GWD-R (draft-gwdrp-ogsa-secure-communication-007)
Category: Recommendation
Open Grid Services Architecture (OGSA®) Working Group

Duane Merrill, UVa

December 27, 2007

Secure Communication Profile 1.0

Status of This Document

This document provides a recommendation to the Grid community on how to secure communications with Web service resources. This profile describes precisely the requirements placed on secure-communication mechanisms and their descriptions to ensure interoperability. Distribution is unlimited.

Copyright Notice

Trademarks

ABSTRACT

This document is an interoperability profile for the secure communication with Web service resources. The requirements stated in this profile are concerned with security mechanisms that can be used to ensure authentication, integrity and confidentiality properties for interaction with such resources. This document serves three primary purposes:

- To provide a point of further refinement for commonly-used security mechanisms profiled within the *WS-I Basic Security Profile 1.0* [WS-I BSP]
- To profile the *WS-Security Policy 1.2* [WS-SecurityPolicy] language to accommodate the inclusion of actual security tokens within policy documents
- To define normative, referenceable, composable policy documents identifying commonly-used security mechanisms.

CONTENTS

1 INTRODUCTION

This document defines the *Secure Communication Profile 1.0* (hereafter, “the Profile”), a set of conformance statements in order to facilitate interoperability with Web service resources having secure communication requirements. The term *resource* is used within the context of this document to connote any logical message-processing entity. (Though the WS-Addressing specification, a single Web service endpoint may expose multiple logical resources.)

Normative profiles are useful tools for understanding and defining the interaction amongst existing Web services specifications in order to achieve interoperability. They are particularly important within the context of secure communication: common treatment of Web services security and addressing specifications (e.g., SSL/TLS, WS-Security and related token profiles, XML-Encryption, XML-Signature, WS-Addressing, etc.) is crucial for real-world interoperability.

More specifically, this profile defines normative policy documents identifying commonly-used secure communication mechanisms and their particulars. These “well-known” policy documents can be referenced by name and composed within resource-specific security policies. The security mechanisms implied by these named policies are well-defined by external profiles that are incorporated by reference, and this document serves as a point of further refinement as necessary for these mechanisms.

By itself, this document is not sufficient to guarantee interoperability of all compliant Web service clients and resources. The purpose of this document is to provide normative profiles of well-known secure-communication mechanisms and their policy descriptions. The Profile does not establish a “lowest-common-denominator” set of security mechanisms that must be supported by all compliant resources, nor is it concerned with any particular vehicle by which a resource’s security policy can be discovered. Rather, the Profile adopts the view that specific secure communication requirements may vary between communities of resource providers and consumers. The intent is for a community to self-select such requirements that are appropriate and then leverage this Profile to achieve interoperability between its members (and/or cleanly discover where interoperability is not possible).

The secure-communication mechanisms referenced within the Profile are tools that are intended to address one or more of the following security properties:

- *Authentication.* It is important to ensure communicating parties that they are indeed communicating with each other and not with imposter(s). This is typically accomplished by having each party cryptographically prove a “fact” about themselves to the other. Although these authenticatable facts are typically in the form of cryptographic identities (e.g., X.509 certificates), other tokens that represent attributes or privileges are equally reasonable. Authentication may be performed at the underlying transport-layer or the SOAP message-layer, or in combination.

  Such authentication facts are often used to facilitate the processes of *authorization* and *auditing*. Authorization and auditing are governed by implementation- and instance-specific policies and are thus out of scope of the Profile. The Profile concerns itself with security tokens in as much as they affect the underlying transport protocol or the SOAP message format.

  For example, security token *type* affects message format, and should be conveyed within the WS-SecurityPolicy documents that describe the communication requirements for a given resource. In some cases, a resource may also use WS-SecurityPolicy to convey additional token *claims*: hints of what must be represented by a given token in order for successful authorization. Token claims are out of scope of the Profile.

- *Integrity.* The Profile accommodates communication scenarios that require that message data be protected in a way that reveals any evidence of tampering. Secure transport-layer protocols can ensure integrity between transport endpoints. In the event that the end-to-end notions of the transport-protocol don't match those of the SOAP message exchange, integrity should be ensured at the message level.

- *Confidentiality.* The Profile accommodates communication scenarios that require that message data not be exposed to third-parties while in transit. Secure transport-layer protocols can ensure confidentiality between transport endpoints. In the event that the end-to-end notions of the transport-protocol don't match those of the SOAP message exchange, confidentiality should be ensured at the message level.

The *WS-SecurityPolicy 1.2* [WS-SecurityPolicy] specification defines a base set of assertions that describe how Web services messages are to be secured. It is an extension of the *Web Services Policy 1.5 – Framework* (WS-Policy), which is a flexible grammar for expressing capabilities, requirements, and general characteristics of Web services-based entities. WS-SecurityPolicy provides a flexible, extensible approach for defining token requirements, cryptographic algorithms, and mechanisms (both at the transport and message levels).

WS-SecurityPolicy uses the notion of "token assertions" to specify the type and usage of security tokens within a message. Unfortunately there is no provision for the embedding of an actual token within a policy's token assertion. The ability for a security policy document to encapsulate actual security tokens is desirable for key-distribution: policy documents are intended for distribution to resource consumers via WS Addressing EPRs or service WSDL. This profile extends the WS-SecurityPolicy specification to enable the inclusion of actual security tokens themselves (e.g., keys, certificates, etc.) within security policy documents.

The remainder of this profile is organized as follows. Section 2, "Document Conventions," describes notational conventions utilized by the Profile. Section 3, "Profile Conformance," explains what it means to be conformant to the Profile. Section 4 describes the extensions to Ws-SecurityPolicy to facilitate the direct inclusion of security tokens within security policy documents. Section 5 describes the global requirements and recommendations put forth by the Profile. Sections 6 and 7 define "well-known", composable transport- and message- level security mechanism profiles, respectively. Section 8 presents an example SOAP message. Note that there is no relationship between the section numbers in this document and those in the referenced profiles and specifications.

2 DOCUMENT CONVENTIONS

This Profile is a *Recommended Profile as Proposed Recommendation,* as defined in the OGSA Profile Definition [OGSA Profile Definition]. Additional document conventions of the Profile are defined normatively in *WS-I Basic Profile 1.1* [WS-I BP], and are briefly summarized below.

2.1 Notational Conventions

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [RFC2119].

Normative statements of requirements in the Profile (i.e., those impacting conformance, as outlined in Section 3, “Conformance Requirements") are presented in the following manner:

> *Rnnnn Statement text here*.

where "*nnnn*" is replaced by a number that is unique among the requirements in the Profile, thereby forming a unique requirement identifier.

Extensibility points in underlying specifications are presented in a similar manner:

> *Ennnn Extensibility Point Name - Description*

where "*nnnn*" is replaced by a number that is unique among the extensibility points in the Profile.

This specification uses a number of namespace prefixes throughout; their associated URIs are listed in the table below:

Table 1 Namespaces used by the Secure Communication Profile

Prefix	Namespace	Specification(s)
ds	http://www.w3.org/2000/09/xmldsig#	[XML-DigSig]
wsse	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd	[WS-S]
wsu	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd	[WS-S]
wsa	http://www.w3.org/2005/08/addressing	[WS-Addressing]
wsp	http://schemas.xmlsoap.org/ws/2004/09/policy	[WS-Policy], [WS-PolicyAttachment]
sp	http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702	[WS-SecurityPolicy]
comm	http://www.ogf.org/ogsa/2007/05/secure-communication	This Document

2.2 Security Considerations

In addition to interoperability requirements (which are made in *Rnnnn* statements and intended to improve interoperability), the Profile makes a number of security considerations intended to improve security. These Security Considerations are presented as follows:

> *Cnnnn Statement text here.*

where "*nnnn*" is replaced by a number that is unique among the security considerations in the Profile, thereby forming a unique security consideration identifier. Each security consideration contains a *SHOULD* or a *MAY* to highlight exactly what is being considered; however, these considerations are informational only and are non-normative.

2.3 Profile Identification and Versioning

This document is identified by a name (in this case, *Secure Communication Profile*) and a version number (here, 1.0). Together, they identify a particular profile instance. Version numbers are composed of a major and minor portion, in the form "major.minor". Version numbers indicate profile instance precedence: higher version numbers indicate a more recent instance that supersedes earlier instances.

3 PROFILE CONFORMANCE

Conformance to the Profile is defined by adherence to the set of requirements defined for a specific target, within the scope of the Profile. This section explains these terms and describes how conformance is defined and used.

3.1 Conformance Requirements

Requirements state the criteria for conformance to the Profile. They typically refer to an existing specification and embody refinements, amplifications, interpretations and clarifications to it in order to improve interoperability. All requirements in the Profile are considered normative, and those in the specifications it references that are in-scope (see Section 3.3, “Conformance Scope") should likewise be considered normative.

Each requirement is individually identified (e.g., R9999) for convenience.

For example;

R9999 Any WIDGET SHOULD be round in shape.

This requirement is identified by "R9999", applies to the target WIDGET (see below), and places a conditional requirement upon widgets; i.e., although this requirement must be met to maintain conformance in most cases, there are some situations where there may be valid reasons for it not being met (which are explained in the requirement itself, or in its accompanying text).

3.2 Conformance Targets

Conformance targets identify what artifacts (e.g., SOAP message, WSDL description, UDDI registry data, etc.) or parties (e.g., SOAP processor, end user, etc.) that the requirements stated within this Profile apply to.

This allows for the definition of conformance in different contexts, to assure unambiguous interpretation of the applicability of requirements, and to allow conformance testing of the specific artifacts (e.g., *POLICY, POLICY_ALTERNATIVE*) and parties (e.g., *INITIATOR*, *SENDER*) defined below.

The Profile discusses elements defined within the *WS-SecurityPolicy 1.2* [WS-SecurityPolicy] profile. The following conformance targets are inherited from those in the WS-SecurityPolicy:

- *POLICY* - A collection of *POLICY_ALTERNATIVEs*. A `<wsp:Policy>` element is used in conjunction with its child `<wsp:ExactlyOne>` element to indicate a policy expression as a union of *POLICY_ALTERNATIVEs*. If there are no children of `<wsp:ExactlyOne>`, there are no admissible policy alternatives (i.e., no behavior is admissible). If there is only one logical *POLICY_ALTERNATIVE*, the compact policy form can be used in which the requisite *POLICY_ASSERTIONs* are placed as direct children of the `<wsp:Policy>` element and the `<wsp:ExactlyOne>` and `<wsp:All>` elements are omitted.
- *POLICY_ALTERNATIVE* - A cohesive collection of *POLICY_ASSERTION*s represented by a `<wsp:All>` element. The `<wsp:All>` element is a child of `<wsp:ExactlyOne>` and indicates a group of *POLICY_ASSERTIONs*. If there are no children of `<wsp:All>`, this is an admissible policy alternative that is empty (i.e., no behavior is specified).
- *POLICY_ASSERTION* - An individual requirement, capability, other property, or a behavior. (E.g., the `<sp:SignedParts>` element is a *SECURITY_BINDING_ASSERTION* indicating which portions of a document are to be signed.)
- *SECURITY_BINDING_ASSERTION* - A *POLICY_ASSERTION* that identifies the type of security binding being used to secure an exchange of messages. A security binding is a

set of properties that together provide enough information to secure a given message exchange.

- *TOKEN_ASSERTION* - A *POLICY_ASSERTION* that describes a token requirement. Token assertions defined within a *SECURITY_BINDING_ASSERTION* are used to satisfy protection requirements.
- *PROFILED_MECHANISM* – A “well-known”, referenceable *POLICY* document containing a *POLICY_ASSERTION.* See Appendix B for *PROFILED_MECHANISM*s defined by this Profile.

This Profile defines the following conformance targets:

- *INSTANCE* – Software that implements a `<wsdl:port>`.
- *RESOURCE* – A logical message-processing *RECIPIENT*, identifiable with an WS-Addressing endpoint reference (EPR). (A *RESOURCE* may have a different cryptographic identity than the *INSTANCE* on which it resides, e.g., when multiple stateful resources are hosted within the same Web services container.)
- *INITIATOR* – The role sending the *initial* message in a message exchange.
- *SENDER* – The role sending a message in a message transfer.
- *RECIPIENT* - The targeted role to process a message in a message transfer. (In the case of a response message transfer, the *INITIATOR* is the *RECIPIENT* and the *RESOURCE* is the *SENDER*.)
- *RESOURCE_SECURITY_POLICY* – A *POLICY* document in conformance with the WS-SecurityPolicy refinements defined by this Profile.
- *RECIPIENT_TRANSPORT_IDENTITY* – a `<wsse:SecurityTokenReference>` placed within the `<wsa:Metadata>` element of an endpoint reference containing an embedded binary security token of type `X509v3` as defined in the *Web Services Security: X.509 Token Profile* [WS-S: X509 TP]. The binary security token must be identified with an `wsu:Id='RecipientTransportIdentity'` attribute.
- *RECIPIENT_MESSAGE_IDENTITY* – a `<wsse:SecurityTokenReference>` placed within the `<wsa:Metadata>` element of an endpoint reference containing an embedded binary security token of type `X509v3`, `PKCS7`, or `X509PKIPathv1` as defined in the *Web Services Security: X.509 Token Profile* [WS-S: X509 TP]. The binary security token must be identified with an `wsu:Id='RecipientMessageIdentity'` attribute.
- *CRITICAL_SIGNING* – *SENDER* signing of the following SOAP message elements in accordance with Section 8 of the WS-I BSP:
  - The entire `<soap:body>` message body.
  - Any WS-Addressing 1.0 – SOAP Binding [WSA-SOAP] message addressing property header elements.
- *CRITICAL_ENCRYPTION* – *SENDER* encryption of the entire `<soap:body>` message body in accordance with Section 9 of the WS-I BSP.
- *MESSAGE_PASSING_INTERMEDIARY* – A message-forwarding *INSTANCE* that receives a message for which it is not the ultimate *RECIPIENT* for the message body.
- *SERVER_TLS* – A normative POLICY document indicating server-authenticated transport layer security.
- *SERVER_TLS_CERT_PROVIDED* – A normative *POLICY* document indicating server-authenticated transport layer security and the presence of an X.509 server certificate to be used for hostname-verification

- *MUTUAL_TLS* – A normative *POLICY* document indicating mutually-authenticated transport layer security.
- *MUTUAL_TLS_CERT_PROVIDED* – A normative *POLICY* document indicating mutually-authenticated transport layer security and the presence of an X.509 server certificate to be used for hostname-verification
- *USERNAME_TOKEN* – A normative *POLICY* document indicating that a Username/Token credential should be supplied in the message security header.
- *PASSWORD_DIGEST* – A normative *POLICY* document indicating that a Username/Token credential utilizing a password digest (a hash of a password, timestamp, and nonce) should be supplied in the message security header.
- *MUTUAL_X509* – A normative *POLICY* document indicating a requirement for secure communication in which both parties have X.509v3 certificates (and corresponding private keys).

3.3 Conformance Scope

The scope of the Profile delineates the technologies that it addresses; in other words, the Profile only attempts to improve interoperability within its own scope. Generally, the Profile’s scope is bounded by the specifications referenced by it (Section 7).

Referenced specifications often provide extension mechanisms and unspecified or open-ended configuration parameters. The Profile defines such extensibility points within referenced specifications, possibly refining them in the process. The extensibility points exposed by the Profile are enumerated in Appendix A. These extensibility points (e.g., mechanisms or parameters) are outside the scope of the Profile, and their use or non-use is not relevant to conformance.

3.4 Claiming Conformance

Claims of conformance to the Profile are the same as normatively described in *WS-I Basic Profile 1.1* [WS-I BP]. The conformance claim URI for this Profile is “http://www.ogf.org/ogsa/2007/05/sp-secure-communication”

4 WS-SECURITYPOLICY EXTENSIONS

This section of the Profile incorporates by reference the *WS-SecurityPolicy 1.2* specification. The Profile defines the following extensibility points from WS-SecurityPolicy:

- E0500 – WS-SecurityPolicy Token Assertion Extensibility – WS-SecurityPolicy allows the extensibility of *TOKEN_ASSERTIONs*.

The E0500 extensibility point is used to supplement the WS-SecurityPolicy specification with the ability to directly embed security tokens within security policy documents.

4.1 Binding Tokens to Token Assertions

WS-SecurityPolicy *TOKEN_ASSERTIONs* specify the types of tokens required during communication. Unfortunately, the WS-SecurityPolicy specification does not provide a way to embed an actual token within a *TOKEN_ASSERTION*. There exist use-cases for which this capability is desirable, for example:

- Key distribution. Consider a WS-Addressing EPR that contains a security policy indicating a resource's requirement for message-level encryption. In this case, it is convenient to furnish the actual recipient's token (e.g., an X.509 certificate containing the necessary public key) within the EPR's security policy.
- Authentication of the resource to the *INITIATOR*. Tokens conveyed in security policy documents can be checked against tokens supplied during transport-level handshakes or signatures supplied within response SOAP messages for extra authentication assurance.

WS-SecurityPolicy specifies that token assertions can carry optional `<sp:Issuer>` or `<sp:IssuerName>` elements to indicate a location from which to obtain the required token. This Profile alternatively allows the inclusion of a `<wsse:SecurityTokenReference>` element within *TOKEN_ASSERTIONs* to indicate that the required token should be obtained locally from the *RESOURCE_SECURITY_POLICY* document.

- R0500 – A WS-SecurityPolicy *TOKEN_ASSERTION* carrying an optional `<wsse:SecurityTokenReference>` MUST NOT additionally specify an `<sp:Issuer>` or an `<sp:IssuerName>` element.
- C0500 – Such a `<wsse:SecurityTokenReference>` within a *TOKEN_ASSERTION* SHOULD be an embedded or direct reference.

5 PROFILE REQUIREMENTS AND RECOMMENDATIONS

This section of the document suggests recommendations for Profile-compliant *SENDER*s and *RECIPIENTs*, and defines the requirements necessary for claiming Profile-compliance.

5.1 Authentication Recommendations

Authentication is a crucial component of secure communication because it exposes imposters and facilitates authorization and auditing.

The types of specific authentication "facts" that the *INITIATOR* must supply to the *RECIPIENT* are specified via policy assertions (such as those defined within this profile). The resource's policy assertions also specify how it will authenticate itself to the *INITIATOR.* Authentication of the *RECIPIENT* to the *SENDER* for one-way communication generally requires encryption.

- C0501 – Transport-level authentication may not be appropriate or sufficient for all use-cases. Message-level authentication SHOULD be used to accommodate:
    - Authentication schemes based upon diverse types of authenticatable facts (e.g., attributes, capabilities, etc.); transport protocols are often restricted to authentication using X.509 identities.
    - Service *INSTANCE*s that expose multiple *RESOURCE*s. For example, a common Web-services container may expose many job activity resources using a single transport-level endpoint. Authentication at the transport-level does not provide sufficient granularity to authenticate the individual activity resource to the *INITIATOR*.

5.2 Integrity Recommendations

In order to provide data integrity during communication, this Profile recommends signed communication. The Profile defines the following integrity recommendations:

- C0502 – In the presence of *MESSAGE_PASSING_INTERMEDIARIES*, the *SENDER* SHOULD perform *CRITICAL_SIGNING* of SOAP messages.

5.3 Confidentiality Recommendations

In order to provide confidentiality during communication, this Profile recommends encrypted communication. The Profile defines the following confidentiality recommendations:

- C0503 – In the presence of *MESSAGE_PASSING_INTERMEDIARIES*, the *SENDER* SHOULD perform *CRITICAL_ENCRYPTION*.

5.4 Policy Requirements

RESOURCE_SECURITY_POLICIES specify the security requirements (and ancillary tokens) for the *RESOURCEs.*

- R0501 – A *RESOURCE_SECURITY_POLICY* MUST reference at least one well-known *PROFILED_MECHANISM* as profiled within this Profile (or within a derivative of this Profile).

Tables 2 and 3 below respectively enumerate the transport-level and message-level *PROFILED_MECHANISM*s defined within this profile.

Table 2 Secure Transport Mechanisms

Mechanism Name	Conformance Target	Policy Reference URI
Server-Authenticated TLS	*SERVER_TLS*	`http://www.ggf.org/ogsa/2007/05/sp-secure-transport#ServerTLS`
Server-Authenticated TLS with Server Certificate Provided	*SERVER_TLS_CERT_PROVIDED*	`http://www.ggf.org/ogsa/2007/05/sp-secure-transport#ServerTLSCertProvided`
Mutually-Authenticated TLS	*MUTUAL_TLS*	`http://www.ggf.org/ogsa/2007/05/sp-secure-transport#MutualTLS`
Mutually-Authenticated TLS with Server Certificate Provided	*MUTUAL_TLS_CERT_PROVIDED*	`http://www.ggf.org/ogsa/2007/05/sp-secure-transport#MutualTLSCertProvided`

Table 3 Secure Message Mechanisms

Mechanism Name	Conformance Target	Policy Reference URI
Username Token	*USERNAME_TOKEN*	`http://www.ogf.org/ogsa/2007/05/sp-secure-soap#UsernameToken`
Password Digest Username Token	*PASSWORD_DIGEST*	`http://www.ogf.org/ogsa/2007/05/sp-secure-soap#PasswordDigest`
Asymmetric X.509 Mutual Authentication	*MUTUAL_X509*	`http://www.ogf.org/ogsa/2007/05/sp-secure-soap#MutualX509`

6 TRANSPORT-LEVEL MECHANISM POLICIES

This section defines several *PROFILED_MECHANISM*s that identify commonly-used transport-level security mechanisms. The transport-level security mechanisms implied by these policies are defined and profiled externally and incorporated by reference.

6.1 References and Extensibility Points

This profile incorporates by reference Section 3, “Transport Layer Mechanisms” of the *WS-I Basic Security Profile Version 1.0* [WS-I BSP] profile and referenced specifications. (Other sections of the WS-I BSP pertain to pertain to SOAP message-level security mechanisms, the requirements of which are considered out of scope of this section.)

The Profile inherits and refines the following extensibility points from the WS-I BSP:

- E0009 – TLS Ciphersuites – TLS allows for the use of arbitrary encryption algorithms. This Profile restricts the set of allowable ciphersuites to those listed in the *WS-SecurityPolicy 1.2* Section 6.1. (As per the WS-I BSP, only TLS Protocol Version 1.0 is incorporated into this profile.)
- E0010 – TLS Extensions – TLS allows for extensions during the handshake phase.
- E0011 – SSL Ciphersuites – SSL allows for the use of arbitrary encryption algorithms. This Profile restricts the set of allowable ciphersuites to those listed in the *WS-SecurityPolicy 1.2* Section 6.1. (As per the WS-I BSP, only SSL Protocol Version 3.0 is incorporated into this profile. SSL 2.0 MUST NOT be used.)
- E0012 – Certificate Authority – The choice of the Certificate Authority is a private agreement between parties.
- E0013 – Certificate Extensions – X.509 allows for arbitrary certificate extensions.

This Profile defines the following extensibility points:

- E0501 – Additional transport-level *PROFILED_MECHANISMs* may be profiled in accordance to the requirements in Section 5.

6.2 Mapping of Algorithm Suites

The TLS and SSL protocols are different versions of the same general transport-layer protocol. The table below illustrates the correspondence between the colloquial protocol name and the negotiated protocol version:

Table 4 Mapping between negotiated TLS versions and their colloquial names

Major Version	Minor Version	Colloquial Name
3	0	SSL 3.0
3	1	TLS 1.0
3	2	TLS 1.1
3	3	TLS 1.2

For convenience, we provide the following mapping between WS-SecurityPolicy algorithm suites and TLS/SSL ciphersuite designations:

Table 5 Mapping between WS-SecurityPolicy algorithm suites and TLS/SSL

WS-SecurityPolicy Algorithm Suite	TLS 1.0/1.1	SSL 3.0
Basic256	TLS_RSA_WITH_AES_256_CBC_SHA	SSL_RSA_WITH_AES_256_CBC_SHA
Basic128	TLS_RSA_WITH_AES_128_CBC_SHA	SSL_RSA_WITH_AES_128_CBC_SHA
TripleDes	TLS_RSA_WITH_3DES_EDE_CBC_SHA	SSL_RSA_WITH_3DES_EDE_CBC_SHA

- R0502 – FIPS-compliant implementations MUST support the FIPS-equivalent versions of the above ciphersuites. (E.g., SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA).

6.3 Server-Authenticated TLS (*SERVER_TLS*) Policy

The *SERVER_TLS* policy is a referenceable transport-level *PROFILED_MECHANISM* that indicates a requirement for server-authenticated transport layer security using SSL/TLS as profiled by the WSI-BSP. It is intended to be referenced by name within a *RESOURCE_SECURITY_POLICY* using a `<wsp:PolicyReference>` element. The normative policy document for the *SERVER_TLS* policy is defined in Appendix B.

- R0503 – The actions upon *RESOURCE*s for which the *SERVER_TLS* policy is advertised MUST support the following:
  - SOAP over HTTPS
  - An SSL or TLS handshake with server authentication
- R0504 –TLS/SSL ClientHello messages MUST indicate maximal supported protocol version no lower than 3.0 (SSL v3.0). This Profile RECOMMENDS that ClientHello messages indicate version 3.2 (TLS v1.1).
- R0505 –TLS/SSL ClientHello messages MUST indicate either TLS_RSA_WITH_AES_256_CBC_SHA or SSL_RSA_WITH_AES_256_CBC_SHA within the list of supported ciphersuites. This Profile RECOMMENDS that ClientHello message also indicate TLS_RSA_WITH_AES_128_CBC_SHA or SSL_RSA_WITH_AES_128_CBC_SHA ciphersuites to allow the *RECIPIENT* the option for more efficient communication.
- R0506 – The *SERVER_TLS* policy MUST be referenced with the policy reference URI `"ServerTLS"`

6.4 Server-Authenticated TLS with Server Certificate Provided (*SERVER_TLS_CERT_PROVIDED*) Policy

The *SERVER_TLS_CERT_PROVIDED* policy is a referenceable transport-level *PROFILED_MECHANISM* that indicates a requirement for server-authenticated transport layer security using SSL/TLS as profiled by the WSI-BSP. It is intended to be referenced by name within a *RESOURCE_SECURITY_POLICY* using a `<wsp:PolicyReference>` element. Such a *RESOURCE_SECURITY_POLICY* must include an X.509 certificate to be used for server hostname-verification. The normative policy document for the *SERVER_TLS_CERT_PROVIDED* policy is defined in Appendix B.

- R0507 – The actions upon a *RESOURCE*s for which the *SERVER_TLS_CERT_PROVIDED* policy is advertised MUST support the following:
  - SOAP over HTTPS
  - An SSL or TLS handshake with server authentication

- R0508 –TLS/SSL ClientHello messages MUST indicate maximal supported protocol version no lower than 3.0 (SSL v3.0). This Profile RECOMMENDS that ClientHello messages indicate version 3.2 (TLS v1.1).
- R0509 –TLS/SSL ClientHello messages MUST indicate either TLS_RSA_WITH_AES_256_CBC_SHA or SSL_RSA_WITH_AES_256_CBC_SHA within the list of supported ciphersuites. This Profile RECOMMENDS that ClientHello message also indicate TLS_RSA_WITH_AES_128_CBC_SHA or SSL_RSA_WITH_AES_128_CBC_SHA ciphersuites to allow the *RECIPIENT* the option for more efficient communication.
- R0510 – A *RESOURCE_SECURITY_POLICY* that references the *SERVER_TLS_CERT_PROVIDED* policy MUST provide a *RECIPIENT_TRANSPORT_IDENTITY* corresponding to the resource's TLS/SSL server certificate.
- R0511 – The *SERVER_TLS_CERT_PROVIDED* policy MUST be referenced with the policy reference URI `"ServerTLSCertProvided"`

Note that in many cases the *RESOURCE_SECURITY_POLICY* itself may be provided from an untrusted source or over an insecure communication channel. Using the *RECIPIENT_TRANSPORT_IDENTITY* for additional hostname verification provides no protection against attacks where *RESOURCE_SECURITY_POLICY* can be compromised.

6.5 Mutually-Authenticated TLS (*MUTUAL_TLS*) Policy

The *MUTUAL_TLS* policy is a referenceable transport-level *PROFILED_MECHANISM* that indicates a requirement for mutually-authenticated transport layer security using SSL/TLS as profiled by the WSI-BSP. It is intended to be referenced by name within a *RESOURCE_SECURITY_POLICY* using a `<wsp:PolicyReference>` element. The normative policy document for the MUTUAL*_TLS* policy is defined in Appendix B.

- R0512 – The actions upon a *RESOURCE*s for which the *MUTUAL_TLS* policy is advertised MUST support the following:
  - SOAP over HTTPS
  - An SSL or TLS handshake with both client and server authentication
- R0513 –TLS/SSL ClientHello messages MUST indicate maximal supported protocol version no lower than 3.0 (SSL v3.0). This Profile RECOMMENDS that ClientHello messages indicate version 3.2 (TLS v1.1).
- R0514 –TLS/SSL ClientHello messages MUST indicate either TLS_RSA_WITH_AES_256_CBC_SHA or SSL_RSA_WITH_AES_256_CBC_SHA within the list of supported ciphersuites. This Profile RECOMMENDS that ClientHello message also indicate TLS_RSA_WITH_AES_128_CBC_SHA or SSL_RSA_WITH_AES_128_CBC_SHA ciphersuites to allow the *RECIPIENT* the option for more efficient communication.
- R0515 – The *MUTUAL_TLS* policy MUST be referenced with the policy reference URI `"MutualTLS"`

6.6 Mutually-Authenticated TLS with Server Certificate Provided (*MUTUAL_TLS_CERT_PROVIDED*) Policy

The *MUTUAL_TLS_CERT_PROVIDED* policy is a referenceable transport-level *PROFILED_MECHANISM* that indicates a requirement for mutually-authenticated transport layer security using SSL/TLS as profiled by the WSI-BSP. It is intended to be referenced by name within a *RESOURCE_SECURITY_POLICY* using a `<wsp:PolicyReference>` element. Such a *RESOURCE_SECURITY_POLICY* must include an X.509 certificate to be used for server hostname-verification. The normative policy document for the *MUTUAL_TLS_CERT_PROVIDED* policy is defined in Appendix B.

- R0516 – The actions upon a *RESOURCE*s for which the *MUTUAL_TLS_CERT_PROVIDED* policy is advertised MUST support the following:
  - SOAP over HTTPS
  - An SSL or TLS handshake with both client and server authentication
- R0517 –TLS/SSL ClientHello messages MUST indicate maximal supported protocol version no lower than 3.0 (SSL v3.0). This Profile RECOMMENDS that ClientHello messages indicate version 3.2 (TLS v1.1).
- R0518 –TLS/SSL ClientHello messages MUST indicate either TLS_RSA_WITH_AES_256_CBC_SHA or SSL_RSA_WITH_AES_256_CBC_SHA within the list of supported ciphersuites. This Profile RECOMMENDS that ClientHello message also indicate TLS_RSA_WITH_AES_128_CBC_SHA or SSL_RSA_WITH_AES_128_CBC_SHA ciphersuites to allow the *RECIPIENT* the option for more efficient communication.
- R0519 – A *RESOURCE_SECURITY_POLICY* that references the *MUTUAL_TLS_CERT_PROVIDED* policy MUST provide a *RECIPIENT_TRANSPORT_IDENTITY* corresponding to the resource's TLS/SSL server certificate.
- R0520 – The *MUTUAL_TLS_CERT_PROVIDED* policy MUST be referenced with the policy reference URI `"MutualTLSCertProvided"`

Note that in many cases the *RESOURCE_SECURITY_POLICY* itself may be provided from an untrusted source or over an insecure communication channel. Using the *RECIPIENT_TRANSPORT_IDENTITY* for additional hostname verification provides no protection against attacks where *RESOURCE_SECURITY_POLICY* can be compromised.

7 MESSAGE-LEVEL MECHANISM POLICIES

This section defines several *PROFILED_MECHANISMS* that identify commonly-used security mechanisms. The message-level security mechanisms implied by these policies are defined and profiled externally and incorporated by reference.

7.1 References and Extensibility Points

This profile incorporates by reference the following sections of *WS-I Basic Security Profile Version 1.0* [WS-I BSP] and referenced specifications:

- Section 4, "SOAP Nodes and Messages"
- Section 5, "Security Headers"
- Section 6, "Timestamps"
- Section 7, "Security Token References"
- Section 8, "XML Signature"
- Section 9, "XML Encryption"
- Section 10, "Binary Security Tokens"
- Section 11, "Username Token"
- Section 12, "X.509 Certificate Token

Other sections of the WS-I BSP are considered out of scope of this section because they either (a) pertain to security token profiles not identified by policies profiled within this document or (b) pertain to transport-level security mechanisms. The Profile inherits and refines the following extensibility points from these sections of the WS-I BSP:

- E0002 – Security Tokens – Security tokens may be specified in additional security token profiles.

This Profile defines the following extensibility points:

- E0502 – Additional message-level *PROFILED_MECHANISMs* may be profiled in accordance to the requirements in Section 5.

7.2 Username-Token (*USERNAME_TOKEN*) Policy

The *USERNAME_TOKEN* policy is a referenceable message-level *PROFILED_MECHANISM* indicating that a Username/Token credential should be supplied in the message security header in accordance with the Section 11 of the *WS-I Basic Security Profile Version 1.0* (WS-I BSP). The normative policy document for the *USERNAME_TOKEN* policy is defined in Appendix B.

- R0521 – The actions upon a *RESOURCE* for which the *USERNAME_TOKEN* policy is advertised MUST support a Username/Token credential in accordance with the WS-I BSP as per the semantics of the *USERNAME_TOKEN* policy document defined in Appendix B.
- R0522 – The *USERNAME_TOKEN* policy MUST be referenced with the policy reference URI `"UsernameToken"`
- R0522 – Any policy document referencing the *USERNAME_TOKEN* policy MUST additionally reference at least one of the following companion *PROFILED_MECHANISM* policies:
  - o A transport-level *PROFILED_MECHANISM* that provides encrypted communication (e.g., *SERVER_TLS*, *MUTUAL_TLS*, etc.)

  - A message-level *PROFILED_MECHANISM* that provides encryption of all `<wsse:UsernameToken>` elements.

7.3 Password Digest Username-Token (*PASSWORD_DIGEST*) Policy

The PASSWORD_DIGEST policy is a referenceable message-level *PROFILED_MECHANISM* indicating that a Username/Token credential utilizing a password digest (a hash of a password, timestamp, and nonce) should be supplied in the message security header in accordance with the Section 11 of the *WS-I Basic Security Profile Version 1.0* (WS-I BSP). The normative policy document for the *PASSWORD_DIGEST* policy is defined in Appendix B.

- R0523 – The actions upon a *RESOURCE* for which the *PASSWORD_DIGEST* policy is advertised MUST support a password-digest Username/Token credential in accordance with the WS-I BSP as per the semantics of the *USERNAME_TOKEN* policy document defined in Appendix B.
- R0524 – The *PASSWORD_DIGEST* policy MUST be referenced with the policy reference URI `”PasswordDigest”`

7.4 Asymmetric X.509 Mutual Authentication (*MUTUAL_X509*) Policy

The *MUTUAL_X509* policy is a referenceable message-level *PROFILED_MECHANISM* indicating a requirement for secure communication in which both parties have X.509v3 certificates (and corresponding private keys).

At a minimum, this policy requires a signature over a *request* message covering the `<soapenv:Body>` message body as well as any *message-addressing headers*. If a *response* message is sent, this policy requires a signature over the *response* message that covers the `<soapenv:Body>` message body.

This profile defines a *message-addressing header* as a child element of the `<soapenv:Header>` that is either declared under the `wsa`: namespace or has a ‘`wsa:IsReferenceParameter=true`’ attribute.

The normative policy document for the *MUTUAL_X509* policy is defined in Appendix B.

- R0525 – The actions upon a *RESOURCE* for which the *MUTUAL_X509* policy is advertised MUST support an X.509-based mutually-authenticated message exchange in accordance with the WS-I BSP as per the semantics of the *MUTUAL_X509* policy document defined in Appendix B.
- R0526 – The enclosing *RESOURCE_SECURITY_POLICY* MUST provide a *RECIPIENT_MESSAGE_IDENTITY* for which the `<sp:X509Token>` element within the *MUTUAL_X509* policy document’s `<sp:RecipientToken>` refers to.
- R0527 – The *MUTUAL_X509* policy MUST be referenced with the policy reference URI `”MutualX509”`
- C0504 – *RESOURCE_SECURITY_POLICIES* that incorporate the *MUTUAL_X509* policy MAY specify additional portions of the message documents to be signed and/or encrypted. These additional requirements are (optionally) specified as XML sibling elements of the `<wsp:PolicyReference>http://www.ggf.org/ogsa/2007/05/sp-secure-soap#MutualX509</wsp:PolicyReference>` element used to reference the *MUTUAL_X509* policy.

The following is an example of a *RESOURCE_SECURITY_POLICY* that specifies additional input and output message protection requirements for encryption:

```
<wsp:Policy>
```

```

    <wsp:PolicyReference>
      http://www.ggf.org/ogsa/2007/05/sp-secure-communication#MutualX509
    </wsp:PolicyReference>

    <wsp:Policy wsu:Id=”SupplementalInputPolicy”>
      <sp:EncryptedParts>
        <sp:Body/>
      </sp:EncryptedParts>
    </wsp:Policy>

    <wsp:Policy wsu:Id=”SupplementalOutputPolicy”>
      <sp:EncryptedParts>
        <sp:Body/>
      </sp:EncryptedParts>
    </wsp:Policy>

  </wsp:Policy>
```

8 EXAMPLE SOAP REQUEST MESSAGE

The following shows an example of an input message to an RNS *RESOURCE* performing a *list* operation that conforms to *MUTUAL_X509* policy. (The Remote Naming Service specification defines a directory/namespace service.)

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope
    xmlns:soapenv=".../envelope/"
    xmlns:xsd=".../XMLSchema"
    xmlns:xsi=".../XMLSchema-instance"
    xmlns:wsu="...-wss-wssecurity-utility-1.0.xsd"
    xmlns:wsse="...-wss-wssecurity-secext-1.0.xsd"
    xmlns:wsa=".../addressing"
    xmlns:ds=".../xmldsig#">
  <soapenv:Header>
    <wsse:Security soapenv:mustUnderstand="1">
      <wsse:BinarySecurityToken
          EncodingType="...-wss-soap-message-security-1.0#Base64Binary"
          ValueType="...-wss-x509-token-profile-1.0#X509v3"
          wsu:Id="CertId-2891833">MIIDqjCCAp...</wsse:BinarySecurityToken>
      <ds:Signature Id="Signature-10923886">
        <ds:SignedInfo>
          <ds:CanonicalizationMethod Algorithm=".../xml-exc-c14n#">
          </ds:CanonicalizationMethod>
          <ds:SignatureMethod Algorithm=".../xmldsig#rsa-sha1">
          </ds:SignatureMethod>
          <ds:Reference URI="#id-28713819">
            <ds:Transforms>
              <ds:Transform Algorithm=".../xml-exc-c14n#">
              </ds:Transform>
            </ds:Transforms>
            <ds:DigestMethod Algorithm=".../xmldsig#sha1">
            </ds:DigestMethod>
            <ds:DigestValue>u+KE5lscRkzx2dTFim8S5Bpn9i4=</ds:DigestValue>
          </ds:Reference>
          <ds:Reference URI="#id-08675309">
            <ds:Transforms>
              <ds:Transform Algorithm=".../xml-exc-c14n#">
              </ds:Transform>
            </ds:Transforms>
            <ds:DigestMethod Algorithm=".../xmldsig#sha1">
            </ds:DigestMethod>
            <ds:DigestValue>sZHtJewew040zT9K76NJ5hKNAoc=</ds:DigestValue>
          </ds:Reference>
          <ds:Reference URI="#id-13320911">
            <ds:Transforms>
              <ds:Transform Algorithm=".../xml-exc-c14n#">
              </ds:Transform>
            </ds:Transforms>
            <ds:DigestMethod Algorithm=".../xmldsig#sha1">
            </ds:DigestMethod>
            <ds:DigestValue>5oHvfCRfo89/PDJ72u97uQa8ds0=</ds:DigestValue>
          </ds:Reference>
        </ds:SignedInfo>
        <ds:SignatureValue>fQ6bwvRjQ8...</ds:SignatureValue>
        <ds:KeyInfo Id="KeyId-29398564">
          <wsse:SecurityTokenReference wsu:Id="STRId-19608393">
            <wsse:Reference URI="#CertId-2891833"
                ValueType="...-wss-x509-token-profile-1.0#X509v3"/>
          </wsse:SecurityTokenReference>
        </ds:KeyInfo>
      </ds:Signature>
    </wsse:Security>
    <wsa:To wsu:Id="id-28713819”>
        https://vcgr.cs.virginia.edu:18080/axis/services/RNSPortType</wsa:To>
    <wsa:Action wsu:Id="id-08675309”>list</wsa:Action>
```

Hiro Kishimoto 1/9/08 6:58 AM
Formatted: Norwegian Bokmal

Hiro Kishimoto 1/9/08 6:58 AM
Formatted: Norwegian Bokmal

Hiro Kishimoto 1/9/08 6:58 AM
Formatted: Polish

Unknown
Field Code Changed

Hiro Kishimoto 1/9/08 6:58 AM
Formatted: Polish

Hiro Kishimoto 1/9/08 6:58 AM
Formatted: Polish

```
(62)     </soapenv:Header>
(63)     <soapenv:Body wsu:Id="id-13320911">
(64)       <list xmlns=".../rns">
(65)         <entry_name_regexp>.*</entry_name_regexp>
(66)       </list>
(67)     </soapenv:Body>
(68)   </soapenv:Envelope>
```

- Lines 01-68: An example input message to an RNS *RESOURCE*.
- Lines 11-58: The WS-S SOAP message security header
- Lines 12-15: The *SENDER*'s X.509 v.3 certificate used to sign the message.
- Lines 17-49: `SignedInfo` description of the signature and canonicalization algorithms used, as well as references to the portions of the SOAP message that are signed. In this case, signing is done in accordance with the SHA1/RSA signature/digest algorithms in accordance with the WSI-BSP. Lines 22-30 indicate the digest used for the signing of the WS-Addressing `To` header. Lines 31-39 indicate the digest used for the signing of the WS-Addressing `Action` header. Lines 40-48 indicate the digest used for the signing of the message body.
- Line 50: The signature of the digests contained within the `SignedInfo` element.
- Lines 42-47: Binding of the X.509 v.3 certificate in Lines 12-15 to the signature.
- Lines 59-60: WS-Addressing `To` header
- Line 61: WS-Addressing Action header.
- Lines 63-67: SOAP message body indicating a wildcard listing of the RNS *RESOURCE*'s entries.

9 CONTRIBUTORS

9.1 Author Information

Duane Merrill
Computer Science Department
University of Virginia
Charlottesville, VA 22903
Email: dgm4d@cs.virginia.edu

9.2 Acknowledgements

We are grateful to colleagues for discussions on the topics covered in this document, in particular (in alphabetical order, with apologies to anybody we've missed) Blair Dillaway, Andrew Grimshaw, Hiro Kishimoto, Mark Morgan, Andreas Savva, and David Snelling.

10 INTELLECTUAL PROPERTY STATEMENT

The OGF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the OGF Secretariat.

The OGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this recommendation. Please address the information to the OGF Executive Director.

11 DISCLAIMER

This document and the information contained herein is provided on an “As Is” basis and the OGF disclaims all warranties, express or implied, including but not limited to any warranty that the use of the information herein will not infringe any rights or any implied warranties of merchantability or fitness for a particular purpose.

12 FULL COPYRIGHT NOTICE

13 REFERENCES

13.1 Normative References

- [RFC2119] S. Bradner (ed.): Key words for use in RFCs to Indicate Requirement Levels, The Internet Engineering Task Force Best Current Practice, March 1997. http://www.ietf.org/rfc/rfc2119
- [HTTP-TLS] E. Rescorla (ed.): HTTP Over TLS, Internet Engineering Task Force, May 2000. http://www.ietf.org/rfc/rfc2818
- [TLS 1.0] T. Dierks, C. Allen (ed.): The TLS Protocol Version 1.0, Internet Engineering Task Force, January 1999. http://www.ietf.org/rfc/rfc2246
- [WS-A Core] M. Gudgin and Marc Hadley (ed.), Web Services Addressing 1.0 - Core, W3C Candidate Recommendation 17 August 2005, http://www.w3.org/TR/2005/CR-ws-addr-core-20050817/
- [WS-I BP 1.1] K. Ballinger, D. Ehnebuske, C. Ferris, M. Gudgin, C.K. Liu, M. Nottingham, and P. Yendluri (ed.): Basic Profile Version 1.1, Web Services Interoperability Organization Final Material, 24 August 2004. http://www.ws-i.org/Profiles/BasicProfile-1.1.html
- [WS-I BSP 1.0] A. Barbir, M. Gudgin, M. McIntosh, and K.S. Morrison (ed.): Basic Security Profile Version 1.0, Web Services Interoperability Organization, Working Group Draft, 17 August 2006. http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0-2006-08-17.html
- [X.509] Information Technology - Open Systems Interconnection - The Directory: Authentication Framework, 08/05. http://www.itu.int/rec/T-REC-X.509-200508-I
- [WS-Policy] A. Vedamuthu, D. Orchard, F. Hirsch, M. Hondo, P. Yendluri, T. Boubez, Ü. Yalçinalp (ed.): Web Services Policy 1.5 – Framework. W3C Candidate Recommendation, 05 June 2007. http://www.w3.org/TR/2007/CR-ws-policy-20070605
- [WS-SecurityPolicy] A. Nadalin, M. Goodner, A. Barbir, H. Granqvist (ed.): WS-SecurityPolicy 1.2. Committee Specification, 30 April 2007. http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702/ws-securitypolicy-1.2-spec-cs.pdf
- [WS-PolicyAttachment] A. Vedamuthu, D. Orchard, F. Hirsch, M. Hondo, P. Yendluri, T. Boubez, Ü. Yalçinalp (ed.): Web Services Policy 1.5 – Attachment. W3C Candidate Recommendation 05 June 2007. http://www.w3.org/TR/2007/CR-ws-policy-attach-20070605

13.2 Non-Normative References

- [WS-S] A. Nadalin, C. Kaler, P. Hallam-Baker and R. Monzillo (ed.): Web Services Security: SOAP Message Security 1.0 (WS-Security 2004), OASIS Standard, 200401, March 2004. http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf
- [XML-DigSig] D. Eastlake, J. Reagle, D. Solo (ed.): XML-Signature Syntax and Processing, W3C Recommendation, Feb 12, 2002. http://www.w3.org/TR/xmldsig-core/
- [XML-Enc] D. Eastlake, J. Reagle (ed.): XML Encryption Syntax and Processing, W3C Recommendation, Dec 10, 2002. http://www.w3.org/TR/xmlenc-core/
- [OGSA Profile Definition] T. Maguire. and D. Snelling: OGSA Profile Definition Version 1.0. Open Grid Forum, Lemont, Illinois, U.S.A., GFD-I.059, January 2006. http://www.ogf.org/gf/docs/?final

APPENDIX A. EXTENSIBILITY POINTS

This section identifies extensibility points for the Profile's component specifications. Except for the use of E0009, E0011, and E0500 as profiled in this document, these mechanisms are out of the scope of the Profile; their use may affect interoperability, and may require private agreement between the parties to a Web service.

In *WS-I Basic Security Profile 1.0* [WS-I BSP]:

- E0002 – Security Tokens – Security tokens may be specified in additional security token profiles.
- E0009 – TLS Ciphersuites – TLS allows for the use of arbitrary encryption algorithms. This Profile restricts the set of allowable ciphersuites to those listed in the *WS-SecurityPolicy 1.2* Section 6.1. (As per the WS-I BSP, only TLS Protocol Version 1.0 is incorporated into this profile.)
- E0010 – TLS Extensions – TLS allows for extensions during the handshake phase.
- E0011 – SSL Ciphersuites – SSL allows for the use of arbitrary encryption algorithms. This Profile restricts the set of allowable ciphersuites to those listed in the *WS-SecurityPolicy 1.2* Section 6.1. (As per the WS-I BSP, only SSL Protocol Version 3.0 is incorporated into this profile. SSL 2.0 MUST NOT be used.)
- E0012 – Certificate Authority – The choice of the Certificate Authority is a private agreement between parties.
- E0013 – Certificate Extensions – X.509 allows for arbitrary certificate extensions.

In *WS-SecurityPolicy 1.2* [WS-SecurityPolicy]:

- E0500 – WS-SecurityPolicy Token Assertion Extensibility – WS-SecurityPolicy allows the extensibility of *TOKEN_ASSERTIONs*.

In *Secure Communication Profile 1.0* (this document):

- E0501 – Additional transport-level binding assertions may be profiled in accordance to the requirements in Section 5.1: Security Mechanism Specifics.
- E0502 – Additional message-level *PROFILED_MECHANISMs* may be profiled in accordance to the requirements in Section 5.

APPENDIX B. NORMATIVE POLICY DOCUMENTS

This appendix defines the normative policy documents introduced by the Profile along with non-normative descriptions.

B.1. *SERVER_TLS* Policy Document

The normative policy document for the *SERVER_TLS* policy is as follows:

```
(01)   <?xml version="1.0" encoding="UTF-8"?>
(02)   <wsp:Policy wsu:Id=”ServerTLS”
(03)       xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
(04)       xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-
    securitypolicy/200702">
(05)     <sp:TransportBinding>
(06)       <wsp:Policy>
(07)         <sp:TransportToken>
(08)           <wsp:Policy>
(09)             <sp:HttpsToken />
(10)           </wsp:Policy>
(11)         </sp:TransportToken>
(12)         <sp:AlgorithmSuite>
(13)           <wsp:Policy>
(14)             <wsp:ExactlyOne>
(15)               <wsp:All>
(16)                 <sp:Basic256 />
(17)               </wsp:All>
(18)               <wsp:All>
(19)                 <sp:Basic128 />
(20)               </wsp:All>
(21)             </wsp:ExactlyOne>
(22)           </wsp:Policy>
(23)         </sp:AlgorithmSuite>
(24)       </wsp:Policy>
(25)     </sp:TransportBinding>
(26)   </wsp:Policy>
```

Below is a detailed non-normative description for the *SERVER_TLS* policy document:

- Lines 02-26: *POLICY* for a `<sp:TransportBinding>` transport binding indicating server-authenticated transport layer security in accordance with this Profile.
- Lines 07-11: Transport token element indicating that the transport binding support the use of HTTPS
- Lines 12-23: Algorithm suite element indicating that either the Basic256 or the Basic128 algorithm suite (see WS-SecurityPolicy Section 6.1) may be used.

B.2. *SERVER_TLS_CERT_PROVIDED* Policy Document

The normative policy document for the *SERVER_TLS_CERT_PROVIDED* policy is as follows:

```
(01)   <?xml version="1.0" encoding="UTF-8"?>
(02)   <wsp:Policy wsu:Id=”ServerTLSCertProvided”
(03)       xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
(04)       xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-
    securitypolicy/200702">
(05)     <sp:TransportBinding>
(06)       <wsp:Policy>
(07)         <sp:TransportToken>
(08)           <wsp:Policy>
(09)             <sp:HttpsToken>
```

```
(10)              <wsse:SecurityTokenReference>
(11)                <wsse:Reference URI='#RecipientTransportIdentity'
(12)                   ValueType=" http://docs.oasis-
   open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3" />
(13)              </wsse:SecurityTokenReference>
(14)            </sp:HttpsToken>
(15)          </wsp:Policy>
(16)        </sp:TransportToken>
(17)        <sp:AlgorithmSuite>
(18)          <wsp:Policy>
(19)            <wsp:ExactlyOne>
(20)              <wsp:All>
(21)                <sp:Basic256 />
(22)              </wsp:All>
(23)              <wsp:All>
(24)                <sp:Basic128 />
(25)              </wsp:All>
(26)            </wsp:ExactlyOne>
(27)          </wsp:Policy>
(28)        </sp:AlgorithmSuite>
(29)      </wsp:Policy>
(30)    </sp:TransportBinding>
(31)  </wsp:Policy>
```

Below is a detailed non-normative description for the *SERVER_TLS_CERT_PROVIDED* policy document:

- Lines 02-31: *POLICY* for a `<sp:TransportBinding>` transport binding indicating server-authenticated transport layer security in accordance with this Profile with the additional inclusion of the *RECEIVER's* X.509 identity certificate.
- Lines 07-16: Transport token element indicating that the transport binding support the use of HTTPS.
- Lines 09-14: The `<sp:HttpsToken>` assertion indicates that the X.509 certificate for the *RECIPIENT* can be found within the enclosing *RESOURCE_SECURITY_POLICY's* `<wsa:Metadata>` element, and should be used for additional hostname verification processing.
- Lines 17-28: Algorithm suite element indicating that either the Basic256 or the Basic128 algorithm suite (see WS-SecurityPolicy Section 6.1) may be used.

B.3. *MUTUAL_TLS* Policy Document

The normative policy document for the *MUTUAL_TLS* policy is as follows:

```
(01)  <?xml version="1.0" encoding="UTF-8"?>
(02)  <wsp:Policy wsu:Id="MutualTLS"
(03)      xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
(04)      xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-
   securitypolicy/200702">
(05)    <sp:TransportBinding>
(06)      <wsp:Policy>
(07)        <sp:TransportToken>
(08)          <wsp:Policy>
(09)            <sp:HttpsToken>
(10)              <wsp:Policy>
(11)                <sp:RequireClientCertificate />
(12)              </wsp:Policy>
(13)            </sp:HttpsToken>
(14)          </wsp:Policy>
(15)        </sp:TransportToken>
(16)        <sp:AlgorithmSuite>
(17)          <wsp:Policy>
(18)            <wsp:ExactlyOne>
```

```
(19)                <wsp:All>
(20)                  <sp:Basic256 />
(21)                </wsp:All>
(22)                <wsp:All>
(23)                  <sp:Basic128 />
(24)                </wsp:All>
(25)              </wsp:ExactlyOne>
(26)            </wsp:Policy>
(27)          </sp:AlgorithmSuite>
(28)        </wsp:Policy>
(29)      </sp:TransportBinding>
(30)    </wsp:Policy>
```

Below is a detailed non-normative description for the *MUTUAL_TLS* policy document:

- Lines 02-30: *POLICY* for a `<sp:TransportBinding>` transport binding indicating server-authenticated transport layer security in accordance with this Profile.
- Lines 07-15: Transport token element indicating that the transport binding support the use of HTTPS
- Lines 09-13: Policy for the `<sp:HttpsToken>` element indicating that the client certificate is required for authentication.
- Lines 16-27: Algorithm suite element indicating that either the Basic256 or the Basic128 algorithm suite (see WS-SecurityPolicy Section 6.1) may be used.

B.4. *MUTUAL_TLS_CERT_PROVIDED* Policy Document

The normative policy document for the *MUTUAL_TLS_CERT_PROVIDED* policy is as follows:

```
(01)    <?xml version="1.0" encoding="UTF-8"?>
(02)    <wsp:Policy wsu:Id=”MutualTLSCertProvided”
(03)        xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
(04)        xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-
    securitypolicy/200702">
(05)      <sp:TransportBinding>
(06)        <wsp:Policy>
(07)          <sp:TransportToken>
(08)            <wsp:Policy>
(09)              <sp:HttpsToken>
(10)                <wsse:SecurityTokenReference>
(11)                  <wsse:Reference URI='#RecipientTransportIdentity'
(12)                     ValueType=" http://docs.oasis-
    open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3" />
(13)                </wsse:SecurityTokenReference>
(14)                <wsp:Policy>
(15)                  <sp:RequireClientCertificate />
(16)                </wsp:Policy>
(17)              </sp:HttpsToken>
(18)            </wsp:Policy>
(19)          </sp:TransportToken>
(20)          <sp:AlgorithmSuite>
(21)            <wsp:Policy>
(22)              <wsp:ExactlyOne>
(23)                <wsp:All>
(24)                  <sp:Basic256 />
(25)                </wsp:All>
(26)                <wsp:All>
(27)                  <sp:Basic128 />
(28)                </wsp:All>
(29)              </wsp:ExactlyOne>
(30)            </wsp:Policy>
(31)          </sp:AlgorithmSuite>
(32)        </wsp:Policy>
(33)      </sp:TransportBinding>
```

```
(34)    </wsp:Policy>
```

Below is a detailed non-normative description for the *MUTUAL_TLS_CERT_PROVIDED* policy document:

- Lines 02-34: *POLICY* for a `<sp:TransportBinding>` transport binding indicating mutually-authenticated transport layer security in accordance with this Profile with the additional inclusion of the *RECEIVER's* X.509 identity certificate.
- Lines 09-17: Policy for the `<sp:HttpsToken>` element indicating that the client certificate is required for authentication and that the X.509 certificate for the *RECIPIENT* can be found within the enclosing *RESOURCE_SECURITY_POLICY's* `<wsa:Metadata>` element, and should be used for additional hostname verification processing.
- Lines 20-31: Algorithm suite element indicating that either the Basic256 or the Basic128 algorithm suite (see WS-SecurityPolicy Section 6.1) may be used.

B.5. *USERNAME_TOKEN* Policy Document

The normative policy document for the *USERNAME_TOKEN* policy is as follows:

```
(01)    <?xml version="1.0" encoding="UTF-8"?>
(02)    <wsp:Policy wsu:Id=”UsernameToken”
(03)        xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
(04)        xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-
    securitypolicy/200702"><wsp:Policy wsu:Id=”UsernameToken”>
(05)      <sp:SupportingTokens>
(06)        <wsp:Policy>
(07)          <sp:UsernameToken/>
(08)        </wsp:Policy>
(09)      </sp:SupportingTokens>
(10)    </wsp:Policy>
```

Hiro Kishimoto 1/9/08 6:58 AM
Formatted: Norwegian Bokmal

Below is a detailed non-normative description for the *USERNAME_TOKEN* policy document:

- Lines 06–10 contain the `<sp:SupportingTokens>` assertion which includes a `<sp:UsernameToken>` indicating that a UsernameToken must be included in the security header.

B.6. *PASSWORD_DIGEST* Policy Document

The normative policy document for the *PASSWORD_DIGEST* policy is as follows:

```
(01)    <?xml version="1.0" encoding="UTF-8"?>
(02)    <wsp:Policy wsu:Id=”PasswordDigest”
(03)        xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
(04)        xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-
    securitypolicy/200702">
(05)      <sp:SupportingTokens>
(06)        <wsp:Policy>
(07)          <sp:UsernameToken>
(08)            <wsp:Policy>
(09)              <sp:HashPassword/>
(10)            </wsp:Policy>
(11)          </sp:UsernameToken>
(12)        </wsp:Policy>
(13)      </sp:SupportingTokens>
(14)    </wsp:Policy>
```

Hiro Kishimoto 1/9/08 6:58 AM
Formatted: Polish

Below is a detailed non-normative description for the *PASSWORD_DIGEST* policy document:

- Lines 05–13: contain the `<sp:SupportingTokens>` assertion which includes a `<sp:UsernameToken>` indicating that a UsernameToken must be included in the security header.
- Line 08 – 10: Sub-policy requiring that the password be protected by combining it with a nonce and timestamp, and then hashing the combination.

B.7. *MUTUAL_X509* Policy Document

The normative policy document for the *MUTUAL_X509* policy is as follows:

```
(01)   <?xml version="1.0" encoding="UTF-8"?>
(02)   <wsp:Policy wsu:Id=”MutualX509”
(03)       xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
(04)       xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-
    securitypolicy/200702">
(05)     <wsp:ExactlyOne>
(06)       <wsp:All>
(07)
(08)         <sp:AsymmetricBinding>
(09)           <wsp:Policy>
(10)             <sp:InitiatorToken>
(11)               <wsp:Policy>
(12)                 <sp:X509Token sp:IncludeToken="http://docs.oasis-
    open.org/ws-sx/ws-securitypolicy/200512/IncludeToken/AlwaysToRecipient">
(13)                   <wsp:Policy>
(14)                     <wsp:ExactlyOne>
(15)                       <wsp:All>
(16)                         <sp:WssX509V3Token11/>
(17)                       </wsp:All>
(18)                       <wsp:All>
(19)                         <sp:WssX509PkiPathV1Token11/>
(20)                       </wsp:All>
(21)                       <wsp:All>
(22)                         <sp:WssX509Pkcs7Token11/>
(23)                       </wsp:All>
(24)                     </wsp:ExactlyOne>
(25)                   </wsp:Policy>
(26)                 </sp:X509Token>
(27)               </wsp:Policy>
(28)             </sp:InitiatorToken>
(29)             <sp:RecipientToken>
(30)               <wsp:Policy>
(31)                 <wsp:ExactlyOne>
(32)                   <wsp:All>
(33)                     <sp:X509Token sp:IncludeToken="http://docs.oasis-
    open.org/ws-sx/ws-securitypolicy/200512/IncludeToken/Never>
(34)                       <wsse:SecurityTokenReference>
(35)                         <wsse:Reference URI='#RecipientMessageIdentity'
    ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-
    token-profile-1.0#X509v3"/>
(36)                       </wsse:SecurityTokenReference>
(37)                       <wsp:Policy>
(38)                         <sp:WssX509V3Token11/>
(39)                       </wsp:Policy>
(40)                     </sp:X509Token>
(41)                   </wsp:All>
(42)                   <wsp:All>
(43)                     <sp:X509Token sp:IncludeToken="http://docs.oasis-
    open.org/ws-sx/ws-securitypolicy/200512/IncludeToken/Never>
(44)                       <wsse:SecurityTokenReference>
(45)                         <wsse:Reference URI='#RecipientMessageIdentity'
    ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-
    token-profile-1.0#X509PKIPathv1"/>
(46)                       </wsse:SecurityTokenReference>
(47)                       <wsp:Policy>
```

Hiro Kishimoto 1/9/08 6:58 AM
Formatted: Polish

Hiro Kishimoto 1/9/08 6:58 AM
Formatted: Norwegian Bokmal

Hiro Kishimoto 1/9/08 6:58 AM
Formatted: Norwegian Bokmal

Hiro Kishimoto 1/9/08 6:58 AM
Formatted: Norwegian Bokmal

```
(48)                         <sp:WssX509PkiPathV1Token11/>
(49)                       </wsp:Policy>
(50)                     </sp:X509Token>
(51)                 </wsp:All>
(52)                   <wsp:All>
(53)                     <sp:X509Token sp:IncludeToken="http://docs.oasis-
    open.org/ws-sx/ws-securitypolicy/200512/IncludeToken/Never>
(54)                       <wsse:SecurityTokenReference>
(55)                         <wsse:Reference URI='#RecipientMessageIdentity'
    ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-
    token-profile-1.0#PKCS7"/>
(56)                       </wsse:SecurityTokenReference>
(57)                       <wsp:Policy>
(58)                         <sp:WssX509Pkcs7Token11/>
(59)                       </wsp:Policy>
(60)                     </sp:X509Token>
(61)                 </wsp:All>
(62)               </wsp:Policy>
(63)             </sp:RecipientToken>
(64)             <sp:AlgorithmSuite>
(65)               <wsp:Policy>
(66)                 <sp:Basic256/>
(67)               </wsp:Policy>
(68)             </sp:AlgorithmSuite>
(69)             <sp:OnlySignEntireHeadersAndBody/>
(70)             <sp:ProtectTokens>
(71)           </wsp:Policy>
(72)         </sp:AsymmetricBinding>
(73)
(74)         <sp:Wss10>
(75)           <wsp:Policy>
(76)             <sp:MustSupportRefKeyIdentifier/>
(77)           </wsp:Policy>
(78)         </sp:Wss10>
(79)
(80)         <wsp:Policy wsu:Id=”RequestPolicy”>
(81)           <sp:SignedParts>
(82)             <sp:Body/>
(83)             <Header namespace="http://www.w3.org/2005/08/addressing"/>
(84)           </sp:SignedParts>
(85)           <sp:SignedElements>
(86)             <sp:XPath/>
(87)               /Envelope/Header/*[@isReferenceParameter="true"]'
(88)             </sp:XPath>
(89)           </sp:SignedElements>
(90)         </wsp:Policy>
(91)
(92)         <wsp:Policy wsu:Id=”ResponsePolicy”>
(93)           <sp:SignedParts>
(94)             <sp:Body/>
(95)           </sp:SignedParts>
(96)         </wsp:Policy>
(97)
(98)       </wsp:All>
(99)     </wsp:ExactlyOne>
(100)  </wsp:Policy>
```

Hiro Kishimoto 1/9/08 6:58 AM
Formatted: Norwegian Bokmal

Hiro Kishimoto 1/9/08 6:58 AM
Formatted: Polish

Below is a detailed non-normative description for the *MUTUAL_X509* policy document:

- Lines 02-100: An encapsulating *POLICY_ASSERTION* comprised of several child *POLICY_ASSERTIONs* that serve to establish a message-level binding policy indicating a mutual authentication with X.509v3 certificates.
- Lines 08-72: A `<sp:AsymmetricBinding>` assertion which indicates that the *SENDER's* token must be used for the message signature and the *RECIPIENT's* token may be used for message encryption.

- Lines 10-28: The Initiator token assertion describes the token required of the *SENDER* by the *RECIPIENT*. Line 23 indicates that this *SENDER*-token is to be included in each message from the *SENDER* to the *RECIPIENT*. Lines 13-25 indicate the *SENDER'*s token can be one of the following:
    - An X.509 v3 certificate capable of signature-verification at a minimum
    - An ordered list of X.509 certificates packaged in a PKIPath
    - A list of X.509 certificates and (optionally) CRLs packaged in a PKCS#7 wrapper
- Lines 29-63: The Recipient token assertion describes a token identifying the RECIPIENT to be used during communication. The *RECIPIENT*-token will be embedded elsewhere within the *SECURE_ENDPOINT_REFERENCE* and may take one of three forms:
    - An X.509 v3 certificate capable of signature-verification at a minimum
    - An ordered list of X.509 certificates packaged in a PKIPath
    - A list of X.509 certificates and (optionally) CRLs packaged in a PKCS#7 wrapper

  This token will not be included in any request message. Instead, according to the `<sp:MustSupportKeyRefIdentifier>` assertion on line 76, a KeyIdentifier must be used to identify the token in any messages where it is used (e.g., for encryption).
- Lines 64-68: Indicate that the Basic256 algorithm suite (as defined in WS-SecurityPolicy Section 6.1) must be used for cryptographic activities.
- Line 69: The `<sp:OnlySignEntireHeadersAndBody>` element indicates that any signing performed must be done on entire message bodies and message headers (as opposed to selective child elements).
- Line 70: The `<sp:ProtectTokens>` element requires token protection which dictates that the signature must cover the X.509 certificate token used to generate that signature. (This enables authentication of the message origin.)
- Lines 80-90: Contains a policy that is attached to request messages to the *RECIPIENT*. Lines 82-83 specify that the message body and any WS-Addressing headers must be signed. Line 87 specifies that any WS-Addressing reference parameter headers (as identified by the `IsReferenceParameter` attribute) must be signed.
- Lines 92-96: Contains a policy that is attached to response messages from the *RECIPIENT*. The message body must be signed.

As specified in Section 4 of WS-SecurityPolicy, multiple message protection assertions (e.g., `<sp:SignedParts>`, `<sp:EncryptedParts>`, etc.) contained within a policy composition are equivalent to a single message protection assertion of the same action containing the union of all specified message parts.

APPENDIX C. REFERENCED SPECIFICATION STATUS AND ADOPTION LEVEL CLASSIFICATION

The classification of this Profile's referenced specifications at the time of writing is shown below:

Table 6 Status of specifications referenced by Secure Communication Profile 1.0

OGSA Referenced Specifications: Secure Communication Profile 1.0														
December 17, 2007	Status							Adoption						
Specification/Profile Name	De Facto	Institutional	Evolving Institutional	Draft Institutional	Consortium	Evolving Consortium	Draft	Ubiquitous	Adopted	Community	Interoperable	Implemented	Unimplemented	Note
Specifications														
None														
WS-Addressing 1.0		X									<	X		IBM, Apache implementing
WS-Policy 1.5 - Framework			X								X			W3C Proposed Recommendation
WS-Policy 1.5 - Attachment			X								X			W3C Proposed Recommendation
WS-Security Policy 1.2		X									X			OASIS Standard
Transport Layer Security		X							X					
HTTP-TLS		X							X					
X.509		X							X					ITU-T recommendation
Profiles														
None														
WS-I Basic Profile 1.1		X						▨	▨	▨	▨	▨	▨	Final Material
WS-I Basic Security Profile 1.0		X						▨	▨	▨	▨	▨	▨	Final Material

Legend:	X	Specification or profile is currently at this status or adoption level
	<	Specification or profile is approaching this status or adoption level
	▨	Status or adoption level is not applicable