the value of the resulting signature.
- `KeyInfo`, which provides the information needed by the recipient to validate the signature, such as a key identifier (label string), the signer's public key, and a reference to where the public key is available. It is assumed that the receiving entity knows how to validate the signature, if this element is omitted.
- `Object`, as with the `EncryptionProperties` element described in XML Encryption, which holds additional information related to the signature.

The following code illustrates an example of XML signature. Element `SignedInfo` is composed of one `SignatureMethod` element, one `CanonicalizationMethod` element, and two `Reference` child elements.

`SignatureMethod` specifies the algorithms for signing and hashing, RSA and SHA-1, respectively. `CanonicalizationMethod` is specified for transforming XML document into standard representation before signing or verifying the document, please refer to [10] for more detail. The data being signed is located by the `URI` attribute of `Reference`, and two `Reference` elements refer to data of `T0` identifier and data of `body` identifier, respectively. `Reference` also contains the algorithm for message digest, indicated by `DigestMethod` (i.e. SHA-1), value of digest wrapped in `DigestValue`, applied the selected algorithm to the referred data. The method of transformation is specified in `Transforms`. The transformation process is necessary for rendering the data suitable for signature. `SignatureValue` is calculated based on `SignedInfo`. The related key information needed for verifying the signature is wrapped in `KeyInfo` element, in which a token identifier is specified.

```
<ds:Signature>
```

```
<ds:SignedInfo>
  <ds:CanonicalizationMethod Algorithm=
    ''Link2001/10/xml-exc-c14n#''/>
  <ds:SignatureMethod Algorithm=
    ''Link2000/09/xmldsig#rsa-shal1''/>

  <ds:Reference URI=''#T0''>
    <ds:Transforms>
      <ds:Transform Algorithm=
        ''Link2001/10/xml-exc-c14n#''/>
    </ds:Transforms>
    <ds:DigestMethod Algorithm=
      ''Link2000/09/xmldsig#sha1''/>
    <ds:DigestValue>LyLsF094hPi4wPU...
    </ds:DigestValue>
  </ds:Reference>

  <ds:Reference URI=''#body''>
    <ds:Transforms>
      <ds:Transform Algorithm=
        ''Link2001/10/xml-exc-c14n#''/>
    </ds:Transforms>
    <ds:DigestMethod Algorithm=
      ''Link2000/09/xmldsig#sha1''/>
    <ds:DigestValue>LyLsF094hPi4wPU...
    </ds:DigestValue>
  </ds:Reference>
</ds:SignedInfo>

<ds:SignatureValue>
Hp1ZkmFZ/2kQLXDJbchm5gK...
</ds:SignatureValue>

<ds:KeyInfo>
  <wsse:SecurityTokenReference>
    <wsse:Reference
      URI=''#X509Token''/>
    </wsse:SecurityTokenReference>
  </ds:KeyInfo>
</ds:Signature>
```

Note: "http://www.w3.org/2000" is replaced by "Link2000" and "http://www.w3.org/2001" is replaced by "Link2001" for neaty illustration respectively.

4.2 Contemporary Strategy for Achieving Trustworthy SOA-Based Web Services

The WS-Security specification [9] is one of standards defined by OASIS (precisely, it is outlined, developed, and driven by a collaboration between IBM and Microsoft, and now being developed by OASIS consortium.) and provides a mechanism for addressing end-to-end security, specifically for data integrity, confidentiality, and data origin authentication features within a SOAP message by leveraging XML-Based cryptographic techniques described above. As well, WS-Security defines how to associate security tokens with SOAP messages.

4.2.1 Token Services for Proof of Identity and beyond

WS-Security defines a generic mechanism for associating security tokens with the message. Specifically it defines the header element, `<wsse:Security>`, as the container for conveying security-related information

with and about a SOAP message. “Associating security tokens” means that one or more tokens are included in `<wsse:Security>` element within the SOAP message header and that referencing mechanisms (e.g. ID Identifier) are introduced to refer to these tokens. Tokens generally are either identification or cryptographic material, i.e. a security token is a representation of security-related information, such as X.509 certificates, Kerberos tickets and so on. Therefore, security tokens can be utilized to identify service endpoint and assert its right.

WS-Security specification [9] defines following types of security tokens for associating them with the SOAP messages, and how they are attached to the messages.

- A simple username token, with use of `UsernameToken`, to pass the username and password (optional) if the Web service is using customer authentication.
- A container for arbitrary binary tokens (non-XML security token), with use of `BinarySecurityToken`, to pass X.509 Certificates and Kerberos Tickets.
- A container for XML-formatted tokens. For security tokens based on XML, the extensibility of the `<wsse:Security>` header allows for these security tokens to be directly inserted into the header.

4.2.2 Cryptographic Services for Data Privacy & Integrity and so on

WS-Security makes use of the XML Signature and XML Encryption and defines how to include digital signatures, message digests, and encrypted data in a SOAP message. Moreover, the specification goes beyond these two underlying specifications by tailoring them for manipulating SOAP messages.

Message confidentiality & integrity and authentication are provided by leveraging XML Encryption and XML Signature in conjunction with security tokens to ensure that messages are transmitted without modification, and to keep messages confidential.

4.2.3 Scenario of Trustworthy SOA-Based Web Services

As a result, WS-Security in essence is a systematic strategy for achieving trustworthy SOA-Based Web Services, which leverages token services and XML-Based cryptographic techniques mentioned in Section 4.1 for comprehensively providing security services needed for dealing with security problems facing Web Services. Fig. 4 depicted the common scenario of trustworthy SOA-Based Web Services.

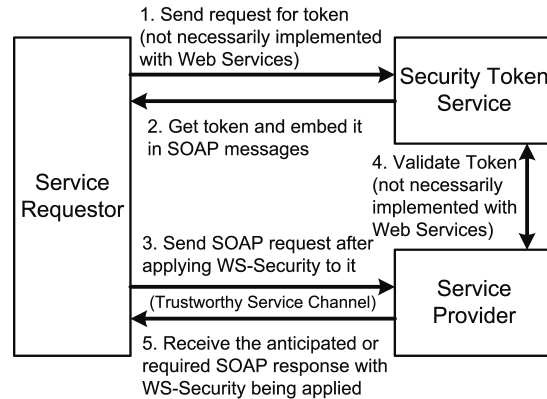


Figure 4. Scenario of Trustworthy SOA-Based Web Services

At last, let’s go through a WS-Security-Based SOAP message (taken from [9] for ease of understanding) with a `<wsse:Security>` header, signature, and encrypted content. For this example, the timestamp and the message body are signed prior to encryption. The entire body of the SOAP message is encrypted. Elements of the SOAP message are explained as follows.

- `BinarySecurityToken` element specifies an X.509 certificate that is encoded as Base64.
- `EncryptedKey` element specifies the key that is used to encrypt the body of the message. Since this is a symmetric key, it is passed in an encrypted form, and the algorithm used to encrypt the symmetric key is also specified in this element. `ReferenceList` is used within `EncryptedKey`, to identify encrypted elements within the message that are encrypted using this symmetric key.
- Signature element has been detailed in Section 4.1.2.
- The message body contain only `EncryptedData` element that has been detailed in Section 4.1.1.

```

<?xml version='1.0' encoding='utf-8'?>
<S11:Envelope xmlns:S11='...'
  xmlns:wsse='...' xmlns:wsu='...'
  xmlns:xenc='...' xmlns:ds='...'>
  <S11:Header>
    <wsse:Security>
      <wsu:Timestamp wsu:Id='T0'>
        <wsu:Created>
          2001-09-13T08:42:00Z</wsu:Created>
        </wsu:Timestamp>

      <wsse:BinarySecurityToken
        ValueType='...#X509v3'
        wsu:Id='X509Token'
  
```

```

    EncodingType='...#Base64Binary'>
    MIEZzCCA9CgAwIBAgIQEmtJZc0rqKh5i...
</wsse:BinarySecurityToken>

<xenc:EncryptedKey>
  <xenc:EncryptionMethod Algorithm=
  'W3C/2001/04/xmlenc#rsa-1_5' />
  <ds:KeyInfo>
    <wsse:KeyIdentifier
      EncodingType='...#Base64Binary'
      ValueType='...#X509v3'>MIGfMa0GCSq...
    </wsse:KeyIdentifier>
  </ds:KeyInfo>
  <xenc:CipherData>
    <xenc:CipherValue>
      d2FpbmdvbGRfE0lm4byV0...
    </xenc:CipherValue>
  </xenc:CipherData>
  <xenc:ReferenceList>
    <xenc:DataReference URI='#enc1' />
  </xenc:ReferenceList>
</xenc:EncryptedKey>

<ds:Signature>
  

Insert signature document exemplified in Section 4.1.2 here.


  </ds:Signature>
</wsse:Security>
</S11:Header>

<S11:Body wsu:Id='body'>
  <xenc:EncryptedData Type=
  'W3C/2001/04/xmlenc#Element'
  wsu:Id='#enc1'>
  

Insert encrypted document exemplified in Section 4.1.1 here.


  </xenc:EncryptedData>
</S11:Body>
</S11:Envelope>

```

Note: "http://www.w3.org" is replaced by "W3C" for neaty illustration.

5 Conclusion

In this paper, it is emphasized to complete two essential tasks for achieving trustworthy SOA-Based Web Services, and it is suggested to implement two types of security services provided (defined) by WS-Security, i.e. token service and cryptographic service, for eventually accomplishing these two tasks. In a word, essential tasks of achieving trustworthy SOA-Based Web Services are accomplished as follows.

1. Trustworthy Service Endpoint: Tasks of authentication and authorization can be accomplished with proper token profiles being implemented. WS-Security allows use of existing token-related technologies, such as X.509 Certificates, Kerberos shared-secret tickets and even customer authentication.
2. Trustworthy Service Channel: Data privacy & integrity, and message authentication can be achieved with use of security tokens, by leveraging XML-Based cryptographic techniques, mainly including

XML Encryption and XML Signature, and especially tailoring these underlying XML-Based techniques for SOAP messages.

Acknowledgments

This work was supported by Natural Science Foundation of Hebei province, China, under Grant No.F2005000515.

References

- [1] W3C Web Services Description Language (WSDL) 1.1, March 2001. <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>
- [2] OASIS universal description, discovery and integration (UDDI), October 2003. <http://uddi.org/pubs/uddi-v3.htm>
- [3] SOAP Version 1.2 Part 1: Messaging framework, June 2003. <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>
- [4] Extensible Markup Language (XML) 1.0, 3rd ed. February 2004. <http://www.w3.org/TR/2004/REC-xml-20040204/>
- [5] Trusted Computing Group. <https://www.trustedcomputinggroup.org/>
- [6] Microsoft Corporation, NGSCB. <http://www.microsoft.com/resources/ngscb/default.aspx>
- [7] XML Encryption Syntax and Processing, December 2002. <http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/>
- [8] XML-Signature Syntax and Processing, February 2002. <http://www.w3.org/TR/2002/REC-xmlsig-core-20020212/>
- [9] WS-Security: SOAP Message Security 1.0 (WS-Security 1.0), March 2004. Available at: <http://docs.oasisopen.org/wss/2004/01/>
- [10] Exclusive XML Canonicalization Version 1.0, July 2002. <http://www.w3.org/TR/xml-exc-c14n/>