Comments on:

Web Services Security: SOAP Message Security, Working Draft 14, Monday, 30 June 2003.

Tim Moses, 7 July 2003

Editorial 1

Abstract:
This specification describes enhancements to ~~the~~ SOAP messaging to provide ~~quality of protection through~~ message integrity, and single message authentication. The~~se~~ specified mechanisms can be used to accommodate a wide variety of security models and encryption technologies.

Editorial 2

This specification also provides a general-purpose mechanism for associating security tokens with message~~s~~ content. No specific type of security token is required; ~~it~~ the specification is designed to be extensible (e.g. support multiple security token formats). For example, a client might provide one format for proof of identity and provide another format for proof that they have a particular business certification.
Additionally, this specification describes how to encode binary security tokens, a framework for XML-based tokens, and ~~describes~~ how to include opaque encrypted keys. It also includes extensibility mechanisms that can be used to further describe the characteristics of the tokens that are included with a message.

Editorial 3

1 Introduction

This specification proposes a standard set of SOAP extensions that can be used when building secure Web services to implement message ~~level~~ content integrity and confidentiality. This specification refer s to this set of extensions as the “Web Services Security Core Language” or “WSS- Core”. This specification is flexible and is designed to be used as the basis for securing Web services within a wide variety of security models including PKI, Kerberos, and SSL. Specifically, this specification provides support for multiple security token formats, multiple trust domains, multiple signature formats, and multiple encryption technologies. The token formats and semantics for using these are defined in the associated profile documents.
This specification provides three main mechanisms: ability to send a security token as part of a message, message integrity, and message confidentiality. These mechanisms by themselves do not provide a complete security solution for Web services. Instead, this specification is a building block that can be used in conjunction with other Web service extensions and higher-level application-specific protocols to accommodate a wide variety of security models and security technologies.
These mechanisms can be used independently (e.g., to pass a security token) or in a tightly coupled manner (e.g., signing and encrypting a message or part of a message and providing a security token or token path associated with the keys used for signing and encryption).

Editorial 4

1.1.1 Requirements

The Web services security language must support a wide variety of security models. The following list identifies the key driving requirements for this specification:

- Multiple security token formats
- Multiple trust domains
- Multiple signature formats
- Multiple encryption technologies
- End-to-end message content ~~level~~ security and not just transport-level security

Editorial 5

Message Confidentiality - *Message Confidentiality* is a property of the message and encryption is the ~~service or~~ mechanism by which this property of the message is provided. **Message Integrity** - *Message Integrity* is a property of the message and digital signature is the ~~service or~~ mechanism by which this property of the message is provided.

Editorial 6

Where the specification requires that ~~the~~ an element~~s~~ be "processed" ~~this~~ it means that the element type MUST be recognized ~~well enough to return~~to the extent that an appropriate error is returned if the element is not supported.

Editorial 7

Message integrity is provided by ~~leveraging~~ XML Signature in conjunction with security tokens to ensure that modifications to messages are ~~received without modifications~~detected. The integrity mechanisms are designed to support multiple signatures, potentially by multiple SOAP roles, and to be extensible to support additional signature formats.

Editorial 8

This document ~~also~~ does not specify any signature appearing outside of `<wsse:Security>` element~~, if any~~.

Editorial 9

The message recipient SHOULD reject a message with an invalid signature ~~determined to be invalid~~, a message that is missing necessary claims and a message whose claims have unacceptable values ~~or unacceptable claims~~ as ~~it is an~~ such messages are unauthorized ~~(~~or malformed~~) message~~.

Editorial 10

processing of the security semantics. That is, they need only "know"

Editorial 11

message either ~~to~~ have or must be able to obtain

Technical 1

503 All compliant implementations MUST be able to process a `<wsse:UsernameToken>`
element.

If this means that ell compliant implementations have to incorporate password management functions, this seems like an unreasonable demand. If it means something else, then it should be clarified. It should be the job of implementation profiles to indicate what functions must be supported by a profile-compliant implementation.

Editorial 12

550 the binary data (e.g., `wsse:Base64Binary`). A new attribute is introduced, as there are
551 issues

Technical 2

558 All compliant implementations MUST be able to support a
`<wsse:BinarySecurityToken>`
559 element.

See comment Technical 1, above.

Editorial 13

580 ~~Subsequent~~ Profile specifications describe rules and processes

Editorial 14

587 However, specific extensions MAY be made to the `wsse:SecurityTokenReference>`
588 element.

The significance of this sentence is not clear. This paragraph should provide more guidance on how to attach an identifier to an XML token.

Editorial 15

613 attribute does not indicate the ID of what is being referenced, that ~~is~~ SHALL be done using a
614 fragment URI in a `<Reference>` element within the `<SecurityTokenReference>`

Editorial 16

634 There are several challenges that implementations face when trying to interoperate. ~~In order to~~
635 P~~p~~rocessing the IDs and references requires the recipient to *understand* the schema. This may be an
636 expensive task and in the general case impossible as there is no way to know the "schema
637 location" for a specific namespace URI. As well, the primary goal of a reference is to uniquely
638 identify the desired token. ID references are, by definition, unique by XML. However, other
639 mechanisms such as "principal name" are not required to be unique and therefore such
640 references may not be unique.

Editorial 17

785 Demonstrating knowledge of a confirmation key associated with a token key - claim ~~supports~~
786 confirms~~ing~~ the ~~other~~ accompanying token claims. Knowledge of a confirmation key may be demonstrated using ~~a~~ that

787 key to create an XML Signature, for example. The relying party acceptance of the claims may
788 depend on its confidence in the token-. Multiple tokens may ~~have~~ contain a key-claim for a signature and
789 may be referenced from the signature using a SecurityTokenReference. A key-claim ~~can~~ may be an
790 X.509 Certificate token, or a Kerberos service ticket token (to give just two examples).

Editorial 18

815 Finally, if a sender wishes to sign a message before encryption, they should alter the order of the
816 signature and encryption elements inside of the `<wsse:Security>` header.

This sentence is quite unclear.

Editorial 19

821 within ~~the~~ one `<wsse:Security>` header block. Senders SHOULD take care to sign all important
822 elements of the message, but care MUST be taken in creating a signing policy that ~~will not to~~requires signing of
823 parts of the message that might legitimately be altered in transit.

Editorial 20

839 exercise care that their transformations do not ~~occur within the scope of~~affect a digitally signed

Editorial 21

858 referenced by the `<SecurityTokenReference>` element not the `<SecurityTokenReference>` element itself-.

Editorial 22

echoed, but instead, it is used to locate the~~a~~ token(s) matching the criteria and rules define~~d~~s by the 861

Editorial 23

958 party MUST either prepend the sub-element ~~in~~to ~~the~~ an existing `<wsse:Security>` header block for the ~~targeted~~ intended
959 recipient~~s that is expected to decrypt these encrypted portions~~ or create a new `<wsse:Security>` header block and insert the sub-element. The combined process of
960 encrypting portion(s) of a message and adding one of these ~~a~~ sub-elements ~~referring to the~~
961 ~~encrypted portion(s)~~ is called an encryption step hereafter. The sub-element ~~should~~ MUST contain
962 ~~enough~~ the information necessary for the recipient to identify ~~which~~ the portions of the message ~~are to be~~ that it is able to decrypt~~ed~~
963 ~~by the recipient~~.

Editorial 24

966 ~~When encrypting elements or element contents within a SOAP envelope, t~~The

Editorial 25

972 <xenc:DataReference> elements inside ~~an~~ one or more <xenc:ReferenceList> element.

Editorial 26

973 Although in XML Encryption, <xenc:ReferenceList> ~~is~~ was originally designed to be used within

Editorial 27

9.2 xenc:EncryptedKey

Need to indicate that multiple symmetric keys may be involved if different recipients are to have access to different message elements.

Alternatively, why even have this discussion here? The Kerberos and X.509 profiles should handle these issues thoroughly.

Technical 3

1058 into a single <Security> header block if they are targeted for the same recipient.

What is supposed to happen if they are not intended to be accessed by the same recipient? They can only go in separate <Security> headers if the recipients can be assigned to separate roles. This may not be the possible.

Editorial 28

1059 When an element or element content inside a SOAP envelope (e.g. ~~of~~ the contents of the <S:Body> element)

Editorial 29

1063 attachment. For example, if an <xenc:EncryptedData> element in an <S:Body> element has an
1064 <xenc:CipherReference> element that refers to an attachment, then the decrypted octet stream

Editorial 30

1092 9.3.2 Decryption

The process should start with identifying any decryption keys that are in the recipient's possession, then identifying any message elements that it is able to decrypt. Just because an element is encrypted, it does not mean that an individual recipient should expect to be able to decrypt it.

Technical 4

10 Security Timestamps

Recognition should be given to the idea of a timestamp from a trusted third party.