Duane Merrill, UVa

Open Grid Services Architecture (OGSA®) Working Group

June 13, 2008

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.

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 versioning timestamps and 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 that facilitate the interoperability of 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.

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 [TLS 1.0], WS-Security [WS-S] and related token profiles, XML-Encryption [XML-Enc], XML-Signature [XML-DigSig], WS-Addressing [WS-A Core], etc.) is crucial for real-world interoperability.

More specifically, this profile refines the *WS-SecurityPolicy 1.2* [WS-SecurityPolicy] specification and serves to define normative, “well-known” policy documents identifying commonly-used secure communication mechanisms. WS-SecurityPolicy provides a flexible, extensible approach for specifying the security tokens, cryptographic algorithms, and protocol mechanisms (both at the transport and message levels) needed to securely communicate with a given Web service resource. This profile refines WS-SecurityPolicy in two ways:

- *Refinement of security semantics*. The WS-SecurityPolicy specification was created to describe the security mechanisms and semantics defined in the WS-Security family of specifications. While the WS-Security specifications support a broad set of requirements and offer a variety of options and approaches, they can lead to interoperability challenges that result from complexity and misinterpretation. This Profile constrains these options and simplifies communication by incorporating the conformance requirements of the WS-I Basic Security Profile 1.0 (WS-I BSP). Thus the secure communication mechanisms described by Profile-compliant policy documents must adhere to the transport (HTTP over TLS) and SOAP message security clarifications and constraints of the WS-I BSP, which are designed to greatly improve the interoperability characteristics of these technologies.
- *Schema refinement for the inclusion of key/token and timestamp information*. 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 (an important process within large, distributed systems that experience dynamic membership) and token-verification (an additional assurance check to verify the identity of the remote resource). The Profile refines the WS-SecurityPolicy specification to enable the inclusion of actual security tokens themselves (e.g., keys, certificates, etc.) within security policy documents. The Profile also refines the WS-Policy specification to profile the inclusion of WS-Security timestamp elements within policy documents to facilitate policy versioning decisions.

Additionally, the 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 for these mechanisms. Schemes for associating such security policy with specific resources (i.e., policy attachment) are out of scope of the Profile.

By itself, this document is not sufficient to guarantee the 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. Rather, the Profile adopts the view that specific secure communication requirements may vary between communities of resource providers and consumers. The intent is for applications and communities to self-select such requirements that are appropriate and then leverage this Profile to achieve interoperability between its participants (and/or cleanly discover where interoperability is not possible).

The secure-communication mechanisms referenced within the Profile are intended to facilitate the following security behaviors:

- *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 message-senders cryptographically prove knowledge of a shared secret (e.g., a password or key) that has been associated with an identity, role, or privilege. Authentication may be performed at the underlying transport-layer or the SOAP message-layer, or in combination.
- *Authorization* and *auditing*. Authenticatable identities, roles, etc. are often manifested as security tokens that can be 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 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]
wsdl	http://schemas.xmlsoap.org/wsdl	[WSDL]
soapenv	http://www.w3.org/2003/05/soap-envelope	[SOAP]
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 conformance 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 messages, XML elements, etc.) or parties (e.g., SOAP processors, end users, 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 mutually-exclusive *POLICY_ALTERNATIVEs*. 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 child element of `<wsp:ExactlyOne>` that is to be treated as a logical alternative to its sibling elements. A *POLICY_ALTERNATIVE* may be manifested as a single *POLICY_ASSERTION* (the compact policy form) or as a `<wsp:All>` element specifying a cohesive group of *POLICY_ASSERTIONs*.
- *POLICY_ASSERTION* - An individual requirement, capability, other property, or a behavior. (E.g., the `<sp:SignedParts>` element is an assertion indicating which portions of a document are to be signed.)
- *SECURITY_BINDING_ASSERTION* - A *POLICY_ASSERTION* identifying a set of properties that together provide enough information to secure message exchanges.

- *TOKEN_ASSERTION* - A *POLICY_ASSERTION* that describes a token requirement. Token assertions defined within a *SECURITY_BINDING_ASSERTION* are used to satisfy protection requirements.
- *POLICY_SUBJECT* – An entity (e.g., an endpoint, message, resource, operation, action, etc.) with which a *POLICY* can be associated.
- *ENDPOINT_POLICY_SUBJECT* – A *POLICY_SUBJECT* indicating the association of a *POLICY* with an entire Web service endpoint (i.e., a service describable by a `<wsdl:binding>` or a `<wsdl:port>`).
- *OPERATION_POLICY_SUBJECT* – A *POLICY_SUBJECT* indicating the association of a *POLICY* with a particular Web service operation (i.e., a message exchange describable by a `<wsdl:operation>`).
- *MESSAGE_POLICY_SUBJECT* – A *POLICY_SUBJECT* indicating the association of a *POLICY* with a particular message (i.e., a message describable by a `<wsdl:input>`, `<wsdl:output>`, or `<wsdl:fault>`).

This Profile defines the following conformance targets:

- *INSTANCE* – A Web service endpoint describable by a `<wsdl:port>`.
- *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* – A logical message-processing *RECIPIENT*, identifiable with a 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.)
- *RESOURCE_SECURITY_POLICY* – A *POLICY* document in conformance with the WS-SecurityPolicy refinements defined by this Profile.
- *PROFILED_MECHANISM* – A "well-known", referenceable *RESOURCE_SECURITY_POLICY.* See Appendix B for *PROFILED_MECHANISM*s 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 `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, and represents an ordered list of one or more X.509 certificates packaged in a PKIPath.
- *CRITICAL_SIGNING* – The *SENDER* signing of the following SOAP message elements in accordance with Section 8 of the WS-I BSP:
    - The entire `<soapenv:body>` message body.
    - Any header elements manifesting WS-Addressing 1.0 – SOAP Binding [WSA-SOAP] message addressing properties. These are child element

of the `<soapenv:Header>` that are either declared under the `wsa:` namespace or have a ‘`wsa:IsReferenceParameter=true`’ attribute.

- *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 certificate to be used for server certificate 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 certificate to be used for server certificate 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, integrity-protected 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/secure-communication”

4 WS-SECURITYPOLICY EXTENSIONS

This section of the Profile incorporates by reference the *WS-SecurityPolicy 1.2* specification (and therefore its parent specification, *WS-Policy 1.5 – Framework* [WS-Policy], as well). The Profile defines the following extensibility points from WS-SecurityPolicy:

- E0500 – WS-SecurityPolicy Token Assertion Extensibility – WS-SecurityPolicy allows the extensibility of *TOKEN_ASSERTIONs*. This extensibility point is used by the Profile to supplement the WS-SecurityPolicy specification with the ability to directly embed security tokens within security token assertions.
- E0501 – WS-Policy Policy Extensibility – WS-Policy allows the extensibility of *POLICY* elements. This extensibility point is used by the Profile to supplement the WS-Policy specification with the ability to add timestamp information to policy documents.

4.1 Binding of 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*.

4.1.1 Use Cases

There exist use-cases for which this capability is desirable, for example:

- *Key distribution*. Key distribution plays an important role in facilitating large, secure distributed systems that experience dynamic membership. Virtually every information security model that supports integrity and confidentiality uses some form of cryptographic key to protect communication. The task of key distribution is to supply cryptographic key(s) to the communicating parties prior to communication. A remote resource's security policy document is an attractive vehicle for distributing its public key (often in the form of a digital certificate); the client will likely need security policy information prior to communication as well. Consider a WSDL document or a WS-Addressing EPR that contains security policy indicating a resource's requirement for message-level encryption. In this case, it is convenient to furnish the recipient's X.509 certificate within the security policy document so that the caller can use it to encrypt messages to the resource.
- *Token verification*. Tokens conveyed in security policy documents can be used to provide extra authentication assurance by checking them against tokens obtained during transport-level handshakes or against signatures obtained within SOAP response messages. For example, this type of security check can be performed at the SSL/TLS level when hostname-verification is not possible: this occurs in scenarios where the remote host has a dynamic network address that cannot be reflected in a static server certificate. The certificate supplied within the TLS handshake's ServerCertificate message can be compared to the one supplied within the security policy document. Token verification facilitates a "defense in depth" strategy that may help clients detect man-in-the-middle attacks.

4.1.2 Security Considerations

The inclusion of security tokens (especially key information) within security policy documents raises several important security issues, particularly when such policy documents are attached to *POLICY_SUBJECT*s having network bindings (e.g., when security policy is attached using WSDL or EPR documents). It is important for an *INITIATOR* to verify the integrity and trustworthiness of such associations. As described above, incorporating a resource's network binding, security policy, and public key into one source location provides convenience and security advantages. However, if the policy source (WSDL/ UDDI/ EPR/ etc.) is untrusted or subject to tampering, the communicating parties may be subject to man-in-the-middle attacks leading to inadvertent disclosure of information or theft of services.

Neither WS-SecurityPolicy nor this Profile defines mechanisms for the integrity-protection of security policy documents. The schemes by which the *INITIATOR* can verify the integrity and trustworthiness of a bound *RESOURCE_SECURITY_POLICY* are specific to the attachment mechanism used to the bind *RESOURCE_SECURITY_POLICY* to its *POLICY_SUBJECT*, and are outside the scope of this document.

An embedded security token may contain, in addition to key material, information signed by an issuing authority that can be used to corroborate the remote identity. For example, an embedded X.509 certificate may contain the remote transport hostname or the remote resource's WS-Naming [WS-Naming] Endpoint Identifier (EPI). In order to trust such corroborating information, the *INITIATOR* should ensure that these signed tokens chain to a properly configured set of trust roots. The process by which an *INITIATOR* ensures the integrity and trustworthiness of an embedded security token is outside the scope of the Profile.

These integrity and trust considerations are folded into the following security recommendation:

- C0500 – An *INITIATOR* SHOULD properly ensure the integrity and trustworthiness of the *RESOURCE_SECURITY_POLICY*, of any embedded security tokens, and of the attachment mechanisms/vehicles by which they are bound to a *POLICY_SUBJECT*.

4.1.3 Schema Refinement

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 document profiles the inclusion of an alternative `<wsse:SecurityTokenReference>` element within *TOKEN_ASSERTIONs* to indicate that the required token should be obtained locally from the *RESOURCE_SECURITY_POLICY* document. The schema outline below illustrates the semantics of this schema refinement, with new semantics shown in bold:

```
<xs:complexType name="TokenAssertionType">
  <xs:sequence>
    <xs:choice minOccurs="0">
      <xs:element name="Issuer"
          type="wsa:EndpointReferenceType"/>
      <xs:element name="IssuerName"
          type="xs:anyURI" />
      <xs:element ref="wsse:SecurityTokenReference"/>
    </xs:choice>

    <xs:any minOccurs="0"
        maxOccurs="unbounded"
        namespace="##other"
        processContents="lax"/>
  </xs:sequence>
  ...
</xs:complexType>
```

The Profile establishes the following requirements and recommendations for enclosing tokens within security policy documents:

- R0500 – A WS-SecurityPolicy *TOKEN_ASSERTION* carrying an optional `<wsse:SecurityTokenReference>` MUST NOT additionally specify an `<sp:Issuer>` or an `<sp:IssuerName>` element.

- C0501 – Such a `<wsse:SecurityTokenReference>` within a *TOKEN_ASSERTION* SHOULD be an embedded or direct reference.

4.2 Adding Timestamp Information to Policy Documents

WS-SecurityPolicy *POLICY* documents specify the secure communication requirements for a *RESOURCE*. Unfortunately, neither WS-Policy nor WS-SecurityPolicy provides a way to timestamp such policy documents.

4.2.1 Use Cases

- The addition of timestamp information allows policy authors to version their policy documents and to optionally specify their expiration. The primary use-case for this versioning capability is the general scenario where a client obtains a security policy document asynchronously from when it will use the respective service. This may happen, for example, when security policy is placed in EPR documents which are then hosted within directory services (e.g., RNS). If policy changes, then there may be instances where one has different copies of policy for the same service and must decide which policy document to abide by. Providing the ability to version security policy documents using timestamps allows a client to determine which copy is the latest version.

4.2.2 Security Considerations

The inclusion of timestamp information within policy documents evokes concerns similar to those for including key information (described in Section 4.1.2). If the policy source (WSDL/ UDDI/ EPR/ etc.) is untrusted or subject to tampering, clients could be persuaded by malicious timestamp information to use malicious policy that might potentially expose them to man-in-the-middle attacks. It is important for *INITIATORs* to verify the integrity and trustworthiness of timestamped policy documents. This recommendation is codified in above in C0500.

Additionally, the Profile does not provide a mechanism for synchronizing time. The assumption is that time is trusted by means outside the scope of this document.

4.2.3 Schema Refinement

As specified in WS-Security, the schema outline for the <wsu:Timestamp> element is as follows:

```
<wsu:Timestamp wsu:Id="...">
  <wsu:Created ValueType="...">...</wsu:Created>
  <wsu:Expires ValueType="...">...</wsu:Expires>
  ...
</wsu:Timestamp>
```

WS-Policy specifies that <wsp:Policy> elements can nest arbitrary child elements. This document profiles the optional inclusion of a <wsu:TimeStamp> element within *POLICY* elements to indicate the creation and expiration times of the policy's security semantics. The schema outline below illustrates the semantics of this schema refinement, with new semantics shown in bold:

```
<xs:element name="Policy">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="tns:OperatorContentType">
        <xs:attribute name="Name" type="xs:anyURI" />
        <xs:anyAttribute namespace="##any" processContents="lax" />
        <xs:sequence>
          <xs:element ref="wsu:Timestamp" minOccurs="0" maxOccurs="1" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
```

```
...
<xs:complexType name="OperatorContentType">
  <xs:sequence>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element ref="tns:Policy" />
      <xs:element ref="tns:All" />
      <xs:element ref="tns:ExactlyOne" />
      <xs:element ref="tns:PolicyReference" />
      <xs:any namespace="##other" processContents="lax" />
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

The Profile establishes the following requirements and recommendations for enclosing timestamp elements within security policy documents:

- R0501 – There MUST be at most one <wsu:Timestamp> direct child element per <wsp:Policy> element.
- R0502 – Instances of the <wsu:Timestamp> element must abide by the requirements of the WS-I BSP, Section 6.
- C0502 – All time references SHOULD be specified using the value type xsd:dateTime.
- C0503 – Implementations SHOULD NOT rely on other applications supporting time resolution finer than milliseconds.

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 *RESOURCE* 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* within response messages.

In some scenarios, the *INITIATOR* may not be able to authenticate the *RESOURCE* before sending application-specific payload data to it. Such cases include session-less request/response message exchanges with message-level authentication (where the *RESOURCE* is authenticated to the *INITIATOR* upon receipt of the response message) as well as truly one-way message patterns. If the *INITIATOR* is concerned with who receives (or can inspect) its message, then it should employ encryption for message confidentiality.

The Profile defines the following authentication recommendations:

- C0504 – 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:

- C0505 – 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:

- C0506 – 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*.

- R0503 – 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_MECHANISMs* defined within this profile.

Table 2 Secure Transport Mechanisms

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

Table 3 Secure Message Mechanisms

Mechanism Name	Conformance Target	Policy Reference URI
Username Token	*USERNAME_TOKEN*	`http://www.ogf.org/ogsa/2007/05/secure-communication#UsernameToken`
Password Digest Username Token	*PASSWORD_DIGEST*	`http://www.ogf.org/ogsa/2007/05/secure-communication#PasswordDigest`
Mutually Authenticated X.509 Binding	*MUTUAL_X509*	`http://www.ogf.org/ogsa/2007/05/secure-communication#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:

- E0502 – 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

It is anticipated that FIPS-compliant implementations would support the FIPS-equivalent versions of the above ciphersuites (e.g., TLS_RSA_FIPS_WITH_AES_256_CBC_SHA).

6.3 Server-Authenticated TLS (*SERVER_TLS*) Policy

The *SERVER_TLS* policy is an endpoint-wide 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.

- R0505 – 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
- R0506 –TLS/SSL ClientHello messages MUST indicate a 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).
- R0507 –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. *ENDPOINT_POLICY_SUBJECT*
- R0508 – The *SERVER_TLS* policy MUST be referenced with the policy reference URI `"http://www.ogf.org/ogsa/2007/05/secure-communication#ServerTLS"`
- R0509 – The *SERVER_TLS* policy MUST apply to an *ENDPOINT_POLICY_SUBJECT*.

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

The *SERVER_TLS_CERT_PROVIDED* policy is an endpoint-wide 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.

- R0510 – The actions upon a *RESOURCE*s for which the *SERVER_TLS_CERT_PROVIDED* policy is advertised MUST support the following:
    - SOAP over HTTPS

  - o An SSL or TLS handshake with server authentication

- R0511 –TLS/SSL ClientHello messages MUST indicate a 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).
- R0512 –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.
- R0513 – 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.
- R0514 – The *SERVER_TLS_CERT_PROVIDED* policy MUST be referenced with the policy reference URI `"http://www.ogf.org/ogsa/2007/05/secure-communication#ServerTLSCertProvided"`
- R0515 – The *SERVER_TLS_CERT_PROVIDED* policy MUST apply to an *ENDPOINT_POLICY_SUBJECT*.

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 an endpoint-wide 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.

- R0516 – The actions upon a *RESOURCE*s for which the *MUTUAL_TLS* policy is advertised MUST support the following:
  - o SOAP over HTTPS
  - o An SSL or TLS handshake with both client and server authentication
- R0517 –TLS/SSL ClientHello messages MUST indicate a 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 – The *MUTUAL_TLS* policy MUST be referenced with the policy reference URI `"http://www.ogf.org/ogsa/2007/05/secure-communication#MutualTLS"`
- R0520 – The *MUTUAL_TLS* policy MUST apply to an *ENDPOINT_POLICY_SUBJECT*.

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

The *MUTUAL_TLS_CERT_PROVIDED* policy is an endpoint-wide 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.

- R0521 – 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
- R0522 –TLS/SSL ClientHello messages MUST indicate a 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).
- R0523 –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.
- R0524 – 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.
- R0525 – The *MUTUAL_TLS_CERT_PROVIDED* policy MUST be referenced with the policy reference URI "`http://www.ogf.org/ogsa/2007/05/secure-communication#MutualTLSCertProvided`"
- R0526 – The *MUTUAL_TLS_CERT_PROVIDED* policy MUST apply to an *ENDPOINT_POLICY_SUBJECT*.

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:

- E0503 – 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 *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 *USERNAME_TOKEN* policy can be associated with *POLICY_SUBJECT*s at the endpoint, operation, or message scope. The normative policy document for the *USERNAME_TOKEN* policy is defined in Appendix B.

- R0527 – Messages for which the *USERNAME_TOKEN* policy is advertised MUST include a Username/Token credential in accordance with the WS-I BSP.
- R0528 – The *USERNAME_TOKEN* policy MUST be referenced with the policy reference URI "`http://www.ogf.org/ogsa/2007/05/secure-communication#UsernameToken`"
- R0529 – The `<sp:SignedEncryptedSupportingTokens>` element within the *USERNAME_TOKEN* policy indicates that the Username/Token credential MUST be digitally signed and encrypted. This requirement can be fulfilled by having the

referencing *RESOURCE_SECURITY_POLICY* additionally reference at least one of the following companion *PROFILED_MECHANISM* policies:

- A transport-level *PROFILED_MECHANISM* that provides confidential and integrity-protected communication (e.g., *SERVER_TLS*, *MUTUAL_TLS*, etc.)
- A message-level *PROFILED_MECHANISM* that provides confidential and integrity-protected communication. (E.g., co-referencing the *CONFIDENTIAL_MUTUAL_X509* policy would indicate encryption of the Username/Token element using the *CONFIDENTIAL_MUTUAL_X509's RECIPIENT* token, and signature using the *INITIATOR* token.)

- R0530 – The *USERNAME_TOKEN* policy MUST apply to either an *ENDPOINT_POLICY_SUBJECT*, an *OPERATION_POLICY_SUBJECT*, or a *MESSAGE_POLICY_SUBJECT*

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 *PASSWORD_DIGEST* policy can be associated with *POLICY_SUBJECT*s at the endpoint, operation, or message scope. The normative policy document for the *PASSWORD_DIGEST* policy is defined in Appendix B.

- R0531 – Messages for which the *PASSWORD_DIGEST* policy is advertised MUST include a password-digest Username/Token credential in accordance with the WS-I BSP.
- R0532 – The *PASSWORD_DIGEST* policy MUST be referenced with the policy reference URI `"http://www.ogf.org/ogsa/2007/05/secure-communication#PasswordDigest"`
- R0533 – The *PASSWORD_DIGEST* policy MUST apply to either an *ENDPOINT_POLICY_SUBJECT*, an *OPERATION_POLICY_SUBJECT*, or a *MESSAGE_POLICY_SUBJECT*

It should be noted that the password digest username-token is susceptible to replay attacks on other services. The digested token is not cryptographically bound to the message that carries it, allowing it to be pasted into other messages to other services (that may have not yet seen the included nonce and timestamp).

7.4 Mutually Authenticated X.509 Binding (*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). These X.509 token requirements are indicated within the policy as an *INITIATOR* and *RECIPIENT* token requirements. The *MUTUAL_X509* policy can be associated with *POLICY_SUBJECT*s at the endpoint or operation scope.

This policy requires *CRITICAL_SIGNING* for applicable messages: signature over the `<soapenv:Body>` message body as well as any WS-Addressing message-addressing headers.

If a message pattern for which this policy is associated with requires multiple messages, the *INITIATOR* token is used for the message signature from the *INITIATOR* to the *RECIPIENT*. The *RECIPIENT* token is used for the response message signature from the *RECIPIENT* to the *INITIATOR*.

The *MUTUAL_X509* policy requires that any referencing *RESOURCE_SECURITY_POLICY* also embed the *RECIPIENT*'s X.509 certificate as a *RECIPIENT_MESSAGE_IDENTITY*. By including the *RECIPIENT's* X.509 binary security token within the policy, the *INITIATOR* can further verify signatures over any response messages. Additionally, the *INITIATOR* is to use this X.509 binary security token for any message-level encryption actions to the *RECIPIENT* that may be specified by supplementary message protection policies (see the example below).

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

- R0535 – Message exchanged with a *RESOURCE* for which the *MUTUAL_X509* policy is advertised MUST have *CRITICAL_SIGNING* performed on them in accordance with the WS-I BSP.
- R0536 – Sign before encrypting: signature MUST be computed over plaintext. The resulting signature can then be encrypted if required by accompanying policy.
- R0537 – 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.
- R0538 – The *MUTUAL_X509* policy MUST be referenced with the policy reference URI `"http://www.ogf.org/ogsa/2007/05/secure-communication#MutualX509"`
- R0539 – The *MUTUAL_X509* policy MUST apply to either an *ENDPOINT_POLICY_SUBJECT* or an *OPERATION_POLICY_SUBJECT.* Additional message protection assertions (e.g., `<sp:SignedParts>`, `<sp:EncryptedParts>`, etc.) can be specified within policies at the same or lower *POLICY_SUBJECT* scope (e.g., *OPERATION_POLICY_SUBJECT* or *MESSAGE_POLICY_SUBJECT* scope) to create an effective policy with additional signed/encrypted element requirements.
- R0540 – All messages for which the *MUTUAL_X509* policy is applicable MUST include `<wsu:Timestamp>` header elements as per WS-Security and the WS-I BSP
- C0507 – *RESOURCE_SECURITY_POLICIES* that incorporate the *MUTUAL_X509* policy MAY specify additional portions of the message documents to be signed and/or encrypted.

Because the *MUTUAL_X509* requires message signature, it can be used by *RESOURCE_SECURITY_POLICIES* to describe holder-of-key subject confirmation semantics for additional authentication tokens. For example, the following *RESOURCE_SECURITY_POLICY* requires that the *INITIATOR* authenticate to the *RECIPIENT* with a holder-of-key SAML assertion. (The XML digital signature provided by the *MUTUAL_X509* policy provides proof that the *INITIATOR* possesses the private key to which the SAML assertion has been bound). The policy also provides message confidentiality, including encryption of any WS-Addressing action headers.

```
<wsp:Policy>

  <wsp:PolicyReference>
    http://www.ogf.org/ogsa/.../secure-communication#MutualX509
  </wsp:PolicyReference>

  <sp:SignedSupportingTokens>
    <wsp:Policy>
      <sp:SamlToken sp:IncludeToken=".../AlwaysToRecipient">
        <wsp:Policy>
          <sp:WssSamlV20Token1.1>
        </wsp:Policy>
      </sp:SamlToken>
    </wsp:Policy>
  </sp:SignedSupportingTokens>

  <wsp:Policy>
    <sp:EncryptedParts>
```

```
        <sp:Body/>
        <Header name="Action" namespace="http://www.w3.org/2005/08/addressing"/>
      </sp:EncryptedParts>
    </wsp:Policy>

</wsp:Policy>
```

8 EXAMPLE SOAP REQUEST MESSAGE

The following shows an example of an input message to an RNS (Resource Naming Service) *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>
```

```
(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

11 DISCLAIMER

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 M. 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-A SOAP] M. Gudgin, M. Hadley, T. Rodgers (ed.), Web Services Addressing 1.0 – SOAP Binding, W3C Candidate Recommendation 9 May 2006, http://www.w3.org/TR/2006/REC-ws-addr-soap-20060509/
- [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
- [WSDL] E. Christensen, F. Curbera, G. Meredith, S. Weerawarana: Web Services Description Language (WSDL) 1.1. W3C Note, March 15, 2001. http://www.w3.org/TR/wsdl
- [SOAP] M. Gudgin, M. Hadley, N. Mendelsohn, J. Moreau, H. Nielsen, A. Karmarkar, Y. Lafon: SOAP Version 1.2 Part 1: Messaging Framework (Second Edition). W3C Note, March 15, 2001. http://www.w3.org/TR/soap12-part1/
- [WS-Naming] A. Grimshaw, D. Snelling: WS-Naming Specification. OGSA Naming Working Group, Open Grid Forum, April 30, 2007. https://forge.gridforum.org/sf/go/doc14420?nav=1

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*.
- E0501 – WS-Policy Policy Extensibility – WS-Policy allows the extensibility of *POLICY* elements.

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

- E0502 – Additional transport-level binding assertions may be profiled in accordance to the requirements in Section 5.1: Security Mechanism Specifics.
- E0503 – 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)
(08)       <sp:TransportToken>
(09)         <wsp:Policy>
(10)           <sp:HttpsToken />
(11)         </wsp:Policy>
(12)       </sp:TransportToken>
(13)
(14)       <sp:AlgorithmSuite>
(15)         <wsp:Policy>
(16)           <wsp:ExactlyOne>
(17)             <sp:Basic256 />
(18)             <sp:Basic128 />
(19)           </wsp:ExactlyOne>
(20)         </wsp:Policy>
(21)       </sp:AlgorithmSuite>
(22)
(23)     </wsp:Policy>
(24)   </sp:TransportBinding>
(25) </wsp:Policy>
```

The *SERVER_TLS* policy can be associated with *POLICY_SUBJECT*s at the endpoint scope. Below is a detailed non-normative description for the *SERVER_TLS* policy document:

- Lines 02-25: *POLICY* for a `<sp:TransportBinding>` transport binding indicating server-authenticated transport layer security in accordance with this Profile.
- Lines 08-12: Transport token element indicating that the transport binding support the use of HTTPS
- Lines 14-21: 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:

```
<?xml version="1.0" encoding="UTF-8"?>
<wsp:Policy wsu:Id=”ServerTLSCertProvided”
    xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
    xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702">
  <sp:TransportBinding>
    <wsp:Policy>

      <sp:TransportToken>
        <wsp:Policy>
          <sp:HttpsToken>
            <wsse:SecurityTokenReference>
```

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

The *SERVER_TLS_CERT_PROVIDED* policy can be associated with *POLICY_SUBJECT*s at the endpoint scope. Below is a detailed non-normative description for the *SERVER_TLS_CERT_PROVIDED* policy document:

- Lines 02-30: *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 08-17: Transport token element indicating that the transport binding support the use of HTTPS.
- Lines 10-15: 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 19-26: 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:

```
<?xml version="1.0" encoding="UTF-8"?>
<wsp:Policy wsu:Id="MutualTLS"
    xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
    xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702">
  <sp:TransportBinding>
    <wsp:Policy>

      <sp:TransportToken>
        <wsp:Policy>
          <sp:HttpsToken>
            <wsp:Policy>
              <sp:RequireClientCertificate />
            </wsp:Policy>
          </sp:HttpsToken>
        </wsp:Policy>
      </sp:TransportToken>

      <sp:AlgorithmSuite>
        <wsp:Policy>
              <wsp:ExactlyOne>
                <sp:Basic256 />
```

```
(22)                 <sp:Basic128 />
(23)               </wsp:ExactlyOne>
(24)         </wsp:Policy>
(25)       </sp:AlgorithmSuite>
(26)
(27)     </wsp:Policy>
(28)   </sp:TransportBinding>
(29) </wsp:Policy>
```

The *MUTUAL_TLS* policy can be associated with *POLICY_SUBJECT*s at the endpoint scope. Below is a detailed non-normative description for the *MUTUAL_TLS* policy document:

- Lines 02-29: *POLICY* for a `<sp:TransportBinding>` transport binding indicating server-authenticated transport layer security in accordance with this Profile.
- Lines 08-16: Transport token element indicating that the transport binding support the use of HTTPS
- Lines 10-14: Policy for the `<sp:HttpsToken>` element indicating that the client certificate is required for authentication.
- Lines 18-25: 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:

```
<?xml version="1.0" encoding="UTF-8"?>
<wsp:Policy wsu:Id=”MutualTLSCertProvided”
    xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
    xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702">
  <sp:TransportBinding>
    <wsp:Policy>

      <sp:TransportToken>
        <wsp:Policy>
          <sp:HttpsToken>
            <wsse:SecurityTokenReference>
              <wsse:Reference URI='#RecipientTransportIdentity'
                 ValueType=" http://docs.oasis-open.org/wss/2004/01/oasis-200401-
                       wss-x509-token-profile-1.0#X509v3" />
            </wsse:SecurityTokenReference>
            <wsp:Policy>
              <sp:RequireClientCertificate />
            </wsp:Policy>
          </sp:HttpsToken>
        </wsp:Policy>
      </sp:TransportToken>

      <sp:AlgorithmSuite>
        <wsp:Policy>
          <wsp:ExactlyOne>
            <sp:Basic256 />
            <sp:Basic128 />
          </wsp:ExactlyOne>
        </wsp:Policy>
      </sp:AlgorithmSuite>

    </wsp:Policy>
  </sp:TransportBinding>
</wsp:Policy>
```

The *MUTUAL_TLS_CERT_PROVIDED* policy can be associated with *POLICY_SUBJECT*s at the endpoint scope. Below is a detailed non-normative description for the *MUTUAL_TLS_CERT_PROVIDED* policy document:

- Lines 02-33: *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 10-18: 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 22-29: 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">
(05)
(06)   <sp:SignedEncryptedSupportingTokens>
(07)     <wsp:Policy>
(08)       <sp:UsernameToken/>
(09)     </wsp:Policy>
(10)   </sp:SignedEncryptedSupportingTokens>
(11)
(12) </wsp:Policy>
```

The *USERNAME_TOKEN* policy can be associated with *POLICY_SUBJECT*s at the endpoint, operation, or message scope. Below is a detailed non-normative description for the *USERNAME_TOKEN* policy document:

- Lines 06–10 contain the `<sp:SignedEncryptedSupportingTokens>` assertion which includes a `<sp:UsernameToken>` indicating that a UsernameToken must be included in the message security header. The `<sp:SignedEncryptedSupportingTokens>` element indicates that the UsernameToken must be integrity and confidentiality protected via security mechanisms either at the transport-level (e.g., *SERVER_TLS*) or at the message-level.

B.6. *PASSWORD_DIGEST* Policy Document

The *PASSWORD_DIGEST* policy can be associated with *POLICY_SUBJECT*s at the endpoint, operation, or message scope. The normative policy document for the *PASSWORD_DIGEST* policy is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<wsp:Policy wsu:Id="PasswordDigest"
    xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
    xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702">

  <sp:SupportingTokens>
    <wsp:Policy>
```

```
(08)       <sp:UsernameToken>
(09)         <wsp:Policy>
(10)           <sp:HashPassword/>
(11)         </wsp:Policy>
(12)       </sp:UsernameToken>
(13)     </wsp:Policy>
(14)   </sp:SupportingTokens>
(15)
(16) </wsp:Policy>
```

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

- Lines 06–14: contain the `<sp:SignedSupportingTokens>` assertion which includes a `<sp:UsernameToken>` indicating that a password-digest UsernameToken must be included in the security header.
- Line 09 – 11: 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 *MUTUAL_X509* policy can be associated with *POLICY_SUBJECT*s at the endpoint or operation scope. The normative policy document for the *MUTUAL_X509* policy is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<wsp:Policy wsu:Id=”MutualX509”
    xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
    xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702">

  <sp:AsymmetricBinding>
    <wsp:Policy>

      <sp:InitiatorToken>
        <wsp:Policy>
          <sp:X509Token sp:IncludeToken="http://docs.oasis-open.org/ws-sx/ws-
                   securitypolicy/200512/IncludeToken/AlwaysToRecipient">
            <wsp:Policy>
              <wsp:ExactlyOne>
                <sp:WssX509V3Token11/>
                <sp:WssX509PkiPathV1Token11/>
                <sp:WssX509Pkcs7Token11/>
              </wsp:ExactlyOne>
            </wsp:Policy>
          </sp:X509Token>
        </wsp:Policy>
      </sp:InitiatorToken>

      <sp:RecipientToken>
        <wsp:Policy>
          <sp:X509Token sp:IncludeToken="http://docs.oasis-open.org/ws-sx/ws-
                   securitypolicy/200512/IncludeToken/Never>
            <wsse:SecurityTokenReference>
              <wsse:Reference URI='#RecipientMessageIdentity'
                   ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
                   wss-x509-token-profile-1.0#X509PKIPathv1"/>
            </wsse:SecurityTokenReference>
            <wsp:Policy>
              <sp:WssX509V3Token11/>
            </wsp:Policy>
          </sp:X509Token>
        </wsp:Policy>
      </sp:RecipientToken>

      <sp:AlgorithmSuite>
        <wsp:Policy>
          <wsp:ExactlyOne>
```

```
(39)             <sp:Basic256 />
(40)             <sp:Basic128 />
(41)           </wsp:ExactlyOne>
(42)         </wsp:Policy>
(43)       </sp:AlgorithmSuite>
(44)
(45)       <sp:OnlySignEntireHeadersAndBody/>
(46)       <sp:IncludeTimestamp/>
(47)       <sp:ProtectTokens>
(48)
(49)     </wsp:Policy>
(50)   </sp:AsymmetricBinding>
(51)
(52)   <sp:Wss10>
(53)     <wsp:Policy>
(54)       <sp:MustSupportRefKeyIdentifier/>
(55)     </wsp:Policy>
(56)   </sp:Wss10>
(57)
(58)   <wsp:Policy>
(59)     <sp:SignedParts>
(60)       <sp:Body/>
(61)       <Header namespace="http://www.w3.org/2005/08/addressing"/>
(62)     </sp:SignedParts>
(63)     <sp:SignedElements>
(64)        <sp:XPath>Envelope/Header/*[@isReferenceParameter="true"]</sp:XPath>
(65)     </sp:SignedElements>
(66)   </wsp:Policy>
(67)
(68) </wsp:Policy>
```

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

- Lines 02-67: An encapsulating *POLICY_ASSERTION* comprised of several child *POLICY_ASSERTIONs* that serve to establish a binding policy indicating secure message-level communication using X.509v3 certificates.
- Lines 06-50: A `<sp:AsymmetricBinding>` assertion which indicates that the *INITIATOR's* token must be used for message signature (and the *RECIPIENT's* token must be used for message encryption if encryption is required by ancillary policy). If the policy is bound to a *POLICY_SUBJECT* with a message exchange pattern having a response message, the response message must use *RECIPIENT's* token for message signature (and the *INITIATOR's* token must be used for message encryption if encryption is required by ancillary policy).
- Lines 09-21: 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 14-16 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 23-34: 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 must be an `X509PKIPathv1` ordered list of one or more certificates beginning with the *RECIPIENT's* identity certificate. The *RECIPIENT's* identity certificate 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 this certificate in any messages where it is used (e.g., for encryption).

- Lines 36-43: Algorithm suite element indicating that either the Basic256 or the Basic128 algorithm suite (see WS-SecurityPolicy Section 6.1) may be used.
- Line 45: The `<sp:OnlySignEntireHeadersAndBody>` element indicates that any signing performed must be done over the entire message body element and/or entire message header elements (as opposed to selective child elements within headers or within the message body).
- Line 46: The `<sp:IncludeTimestamp>` element indicates the required inclusion of `<wsu:Timestamp>` elements within message headers.
- Line 47: 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 58-66: Message protection requirements for the endpoint or operation to which this policy is applied to. Lines 60-61 specify that the message body and any WS-Addressing headers must be signed. Line 64 specifies that any WS-Addressing reference parameter headers (as identified by the `IsReferenceParameter` attribute) must be signed.

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