# DYNAMIC NEGOTIATION OF SECURITY ARRANGEMENTS BETWEEN WEB SERVICES

## COPYRIGHT NOTICE

**[0001]** A portion of the disclosure of this patent document contains material that is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent disclosure as it appears in the Patent and Trademark Office patent file or records, but otherwise reserves all copyright rights whatsoever.

## REFERENCE TO COMPUTER PROGRAM LISTING APPENDIX

**[0002]** A computer program listing appendix appears immediately before the claims. The computer program listing appendix includes the following program excerpts:

| | |
|---|---|
| SecuritySenderReceiverInfo.XSD | (schema for negotiation inputs.) |
| SecurityContractKeyInfo.XSD | (schema for keys used for security.) |
| SecurityContract.XSD | (schema for security contract output from negotiation.) |
| CommunitySecurityTemplatesInfo.XML | (schema for negotiation inputs.) |
| SecuritySenderInfo.XML | (sender info in example.) |
| SecurityReceiverInfo.XML | (sender info in example.) |
| ComputeSecurityContract.XML | (computed security contract in example.) |

## BACKGROUND OF THE INVENTION

**[0003]** The present invention relates to computer-based devices and methods to negotiate and implement security arrangements between two or more Web Services. More particularly, it relates to devices and methods that specify input and output interfaces, compute and generate a security contract consistent with inputs, and implement security in accordance with negotiated security arrangements. Particular aspects of the present invention are described in the claims, specification and drawings.

**[0004]** Business-to-business (B2B) and application-to-application (A2A) electronic commerce are replacing former protocols for electronic data interchange (EDI). As businesses strive to improve their efficiency with B2B and A2A systems, a number of incompatible platforms and competing standards have emerged. Among compatible standards, gaps remain to

be filled. For instance, the industry has defined what a simple web service is. Standards related to simple web service include UDDI, WSDL, XSDL and SOAP. However, these standards do not fully meet the security, reliability, manageability, and choreography requirements for practical B2B and A2A electronic commerce. Security in particular presents numerous options and configuration issues. Collaborative web services and their security needs are expected to evolve as non-web businesses do. There is no any comprehensive or unified device or method that dynamically resolves and updates security options and configurations as web services evolve.

**[0005]** There are a number of industry initiatives to extend standards applicable to B2B and A2A electronic commerce. Choreography efforts include ebXML/BPSS from OASIS, WSFL from IBM, and XLANG from Microsoft. Conversation efforts include ebXML/TRP from OASIS and Microsoft's WS-routing. The dominant security effort is WS-security from IBM and Microsoft, there is also a complementary security effort in OASIS called SAML. For reliability, there are proposals from Microsoft, ebXML/TRP from OASIS, and HTTPR from IBM. W3C is addressing standardization in all of these areas. Key industry players have formed a rival consortium called WSI. However, they have not addressed the dynamic security negotiation issue.

**[0006]** Accordingly, an opportunity arises to develop methods and devices that dynamically resolve security option and configuration issues for trading partners.

## SUMMARY OF THE INVENTION

**[0007]** The present invention relates to computer-based devices and methods negotiate and implement security arrangements between two or more web services. More particularly, it relates to devices and methods that specify input and output interfaces, computation and generation of a security contract consistent with inputs, and implementation of security in accordance with negotiated security arrangements. Particular aspects of the present invention are described in the claims, specification and drawings.

## BRIEF DESCRIPTION OF THE DRAWINGS

**[0008]** Figure 1 illustrates communities and networks of communities, which are one environment in which computer-assisted, dynamic negotiation of security arrangements is useful.

**[0009]** Figure 2 depicts negotiation and implementation of security arrangements.

**[0010]** Figure 3 illustrates reconciling preferences among algorithm types.

[0011] Figure 4 illustrates alternative embodiments for obtaining receiver's information when the sender is local to calculations of the security arrangements.

[0012] Figure 5 illustrates one network of program logic and resources that can be used to implement aspects of the present invention.

## DETAILED DESCRIPTION

[0013] The following detailed description is made with reference to the figures. Preferred embodiments are described to illustrate the present invention, not to limit its scope, which is defined by the claims. Those of ordinary skill in the art will recognize a variety of equivalent variations on the description that follows.

[0014] Figure 1 illustrates communities and networks of communities, which are one environment in which computer-assisted, dynamic negotiation of security arrangements is useful. Among these communities, a community maintains a local registry that includes information such as users, companies, services and connectors that are part of the community. The community can be a marketplace, an enterprise or a sub enterprise. Communities can belong to one or more community networks. Typically, communities and networks have some common business interest. Interoperation is between member communities in one or more community networks. The networks include a gold marketplace network 1, a precious metal marketplace network 2, a private network 3 and a global trading web network 4. In this illustration, the gold marketplace network 1 and the precious metal marketplace network 2 are contained within the global trading web network 4. The precious metals marketplace network 2 includes gold and silver marketplaces 14, 13. Gold marketplace customers can trade silver in the silver marketplace 13 and silver marketplace customers can trade in gold 14. One community, PQR Enterprise 17 belongs to the gold marketplace network 1, the private network 3 and the global trading web network 4; another community, ABC Big Supplier 18 belongs to the private network 3. In this illustration, XYZ Gold 14 is a marketplace or community for trading gold. Enterprises belong to this community. Enterprises like PQR Enterprise 17 that have formed a community by themselves belong to the gold marketplace network 1. These communities are part of the gold marketplace network 1, and the global trading web network 4. Small supplier 15 is part of the gold marketplace community. Other enterprises 16 are communities that are part of the gold marketplace community network 1. The connections between XYZ Gold 14 and other gold marketplace entities 15-17 indicate that the gold marketplace requires all traffic between enterprises (communities or otherwise) transacting gold trading to be routed through XYZ

Gold14, for instance, to collect billing and business intelligence information. PQR Enterprise 17 is a community is part of the gold marketplace and also part of local private network with supplier 18. Small supplier 15 may be an individual small supplier that does not want to form a community by itself and instead registers its metadata, such as users, organizations, services and transformations, in the registry of the gold marketplace. On the other hand, ABC Big Supplier 18 has formed a private network of its own, for instance because it wants to keep its metadata, internal back office systems and transformations hidden from general public access because they were developed at considerable cost. Because PRQ 17 is a customer of ABC 18, it participates in the private network 3. Financial service provider DEF Financial 12 wants to provide financial services to anyone in the global trading web network 4, such forms a community of its own and registers with the global trading web root 11. A network of communities makes available a global registry of communities. The global registry permits lookup of the community and determination of one or more routes to that community, or to external connectors through which the electronic commerce documents bound for the community may be routed. Documents routed from one community to another may be routed directly between external connectors for the two communities or indirectly through one or more intermediary communities. Business and security rules for transactions involving the communities also can be defined and maintained in community registries. In general, figure 1 illustrates the mixed loyalties of entities and communities that create an impetus for interoperability among electronic commerce platforms.

**[0015]** Connector is a general term for applications that communicate with other applications. Connectors may communicate on a peer-to-peer (P2P) basis or on a directed basis through other connectors that function as hubs, gateways, external ports, central connectors, etc. Connectors that communicate P2P are able to communicate with other connectors that use the same transport/envelope protocols. Connectors that communicate P2P optionally may enlist the assistance of other hub connectors that perform translation services, when trying to communicate with a connector that does not use the same transport/envelope protocol. Connectors that communicate on a directed basis communicate through hub connectors according to routing rules. Routing rules among connectors can be mapped in a directed graph, supporting one or more hub and spoke topologies for one or more transport/envelope protocols. A hub and spoke topology directs communications along spokes to hubs, in one or more tiers. This facilitates centralized services such as billing, business intelligence collection, tracking, auditing, accounting, or others. Multiple hub and spoke organizations may overlay the same connectors to support different transport/envelope protocols and technologies, as suggested by Figure 2. For instance, a stronger hub and spoke organization may be required to use Sonic as a transport

technology than to use HTTP or HTTPS. Optionally, communication routes may depend on whether the source and destination are part of the same community. Within a sub-community (which may include the whole community), centralized functions may be unneeded and P2P communications permitted among connectors that otherwise are directed to communicate with parent connectors when communicating with destinations in other sub-communities.

**[0016]** Connectors may be labeled simple connectors (sometimes simply called connectors), hubs (sometimes called gateways or routers) or central connectors. Alternatively, they may be described functionally. Simple connectors are directed to communicate via hub connectors, except when they are permitted to communicate P2P among connectors in the same sub-community. So-called hubs are used by connectors that are explicitly directed or linked to them. Hubs may serve more than one function and, accordingly, may appear more than once in a route from a source to a destination. Hubs forward electronic commerce documents or messages. Hubs also may translate among transport protocols that support a common envelope protocol. For instance, a hub may translate envelope protocols and also implement a different transport protocol upon transmission than upon receipt. A central connector is a special case of a hub, which can be used by connectors that are not explicitly directed or linked to them. A central connector is useful, for instance, to carry out translation functions when traversing connectors from a source according to routing rules does not lead to any hub that supports the transport/envelope protocol used by the destination.

**[0017]** A schema and a process flow provide an overview of security arrangements according to aspects of the present invention. In this context, negotiation of security arrangements is carried out by a computer-based process that uses security profiles of sending and receiving services to determine a mutually agreeable security arrangement. Preferably, this security arrangement is negotiated or potentially updated regularly, without user intervention. This arrangement may be negotiated, updated or checked for validity at a user request or without user intervention whenever messages are exchanged or on some other periodic or occasional basis, such as monthly, weekly, daily, on occurrence of an event that impacts exchange of messages between a particular sender and receiver (e.g., a software component failure or a change in security preferences), when a previously negotiated arrangement fails, or on some other periodic or occasional basis. The schema SecuritySenderReceiverInfo.XSD, in the source code appendix, describes some inputs to negotiation of security arrangements. The schema SecurityContract.XSD, also in the source code appendix, describes one embodiment of negotiated security arrangements, in a so-called security interoperability contract document

("SCID"). The process flow Figure 1 can be used to describe negotiation and implementation of security arrangements.

**[0018]** The schema SecuritySenderReceiverInfo.XSD, in the source code appendix, can be used to validate a plurality input files to negotiation of security arrangements. In this embodiment, the machine-readable input files are XML documents. In other embodiments, other data structures may be used to store the same information, for instance a tree structure modeled after the XML code. The schema SecuritySenderReceiverInfo.XSD is best understood by loading the file into an integrated development environment (IDE) such as XML Spy TM, which provides several alternative views of the schema, including a documentation generation view. Sender and receiver security interoperability contract document information blocks are defined by this schema. Viewed in Spy's schema design view, SecuritySenderReceiverInfo.XSD includes several components that are used to define sender and receiver security information. The CommunitySecurityPolicyPreference component states the community preferences to sign the header, encrypt the credential, and credential preferences. It can be used to specify a default value for a whole community or it could be adapted to specify a default value for a collaboration partner (CP). The SAMsgSecurityPolicy component allows specification of signature and encryption preferences and authentication options. Message exchanged between services may have multiple parts. Signature and encryption policies can be applied to the whole message or individual parts. This approach can readily be extended to applying signature and encryption policies to elements within the parts. The PublicKeys component identifies key records for this CP. The ConnectorCapability component provides routing information to a resource that implements part of the security arrangement, such as a connector name. It includes connector capability parameters such as encryption capability, signature capability, an encryption public key party, and signing public key party. Public key party can be the sender's CP, the receiver's CP, or the owner of the connector, depending on whether signing or encryption is involved. If the public key party is not defined, the key of the message sender can be used for signing and the key of the message receiver can be used for encryption. The SecurityContainer component can be used to carry additional objects that are useful for security. The SendingCPSecurityPolicyProfile component includes the sending CP's available credentials information. The CPSendServicesSecurityPolicy and CPRecvServicesSecurityPolicy component include sets of security policies for the sending and receiving services, respectively. The services preferences and overrides can be defined here.

**[0019]** The schema SecurityContract.XSD, also in the source code appendix, can be used as a model for preparing a machine-readable security interoperability contract document. In this

embodiment, the machine-readable document is an XML document. In other embodiments, other data structures may be used to store the same information, for instance a tree structure modeled after the XML code. This schema defines policies and channels for security policies. A security channel defines resources and routes to resources that carry out security algorithms, such as signature, encryption and authentication algorithms. It also may include non-repudiation and authorization resources.

**[0020]** The process flow Figure 2 can be used to describe negotiation and implementation of security arrangements. In one embodiment, the preferences of the sending and receiving services are maintained in a registry 201. This registry may be accessible to the sending and receiving services, so that either service can compute security arrangements, or it may be available to a security arrangement computing service that is accessible to one or both of the sending and receiving services. The sending and receiving services may maintain their own registries. Or, a protocol may be developed for the sending and receiving services to exchange their security preferences as part of the negotiation of security arrangements. A registry 201 further may maintain information regarding default preferences of a collaboration partner that owns a service or a community to which a collaboration partner belongs, or both. Default preferences may be overridden by service-specific preferences, in general, or certain default preferences may be given precedence over service-specific preferences. Default preferences of collaboration partners may be treated differently than default preferences of the community. Input statements of security arrangement preferences are taken from the registry 201 or another source and acted upon by a security arrangements computing service 202. In one embodiment, this computing service is a security contract builder. A set of security arrangements are output 203. These arrangements may be confirmed with the sending and receiving services, may be subject to a veto by the sending or receiving service, or may be trusted by the sending and receiving services. The sending service or another service responsive to the sending service 205 uses the security arrangements 203 to process the document 204 for transmission to the receiving service 209. In some circumstances, the security arrangements will call for obtaining an assertion from a trusted assertion service 206. For instance, the sending and receiving services may agree to use a SAML service to generate authentication assertions. The security arrangement 203 would call for generation of a SAML assertion and the sending service 205 would obtain a SAML assertion from a SAML server 206. In another embodiment, an electronic notarization might be provided by a trusted service 206. Banks or security authorities might be trusted to generate authentication assertions, in a function analogous to notarization. In some circumstances, the security arrangements will call for obtaining public keys used in asymmetrical

signing or encryption from a public keys source 208. For instance, the sending and receiving services may agree to use an XKMS service to exchange public keys. The security arrangement 203 would specify the XKMS service address as the source of public keys. The sending service 205 and the receiving service 209 both would access the agreed keys source 208. In accordance with the security arrangements 203 the sending service 205 communicates the document 204 through a network 207 to the receiving party 209. The routing and transport through the network 207 may be part of the security arrangements or, preferably, may be handled by a secure transport infrastructure. The security arrangements 203 may be provided by the computing service 202 to the receiving party 209 or otherwise made accessible to the receiving party, independent of the message carrying the document 204. Alternatively, the security arrangements 203 may be included with the document 204 according to a prearranged protocol. For instance, it may be part of the message header or it may be a separate part of the message. The prearranged protocol may call for the message header or message part to be signed and/or encrypted using the parties' respective keys. With this process flow and schemas above in mind, an example from the source code appendix may be explained.

**[0021]** The files SecuritySenderInfo.XML, SecurityReceiverInfo.XML, and ComputeSecurityContract.XML provide an example of sender and receiver preferences and a resulting computed security arrangement. The sender and receiver preferences are stated in XML code conforming to the XML schema explained above. The computed security arrangement is stated in an interoperability security contract document conforming to the SecurityContract.XSD schema in the source code appendix.

**[0022]** In this example, the sender preferences information includes community preferences and service preferences. The community preferences address security algorithms, preferences to sign the header, encrypt the credential and for selection among available credentials. The community preferences also may rank order the security algorithms or otherwise indicate preference among the security algorithms. A similar set of preferences might be provided for a collaboration partner, either instead of more in addition to preferences for a community. In this example, the community has six sets of signature algorithms options in elements named XMLSignatureAlgorithmTemplate and three sets of encryption algorithms options in elements named XMLEncryptionAlgorithmTemplate. These sets of options are templates. More than one template of options can be provided for a particular algorithm. Use of templates simplifies configuration of options and increases the likelihood consistent option sets will be selected by sending and receiving services. The community in this example prefers not to sign headers or encrypt credentials and accepts basic credentials. In general, a community or a

collaboration partner may have preferences for any security arrangement options that the service can select, or the community or collaboration partner may have preferences for only some options. Community preferences in a sender's preference file should correspond to community preferences stated elsewhere, such as in a registry entry for community preferences. The file CommunitySecurityTemplatesPreferences.XML is an example of a file used to record some or all of a community's security preferences.

**[0023]** The service (sending service in this example) records in SAMsgSecurityPolicy its preferences for handling message parts, for signature and encryption of the message as a whole, and for authentication. Messages may have several parts. Corresponding to a message part, a service may identify the message part and express a preference for signing or not signing or for encrypting or not encrypting a message part. In this embodiment, a preference for a category of algorithm, such as a general algorithm or in XML-specific algorithm can be selected. In other embodiments, the service might not specify a category of algorithm or it might specify a specific algorithm.

**[0024]** Other arrangements for security are also covered by this example. The receiver's (buyer's) public key, in an X509 format, is use for signature and authentication. Two resources, so-called connectors, are identified for the sending service to use for signing and encryption. The sender's available credentials are identified as basic and X509 credentials. The sending service's security arrangement preferences are rank ordered from one to three under SecurityPolicyTemplatePreference. In this example, the three encryption preferences are all for XML-specific encryption. These and other details of this example are found in the source code appendix file SecuritySenderInfo.XML.

**[0025]** Receiving party preferences are found in the source code appendix file SecurityReceiverInfo.XML. In general, the elements of the receiving party's preference profile are very similar to those of the sending party, even using the same element types from the schema. Significant differences are found in authentication and authorization, since the logic applicable to authentication and authorization depends on whether you are presenting your credentials or determining whether to accept what is presented. For instance, the SendingCPSecurityProfile of the sending party lists available credentials. This element is not part of the receiving party's preferences. This issue is addressed by the receiving party's CPRecvServicesSecurityPolicy, which identifies AcceptedCredentials.

**[0026]** In this example, two types of preferences are stated that the security arrangements logic reconciles. One type of preferences is among algorithm templates. The element SecurityPolicyTemplatePreference appears twice in each of the sending and receiving services'

preferences, setting forth community and service-specific preferences among algorithms. Figure 3 illustrates reconciling preferences among algorithm types. Stacks 301 and 302 represent sending and receiving preferences. Suppose A is the most secure and G the least secure. In the two preference stacks 301, 302, preference B and D match. A decision rule for choosing between B or D might take into account one or both stacks of preferences. For instance, the receiving service's preference (D) for signature or the sending service's preference (B) for encryption might be selected from among the matches. Taking both preferences into account, the most secure (B) or the least secure (D) might be selected. In another embodiment, the respective services might weight or score their preferences and a combined weighting or score may be used to take into account both preferences. The second type of preferences is for whether or not to sign or encrypt a part of a message. What to sign or encrypt is addressed by the SAMsgPart elements of SAMsgSecurityPolicy. The message parts in the example are Order and Image. In this example, sender and receiver preferences match, for signing and encrypting the Order and only encrypting the Image. Preferences would not match if the receiver wanted the Image signed, as well as the Order. Then, a decision rule would be needed to resolve the mismatch. The available decision rules could include: receiver wins, sender wins, highest requirement wins or lowest requirement wins. One type of preference reconciliation determines whether to apply a security measure. The other type selects among option templates, when the security measure is applied.

**[0027]** A set of computed security arrangements for this example appear in ComputeSecurityContract.XML, which is partially reproduced below:

```
<SecurityContractICD ... >
  <SecurityPolicies>
    <SignaturePolicies>
      <XMLDsigPolicy PolicyId="P-XMLSignatureRSA-MD5-C14N">
      <SignaturePolicyAlgorithm>...</SignaturePolicyAlgorithm>
        <SignatureAlg...>MD5withRSA</SignatureAlg...>
        <HashFunction>MD5</HashFunction>
        <Canonical ...>...14n-20001026</Canonical ...>
        <Transform>...#RoutingSignatureT...</Transform>
      </XMLDsigPolicy>
    </SignaturePolicies>
    <EncryptionPolicies>
```

```
<XMLEncryptionPolicy PolicyId="P-XMLEncrypt3DES-RSA-2048">
<EncryptionPolicyAlgorithm>http://www.w3.org/2001/04/xmlenc#</EncryptionPolicyAlgorithm>
<EncryptionMethod>http://www.w3.org/2001/04/xmlenc#3des-cbc</EncryptionMethod>
<KeySize>2048</KeySize>
<KeyEncryptionMethod>http://www.w3.org/2001/04/xmlenc#rsa-1_5</KeyEncryptionMethod>
</XMLEncryptionPolicy>
</EncryptionPolicies>
<EncryptionKeyInfo KeyOwner="x-ccns:commerceone.com:CollaborationParty::sellParty">
<PublicKeyID>DefaultTestCert</PublicKeyID>
<X509Data> <X509Certificate>LS0tLS1... ==</X509Certificate>
</X509Data>
</EncryptionKeyInfo>
</SecurityPolicies>
<SecurityChannel channelId="CHANNEL1" sourceConnector="x-ccns:cup.commerceone.com:connector::centerSell" targetConnector="x-ccns:cup.commerceone.com:connector::centerSell">
<Confidential AlgorithmId="P-XMLEncrypt3DES-RSA-2048">
<PublicKeyName KeyOwner="x-ccns:commerceone.com:CollaborationParty::sellParty">DefaultTestCert</PublicKeyName>
<MessagePart PartName="Order" isOptional="false"/>
<MessagePart PartName="Image" isOptional="false"/>
</Confidential>
</SecurityChannel>
<SecurityChannel channelId="CHANNEL2" sourceConnector="x-ccns:cup.commerceone.com:connector::buy" targetConnector="x-ccns:cup.commerceone.com:connector::sell">
<Integrity AlgorithmId="P-XMLSignatureRSA-MD5-C14N">
<PublicKeyName KeyOwner="OwnerA">BuyerPublicKey</PublicKeyName>
<MessagePart PartName="Order" isOptional="false"/>
</Integrity>
</SecurityChannel>
</SecurityContractICD>
```

This set of security arrangements has two major sections for security policy and security channels. In this example, there is one security policy applicable to the entire message and multiple security channels to implement parts of the security policy. The security policy section sets out the signature policy, and encryption policy and encryption key information. It also may set out policies regarding authentication, authorization and non-repudiation of origin or receipt. In this embodiment, the same signature and encryption policy is applied to all parts of the document. In other embodiments, multiple algorithms could be applied to different parts. The algorithm selected for signature, encryption and authentication are abstracted through templates containing options sets, simplifying the selection of algorithms. Selected algorithms are associated with logic and resources, so different services or processes can be used for signing/verifying and encrypting/decrypting different parts of a message. A public key or certificate can be transmitted in the encryption key element of the security policy section. The security channel section describes services or connectors involved in applying security policies. For a particular policy, the channel section identifies a source connector that requires assistance in applying a security policy (e.g., the sending service requesting encryption), and a target connector that applies the security policy or acts as an intermediary to logic and resources that apply the security policy. For a particular security policy, such as signing, encryption, authentication, authorization or non-repudiation, specific information required to carry out the security policy is provided in the security channel section.

**[0028]** The data used to determine security arrangements can be categorized as message and activity related data, CP-service related data, security algorithms related data, routing related data, encryption key related data and configuration data. Some additional detail regarding use of these categories of is described below. Message and activity related data relates to digital signatures, encryption, non-repudiation, and authorization. For non-repudiation, a receiver may require non-repudiation measures for a sender, amounting to a trusted party verification of the sender's message to receiver. Similarly, a sender may require non-repudiation measures for a receiver, amounting to a trusted party verification of receipt of sender's message by the receiver. Beyond the description above, it should be mentioned that signatures and encryption can be applied on an element basis, to particular items of data, if fine granularity is desired. In addition, overrides can be specified for pairs of sending and receiving services. For instance, a pre-existing or proven relationship can be treated differently than an entirely new relationship. Overrides to security policies can be implemented to cautiously reduce (or increase, as warranted) security requirements in particular cases.

**[0029]** CP-related data includes authentication and authorization data. Authorization is the process of granting or denying access to a network resource. Authorization to access most computer security systems is a two-step process. The first stage is authentication, which ensures that a principal (user, process, application or service) is who it claims to be. The second stage is authorization, which allows the principal access to various resources based on their identity. Authorization is also called access control. Access control is used to authorize access to website resources. It manages information about users, groups of users, and the roles assigned to users. SAML provides an XML-based means to share information about security events (authentication and authorization) and attributes (e.g. credit rating) in a SOAP message. This SAML data can then be sent to a third-party, and this enables 'distributed trust', whereby the user signs on once, but can re-use their authentication or authorization details. With SAML or a similar trusted party technology, the issuing authority decides whether to grant the request by subject services or sender, for access type to resource web service, given the evidence provided by the requestor. The authorization decision allows or denies a subject access to a specific resource. SAML is useful option for web services security, but it requires an initial degree of trust and technical resources. In instances when SAML is unavailable or not preferred, other approaches such as ID/password and a table of privileges associated with an ID can be used. The present invention is not limited by the authorization technology used, but extends more abstractly to selection among presently available or hereafter invented technologies. With either SAML authorization or ID/password technologies, the authorization data can be encrypted and built into the message.

**[0030]** Security algorithms related data includes algorithms and configuration options for signature, encryption and non-repudiation. As the schema illustrates, signature algorithms options (XML or non-XML) may include use of XMLDsig, choice of a Canonicalization algorithm, a signature method and a digest algorithm. Encryption / decryption options (XML or non-XML) may include key size, key and method. Default may be inherited by a service, either overriding the services preferences or being overridden. In addition, specific overrides can be specified for CP pairs, as described above. Option templates, also described above, simplify negotiation of security arrangements. Different options will apply to XML and non-XML algorithms, signature algorithms for example. XML signature algorithms, e.g., XMLDisg, my offer options for method, Canonicalization, transform and digest, while non-XML algorithms, e.g., PCKS#7, may have options for signature and digest methods, only. Use of community standard security templates are preferred, to ensure that there is at least one match between preference lists of the respective services. A community may require all CP's or all services

operating in the community to support a particular community standard security option set, to assure that messages can be exchanged within the community.

**[0031]** Routing related data includes how to access logic and resources that implement authentication / verification, signing / verification, and encryption / decryption. Any type of access information may be used, such as a universal resource name (URN) or universal resource locator (URL). As discussed in one of the prior applications referred to above, a message may take multiple hops through connectors for translation or other value-added services. Accordingly, multiple route steps may be associated with any action. Security typically will need to be reapplied after any translation or other value-added service.

**[0032]** Encryption key related data is generally discussed above.

**[0033]** Configuration data includes default (e.g., community or collaboration partner) preferences and credential preferences.

**[0034]** Figure 4 illustrates alternative embodiments for obtaining receiver's information when the sender is local to calculations of the security arrangements. In the figure, local 431 and remote 432 registries are indicated. In this example, the sender is local and the receiver remote. The sender's data is current and complete in the local registry 431. The sender's information is collected 421 and made available to the logic and resources that compute the security arrangements 411. The receiver's data may be current and complete, for instance if the receiver is in the same community as the sender and there is a community-wide registry, or if the receiver's information has been recently obtained and locally cached. Depending on where the receiver's information can be found, 431 or 432, a process 422 or 423 is invoked to collect the receiver information and make it available to the logic that computes security arrangements. A set of security arrangements 401 result.

**[0035]** Figure 5 illustrates one network of program logic and resources that can be used to implement aspects of the present invention. The logic components of this network include: send side collection 551, receive side collection 552, data object manager 541, routing manager 542, credential negotiator 531, template negotiator 532, connector manager 533, authentication manager 521, policy manager 522, public key manager 523, algorithm manager 524, policy builder 511, channel builder 512 and security arrangements document builder 501.

**[0036]** One embodiment of program logic operative in a community of collaboration partners to generate security arrangements can be described as follows: Collect the receiver security information, including an attribute assertion to authenticate the sender CP. Collect the sender security information. Look into routing block to find all connectors information to implement security measures. Get capability parameters for each connector. Walk through the

routing chain to find which connector-pair to use for authentication, signature, and encryption. Get the recever's service-activity-message object. This may include getting a SAMsgSecurityPolicy object from the receiver. This will have multiple parts and it can have signature and encryption policies for the whole message. It also may include getting a SAMsgSecurityPolicy object from the sender, and match the override options the SAMsgSecurityPolicy object accordingly. (Override decision tables are discussed below.) From the SAMsgSecurityPolicy object, find all algorithms required for this message, and build RequiredAlgorithmList. Get community preference objects for both SenderInfo and ReceiverInfo. This may include getting a CommunitySecurityTemplatesPreference object of the sender, which includes security algorithm templates, and community security policy preferences. It also may include getting a CommunitySecurityTemplatesPreference object of the receiver, if not the same community. If they are in the same community, it may be sufficient to set an object pointer. Get CP-Service objects for both sender and receiver services and get CP objects for corresponding communities. This may include building the CPSecurityPolicyPreference of the sender and receiver. Based on the sender and receiver preferences and the decision rules in a RequiredAlgorithmList, select from the preference lists, and build a RequiredTemplateObjectList. If the services' respective preference list do not match on any algorithm, community defaults may generate a match. Get a ServiceAuthentication object for the receiver service. This will have one or more authentication method specified, including accepted credentials and authentication mode. Match the credential from ServiceAuthentication object and available credentials from CPSecurityPolicyPreference of the sender. If there is more than one match, then get the one that matches CredentialPreference from CPSecurityPolicyPreference of the receiver, or from CommunitySecurityTemplatesPreference corresponding to the receiver. Get the value of SignMessageHeader and EncryptCredential from either the CPSecurityPolicyPreference of the receiver or from CommunitySecurityTemplatesPreference object of the receiver. If no value is specified in either place, set it to a default such as false or true. Using the available sender's credential selected by the receiver, the authentication mode specified in the ServiceAuthentication object for the Receiver the SignMessageHeader Boolean attribute, and the EncryptCredential to build the authentication algorithm. Base on the connector's PublicKeyCapability to get the proper key. This may include getting the encryption key of the sender, if an encryption is required, and getting signature key ID of the receiver, if a signature is required. Get the authentication key ID of the receiver, if an X509 authentication is required. Build the policy section of the security

arrangements. Find the connector for the channel section and build the channel section of the security arrangements.

**[0037]** Decision tables may be used to implement the type of preference reconciliation related to whether to sign or encrypt part of a message. Again, decisions could be biased to accept preference not to sign or to accept the receiver's preference, or just the opposite. Some decision tables that could be used to implement possible decision rules follow:

| | | **Sender Preference** | | |
|---|---|---|---|---|
| | | Signature Required | Signature Optional | No Signature |
| **Receiver Preference** | Signature Required | Sign | Sign | Error |
| | Signature Optional | Sign | Don't Sign | Don't Sign |
| | No Signature | Error | Don't Sign | Don't Sign |

| | | **Sender** | | |
|---|---|---|---|---|
| | | Encryption Required | Encryption Optional | No Encryption |
| **Receiver** | Encryption Required | Encrypt | Encrypt | Error |
| | Encryption Optional | Encrypt | Don't Encrypt | Don't Encrypt |
| | No Encryption | Error | Don't Encrypt | Don't Encrypt |

| | | **Sender** | | |
|---|---|---|---|---|
| | | Signature Required | Signature Optional | No Signature |
| **Receiver** | Signature Required | Sign | Sign | Sign |
| | Signature Optional | Sign | Don't Sign | Don't Sign |
| | No Signature | Don't Sign | Don't Sign | Don't Sign |

| | | **Sender** | | |
|---|---|---|---|---|
| | | Encryption Required | Encryption Optional | No Encryption |
| **Receiver** | Encryption Required | Encrypt | Encrypt | Encrypt |
| | Encryption Optional | Encrypt | Don't Encrypt | Don't Encrypt |
| | No Encryption | Don't Encrypt | Don't Encrypt | Don't Encrypt |

**[0038]** The present invention is readily extended to support signing and encryption at intermediate connectors along a path between a sender and receiver. It is useful to be able to

sign and encrypt documents at connectors along a routing path that are not the message originators or final receivers. This may be useful for gateways, routers and central connectors. For gateways, signing and encryption may need to be performed by a gateway if signed/encrypted message data is transformed from one envelope protocol to another. For routers and central connectors, it may be desirable to use a single entry/exit point into the enterprise for external communities. A router or central connector may act as the central security hub and perform or organize security operations on behalf of the entire enterprise. This may simplifies the PKI management and other administrative burdens. This functionality can be configured by setting up the security capabilities of connectors in the enterprise's part of a community. A connector can be configured on an envelope/transport protocol basis to have signing capability or encryption capability and can be linked to signing and encryption capabilities of the collaboration partner at other connectors. In the case of gateways and routers, you could configure the connector to use the key of the CP owner or the gateway/router connector.

**[0039]** From the preceding description, it will be apparent to those of skill in the art that a wide variety of systems and methods can be constructed from aspects and components of the present invention. One embodiment is a method of dynamically determining security options for exchange of one or more messages between sending and receiving services. This method uses sender and receiver security preferences, which may take the form of machine security profiles for first and second services. The security profiles may identify security options/elements and option subsets that are acceptable to the respective services. The options may include requirements to sign or encrypt one or more parts of the message, signing option subsets corresponding to one or more signing algorithms, encryption option subsets corresponding to one or more encryption algorithms, identification of signing and encryption keys and identification of an authentication algorithm. The dynamic method includes accessing the security profiles and selecting a particular option set that is acceptable to the respective services. Optionally, this option set can be used to communicate a message between the respective services. Several options and aspects of the present invention can be added to this embodiment. Security profiles can be maintained in one or more registries that are accessible to security logic of the first and second services. Default option subsets and/or preferences can be specified in community or collaboration partner security profiles and may be copied into service security profiles. Requirements to sign or encrypt can be applied to the parts of the message or to a message as a whole. Signature and encryption algorithms may be applied to a message as a whole, reducing complexity. Signing and encryption keys may be symmetrical or asymmetrical. Authentication

may be carried out by a trusted agent, such as a SAML server, before communicating the message between the respective services. Authentication by a trusted agent may be evidenced by authentication assertion. Alternatively, authentication may include submitting credentials for examination by the receiving service. These credentials may be part of the message or may be transmitted in addition to the message. In addition to authentication, authorization may be addressed by security arrangements. The security profiles may include identification of at least one authorization algorithm to establish a sending service's privileges. This authorization may be implemented by a trusted agent before communicating the message or by submitting credentials to the service receiving the message. A further aspect of the present invention is taken into account preferences of the respective services among option subsets for signing and/or encryption. Preferences of one or both of the services may be taken into account. Any of the decision rules discussed above may be applied, including receiver wins, sender wins, most secure wins, least secure wins or a weighted factoring of both services' preferences. Determination of security arrangements may include determining resources to be used by the respective parties to implement any combination of signatures, encryption, authentication, authorization or non-repudiation. Resources, algorithms and option says may be packaged into security channels. A security channel may implement a single aspect of security.

**[0040]** While the present invention is disclosed by reference to the preferred embodiments and examples detailed above, it is understood that these examples are intended in an illustrative rather than in a limiting sense. Computer-assisted processing is implicated in the described embodiments. Accordingly, the present invention may be embodied in methods for computer-assisted processing, systems including logic to implement the methods, media impressed with logic to carry out the methods, data streams impressed with logic to carry out the methods, or computer-accessible processing services. It is contemplated that modifications and combinations will readily occur to those skilled in the art, which modifications and combinations will be within the spirit of the invention and the scope of the following claims.

**[0041]** We claim as appears after the appendix.

# COMPUTER PROGRAM LISTING APPENDIX

## SecuritySenderReceiverInfo.XSD

```
<?xml version="1.0" encoding="UTF-8" ?>
- <!--
edited with XML Spy v4.4 U (http://www.xmlspy.com) by Symon Chang (Commerce One)
  -->
- <xs:schema
    targetNamespace="publicid:com.commerceone:schemas/contract/helperi
    nfo/v1_0/SecuritySenderReceiverInfo.xsd"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:sicd="publicid:com.commerceone:schemas/soapextension/contrac
    t/security/v1_0/SecurityContract.xsd"
    xmlns:sicdr="publicid:com.commerceone:schemas/contract/helperinfo/v
    1_0/SecuritySenderReceiverInfo.xsd"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
    xmlns="publicid:com.commerceone:schemas/contract/helperinfo/v1_0/
    SecuritySenderReceiverInfo.xsd" elementFormDefault="qualified"
    attributeFormDefault="unqualified">
    <xs:import
      namespace="publicid:com.commerceone:schemas/soapextension/co
      ntract/security/v1_0/SecurityContract.xsd"
      schemaLocation="http://schemas.commerceone.com/schemas/soape
      xtension/contract/security/v1_0/SecurityContract.xsd" />
    <xs:import namespace="urn:oasis:names:tc:SAML:1.0:assertion"
      schemaLocation="http://www.oasis-
      open.org/committees/security/docs/cs-sstc-schema-assertion-
      01.xsd" />
  - <!--
  Sender Security ICD Infomation Block
    -->
  - <xs:element name="SecuritySenderInfo"
      type="SecuritySenderInfoType">
    - <xs:annotation>
        <xs:documentation>The root for all ICD security policyinfo from
          the sender.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <!--
  Receiver Security ICD Infomation Block
    -->
  - <xs:element name="SecurityReceiverInfo"
      type="SecurityReceiverInfoType">
    - <xs:annotation>
        <xs:documentation>The root for all ICD security policy info from
          the Recevier.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <!--
  Main Elements
    -->
  - <xs:element name="CommunitySecurityTemplatesPreference">
    - <xs:annotation>
```

```
<xs:documentation>Security Policy for this community,
  including Security Algorithm Templates, Community
  Ssecurity Policy Preferences and Community Security Policy
  Preference.</xs:documentation>
</xs:annotation>
- <xs:complexType>
  - <xs:sequence minOccurs="0">
      <xs:element ref="sicdr:SecurityAlgorithmTemplates"
        minOccurs="0" />
    - <xs:element name="CommunitySecurityPolicyPreference"
        type="sicdr:ConfiguredPreferencePolicyType"
        minOccurs="0">
      - <xs:annotation>
          <xs:documentation>The preference will be sign
            header, encrypt credential, and credential
            preference, etc. It will be the default value for
            the whole community.</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element ref="sicdr:SecurityPolicyTemplatePreference"
        minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>
</xs:element>
- <xs:element name="SendingCPSecurityPolicyProfile">
  - <xs:annotation>
      <xs:documentation>The Security Profile for the sending
        Collaboration Party. It has CP's Available Credentials
        info.</xs:documentation>
    </xs:annotation>
  - <xs:complexType>
    - <xs:sequence minOccurs="0">
        <xs:element name="AvailableCredentials"
          type="sicdr:CredentialTypes" maxOccurs="unbounded" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
- <xs:element name="SAMsgSecurityPolicy"
    type="sicdr:SAMsgPartsType">
  - <xs:annotation>
      <xs:documentation>Each Server/Active/Message have multiple
        parts and it can have signature and encryption policies for
        the whole message. The authentication is defined at the
        service level.</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="PublicKeys" type="sicd:PublicKeyType">
  - <xs:annotation>
      <xs:documentation>Public key records for this CP. The KeyID
        will be the unique key for the public key records. The
        location will match all connectors within this
        request.</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="ConnectorCapability">
```

```
- <xs:annotation>
    <xs:documentation>Connector Capability
      Parameters</xs:documentation>
  </xs:annotation>
- <xs:complexType>
  - <xs:sequence>
    - <xs:element name="EncryptionCapability"
        type="xs:boolean">
      - <xs:annotation>
          <xs:documentation>Yes or No flag. The document
            encryption/decryption can be performed at this
            connector or not.</xs:documentation>
        </xs:annotation>
      </xs:element>
    - <xs:element name="SignatureCapability"
        type="xs:boolean">
      - <xs:annotation>
          <xs:documentation>Yes or No flag. The signing the
            message or verify the signature can be
            performed at this connector or
            not.</xs:documentation>
        </xs:annotation>
      </xs:element>
    - <xs:element name="EncryptionPublicKeyParty"
        type="sicd:CollaberationPartyID" minOccurs="0">
      - <xs:annotation>
          <xs:documentation>The Public Key party that is
            used for encryption. This can be either sender's
            CP or the owner of the connector. If it is not
            defined, then the key of message receiver at
            this connector location will be used for
            encryption.</xs:documentation>
        </xs:annotation>
      </xs:element>
    - <xs:element name="SigningPublicKeyParty"
        type="sicd:CollaberationPartyID" minOccurs="0">
      - <xs:annotation>
          <xs:documentation>The Public Key party that is
            used for signing. This can be any CP or the
            owner of connector. If it is not defined, then
            the Key of message sender at this location will
            be used for signature.</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="ConnectorName" type="xs:string"
      use="optional" />
  </xs:complexType>
</xs:element>
- <!--
Main Complex Types
-->
- <xs:complexType name="SendingServicesSecurityPolicyType">
  - <xs:annotation>
```

```
        <xs:documentation>Servcies security policy for each CP. This
          can be the policy for either sending or receiving
          service.</xs:documentation>
      </xs:annotation>
    - <xs:sequence minOccurs="0">
        <xs:element ref="sicdr:SecurityPolicyTemplatePreference"
          minOccurs="0" maxOccurs="unbounded" />
      - <xs:element ref="ServiceAuthentication" minOccurs="0">
        - <xs:annotation>
            <xs:documentation>Authentication method for a given
              service, including Accepted Credentials and
              Authentication Mode.</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  - <xs:complexType name="ReceivingServicesSecurityPolicyType">
    - <xs:annotation>
        <xs:documentation>Receiving Services security policy for each
          CP</xs:documentation>
      </xs:annotation>
    - <xs:complexContent>
      - <xs:extension base="sicdr:SendingServicesSecurityPolicyType">
        - <xs:sequence minOccurs="0">
          - <xs:element ref="sicd:Authorization" minOccurs="0">
            - <xs:annotation>
                <xs:documentation>SAML Attribute Assertion
                  for the end connector to use. This will be a
                  data type from SMAL
                  Standard.</xs:documentation>
              </xs:annotation>
            </xs:element>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  - <!--
    Simple Types
    -->
  - <xs:simpleType name="AuthenticateCapabilityTypes">
    - <xs:annotation>
        <xs:documentation>The Authenticate Capability for the
          connector.</xs:documentation>
      </xs:annotation>
    - <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="LOCAL" />
        <xs:enumeration value="REMOTE" />
        <xs:enumeration value="BOTH" />
        <xs:enumeration value="NONE" />
      </xs:restriction>
    </xs:simpleType>
  - <xs:simpleType name="OverrideTypes">
    - <xs:annotation>
```

```
            <xs:documentation>Type of the override rules. This is used for
              matching.</xs:documentation>
          </xs:annotation>
        - <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="Required" />
            <xs:enumeration value="Optional" />
            <xs:enumeration value="NotRequired" />
          </xs:restriction>
        </xs:simpleType>
      - <xs:simpleType name="NegotiationRuleTypes">
        - <xs:annotation>
            <xs:documentation>Type of the Negotiation Rules: when there
              are multiple matches during the algorithm negotiation, the
              rule will determine which algorithm will be picked. It can be
              receiver wins, sender wins, highest requirement wins or
              lowest requirement wins.</xs:documentation>
          </xs:annotation>
        - <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="ReceiverWins" />
            <xs:enumeration value="SenderWins" />
            <xs:enumeration value="MoreSecurityWins" />
            <xs:enumeration value="LessSecurityWins" />
          </xs:restriction>
        </xs:simpleType>
      - <xs:simpleType name="CategoryTypes">
        - <xs:annotation>
            <xs:documentation>Type of the algorithm
              category</xs:documentation>
          </xs:annotation>
        - <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="XMLSignature" />
            <xs:enumeration value="Signature" />
            <xs:enumeration value="XMLEncryption" />
            <xs:enumeration value="Encryption" />
            <xs:enumeration value="NonRepudiation" />
            <xs:enumeration value="NonRepudiationReceipt" />
          </xs:restriction>
        </xs:simpleType>
      - <xs:simpleType name="CredentialTypes">
        - <xs:annotation>
            <xs:documentation>Type of the credential
              algorithm</xs:documentation>
          </xs:annotation>
        - <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="BASIC" />
            <xs:enumeration value="X509" />
            <xs:enumeration value="BASE64_BINARY" />
            <xs:enumeration value="ANONYMOUS" />
            <xs:enumeration value="NONE" />
          </xs:restriction>
        </xs:simpleType>
      - <!--
        Elements and Complex types
        -->
```

```
- <xs:element name="ServiceAuthentication">
  - <xs:annotation>
      <xs:documentation>Authentication method for a given service,
        including Accepted Credentials and Authentication
        Mode.</xs:documentation>
    </xs:annotation>
  - <xs:complexType>
    - <xs:sequence>
      - <xs:element name="AcceptedCredentials"
          type="sicdr:CredentialTypes" maxOccurs="5">
        - <xs:annotation>
            <xs:documentation>Multiple credentiails can be
              accpeted for a given
              service.</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element ref="sicd:AuthenticateMode" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
- <xs:complexType name="XMLSignatureAlgorithmTemplateType">
  - <xs:annotation>
      <xs:documentation>Define XMLDsig type of policy and
        algorithms</xs:documentation>
    </xs:annotation>
  - <xs:complexContent>
    - <xs:extension
        base="sicdr:Abstract_SecurityAlgorithmTemplateType">
      - <xs:sequence>
          <xs:element ref="sicd:XMLDsigPolicy" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
- <xs:element name="XMLSignatureAlgorithmTemplate"
    type="sicdr:XMLSignatureAlgorithmTemplateType">
  - <xs:annotation>
      <xs:documentation>This is for XML only
        signature.</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="TemplateDescription" type="xs:string">
  - <xs:annotation>
      <xs:documentation>This element is not used. It is a placeholder
        to circumvent a Castor bug.</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:complexType name="XMLEncryptionAlgorithmTemplateType">
  - <xs:annotation>
      <xs:documentation>Define XMLEnc type of policy and
        algorithms</xs:documentation>
    </xs:annotation>
  - <xs:complexContent>
```

```
      - <xs:extension
           base="sicdr:Abstract_SecurityAlgorithmTemplateType">
         - <xs:sequence>
             <xs:element ref="sicd:XMLEncryptionPolicy" />
           </xs:sequence>
         </xs:extension>
       </xs:complexContent>
     </xs:complexType>
   - <xs:element name="XMLEncryptionAlgorithmTemplate"
       type="sicdr:XMLEncryptionAlgorithmTemplateType">
     - <xs:annotation>
         <xs:documentation>This is for XML only
           encryption.</xs:documentation>
       </xs:annotation>
     </xs:element>
   - <xs:complexType name="Abstract_SecurityAlgorithmTemplateType"
       abstract="true">
     - <xs:annotation>
         <xs:documentation>The template will be used by the SIC
           Builder during run-time algorithm
           matching.</xs:documentation>
       </xs:annotation>
     - <xs:sequence minOccurs="0">
       - <xs:element name="Category" type="sicdr:CategoryTypes"
           minOccurs="0">
         - <xs:annotation>
             <xs:documentation>The category of security algorithm
               template.</xs:documentation>
           </xs:annotation>
         </xs:element>
       </xs:sequence>
       <xs:attribute name="Name" type="xs:string" use="optional" />
       <xs:attribute name="ID" type="xs:string" use="optional" />
     </xs:complexType>
   - <xs:complexType name="SecurityAlgorithmPreferenceType"
       abstract="false">
     - <xs:annotation>
         <xs:documentation>The preference of each security algorithm
           policy.</xs:documentation>
       </xs:annotation>
     - <xs:complexContent>
       - <xs:extension
           base="sicdr:Abstract_SecurityAlgorithmTemplateType">
         - <xs:sequence>
             <xs:element name="Preference" type="xs:short" />
           </xs:sequence>
         </xs:extension>
       </xs:complexContent>
     </xs:complexType>
   - <xs:element name="SecurityAlgorithmTemplates">
     - <xs:annotation>
         <xs:documentation>Each Community will have a set of
           recommended Security Algorithm for CP to select
           from.</xs:documentation>
```

```
    </xs:annotation>
  - <xs:complexType>
    - <xs:sequence>
        <xs:element ref="sicdr:XMLSignatureAlgorithmTemplate"
          maxOccurs="unbounded" />
        <xs:element ref="sicdr:XMLEncryptionAlgorithmTemplate"
          maxOccurs="unbounded" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
- <xs:complexType name="ConfiguredPreferencePolicyType">
  - <xs:annotation>
      <xs:documentation>Define some configurable policy preference.
        This can be either at whole community level or at the CP
        level.</xs:documentation>
    </xs:annotation>
  - <xs:sequence>
    - <xs:element name="SignMessageHeader" type="xs:boolean"
        minOccurs="0">
      - <xs:annotation>
          <xs:documentation>If set then the Header and ICD
            Block will be signed.</xs:documentation>
        </xs:annotation>
      </xs:element>
    - <xs:element name="EncryptCredential" type="xs:boolean"
        minOccurs="0">
      - <xs:annotation>
          <xs:documentation>If set then the credential header
            will be encrypted. This only apply to the non-SAML
            type of of credential, where Authentication mode is
            TARGET.</xs:documentation>
        </xs:annotation>
      </xs:element>
    - <xs:element name="CredentialPreference"
        type="sicdr:CredentialTypes" minOccurs="0">
      - <xs:annotation>
          <xs:documentation>Select one from BASIC and X509. It
            will be used, when there are multiple credential
            matched condition. SICB will pick the one matches
            to this field first.</xs:documentation>
        </xs:annotation>
      </xs:element>
    - <xs:element name="NegotiationRule"
        type="NegotiationRuleTypes" minOccurs="0">
      - <xs:annotation>
          <xs:documentation>When there are multiple matches
            during the algorithm negotiation, the rule will
            determine which algorithm will be picked. It can be
            receiver wins, sender wins, highest requirement
            wins or lowest requirement
            wins.</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
```

```
- <xs:element name="SecurityPolicyTemplatePreference"
    type="sicdr:SecurityAlgorithmPreferenceType">
  - <xs:annotation>
      <xs:documentation>The preference will be signature, XML
        signature, encryption, XML encryption, etc. It can have any
        number of preferences in each category.</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:complexType name="SAMsgPartElementType">
  - <xs:annotation>
      <xs:documentation>The element within the part from
        Server/Activity/Message.</xs:documentation>
    </xs:annotation>
  - <xs:simpleContent>
    - <xs:extension base="xs:string">
        <xs:attribute name="Signature" type="xs:boolean"
          use="optional" />
        <xs:attribute name="SignatureType" type="xs:anyURI"
          use="optional" />
        <xs:attribute name="Encryption" type="xs:boolean"
          use="optional" />
        <xs:attribute name="EncryptionType" type="xs:anyURI"
          use="optional" />
        <xs:attribute name="BlockId" type="xs:short"
          use="optional" />
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
- <xs:complexType name="SAMsgPartType">
  - <xs:annotation>
      <xs:documentation>The part within a
        message.</xs:documentation>
    </xs:annotation>
  - <xs:sequence minOccurs="0">
    - <xs:element name="SAMsgPartElement" minOccurs="0"
        maxOccurs="unbounded">
      - <xs:annotation>
          <xs:documentation>This is for each
            CP/Service/Activity/Message. The element is
            defined by using XPath. If an element within the
            part is defined, then the attributes of the element
            will be used to determinewhether the element will
            be signed or encrypted.</xs:documentation>
        </xs:annotation>
      - <xs:complexType>
        - <xs:simpleContent>
            <xs:extension base="sicdr:SAMsgPartElementType"
              />
          </xs:simpleContent>
        </xs:complexType>
      </xs:element>
    - <xs:element name="PartSignatureAlgCategory"
        type="sicdr:SAMsgPartSignatureAlgorithmType"
        minOccurs="0">
      - <xs:annotation>
```

```
            <xs:documentation>If the signature algorithm is
              defined, then the whole part will be
              signed.</xs:documentation>
          </xs:annotation>
        </xs:element>
      - <xs:element name="PartEncryptionAlgCategory"
          type="sicdr:SAMsgPartEncryptionAlgorithmType"
          minOccurs="0">
        - <xs:annotation>
            <xs:documentation>If the encryption algorithm is
              defined, then the whole part will be
              encrypted.</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="PartName" type="xs:string" use="required" />
      <xs:attribute name="SignatureType" type="xs:anyURI"
        use="optional" />
      <xs:attribute name="EncryptionType" type="xs:anyURI"
        use="optional" />
      <xs:attribute name="BlockId" type="xs:short" use="optional" />
      <xs:attribute name="isOptional" type="xs:boolean" use="optional"
        default="false" />
    </xs:complexType>
  - <xs:complexType name="SAMsgPartsType">
    - <xs:annotation>
        <xs:documentation>The root for parts in a message for each
          CP/Service/Activity.</xs:documentation>
      </xs:annotation>
    - <xs:sequence>
      - <xs:element name="SAMsgPart" type="sicdr:SAMsgPartType"
          minOccurs="0" maxOccurs="unbounded">
        - <xs:annotation>
            <xs:documentation>This is for each
              CP/Service/Activity. Each message part has
              multiple elements and it can have signature or
              encryption policies for the message
              part.</xs:documentation>
          </xs:annotation>
        </xs:element>
      - <xs:element name="SAMsgSignatureAlgCategory"
          type="sicdr:SAMsgSignatureAlgorithmType" minOccurs="0">
        - <xs:annotation>
            <xs:documentation>If the signature policy is defined,
              then the whole message will be
              signed.</xs:documentation>
          </xs:annotation>
        </xs:element>
      - <xs:element name="SAMsgEncryptionAlgCategory"
          type="sicdr:SAMsgEncryptionAlgorithmType" minOccurs="0">
        - <xs:annotation>
            <xs:documentation>If the encryption policy is defined,
              then the whole message will be
              encrypted.</xs:documentation>
          </xs:annotation>
```

```
        </xs:element>
      </xs:sequence>
      <xs:attribute name="MessageName" type="xs:string" use="optional"
        />
    </xs:complexType>
  - <xs:complexType name="Abstract_CPMessageSecurityAlgorithmType"
      abstract="true">
    - <xs:annotation>
        <xs:documentation>This type will have Encryption or Signature
          Algorithms.</xs:documentation>
      </xs:annotation>
      <xs:attribute name="Override" type="sicdr:OverrideTypes"
        use="optional" />
    </xs:complexType>
  - <xs:complexType name="SAMsgPartEncryptionAlgorithmType">
    - <xs:annotation>
        <xs:documentation>Define the encryption policy for each part
          within a message per
          CP/Service/Activity</xs:documentation>
      </xs:annotation>
    - <xs:complexContent>
      - <xs:extension
          base="sicdr:Abstract_CPMessageSecurityAlgorithmType">
        - <xs:sequence>
          - <xs:choice>
              <xs:element name="XMLEncryptionAlgCategory"
                type="sicdr:CategoryTypes"
                fixed="XMLEncryption" minOccurs="0" />
              <xs:element name="EncryptionAlgCategory"
                type="sicdr:CategoryTypes" fixed="Encryption"
                minOccurs="0" />
            </xs:choice>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  - <xs:complexType name="SAMsgEncryptionAlgorithmType">
    - <xs:annotation>
        <xs:documentation>Define the category of the encryption policy
          for the whole message per CP/Service/Activity. In this case,
          both XML and Non-XML must be defined.</xs:documentation>
      </xs:annotation>
    - <xs:complexContent>
      - <xs:extension
          base="sicdr:Abstract_CPMessageSecurityAlgorithmType">
        - <xs:sequence>
            <xs:element name="XMLEncryptionAlgCategory"
              type="sicdr:CategoryTypes" fixed="XMLEncryption"
              />
            <xs:element name="EncryptionAlgCategory"
              type="sicdr:CategoryTypes" fixed="Encryption"
              minOccurs="0" />
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
```

```
    </xs:complexType>
  - <xs:complexType name="SAMsgPartSignatureAlgorithmType">
    - <xs:annotation>
        <xs:documentation>Define the signature policy for each part
          within a message per
          CP/Service/Activity</xs:documentation>
      </xs:annotation>
    - <xs:complexContent>
      - <xs:extension
          base="sicdr:Abstract_CPMessageSecurityAlgorithmType">
        - <xs:sequence>
          - <xs:choice>
              <xs:element name="SignatureAlgCategory"
                type="sicdr:CategoryTypes" fixed="Signature"
                minOccurs="0" />
              <xs:element name="XMLSignatureAlgCategory"
                type="sicdr:CategoryTypes"
                fixed="XMLSignature" minOccurs="0" />
            </xs:choice>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  - <xs:complexType name="SAMsgSignatureAlgorithmType">
    - <xs:annotation>
        <xs:documentation>Define the category of the signature policy
          for the whole message per CP/Service/Activity. In this case,
          only the XML Signature algorithm will be
          defined.</xs:documentation>
      </xs:annotation>
    - <xs:complexContent>
      - <xs:extension
          base="sicdr:Abstract_CPMessageSecurityAlgorithmType">
        - <xs:sequence>
            <xs:element name="XMLSignatureAlgCategory"
              type="sicdr:CategoryTypes" fixed="XMLSignature" />
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  - <xs:element name="CPSendServicesSecurityPolicy">
    - <xs:annotation>
        <xs:documentation>For the CP/Sending Service will have a set
          of security policy. The policy is defined per Sending CP's
          preferences. This will be used for
          override.</xs:documentation>
      </xs:annotation>
    - <xs:complexType>
      - <xs:complexContent>
        - <xs:extension
            base="sicdr:SendingServicesSecurityPolicyType">
          - <xs:attribute name="AuthenticateParty"
              type="sicd:CollaberationPartyID" use="optional">
            - <xs:annotation>
```

```
                    <xs:documentation>Host service delegate
                      party ID or the Sender's party
                      ID</xs:documentation>
                  </xs:annotation>
                </xs:attribute>
                <xs:attribute name="DelegateFlag" type="xs:boolean"
                  use="optional" default="false" />
              </xs:extension>
            </xs:complexContent>
          </xs:complexType>
        </xs:element>
      - <xs:complexType name="SecuritySenderInfoType">
        - <xs:complexContent>
          - <xs:extension base="SecurityInfoType">
            - <xs:sequence minOccurs="0">
                <xs:element ref="sicdr:SendingCPSecurityPolicyProfile"
                  minOccurs="0" />
              - <xs:element ref="CPSendServicesSecurityPolicy"
                  minOccurs="0">
                - <xs:annotation>
                    <xs:documentation>For the CP/Sending
                      Service will have a set of security policy.
                      The policy is defined perSending CP's
                      preferences. This will be used for
                      override.</xs:documentation>
                  </xs:annotation>
                </xs:element>
              </xs:sequence>
            </xs:extension>
          </xs:complexContent>
        </xs:complexType>
      - <xs:element name="CPRecvServicesSecurityPolicy"
          type="sicdr:ReceivingServicesSecurityPolicyType">
        - <xs:annotation>
            <xs:documentation>For each CP, every Receiving Service will
              have a set of security policy. The policy is defined per
              Receiving CP's requirements.</xs:documentation>
          </xs:annotation>
        </xs:element>
      - <xs:complexType name="SecurityReceiverInfoType">
        - <xs:complexContent>
          - <xs:extension base="sicdr:SecurityInfoType">
            - <xs:sequence minOccurs="0">
              - <xs:element ref="sicdr:CPRecvServicesSecurityPolicy"
                  minOccurs="0">
                - <xs:annotation>
                    <xs:documentation>For each CP, every
                      Receiving Service will have a set of
                      security policy. The policy is defined per
                      Receiving CP's
                      requirements.</xs:documentation>
                  </xs:annotation>
                </xs:element>
              </xs:sequence>
```

```
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  - <xs:complexType name="SecurityInfoType">
    - <xs:sequence minOccurs="0">
      - <xs:element
          ref="sicdr:CommunitySecurityTemplatesPreference"
          minOccurs="0">
        - <xs:annotation>
            <xs:documentation>Security Policy for this community,
              including Security Algorithm Templates, default
              security Policy Templates and Community Security
              Policy Preference.</xs:documentation>
          </xs:annotation>
        </xs:element>
      - <xs:element ref="sicdr:SAMsgSecurityPolicy" minOccurs="0">
        - <xs:annotation>
            <xs:documentation>Each Server/Active/Message have
              multiple parts and it can have signature and
              encryption policies for the whole message. The
              authentication is defined at the service
              level.</xs:documentation>
          </xs:annotation>
        </xs:element>
      - <xs:element ref="sicdr:PublicKeys" minOccurs="0"
          maxOccurs="unbounded">
        - <xs:annotation>
            <xs:documentation>Public key records for this CP. The
              KeyID will be the unique key for the public key
              records. The location will match all connectors
              within this request.</xs:documentation>
          </xs:annotation>
        </xs:element>
      - <xs:element ref="sicdr:ConnectorCapability" minOccurs="0"
          maxOccurs="unbounded">
        - <xs:annotation>
            <xs:documentation>Connector Capability
              Parameters</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element ref="sicd:SecurityContainer" minOccurs="0"
          maxOccurs="unbounded" />
      </xs:sequence>
      <xs:attribute name="passcode" type="xs:base64Binary"
        use="optional" />
    </xs:complexType>
  </xs:schema>
```

# SecurityContractKeyInfo.XSD

```
<?xml version="1.0" encoding="UTF-8" ?>
- <!--
edited with XML Spy v4.4 U (http://www.xmlspy.com) by Symon Chang (Commerce One)
-->
- <xs:schema
    targetNamespace="publicid:com.commerceone:schemas/soapextension/contract/security/v1_0/SecurityContract.xsd"
    xmlns:sicd="publicid:com.commerceone:schemas/soapextension/contract/security/v1_0/SecurityContract.xsd"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified" attributeFormDefault="unqualified"
    version="1.0">
  - <xs:simpleType name="CollaborationPartyID">
    - <xs:annotation>
        <xs:documentation>This is the Collaboration Partner's
          ID</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string" />
    </xs:simpleType>
  - <xs:simpleType name="KeyUsageTypes">
    - <xs:annotation>
        <xs:documentation>Key is used for signature, encryption,
          and/or authentication.</xs:documentation>
      </xs:annotation>
    - <xs:restriction base="xs:NMTOKENS">
        <xs:enumeration value="AUTHENTICATION" />
        <xs:enumeration value="ENCRYPTION" />
        <xs:enumeration value="SIGNATURE" />
        <xs:enumeration value="SSL" />
      </xs:restriction>
    </xs:simpleType>
  - <xs:simpleType name="KeyAlgorithmTypes">
    - <xs:annotation>
        <xs:documentation>Key is RSA or DSA type of
          key.</xs:documentation>
      </xs:annotation>
    - <xs:restriction base="xs:NMTOKENS">
        <xs:enumeration value="RSA" />
        <xs:enumeration value="DSA" />
      </xs:restriction>
    </xs:simpleType>
  - <xs:simpleType name="AuthenticateModeTypes">
    - <xs:annotation>
        <xs:documentation>The location of where the authentication
          takes place. NONE means neither source nor target
          connector will perform the authentication. This may be the
          case of letting foreign connector to perform the
          authentication.</xs:documentation>
      </xs:annotation>
    - <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="SOURCE" />
        <xs:enumeration value="TARGET" />
```

```
        <xs:enumeration value="NONE" />
      </xs:restriction>
    </xs:simpleType>
  - <xs:element name="PublicKey" type="sicd:PublicKeyType">
    - <xs:annotation>
        <xs:documentation>The Public Key record. Each public key will
          have partyID, KeyInfo, description and
          usages.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="EncryptionKeyInfo">
    - <xs:annotation>
        <xs:documentation>The KeyInfo that has both PublicKeyID and
          X509Data for encryption.</xs:documentation>
      </xs:annotation>
    - <xs:complexType>
      - <xs:complexContent>
        - <xs:extension base="sicd:KeyInfoType">
            <xs:attribute name="KeyOwner"
              type="sicd:CollaberationPartyID" use="optional" />
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
    </xs:element>
  - <xs:complexType name="PublicKeyType">
    - <xs:annotation>
        <xs:documentation>The Public Key record, including PartyID,
          KeyInfo, Usages and Description.</xs:documentation>
      </xs:annotation>
    - <xs:sequence>
        <xs:element ref="sicd:PartyID" />
      - <xs:element ref="sicd:EncryptionKeyInfo">
        - <xs:annotation>
            <xs:documentation>The KeyInfo block that has KeyID
              and X509 Data.</xs:documentation>
          </xs:annotation>
        </xs:element>
      - <xs:element ref="sicd:KeyTypeUsage" maxOccurs="4">
        - <xs:annotation>
            <xs:documentation>Key is used for signature,
              encryption, and/or
              authentication.</xs:documentation>
          </xs:annotation>
        </xs:element>
      - <xs:element name="KeyAlgorithm"
          type="sicd:KeyAlgorithmTypes" minOccurs="0">
        - <xs:annotation>
            <xs:documentation>The Key is RSA or DSA
              key</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element ref="sicd:Description" minOccurs="0" />
      - <xs:element name="Location" type="xs:string" minOccurs="0">
        - <xs:annotation>
```

```
            <xs:documentation>The connector ID that key the
              Private Key.</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  - <xs:element name="PartyID" type="sicd:CollaberationPartyID">
    - <xs:annotation>
        <xs:documentation>Trading partner ID or Collaboration Partner
          ID in UUID format.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="Description" type="xs:string">
    - <xs:annotation>
        <xs:documentation>The description of the
          key</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="KeyTypeUsage" type="sicd:KeyUsageTypes">
    - <xs:annotation>
        <xs:documentation>Key is used for signature, encryption,
          and/or authentication.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="KeyInfo">
    - <xs:annotation>
        <xs:documentation>The KeyInfo object is from the XMLDsig
          ds:KeyInfo object. However, within SICD we only use Public
          Key ID field.</xs:documentation>
      </xs:annotation>
    - <xs:complexType>
      - <xs:sequence>
          <xs:element ref="sicd:PublicKeyID" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  - <xs:element name="PublicKeyID" type="xs:string">
    - <xs:annotation>
        <xs:documentation>The Public Key ID is a unique key ID (UUID
          or from XMKS server).</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="PublicKeyName" type="sicd:PublicKeyNameType">
    - <xs:annotation>
        <xs:documentation>The Name of the Public Key. It is same as
          the PublicKeyID but has owner name as the optional
          attribute.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:complexType name="PublicKeyNameType">
    - <xs:simpleContent>
      - <xs:extension base="xs:string">
          <xs:attribute name="KeyOwner"
            type="sicd:CollaberationPartyID" use="optional" />
```

```
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  - <xs:complexType name="KeyInfoType">
    - <xs:annotation>
        <xs:documentation>This is for Encryption. The KeyInfo object is
          from the XMLDsig ds:KeyInfo object. However, within SICD
          we only use Public Key ID and X509 Certificate two
          fields.</xs:documentation>
      </xs:annotation>
    - <xs:sequence>
        <xs:element ref="sicd:PublicKeyID" />
      - <xs:element name="X509Data" minOccurs="0">
        - <xs:complexType>
          - <xs:sequence>
              <xs:element name="X509Certificate"
                type="xs:base64Binary" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  - <!--
   Policy Types
    -->
  - <xs:complexType name="Abstract_PolicyType" abstract="true">
    - <xs:annotation>
        <xs:documentation>This is the abstract policy for all security
          policy related algorithm. The ID is the Template Name for
          the Algorithm.</xs:documentation>
      </xs:annotation>
      <xs:attribute name="PolicyId" type="xs:string" use="optional" />
    </xs:complexType>
  - <xs:complexType name="Abstract_CredentialPolicyType"
      abstract="true">
    - <xs:annotation>
        <xs:documentation>This is the abstract policy for authentication
          credential policy algorithm.</xs:documentation>
      </xs:annotation>
    - <xs:complexContent>
      - <xs:extension base="sicd:Abstract_PolicyType">
        - <xs:sequence>
            <xs:element name="CredentialPolicyAlgorithm"
              type="xs:string" />
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  - <xs:element name="AuthenticateImplementation" type="xs:string">
    - <xs:annotation>
        <xs:documentation>Optional for different implementation, such
          as SAML, SecureID, or Kerberos.</xs:documentation>
      </xs:annotation>
    </xs:element>
```

```
- <xs:element name="AuthenticateMode"
    type="sicd:AuthenticateModeTypes">
  - <xs:annotation>
      <xs:documentation>The location of where the authentication
        takes place. It can be either SOURCE connector or TARGET
        connector. SOURCE means the sender's local connectors will
        perform SAML Single Sign-On type of authentication.
        TARGET means the connector on the receiving end will
        perform the authentication. NONE means neither source nor
        target connector will perform the authentication. This may
        be the case of letting foreign connector to perform the
        authentication.</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:complexType name="AuthenticationCredentialPolicyType">
  - <xs:annotation>
      <xs:documentation>This authentication and credential policy
        will work for Basic and X509.</xs:documentation>
    </xs:annotation>
  - <xs:complexContent>
    - <xs:extension base="sicd:Abstract_CredentialPolicyType">
      - <xs:sequence minOccurs="0">
          <xs:element ref="sicd:AuthenticateMode" />
          <xs:element ref="sicd:AuthenticateImplementation"
            minOccurs="0" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
- <xs:complexType name="AnonymousCredentialPolicyType">
  - <xs:annotation>
      <xs:documentation>This is an anonymous credential policy type
        that has no credential.</xs:documentation>
    </xs:annotation>
  - <xs:complexContent>
    - <xs:restriction base="sicd:Abstract_CredentialPolicyType">
      - <xs:sequence>
          <xs:element name="CredentialPolicyAlgorithm"
            type="xs:string" fixed="Anonymous" />
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
- <xs:complexType name="BasicCredentialPolicyType">
  - <xs:annotation>
      <xs:documentation>This is a basic credential policy type that
        uses ID and password as credential.</xs:documentation>
    </xs:annotation>
  - <xs:complexContent>
      <xs:extension base="sicd:AuthenticationCredentialPolicyType"
        />
    </xs:complexContent>
  </xs:complexType>
- <xs:complexType name="X509CredentialPolicyType">
```

```
- <xs:annotation>
    <xs:documentation>This is a X509 credential policy
      type.</xs:documentation>
  </xs:annotation>
- <xs:complexContent>
    <xs:extension base="sicd:AuthenticationCredentialPolicyType"
      />
  </xs:complexContent>
</xs:complexType>
- <xs:complexType name="BASE64_BINARYCredentialPolicyType">
- <xs:annotation>
    <xs:documentation>This is a BASE64_BINARY_CREDENTIAL
      policy type.</xs:documentation>
  </xs:annotation>
- <xs:complexContent>
  - <xs:extension base="sicd:AuthenticationCredentialPolicyType">
    - <xs:sequence>
        <xs:element name="valueType" type="xs:QName" />
        <xs:element name="encodingType" type="xs:QName" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
- <xs:complexType name="Abstract_EncryptionPolicyType"
    abstract="true">
- <xs:annotation>
    <xs:documentation>This is the abstract policy for Encryption
      policy algorithm.</xs:documentation>
  </xs:annotation>
- <xs:complexContent>
  - <xs:extension base="sicd:Abstract_PolicyType">
    - <xs:sequence>
        <xs:element name="EncryptionPolicyAlgorithm"
          type="xs:string" />
        <xs:element name="EncryptionMethod" type="xs:string"
          />
        <xs:element ref="sicd:KeySize" />
        <xs:element ref="sicd:SymmetryKeySize" minOccurs="0"
          />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
- <xs:complexType name="EncryptionPolicyType">
- <xs:annotation>
    <xs:documentation>This encryption policy will work for both
      XMLEnc and PKCS#7.</xs:documentation>
  </xs:annotation>
- <xs:complexContent>
  - <xs:extension base="sicd:Abstract_EncryptionPolicyType">
    - <xs:sequence>
        <xs:element name="KeyEncryptionMethod"
          type="xs:string" minOccurs="0" />
      </xs:sequence>
```

```
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  - <xs:element name="KeySize">
    - <xs:annotation>
        <xs:documentation>This is the asymmetry encryption or
          symmetry key size, depends which algorithm is used. For an
          asymmetry case, this will be the asymmetry key size, and
          the symmetry key size is defined on the SymmetryKeySize
          field.</xs:documentation>
      </xs:annotation>
    - <xs:simpleType>
      - <xs:restriction base="xs:short">
          <xs:minInclusive value="56" />
          <xs:maxExclusive value="4096" />
        </xs:restriction>
      </xs:simpleType>
    </xs:element>
  - <xs:element name="SymmetryKeySize">
    - <xs:annotation>
        <xs:documentation>This is the symmetry encryption key size, if
          the asymmetry algorithm is used.</xs:documentation>
      </xs:annotation>
    - <xs:simpleType>
      - <xs:restriction base="xs:short">
          <xs:minInclusive value="56" />
          <xs:maxExclusive value="4096" />
        </xs:restriction>
      </xs:simpleType>
    </xs:element>
  - <xs:complexType name="XMLEncryptionPolicyType">
    - <xs:annotation>
        <xs:documentation>This will work for any encryption policy
          type.</xs:documentation>
      </xs:annotation>
    - <xs:complexContent>
      - <xs:extension base="sicd:Abstract_EncryptionPolicyType">
        - <xs:sequence>
            <xs:element name="KeyEncryptionMethod"
              type="xs:string"
              default="http://www.w3.org/2001/04/xmlenc#rsa
              -1_5" />
            <xs:element name="DecryptionTransform"
              type="xs:string" minOccurs="0" />
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  - <xs:complexType name="Abstract_SignaturePolicyType" abstract="true">
    - <xs:annotation>
        <xs:documentation>This is the abstract policy for Digital
          Signature policy algorithm.</xs:documentation>
      </xs:annotation>
    - <xs:complexContent>
```

```
- <xs:extension base="sicd:Abstract_PolicyType">
  - <xs:sequence>
      <xs:element name="SignaturePolicyAlgorithm"
        type="xs:string" />
      <xs:element name="SignatureAlgorithm"
        type="xs:string" />
      <xs:element name="HashFunction" type="xs:string" />
    </xs:sequence>
  </xs:extension>
</xs:complexContent>
</xs:complexType>
- <xs:complexType name="SignaturePolicyType">
  - <xs:annotation>
      <xs:documentation>This will work for any digital signature
        policy type.</xs:documentation>
    </xs:annotation>
  - <xs:complexContent>
      <xs:extension base="sicd:Abstract_SignaturePolicyType" />
    </xs:complexContent>
</xs:complexType>
- <xs:complexType name="XMLDsigPolicyType">
  - <xs:annotation>
      <xs:documentation>This is for XMLDsig
        policy.</xs:documentation>
    </xs:annotation>
  - <xs:complexContent>
    - <xs:extension base="sicd:SignaturePolicyType">
      - <xs:sequence>
          <xs:element name="CanonicalizationMethod"
            type="xs:string" minOccurs="0" />
          <xs:element name="Transform" type="xs:string"
            minOccurs="0" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
</xs:complexType>
- <!--
Message Part
-->
- <xs:complexType name="PartElementType">
  - <xs:annotation>
      <xs:documentation>Xpath is used to define the element within
        the part of the message.</xs:documentation>
    </xs:annotation>
  - <xs:simpleContent>
    - <xs:extension base="xs:string">
        <xs:attribute name="Type" type="xs:anyURI" use="optional"
          />
        <xs:attribute name="BlockId" type="xs:short"
          use="optional" />
      </xs:extension>
    </xs:simpleContent>
</xs:complexType>
- <xs:complexType name="MessagePartsType">
```

```
- <xs:annotation>
    <xs:documentation>The part within a message. URI is used to
      define the part.</xs:documentation>
  </xs:annotation>
- <xs:sequence>
  - <xs:element name="PartElement" type="sicd:PartElementType"
      minOccurs="0" maxOccurs="unbounded">
    - <xs:annotation>
        <xs:documentation>The element within the part. It is
          only apply to XML type of message
          part.</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="PartName" type="xs:string" use="required" />
  <xs:attribute name="Type" type="xs:anyURI" use="optional" />
  <xs:attribute name="AlgorithmId" type="xs:string" use="optional" />
  <xs:attribute name="BlockId" type="xs:short" use="optional" />
  <xs:attribute name="isOptional" type="xs:boolean" use="optional"
    default="false" />
</xs:complexType>
- <xs:element name="MessagePart" type="sicd:MessagePartsType">
  - <xs:annotation>
      <xs:documentation>The part within the message. The
        AlgorithmId is for this part. If the AlgorithmId is not
        defined, then parent's AlgorithmId will be
        used.</xs:documentation>
    </xs:annotation>
  </xs:element>
</xs:schema>
```

# SecurityContract.XSD

```
<?xml version="1.0" encoding="UTF-8" ?>
- <!--
edited with XML Spy v4.4 U (http://www.xmlspy.com) by Chong Hsu (Commerce One)
-->
  - <!--
  Security Interop Contract Document

  Created by: Symon Chang
  Copyright 2002 Commerce One, Inc.
-->
- <xs:schema
  targetNamespace="publicid:com.commerceone:schemas/soapextension/contract/security/v1_0/SecurityContract.xsd"
  xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
  xmlns:sicd="publicid:com.commerceone:schemas/soapextension/contract/security/v1_0/SecurityContract.xsd"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  elementFormDefault="qualified" attributeFormDefault="unqualified"
  version="1.0">
  - <!--
  imports
  -->
    - <!--

      <xs:import
      namespace="publicid:com.commerceone:schemas/soapextension/contract/v1_0/InteroperabilityContract.xsd"
      schemaLocation="http://schemas.commerceone.com/schemas/soapextension/contract/v1_0/InteroperabilityContract.xsd"/>

    -->
    <xs:import namespace="urn:oasis:names:tc:SAML:1.0:assertion"
      schemaLocation="http://www.oasis-open.org/committees/security/docs/cs-sstc-schema-assertion-01.xsd" />
  - <!--
  includes
  -->
    <xs:include schemaLocation="SecurityContractKeyInfo.xsd" />
  - <!--
  Schema for Security Policies
  -->
  - <!--
  top element
  -->
  - <xs:element name="SecurityContractICD"
      type="sicd:SecurityContractType">
    - <xs:annotation>
        <xs:documentation>The Security Interop Contract agreement. It defines Policies and channels for security policies.</xs:documentation>
      </xs:annotation>
```

```
    </xs:element>
  - <!--
   Schema for Security Policies
    -->
  - <!--
   Define Crdetential Policies
    -->
  - <xs:element name="BasicCredentialPolicy"
      type="sicd:BasicCredentialPolicyType">
    - <xs:annotation>
        <xs:documentation>The credential and authentication algorithm
          policy for ID and Password.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="X509CredentialPolicy"
      type="sicd:X509CredentialPolicyType">
    - <xs:annotation>
        <xs:documentation>The credential and authentication algorithm
          policy for X.509 Certificate.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="AnonymousCredentialPolicy"
      type="sicd:AnonymousCredentialPolicyType">
    - <xs:annotation>
        <xs:documentation>The credential and authentication algorithm
          policy for no credential.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="BASE64_BINARYCredentialPolicy"
      type="sicd:BASE64_BINARYCredentialPolicyType">
    - <xs:annotation>
        <xs:documentation>The credential and authentication algorithm
          policy for BASE64_BINARY_CREDENTIAL</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="AuthenticationPolicies">
    - <xs:annotation>
        <xs:documentation>The abstraction for credential and
          authentication algorithm policy.</xs:documentation>
      </xs:annotation>
    - <xs:complexType>
      - <xs:sequence>
          <xs:element ref="sicd:BasicCredentialPolicy" minOccurs="0"
            maxOccurs="unbounded" />
          <xs:element ref="sicd:X509CredentialPolicy" minOccurs="0"
            maxOccurs="unbounded" />
          <xs:element ref="sicd:BASE64_BINARYCredentialPolicy"
            minOccurs="0" maxOccurs="unbounded" />
          <xs:element ref="sicd:AnonymousCredentialPolicy"
            minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  - <!--
   Define Encryption Policies
```

```
    -->
  - <xs:element name="EncryptionPolicy"
      type="sicd:EncryptionPolicyType">
    - <xs:annotation>
        <xs:documentation>The encryption algorithm and policy, such
          as PCSK#7, or S/MIME.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="XMLEncryptionPolicy"
      type="sicd:XMLEncryptionPolicyType">
    - <xs:annotation>
        <xs:documentation>The encryption algorithm and policy for
          XMLEnc.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="EncryptionPolicies">
    - <xs:annotation>
        <xs:documentation>The group of encryption algorithms and
          policies for XMLEnc, PCSK#7, or S/MIME. The PolicyID will
          be the TemplateID in the Registry. This ID will be used in
          the Channel Section as AlgorithmID to identify which
          encryption policy algorithm will be used.</xs:documentation>
      </xs:annotation>
    - <xs:complexType>
      - <xs:sequence>
          <xs:element ref="sicd:XMLEncryptionPolicy" minOccurs="0"
            maxOccurs="unbounded" />
          <xs:element ref="sicd:EncryptionPolicy" minOccurs="0"
            maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  - <!--
  Digital Signature Policy
    -->
  - <xs:element name="XMLDsigPolicy" type="sicd:XMLDsigPolicyType">
    - <xs:annotation>
        <xs:documentation>The signature algorithm and policy for
          XMLDsig.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="SignaturePolicy" type="sicd:SignaturePolicyType">
    - <xs:annotation>
        <xs:documentation>The signature algorithm and policy for
          XMLDsig, PCSK#7 or S/MIME.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="SignaturePolicies">
    - <xs:annotation>
        <xs:documentation>The group of digital signature algorithms
          and policies for XMLDsig, PCKS#7, or S/MIME. The Policy ID
          will be the TemplateID in the Registry. This Policy ID will be
          used in the Channel Section as AlgorithmID to identify
          which sinature policy algorithm will be
          used.</xs:documentation>
```

```
      </xs:annotation>
    - <xs:complexType>
      - <xs:sequence>
          <xs:element ref="sicd:XMLDsigPolicy" minOccurs="0"
            maxOccurs="unbounded" />
          <xs:element ref="sicd:SignaturePolicy" minOccurs="0"
            maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  - <!--
   Non-repudiation
    -->
  - <xs:element name="NonRepudiationPolicy"
      type="sicd:SignaturePolicyType"
      substitutionGroup="sicd:NonRepudiationPolicies">
    - <xs:annotation>
        <xs:documentation>The non-repudiation algorithm and policy
          that use digital signature.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="NonRepudiationPolicies"
      type="sicd:Abstract_PolicyType" abstract="true">
    - <xs:annotation>
        <xs:documentation>The policy and algorithm for non-
          repudiation of origin.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="NonRepudiationReceiptPolicy"
      type="sicd:SignaturePolicyType"
      substitutionGroup="sicd:NonRepudiationReceiptPolicies">
    - <xs:annotation>
        <xs:documentation>The non-repudiation algorithm and policy
          that use digital signature.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="NonRepudiationReceiptPolicies"
      type="sicd:Abstract_PolicyType" abstract="true">
    - <xs:annotation>
        <xs:documentation>The policy and algorithm for non-
          repudiation of receipt.</xs:documentation>
      </xs:annotation>
    </xs:element>
  - <xs:element name="SecurityPolicies">
    - <xs:annotation>
        <xs:documentation>The security Policies section. It defines all
          policy related security policies.</xs:documentation>
      </xs:annotation>
    - <xs:complexType>
      - <xs:sequence>
          <xs:element ref="sicd:AuthenticationPolicies" minOccurs="0"
            />
          <xs:element ref="sicd:SignaturePolicies" minOccurs="0" />
          <xs:element ref="sicd:EncryptionPolicies" minOccurs="0" />
```

```
            <xs:element ref="sicd:NonRepudiationPolicies"
              minOccurs="0" maxOccurs="unbounded" />
            <xs:element ref="sicd:NonRepudiationReceiptPolicies"
              minOccurs="0" maxOccurs="unbounded" />
            <xs:element ref="sicd:EncryptionKeyInfo" minOccurs="0"
              maxOccurs="unbounded" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    - <!--
     Schema for Channel
      -->
    - <xs:complexType name="KeyAlgorithmType">
      - <xs:annotation>
          <xs:documentation>The root for Integrity and Confidential
            blocks. All these two types of block within the Security
            channel have to have PublicKeyID and Algorithm Id, so does
            the signing and encryption policy within the Credential
            block.</xs:documentation>
        </xs:annotation>
      - <xs:sequence>
          <xs:element ref="sicd:PublicKeyName" />
        </xs:sequence>
        <xs:attribute name="AlgorithmId" type="xs:string" use="optional" />
      </xs:complexType>
    - <xs:complexType name="KeyMessagePartsType">
      - <xs:annotation>
          <xs:documentation>The root for parts in a message. It also
            define the KeyInfo and the algorithm policy for all
            parts.</xs:documentation>
        </xs:annotation>
      - <xs:complexContent>
        - <xs:extension base="sicd:KeyAlgorithmType">
          - <xs:sequence minOccurs="0">
              <xs:element ref="sicd:MessagePart" minOccurs="0"
                maxOccurs="unbounded" />
            </xs:sequence>
            <xs:attribute name="SequenceID" type="xs:short"
              use="optional" />
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
    - <xs:element name="Credential">
      - <xs:annotation>
          <xs:documentation>The credential and authentication policy.
            Note that the CredentailEncryptionAlgorithm is here. This is
            due to authentication will be preformed before the
            decryption at inbound.</xs:documentation>
        </xs:annotation>
      - <xs:complexType>
        - <xs:sequence minOccurs="0">
          - <xs:choice minOccurs="0">
            - <xs:element name="PartyID"
                type="sicd:CollaberationPartyID" minOccurs="0">
```

```
- <xs:annotation>
    <xs:documentation>The party ID that is used
      for Basic credential.</xs:documentation>
  </xs:annotation>
</xs:element>
- <xs:element ref="sicd:PublicKeyName" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>The key that is used for
        X.509 credential.</xs:documentation>
    </xs:annotation>
  </xs:element>
</xs:choice>
- <xs:element name="CredentialEncryptionAlgorithm"
    type="sicd:KeyAlgorithmType" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>The Encryption Algorithm that
        is used to encrypt the credential. This will only
        be used when the Authentication mode is
        TARGET.</xs:documentation>
    </xs:annotation>
  </xs:element>
</xs:sequence>
<xs:attribute name="AlgorithmId" type="xs:string"
  use="required" />
<xs:attribute name="SequenceID" type="xs:short"
  use="optional" />
<xs:attribute name="DelegationFlag" type="xs:boolean"
  use="optional" default="false" />
</xs:complexType>
</xs:element>
- <xs:element name="Confidential">
  - <xs:annotation>
      <xs:documentation>The encryption security policy. The
        AlgorithmId will be the tmeplateID from the Registry. If the
        AlgorithmId is defined and no message parts, then the
        whole message will be encrypted. In this case, if there are
        Non-XML parts, then the NonXMLAlgorithmID will be
        defined, too.</xs:documentation>
    </xs:annotation>
  - <xs:complexType>
    - <xs:complexContent>
      - <xs:extension base="sicd:KeyMessagePartsType">
          <xs:attribute name="NonXMLAlgorithmId"
            type="xs:string" use="optional" />
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>
- <xs:element name="Integrity">
  - <xs:annotation>
      <xs:documentation>The digital signature security policy. The
        AlgorithmId will be the tmeplateID from the Registry. If the
        AlgorithmID is defined, and no message parts then the
        whole message will be signed.</xs:documentation>
    </xs:annotation>
```

```
- <xs:complexType>
  - <xs:complexContent>
    - <xs:extension base="sicd:KeyMessagePartsType">
      - <xs:sequence minOccurs="0">
        - <xs:element name="HeaderSignatureAlgorithm"
            type="sicd:KeyAlgorithmType" minOccurs="0">
          - <xs:annotation>
              <xs:documentation>The Signature
                Algorithm that is used to sign the
                header credntial.</xs:documentation>
            </xs:annotation>
          </xs:element>
        </xs:sequence>
        <xs:attribute name="NonXMLAlgorithmId"
          type="xs:string" use="optional" />
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
- <xs:element name="NonRepudiation">
  - <xs:annotation>
      <xs:documentation>The non-repudiation of orgin
        policy.</xs:documentation>
    </xs:annotation>
  - <xs:complexType>
    - <xs:sequence>
        <xs:element name="NROSignPart"
          type="sicd:KeyMessagePartsType" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
- <xs:element name="NonRepudiationReceipt">
  - <xs:annotation>
      <xs:documentation>The non-repudiation of receipt
        policy.</xs:documentation>
    </xs:annotation>
  - <xs:complexType>
    - <xs:sequence>
        <xs:element name="NRRSignPart"
          type="sicd:KeyMessagePartsType" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
- <xs:element name="Authorization">
  - <xs:annotation>
      <xs:documentation>The SAML attribute assertion for the
        sending CP that will be pass to the receiving service. This
        will be shown in the end-to-end security
        channel.</xs:documentation>
    </xs:annotation>
  - <xs:complexType>
    - <xs:sequence minOccurs="0">
        <xs:element ref="saml:Assertion" minOccurs="0"
          maxOccurs="unbounded" />
```

```
        </xs:sequence>
        <xs:attribute name="RequireSubscription" type="xs:boolean"
          use="required" />
      </xs:complexType>
    - <!--
     saml:AttributeStatementType">
     -->
    </xs:element>
  - <xs:element name="SecurityContainer">
    - <xs:annotation>
        <xs:documentation>This will be the container for those piggy
          back security related objects.</xs:documentation>
      </xs:annotation>
    - <xs:complexType>
      - <xs:sequence minOccurs="0">
        - <xs:element name="MMLCredential" minOccurs="0">
          - <xs:complexType>
            - <xs:sequence minOccurs="0">
                <xs:element name="MarketParticipantID"
                  type="xs:string" minOccurs="0" />
                <xs:element name="TPName" type="xs:string"
                  minOccurs="0" />
                <xs:element name="TPShortName"
                  type="xs:string" minOccurs="0" />
                <xs:element name="TPRoleName"
                  type="xs:string" minOccurs="0"
                  maxOccurs="unbounded" />
              </xs:sequence>
            </xs:complexType>
          </xs:element>
          <xs:element name="PiggybackObject" type="xs:anyType"
            minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  - <xs:element name="SecurityChannel">
    - <xs:annotation>
        <xs:documentation>The Security Channel defines the from
          connector and to connector, and what to do within the
          channel, such as authentication, encryption and digital
          signature.</xs:documentation>
      </xs:annotation>
    - <xs:complexType>
      - <xs:sequence>
          <xs:element ref="sicd:Credential" minOccurs="0" />
          <xs:element ref="sicd:Confidential" minOccurs="0" />
          <xs:element ref="sicd:Integrity" minOccurs="0" />
        - <xs:element ref="sicd:Authorization" minOccurs="0">
          - <xs:annotation>
              <xs:documentation>The SAML attribute assertion
                for the sending CP that will be pass to the
                receiving service. This will be shown in the
                end-to-end security
                channel.</xs:documentation>
```

```
            </xs:annotation>
          </xs:element>
          <xs:element ref="sicd:NonRepudiation" minOccurs="0" />
          <xs:element ref="sicd:NonRepudiationReceipt"
            minOccurs="0" />
        - <xs:element ref="sicd:SecurityContainer" minOccurs="0">
          - <xs:annotation>
              <xs:documentation>This will be the container for
                those piggy back security related
                objects.</xs:documentation>
            </xs:annotation>
          </xs:element>
        </xs:sequence>
        <xs:attribute name="channelId" type="xs:string" use="optional"
          />
        <xs:attribute name="sourceConnector" type="xs:string"
          use="required" />
        <xs:attribute name="targetConnector" type="xs:string"
          use="required" />
      </xs:complexType>
    </xs:element>
  - <xs:complexType name="SecurityContractType">
    - <xs:sequence>
        <xs:element ref="sicd:SecurityPolicies" />
        <xs:element ref="sicd:SecurityChannel"
          maxOccurs="unbounded" />
      </xs:sequence>
    </xs:complexType>
  </xs:schema>
```

# CommunitySecurityTemplatesInfo.XML

```
<?xml version="1.0" encoding="UTF-8" ?>
- <!--
edited with XML Spy v4.4 U (http://www.xmlspy.com) by Symon Chang (Commerce One)
-->
- <sicdr:CommunitySecurityTemplatesPreference
    xmlns:sicdr="publicid:com.commerceone:schemas/contract/helperinfo/v1_0/SecuritySenderReceiverInfo.xsd"
    xmlns:sicd="publicid:com.commerceone:schemas/soapextension/contract/security/v1_0/SecurityContract.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="publicid:com.commerceone:schemas/contract/helperinfo/v1_0/SecuritySenderReceiverInfo.xsd
    http://schemas.commerceone.com/schemas/contract/helperinfo/v1_0/SecuritySenderReceiverInfo.xsd">
  - <sicdr:SecurityAlgorithmTemplates>
    - <sicdr:XMLSignatureAlgorithmTemplate Name="DSA-SHA1-C14N"
        ID="bb587fa0-b980-11d6-b8e9-c40beac518e7">
      <sicdr:Category>XMLSignature</sicdr:Category>
      - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureDSA-SHA1-C14N">
          <sicd:SignaturePolicyAlgorithm>http://www.w3.org/2000/09/xmldsig#</sicd:SignaturePolicyAlgorithm>
          <sicd:SignatureAlgorithm>SHA1withDSA</sicd:SignatureAlgorithm>
          <sicd:HashFunction>SHA1</sicd:HashFunction>
          <sicd:CanonicalizationMethod>http://www.w3.org/TR/2000/CR-xml-c14n-20001026</sicd:CanonicalizationMethod>
          <sicd:Transform>http://msdn.microsoft.com/ws/2002/01/Security#RoutingSignatureTransform</sicd:Transform>
        </sicd:XMLDsigPolicy>
      </sicdr:XMLSignatureAlgorithmTemplate>
    - <sicdr:XMLSignatureAlgorithmTemplate Name="DSA-SHA1-EXC14N"
        ID="bb587fa0-b980-11d6-b8e6-c40beac518e7">
      <sicdr:Category>XMLSignature</sicdr:Category>
      - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureDSA-SHA1-EXC14N">
          <sicd:SignaturePolicyAlgorithm>http://www.w3.org/2000/09/xmldsig#</sicd:SignaturePolicyAlgorithm>
          <sicd:SignatureAlgorithm>SHA1withDSA</sicd:SignatureAlgorithm>
          <sicd:HashFunction>SHA1</sicd:HashFunction>
          <sicd:CanonicalizationMethod>http://www.w3.org/2001/10/xml-exc-c14n#</sicd:CanonicalizationMethod>
          <sicd:Transform>http://msdn.microsoft.com/ws/2002/
```

```
            01/Security#RoutingSignatureTransform</sicd:Transfor
            m>
          </sicd:XMLDsigPolicy>
        </sicdr:XMLSignatureAlgorithmTemplate>
      - <sicdr:XMLSignatureAlgorithmTemplate Name="RSA-MD5-C14N"
          ID="bb587fa0-b980-11d6-b8e8-c40beac518e7">
          <sicdr:Category>XMLSignature</sicdr:Category>
        - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureRSA-MD5-C14N">

            <sicd:SignaturePolicyAlgorithm>http://www.w3.org/2000
            /09/xmldsig#</sicd:SignaturePolicyAlgorithm>

            <sicd:SignatureAlgorithm>MD5withRSA</sicd:SignatureAlgo
            rithm>
          <sicd:HashFunction>MD5</sicd:HashFunction>

            <sicd:CanonicalizationMethod>http://www.w3.org/TR/20
            00/CR-xml-c14n-
            20001026</sicd:CanonicalizationMethod>

            <sicd:Transform>http://msdn.microsoft.com/ws/2002/
            01/Security#RoutingSignatureTransform</sicd:Transfor
            m>
          </sicd:XMLDsigPolicy>
        </sicdr:XMLSignatureAlgorithmTemplate>
      - <sicdr:XMLSignatureAlgorithmTemplate Name="RSA-MD5-EXC14N"
          ID="bb587fa0-b980-11d6-b8e5-c40beac518e7">
          <sicdr:Category>XMLSignature</sicdr:Category>
        - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureRSA-MD5-
            EXC14N">

            <sicd:SignaturePolicyAlgorithm>http://www.w3.org/2000
            /09/xmldsig#</sicd:SignaturePolicyAlgorithm>

            <sicd:SignatureAlgorithm>MD5withRSA</sicd:SignatureAlgo
            rithm>
          <sicd:HashFunction>MD5</sicd:HashFunction>

            <sicd:CanonicalizationMethod>http://www.w3.org/2001/
            10/xml-exc-c14n#</sicd:CanonicalizationMethod>

            <sicd:Transform>http://msdn.microsoft.com/ws/2002/
            01/Security#RoutingSignatureTransform</sicd:Transfor
            m>
          </sicd:XMLDsigPolicy>
        </sicdr:XMLSignatureAlgorithmTemplate>
      - <sicdr:XMLSignatureAlgorithmTemplate Name="RSA-SHA1-C14N"
          ID="bb587fa0-b980-11d6-b8e7-c40beac518e7">
          <sicdr:Category>XMLSignature</sicdr:Category>
        - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureRSA-SHA1-C14N">

            <sicd:SignaturePolicyAlgorithm>http://www.w3.org/2000
            /09/xmldsig#</sicd:SignaturePolicyAlgorithm>

            <sicd:SignatureAlgorithm>SHA1withRSA</sicd:SignatureAlg
            orithm>
          <sicd:HashFunction>SHA1</sicd:HashFunction>
```

```
            <sicd:CanonicalizationMethod>http://www.w3.org/TR/20
            00/CR-xml-c14n-
            20001026</sicd:CanonicalizationMethod>

            <sicd:Transform>http://msdn.microsoft.com/ws/2002/
            01/Security#RoutingSignatureTransform</sicd:Transfor
            m>
          </sicd:XMLDsigPolicy>
        </sicdr:XMLSignatureAlgorithmTemplate>
      - <sicdr:XMLSignatureAlgorithmTemplate Name="RSA-SHA1-EXC14N"
          ID="bb587fa0-b980-11d6-b8e4-c40beac518e7">
          <sicdr:Category>XMLSignature</sicdr:Category>
        - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureRSA-SHA1-
            EXC14N">

            <sicd:SignaturePolicyAlgorithm>http://www.w3.org/2000
            /09/xmldsig#</sicd:SignaturePolicyAlgorithm>

            <sicd:SignatureAlgorithm>SHA1withRSA</sicd:SignatureAlg
            orithm>
          <sicd:HashFunction>SHA1</sicd:HashFunction>

            <sicd:CanonicalizationMethod>http://www.w3.org/2001/
            10/xml-exc-c14n#</sicd:CanonicalizationMethod>

            <sicd:Transform>http://msdn.microsoft.com/ws/2002/
            01/Security#RoutingSignatureTransform</sicd:Transfor
            m>
          </sicd:XMLDsigPolicy>
        </sicdr:XMLSignatureAlgorithmTemplate>
      - <sicdr:XMLEncryptionAlgorithmTemplate Name="3DES-RSA-2048"
          ID="bb587fa0-b980-11d6-b8ea-c40beac518e7">
          <sicdr:Category>XMLEncryption</sicdr:Category>
        - <sicd:XMLEncryptionPolicy PolicyId="P-XMLEncrypt3DES-RSA-
            2048">

            <sicd:EncryptionPolicyAlgorithm>http://www.w3.org/200
            1/04/xmlenc#</sicd:EncryptionPolicyAlgorithm>

            <sicd:EncryptionMethod>http://www.w3.org/2001/04/x
            mlenc#3des-cbc</sicd:EncryptionMethod>
          <sicd:KeySize>2048</sicd:KeySize>

            <sicd:KeyEncryptionMethod>http://www.w3.org/2001/0
            4/xmlenc#rsa-1_5</sicd:KeyEncryptionMethod>
          </sicd:XMLEncryptionPolicy>
        </sicdr:XMLEncryptionAlgorithmTemplate>
      - <sicdr:XMLEncryptionAlgorithmTemplate Name="AES-128-RSA-2048"
          ID="bb587fa0-b980-11d6-b8eb-c40beac518e7">
          <sicdr:Category>XMLEncryption</sicdr:Category>
        - <sicd:XMLEncryptionPolicy PolicyId="P-XMLEncryptAES-128-RSA-
            2048">

            <sicd:EncryptionPolicyAlgorithm>http://www.w3.org/200
            1/04/xmlenc#</sicd:EncryptionPolicyAlgorithm>
```

```
            <sicd:EncryptionMethod>http://www.w3.org/2001/04/x
            mlenc#aes128-cbc</sicd:EncryptionMethod>
          <sicd:KeySize>2048</sicd:KeySize>
          <sicd:SymmetryKeySize>128</sicd:SymmetryKeySize>

            <sicd:KeyEncryptionMethod>http://www.w3.org/2001/0
            4/xmlenc#rsa-1_5</sicd:KeyEncryptionMethod>
        </sicd:XMLEncryptionPolicy>
      </sicdr:XMLEncryptionAlgorithmTemplate>
    - <sicdr:XMLEncryptionAlgorithmTemplate Name="DES-RSA-1024"
        ID="bb587fa0-b980-11d6-b8ec-c40beac518e7">
        <sicdr:Category>XMLEncryption</sicdr:Category>
      - <sicd:XMLEncryptionPolicy PolicyId="P-XMLEncryptDES-RSA-
          1024">

            <sicd:EncryptionPolicyAlgorithm>http://www.w3.org/200
            1/04/xmlenc#</sicd:EncryptionPolicyAlgorithm>

            <sicd:EncryptionMethod>http://www.commerceone.com/
            security/xmlenc#des</sicd:EncryptionMethod>
          <sicd:KeySize>1024</sicd:KeySize>

            <sicd:KeyEncryptionMethod>http://www.w3.org/2001/0
            4/xmlenc#rsa-1_5</sicd:KeyEncryptionMethod>
        </sicd:XMLEncryptionPolicy>
      </sicdr:XMLEncryptionAlgorithmTemplate>
    </sicdr:SecurityAlgorithmTemplates>
  - <sicdr:CommunitySecurityPolicyPreference>
      <sicdr:SignMessageHeader>false</sicdr:SignMessageHeader>
      <sicdr:EncryptCredential>false</sicdr:EncryptCredential>
      <sicdr:CredentialPreference>BASIC</sicdr:CredentialPreference>
    </sicdr:CommunitySecurityPolicyPreference>
  - <sicdr:SecurityPolicyTemplatePreference Name="RSA-SHA1-EXC14N"
      ID="bb587fa0-b980-11d6-b8e4-c40beac518e7">
      <sicdr:Category>XMLSignature</sicdr:Category>
      <sicdr:Preference>101</sicdr:Preference>
    </sicdr:SecurityPolicyTemplatePreference>
  - <sicdr:SecurityPolicyTemplatePreference Name="RSA-MD5-EXC14N"
      ID="bb587fa0-b980-11d6-b8e5-c40beac518e7">
      <sicdr:Category>XMLSignature</sicdr:Category>
      <sicdr:Preference>102</sicdr:Preference>
    </sicdr:SecurityPolicyTemplatePreference>
  - <sicdr:SecurityPolicyTemplatePreference Name="RSA-SHA1-C14N"
      ID="bb587fa0-b980-11d6-b8e7-c40beac518e7">
      <sicdr:Category>XMLSignature</sicdr:Category>
      <sicdr:Preference>104</sicdr:Preference>
    </sicdr:SecurityPolicyTemplatePreference>
  - <sicdr:SecurityPolicyTemplatePreference Name="RSA-MD5-C14N"
      ID="bb587fa0-b980-11d6-b8e8-c40beac518e7">
      <sicdr:Category>XMLSignature</sicdr:Category>
      <sicdr:Preference>105</sicdr:Preference>
    </sicdr:SecurityPolicyTemplatePreference>
  - <sicdr:SecurityPolicyTemplatePreference Name="3DES-RSA-2048"
      ID="bb587fa0-b980-11d6-b8ea-c40beac518e7">
```

```
        <sicdr:Category>XMLEncryption</sicdr:Category>
        <sicdr:Preference>107</sicdr:Preference>
      </sicdr:SecurityPolicyTemplatePreference>
    - <sicdr:SecurityPolicyTemplatePreference Name="AES-128-RSA-2048"
        ID="bb587fa0-b980-11d6-b8eb-c40beac518e7">
        <sicdr:Category>XMLEncryption</sicdr:Category>
        <sicdr:Preference>108</sicdr:Preference>
      </sicdr:SecurityPolicyTemplatePreference>
    - <sicdr:SecurityPolicyTemplatePreference Name="DES-RSA-1024"
        ID="bb587fa0-b980-11d6-b8ec-c40beac518e7">
        <sicdr:Category>XMLEncryption</sicdr:Category>
        <sicdr:Preference>109</sicdr:Preference>
      </sicdr:SecurityPolicyTemplatePreference>
    </sicdr:CommunitySecurityTemplatesPreference>
```

## SecuritySenderInfo.XML

```
<?xml version="1.0" encoding="UTF-8" ?>
- <!--
edited with XML Spy v4.4 U (http://www.xmlspy.com) by Symon Chang (Commerce One)
-->
- <!--
Sample XML file generated by XML Spy v4.4 U (http://www.xmlspy.com)
-->
- <SecuritySenderInfo
    xmlns="publicid:com.commerceone:schemas/contract/helperinfo/v1_0/
    SecuritySenderReceiverInfo.xsd"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
    xmlns:sicd="publicid:com.commerceone:schemas/soapextension/contrac
    t/security/v1_0/SecurityContract.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="publicid:com.commerceone:schemas/contract/helpe
    rinfo/v1_0/SecuritySenderReceiverInfo.xsd
    C:\platform\core\main\wse\schema\contract\helperinfo\v1_0\Securit
    ySenderReceiverInfo.xsd">
  - <CommunitySecurityTemplatesPreference>
    - <SecurityAlgorithmTemplates>
      - <XMLSignatureAlgorithmTemplate Name="DSA-SHA1-C14N"
          ID="bb587fa0-b980-11d6-b8e9-c40beac518e7">
          <Category>XMLSignature</Category>
        - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureDSA-SHA1-
            C14N">

              <sicd:SignaturePolicyAlgorithm>http://www.w3.org/
              2000/09/xmldsig#</sicd:SignaturePolicyAlgorithm>

              <sicd:SignatureAlgorithm>SHA1withDSA</sicd:Signatu
              reAlgorithm>
            <sicd:HashFunction>SHA1</sicd:HashFunction>

              <sicd:CanonicalizationMethod>http://www.w3.org/TR
              /2000/CR-xml-c14n-
              20001026</sicd:CanonicalizationMethod>

              <sicd:Transform>http://msdn.microsoft.com/ws/20
              02/01/Security#RoutingSignatureTransform</sicd:
              Transform>
          </sicd:XMLDsigPolicy>
        </XMLSignatureAlgorithmTemplate>
      - <XMLSignatureAlgorithmTemplate Name="DSA-SHA1-EXC14N"
          ID="bb587fa0-b980-11d6-b8e6-c40beac518e7">
          <Category>XMLSignature</Category>
        - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureDSA-SHA1-
            EXC14N">

              <sicd:SignaturePolicyAlgorithm>http://www.w3.org/
              2000/09/xmldsig#</sicd:SignaturePolicyAlgorithm>

              <sicd:SignatureAlgorithm>SHA1withDSA</sicd:Signatu
              reAlgorithm>
```

```
            <sicd:HashFunction>SHA1</sicd:HashFunction>

              <sicd:CanonicalizationMethod>http://www.w3.org/20
              01/10/xml-exc-c14n#</sicd:CanonicalizationMethod>

              <sicd:Transform>http://msdn.microsoft.com/ws/20
              02/01/Security#RoutingSignatureTransform</sicd:
              Transform>
            </sicd:XMLDsigPolicy>
          </XMLSignatureAlgorithmTemplate>
        - <XMLSignatureAlgorithmTemplate Name="RSA-MD5-C14N"
            ID="bb587fa0-b980-11d6-b8e8-c40beac518e7">
            <Category>XMLSignature</Category>
          - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureRSA-MD5-
              C14N">

              <sicd:SignaturePolicyAlgorithm>http://www.w3.org/
              2000/09/xmldsig#</sicd:SignaturePolicyAlgorithm>

              <sicd:SignatureAlgorithm>MD5withRSA</sicd:Signatur
              eAlgorithm>
            <sicd:HashFunction>MD5</sicd:HashFunction>

              <sicd:CanonicalizationMethod>http://www.w3.org/TR
              /2000/CR-xml-c14n-
              20001026</sicd:CanonicalizationMethod>

              <sicd:Transform>http://msdn.microsoft.com/ws/20
              02/01/Security#RoutingSignatureTransform</sicd:
              Transform>
            </sicd:XMLDsigPolicy>
          </XMLSignatureAlgorithmTemplate>
        - <XMLSignatureAlgorithmTemplate Name="RSA-MD5-EXC14N"
            ID="bb587fa0-b980-11d6-b8e5-c40beac518e7">
            <Category>XMLSignature</Category>
          - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureRSA-MD5-
              EXC14N">

              <sicd:SignaturePolicyAlgorithm>http://www.w3.org/
              2000/09/xmldsig#</sicd:SignaturePolicyAlgorithm>

              <sicd:SignatureAlgorithm>MD5withRSA</sicd:Signatur
              eAlgorithm>
            <sicd:HashFunction>MD5</sicd:HashFunction>

              <sicd:CanonicalizationMethod>http://www.w3.org/20
              01/10/xml-exc-c14n#</sicd:CanonicalizationMethod>

              <sicd:Transform>http://msdn.microsoft.com/ws/20
              02/01/Security#RoutingSignatureTransform</sicd:
              Transform>
            </sicd:XMLDsigPolicy>
          </XMLSignatureAlgorithmTemplate>
        - <XMLSignatureAlgorithmTemplate Name="RSA-SHA1-C14N"
            ID="bb587fa0-b980-11d6-b8e7-c40beac518e7">
            <Category>XMLSignature</Category>
```

```
- <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureRSA-SHA1-
    C14N">
      <sicd:SignaturePolicyAlgorithm>http://www.w3.org/
      2000/09/xmldsig#</sicd:SignaturePolicyAlgorithm>
      <sicd:SignatureAlgorithm>SHA1withRSA</sicd:Signatu
      reAlgorithm>
    <sicd:HashFunction>SHA1</sicd:HashFunction>
      <sicd:CanonicalizationMethod>http://www.w3.org/TR
      /2000/CR-xml-c14n-
      20001026</sicd:CanonicalizationMethod>
      <sicd:Transform>http://msdn.microsoft.com/ws/20
      02/01/Security#RoutingSignatureTransform</sicd:
      Transform>
  </sicd:XMLDsigPolicy>
</XMLSignatureAlgorithmTemplate>
- <XMLSignatureAlgorithmTemplate Name="RSA-SHA1-EXC14N"
    ID="bb587fa0-b980-11d6-b8e4-c40beac518e7">
  <Category>XMLSignature</Category>
  - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureRSA-SHA1-
      EXC14N">
      <sicd:SignaturePolicyAlgorithm>http://www.w3.org/
      2000/09/xmldsig#</sicd:SignaturePolicyAlgorithm>
      <sicd:SignatureAlgorithm>SHA1withRSA</sicd:Signatu
      reAlgorithm>
    <sicd:HashFunction>SHA1</sicd:HashFunction>
      <sicd:CanonicalizationMethod>http://www.w3.org/20
      01/10/xml-exc-c14n#</sicd:CanonicalizationMethod>
      <sicd:Transform>http://msdn.microsoft.com/ws/20
      02/01/Security#RoutingSignatureTransform</sicd:
      Transform>
  </sicd:XMLDsigPolicy>
</XMLSignatureAlgorithmTemplate>
- <XMLEncryptionAlgorithmTemplate Name="3DES-RSA-2048"
    ID="bb587fa0-b980-11d6-b8ea-c40beac518e7">
  <Category>XMLEncryption</Category>
  - <sicd:XMLEncryptionPolicy PolicyId="P-XMLEncrypt3DES-RSA-
      2048">
      <sicd:EncryptionPolicyAlgorithm>http://www.w3.org/
      2001/04/xmlenc#</sicd:EncryptionPolicyAlgorithm>
      <sicd:EncryptionMethod>http://www.w3.org/2001/0
      4/xmlenc#3des-cbc</sicd:EncryptionMethod>
    <sicd:KeySize>2048</sicd:KeySize>
      <sicd:KeyEncryptionMethod>http://www.w3.org/200
      1/04/xmlenc#rsa-1_5</sicd:KeyEncryptionMethod>
  </sicd:XMLEncryptionPolicy>
</XMLEncryptionAlgorithmTemplate>
```

```
- <XMLEncryptionAlgorithmTemplate Name="AES-128-RSA-2048"
    ID="bb587fa0-b980-11d6-b8eb-c40beac518e7">
    <Category>XMLEncryption</Category>
  - <sicd:XMLEncryptionPolicy PolicyId="P-XMLEncryptAES-128-
      RSA-2048">
      <sicd:EncryptionPolicyAlgorithm>http://www.w3.org/
      2001/04/xmlenc#</sicd:EncryptionPolicyAlgorithm>
      <sicd:EncryptionMethod>http://www.w3.org/2001/0
      4/xmlenc#aes128-cbc</sicd:EncryptionMethod>
    <sicd:KeySize>2048</sicd:KeySize>
    <sicd:SymmetryKeySize>128</sicd:SymmetryKeySize>
      <sicd:KeyEncryptionMethod>http://www.w3.org/200
      1/04/xmlenc#rsa-1_5</sicd:KeyEncryptionMethod>
    </sicd:XMLEncryptionPolicy>
  </XMLEncryptionAlgorithmTemplate>
- <XMLEncryptionAlgorithmTemplate Name="DES-RSA-1024"
    ID="bb587fa0-b980-11d6-b8ec-c40beac518e7">
    <Category>XMLEncryption</Category>
  - <sicd:XMLEncryptionPolicy PolicyId="P-XMLEncryptDES-RSA-
      1024">
      <sicd:EncryptionPolicyAlgorithm>http://www.w3.org/
      2001/04/xmlenc#</sicd:EncryptionPolicyAlgorithm>
      <sicd:EncryptionMethod>http://www.commerceone.c
      om/security/xmlenc#des</sicd:EncryptionMethod>
    <sicd:KeySize>1024</sicd:KeySize>
      <sicd:KeyEncryptionMethod>http://www.w3.org/200
      1/04/xmlenc#rsa-1_5</sicd:KeyEncryptionMethod>
    </sicd:XMLEncryptionPolicy>
  </XMLEncryptionAlgorithmTemplate>
</SecurityAlgorithmTemplates>
- <CommunitySecurityPolicyPreference>
    <SignMessageHeader>false</SignMessageHeader>
    <EncryptCredential>false</EncryptCredential>
    <CredentialPreference>BASIC</CredentialPreference>
</CommunitySecurityPolicyPreference>
- <SecurityPolicyTemplatePreference Name="RSA-SHA1-EXC14N"
    ID="bb587fa0-b980-11d6-b8e4-c40beac518e7">
    <Category>XMLSignature</Category>
    <Preference>101</Preference>
</SecurityPolicyTemplatePreference>
- <SecurityPolicyTemplatePreference Name="RSA-MD5-EXC14N"
    ID="bb587fa0-b980-11d6-b8e5-c40beac518e7">
    <Category>XMLSignature</Category>
    <Preference>102</Preference>
</SecurityPolicyTemplatePreference>
- <SecurityPolicyTemplatePreference Name="DSA-SHA1-EXC14N"
    ID="bb587fa0-b980-11d6-b8e6-c40beac518e7">
    <Category>XMLSignature</Category>
    <Preference>103</Preference>
```

```
    </SecurityPolicyTemplatePreference>
  - <SecurityPolicyTemplatePreference Name="RSA-SHA1-C14N"
      ID="bb587fa0-b980-11d6-b8e7-c40beac518e7">
      <Category>XMLSignature</Category>
      <Preference>104</Preference>
    </SecurityPolicyTemplatePreference>
  - <SecurityPolicyTemplatePreference Name="RSA-MD5-C14N"
      ID="bb587fa0-b980-11d6-b8e8-c40beac518e7">
      <Category>XMLSignature</Category>
      <Preference>105</Preference>
    </SecurityPolicyTemplatePreference>
  - <SecurityPolicyTemplatePreference Name="DSA-SHA1-C14N"
      ID="bb587fa0-b980-11d6-b8e9-c40beac518e7">
      <Category>XMLSignature</Category>
      <Preference>106</Preference>
    </SecurityPolicyTemplatePreference>
  - <SecurityPolicyTemplatePreference Name="3DES-RSA-2048"
      ID="bb587fa0-b980-11d6-b8ea-c40beac518e7">
      <Category>XMLEncryption</Category>
      <Preference>107</Preference>
    </SecurityPolicyTemplatePreference>
  - <SecurityPolicyTemplatePreference Name="AES-128-RSA-2048"
      ID="bb587fa0-b980-11d6-b8eb-c40beac518e7">
      <Category>XMLEncryption</Category>
      <Preference>108</Preference>
    </SecurityPolicyTemplatePreference>
  - <SecurityPolicyTemplatePreference Name="DES-RSA-1024"
      ID="bb587fa0-b980-11d6-b8ec-c40beac518e7">
      <Category>XMLEncryption</Category>
      <Preference>109</Preference>
    </SecurityPolicyTemplatePreference>
  </CommunitySecurityTemplatesPreference>
- <SAMsgSecurityPolicy>
  - <SAMsgPart PartName="Order" isOptional="false">
    - <PartSignatureAlgCategory>
        <XMLSignatureAlgCategory>XMLSignature</XMLSignatureAl
        gCategory>
      </PartSignatureAlgCategory>
    - <PartEncryptionAlgCategory>
        <XMLEncryptionAlgCategory>XMLEncryption</XMLEncryptio
        nAlgCategory>
      </PartEncryptionAlgCategory>
    </SAMsgPart>
  - <SAMsgPart PartName="Image" isOptional="false">
    - <PartEncryptionAlgCategory>
        <XMLEncryptionAlgCategory>XMLEncryption</XMLEncryptio
        nAlgCategory>
      </PartEncryptionAlgCategory>
    </SAMsgPart>
  </SAMsgSecurityPolicy>
- <PublicKeys>
```

```
<sicd:PartyID>x-ccns:commerceone.com:CollaborationParty::buyParty</sicd:PartyID>
- <sicd:EncryptionKeyInfo KeyOwner="OwnerA">
<sicd:PublicKeyID>BuyerPublicKey</sicd:PublicKeyID>
- <sicd:X509Data>
<sicd:X509Certificate>LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tTUlJREZEQ0NBZnlnQXdJQkFnSUVQT0ZQSVRBTkJna3Foa2lHOXcwQkFRVUZBREI2TVFzd0NRWURWUVFHRXdKVlV6RVZNQk1HQTFVRUNoTU1RMjl0YldWeVkyVWdUMjVsTVMwd0t3WURWUVFMRXlSVWFHbHpJRU5CSUdseklHWnZjaUIwWlhOMGFXNW5JSEIxY25CdmMyVnpJRzl1YkhreEpUQWpCZ05WQkFNVUhFTnZiVzFsY21OblpXNWpiR1VnUTBFZ01CNFhEVEUwTURJd05URTBNVGMxTlRFME1UWTFNek0zV2pCb01SZ3dGZ1lEVlFRREV3OUVZWFpwWkNCVVpYTjBJRUl3TURJeEVqQVFCZ05WQkFjVENVTjFjR1Z5ZEdsdWJ6RVVNQklHQTFVRUN4TUxSVzVuYVc1bFpYSnBibWN4RlRBVEJnTlZCQW9URVODmJXMWxjbU5sSUU5dVpURUxNQWtHQTFVRUJoTUNWVk13Z1o4d0RRWUpLb1pJaHZjTkFRRUJCUUFEZ1kwQU1JR0pBb0dCQU5nc2pTQkxjcFp2QnVDQ2lTTHR3RGFkaFZEMGNLRXJuQ3M2azg5UEhSUGJSMFdYOHBDUzByZWxIMkcyaDMxNU5vNGkzQVNidHZhYmdHellRVFNiR2EzcWtNYmVLNDZTSGxtTkJOTUp2YUkvMmZVQlBxdkkzejlLTVJSTGh3eUhCMEdFNmUvSzdnVGZkSUo0MUJobTZzSmcwYzJqZ041cWt1d3FZQkV4eWN1MUFnTUJBQUdqT0RBMk1DY0dBMVVkRVFRZ01CNkJIRzE1VkdWemRFVnRZV2xzUUdOdmJXMWpaWEpqWlc5dVpTNWpiMjB3Q3dZRFZSMFBCQVFEQWdYZ01BMEdDU3FHU0liM0RRRUJCUVVBQTRJQkFRQ0UrNEVaUWZYZWpmVnBsbXEzZnFtUjJZSGZhczErXA0MUg4UWRmNmRESXBiYkZ2OUxocnorYkc2c2hWQlptMVpYVXphaHl6N2Q3Z2U3V0MxR2FZVjFHYldFTXJMUkZkeXM2c1VlQkZNbHZuNkZPRjNqOHdMY3JuN2FFN3pRMEMwa2U5LzVVNVBHTnlaZWVaUGNLNTlKM0hPdWpzbXUvaENPVW1OOXZVM2M3MHVjMmhRaE96aExxJQ0VlQ2VTRDFCd2hEMXNkdXZmNnVOanAzUGp2eUpCaklTeDVxY2UwS25oQmxpcDR3ejRNTWxpdEtTdkFXSElqR1Bvb0w0N01ac3I4N3RLamJHaTgxcWJrQ3hiYlZ1dEloYmkzZDRnaW1Ockc1RXJ0dUUxNmwvRW9GUkJLU2VRTXd2cFdGUlliN2YreWtKVGE5ZVRLaWF4R2hOcDR4dnc5LS0tLS1FTkQgQ0VSVElGSUNBVEUtLS0tLQ==</sicd:X509Certificate>
</sicd:X509Data>
</sicd:EncryptionKeyInfo>
<sicd:KeyTypeUsage>SIGNATURE</sicd:KeyTypeUsage>
<sicd:KeyTypeUsage>AUTHENTICATION</sicd:KeyTypeUsage>
<sicd:KeyAlgorithm>RSA</sicd:KeyAlgorithm>
<sicd:Description>String</sicd:Description>
<sicd:Location>String</sicd:Location>
</PublicKeys>
- <ConnectorCapability ConnectorName="x-ccns:cup.commerceone.com:connector::buy">
<EncryptionCapability>false</EncryptionCapability>
<SignatureCapability>true</SignatureCapability>
</ConnectorCapability>
```

```
- <ConnectorCapability ConnectorName="x-
    ccns:cup.commerceone.com:connector::centerBuyI">
    <EncryptionCapability>true</EncryptionCapability>
    <SignatureCapability>false</SignatureCapability>
    <EncryptionPublicKeyParty>x-
      ccns:commerceone.com:CollaborationParty::buyParty</Encryptio
      nPublicKeyParty>
  </ConnectorCapability>
- <SendingCPSecurityPolicyProfile>
    <AvailableCredentials>BASIC</AvailableCredentials>
    <AvailableCredentials>X509</AvailableCredentials>
  </SendingCPSecurityPolicyProfile>
- <CPSendServicesSecurityPolicy AuthenticateParty="x-
    ccns:commerceone.com:CollaborationParty::buyParty">
  - <SecurityPolicyTemplatePreference Name="3DES-RSA-2048">
      <Category>XMLEncryption</Category>
      <Preference>1</Preference>
    </SecurityPolicyTemplatePreference>
  - <SecurityPolicyTemplatePreference Name="AES-128-RSA-2048">
      <Category>XMLEncryption</Category>
      <Preference>2</Preference>
    </SecurityPolicyTemplatePreference>
  - <SecurityPolicyTemplatePreference Name="RC2-128-RSA-2048">
      <Category>XMLEncryption</Category>
      <Preference>3</Preference>
    </SecurityPolicyTemplatePreference>
  </CPSendServicesSecurityPolicy>
</SecuritySenderInfo>
```

# SecurityReceiverInfo.XML

```
<?xml version="1.0" encoding="UTF-8" ?>
- <!--
edited with XML Spy v4.4 U (http://www.xmlspy.com) by Symon Chang (Commerce One)
-->
- <SecurityReceiverInfo
    xmlns="publicid:com.commerceone:schemas/contract/helperinfo/v1_0/
    SecuritySenderReceiverInfo.xsd"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
    xmlns:sicd="publicid:com.commerceone:schemas/soapextension/contrac
    t/security/v1_0/SecurityContract.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="publicid:com.commerceone:schemas/contract/helpe
    rinfo/v1_0/SecuritySenderReceiverInfo.xsd
    C:\platform\core\main\wse\schema\contract\helperinfo\v1_0\Securit
    ySenderReceiverInfo.xsd">
  - <CommunitySecurityTemplatesPreference>
    - <SecurityAlgorithmTemplates>
      - <XMLSignatureAlgorithmTemplate Name="DSA-SHA1-C14N"
          ID="bb587fa0-b980-11d6-b8e9-c40beac518e7">
          <Category>XMLSignature</Category>
        - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureDSA-SHA1-
            C14N">

              <sicd:SignaturePolicyAlgorithm>http://www.w3.org/
              2000/09/xmldsig#</sicd:SignaturePolicyAlgorithm>

              <sicd:SignatureAlgorithm>SHA1withDSA</sicd:Signatu
              reAlgorithm>
            <sicd:HashFunction>SHA1</sicd:HashFunction>

              <sicd:CanonicalizationMethod>http://www.w3.org/TR
              /2000/CR-xml-c14n-
              20001026</sicd:CanonicalizationMethod>

              <sicd:Transform>http://msdn.microsoft.com/ws/20
              02/01/Security#RoutingSignatureTransform</sicd:
              Transform>
          </sicd:XMLDsigPolicy>
        </XMLSignatureAlgorithmTemplate>
      - <XMLSignatureAlgorithmTemplate Name="DSA-SHA1-EXC14N"
          ID="bb587fa0-b980-11d6-b8e6-c40beac518e7">
          <Category>XMLSignature</Category>
        - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureDSA-SHA1-
            EXC14N">

              <sicd:SignaturePolicyAlgorithm>http://www.w3.org/
              2000/09/xmldsig#</sicd:SignaturePolicyAlgorithm>

              <sicd:SignatureAlgorithm>SHA1withDSA</sicd:Signatu
              reAlgorithm>
            <sicd:HashFunction>SHA1</sicd:HashFunction>
```

```
                <sicd:CanonicalizationMethod>http://www.w3.org/20
                01/10/xml-exc-c14n#</sicd:CanonicalizationMethod>

                <sicd:Transform>http://msdn.microsoft.com/ws/20
                02/01/Security#RoutingSignatureTransform</sicd:
                Transform>
            </sicd:XMLDsigPolicy>
        </XMLSignatureAlgorithmTemplate>
    - <XMLSignatureAlgorithmTemplate Name="RSA-MD5-C14N"
        ID="bb587fa0-b980-11d6-b8e8-c40beac518e7">
        <Category>XMLSignature</Category>
      - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureRSA-MD5-
            C14N">

                <sicd:SignaturePolicyAlgorithm>http://www.w3.org/
                2000/09/xmldsig#</sicd:SignaturePolicyAlgorithm>

                <sicd:SignatureAlgorithm>MD5withRSA</sicd:Signatur
                eAlgorithm>
              <sicd:HashFunction>MD5</sicd:HashFunction>

                <sicd:CanonicalizationMethod>http://www.w3.org/TR
                /2000/CR-xml-c14n-
                20001026</sicd:CanonicalizationMethod>

                <sicd:Transform>http://msdn.microsoft.com/ws/20
                02/01/Security#RoutingSignatureTransform</sicd:
                Transform>
            </sicd:XMLDsigPolicy>
        </XMLSignatureAlgorithmTemplate>
    - <XMLSignatureAlgorithmTemplate Name="RSA-MD5-EXC14N"
        ID="bb587fa0-b980-11d6-b8e5-c40beac518e7">
        <Category>XMLSignature</Category>
      - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureRSA-MD5-
            EXC14N">

                <sicd:SignaturePolicyAlgorithm>http://www.w3.org/
                2000/09/xmldsig#</sicd:SignaturePolicyAlgorithm>

                <sicd:SignatureAlgorithm>MD5withRSA</sicd:Signatur
                eAlgorithm>
              <sicd:HashFunction>MD5</sicd:HashFunction>

                <sicd:CanonicalizationMethod>http://www.w3.org/20
                01/10/xml-exc-c14n#</sicd:CanonicalizationMethod>

                <sicd:Transform>http://msdn.microsoft.com/ws/20
                02/01/Security#RoutingSignatureTransform</sicd:
                Transform>
            </sicd:XMLDsigPolicy>
        </XMLSignatureAlgorithmTemplate>
    - <XMLSignatureAlgorithmTemplate Name="RSA-SHA1-C14N"
        ID="bb587fa0-b980-11d6-b8e7-c40beac518e7">
        <Category>XMLSignature</Category>
      - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureRSA-SHA1-
            C14N">
```

```
                <sicd:SignaturePolicyAlgorithm>http://www.w3.org/
                2000/09/xmldsig#</sicd:SignaturePolicyAlgorithm>

                <sicd:SignatureAlgorithm>SHA1withRSA</sicd:Signatu
                reAlgorithm>
              <sicd:HashFunction>SHA1</sicd:HashFunction>

                <sicd:CanonicalizationMethod>http://www.w3.org/TR
                /2000/CR-xml-c14n-
                20001026</sicd:CanonicalizationMethod>

                <sicd:Transform>http://msdn.microsoft.com/ws/20
                02/01/Security#RoutingSignatureTransform</sicd:
                Transform>
            </sicd:XMLDsigPolicy>
          </XMLSignatureAlgorithmTemplate>
        - <XMLSignatureAlgorithmTemplate Name="RSA-SHA1-EXC14N"
            ID="bb587fa0-b980-11d6-b8e4-c40beac518e7">
            <Category>XMLSignature</Category>
          - <sicd:XMLDsigPolicy PolicyId="P-XMLSignatureRSA-SHA1-
              EXC14N">

                <sicd:SignaturePolicyAlgorithm>http://www.w3.org/
                2000/09/xmldsig#</sicd:SignaturePolicyAlgorithm>

                <sicd:SignatureAlgorithm>SHA1withRSA</sicd:Signatu
                reAlgorithm>
              <sicd:HashFunction>SHA1</sicd:HashFunction>

                <sicd:CanonicalizationMethod>http://www.w3.org/20
                01/10/xml-exc-c14n#</sicd:CanonicalizationMethod>

                <sicd:Transform>http://msdn.microsoft.com/ws/20
                02/01/Security#RoutingSignatureTransform</sicd:
                Transform>
            </sicd:XMLDsigPolicy>
          </XMLSignatureAlgorithmTemplate>
        - <XMLEncryptionAlgorithmTemplate Name="3DES-RSA-2048"
            ID="bb587fa0-b980-11d6-b8ea-c40beac518e7">
            <Category>XMLEncryption</Category>
          - <sicd:XMLEncryptionPolicy PolicyId="P-XMLEncrypt3DES-RSA-
              2048">

                <sicd:EncryptionPolicyAlgorithm>http://www.w3.org/
                2001/04/xmlenc#</sicd:EncryptionPolicyAlgorithm>

                <sicd:EncryptionMethod>http://www.w3.org/2001/0
                4/xmlenc#3des-cbc</sicd:EncryptionMethod>
              <sicd:KeySize>2048</sicd:KeySize>

                <sicd:KeyEncryptionMethod>http://www.w3.org/200
                1/04/xmlenc#rsa-1_5</sicd:KeyEncryptionMethod>
            </sicd:XMLEncryptionPolicy>
          </XMLEncryptionAlgorithmTemplate>
        - <XMLEncryptionAlgorithmTemplate Name="AES-128-RSA-2048"
            ID="bb587fa0-b980-11d6-b8eb-c40beac518e7">
```

```
<Category>XMLEncryption</Category>
- <sicd:XMLEncryptionPolicy PolicyId="P-XMLEncryptAES-128-
    RSA-2048">
    <sicd:EncryptionPolicyAlgorithm>http://www.w3.org/
    2001/04/xmlenc#</sicd:EncryptionPolicyAlgorithm>
    <sicd:EncryptionMethod>http://www.w3.org/2001/0
    4/xmlenc#aes128-cbc</sicd:EncryptionMethod>
  <sicd:KeySize>2048</sicd:KeySize>
  <sicd:SymmetryKeySize>128</sicd:SymmetryKeySize>
    <sicd:KeyEncryptionMethod>http://www.w3.org/200
    1/04/xmlenc#rsa-1_5</sicd:KeyEncryptionMethod>
  </sicd:XMLEncryptionPolicy>
</XMLEncryptionAlgorithmTemplate>
- <XMLEncryptionAlgorithmTemplate Name="DES-RSA-1024"
    ID="bb587fa0-b980-11d6-b8ec-c40beac518e7">
  <Category>XMLEncryption</Category>
  - <sicd:XMLEncryptionPolicy PolicyId="P-XMLEncryptDES-RSA-
      1024">
    <sicd:EncryptionPolicyAlgorithm>http://www.w3.org/
    2001/04/xmlenc#</sicd:EncryptionPolicyAlgorithm>
    <sicd:EncryptionMethod>http://www.commerceone.c
    om/security/xmlenc#des</sicd:EncryptionMethod>
  <sicd:KeySize>1024</sicd:KeySize>
    <sicd:KeyEncryptionMethod>http://www.w3.org/200
    1/04/xmlenc#rsa-1_5</sicd:KeyEncryptionMethod>
  </sicd:XMLEncryptionPolicy>
</XMLEncryptionAlgorithmTemplate>
</SecurityAlgorithmTemplates>
- <CommunitySecurityPolicyPreference>
  <SignMessageHeader>false</SignMessageHeader>
  <EncryptCredential>false</EncryptCredential>
  <CredentialPreference>BASIC</CredentialPreference>
</CommunitySecurityPolicyPreference>
- <SecurityPolicyTemplatePreference Name="RSA-SHA1-EXC14N"
    ID="bb587fa0-b980-11d6-b8e4-c40beac518e7">
  <Category>XMLSignature</Category>
  <Preference>101</Preference>
</SecurityPolicyTemplatePreference>
- <SecurityPolicyTemplatePreference Name="RSA-MD5-EXC14N"
    ID="bb587fa0-b980-11d6-b8e5-c40beac518e7">
  <Category>XMLSignature</Category>
  <Preference>102</Preference>
</SecurityPolicyTemplatePreference>
- <SecurityPolicyTemplatePreference Name="DSA-SHA1-EXC14N"
    ID="bb587fa0-b980-11d6-b8e6-c40beac518e7">
  <Category>XMLSignature</Category>
  <Preference>103</Preference>
</SecurityPolicyTemplatePreference>
```

```
- <SecurityPolicyTemplatePreference Name="RSA-SHA1-C14N"
    ID="bb587fa0-b980-11d6-b8e7-c40beac518e7">
    <Category>XMLSignature</Category>
    <Preference>104</Preference>
  </SecurityPolicyTemplatePreference>
- <SecurityPolicyTemplatePreference Name="RSA-MD5-C14N"
    ID="bb587fa0-b980-11d6-b8e8-c40beac518e7">
    <Category>XMLSignature</Category>
    <Preference>105</Preference>
  </SecurityPolicyTemplatePreference>
- <SecurityPolicyTemplatePreference Name="DSA-SHA1-C14N"
    ID="bb587fa0-b980-11d6-b8e9-c40beac518e7">
    <Category>XMLSignature</Category>
    <Preference>106</Preference>
  </SecurityPolicyTemplatePreference>
- <SecurityPolicyTemplatePreference Name="3DES-RSA-2048"
    ID="bb587fa0-b980-11d6-b8ea-c40beac518e7">
    <Category>XMLEncryption</Category>
    <Preference>107</Preference>
  </SecurityPolicyTemplatePreference>
- <SecurityPolicyTemplatePreference Name="AES-128-RSA-2048"
    ID="bb587fa0-b980-11d6-b8eb-c40beac518e7">
    <Category>XMLEncryption</Category>
    <Preference>108</Preference>
  </SecurityPolicyTemplatePreference>
- <SecurityPolicyTemplatePreference Name="DES-RSA-1024"
    ID="bb587fa0-b980-11d6-b8ec-c40beac518e7">
    <Category>XMLEncryption</Category>
    <Preference>109</Preference>
  </SecurityPolicyTemplatePreference>
</CommunitySecurityTemplatesPreference>
- <SAMsgSecurityPolicy>
  - <SAMsgPart PartName="Order" isOptional="false">
    - <PartSignatureAlgCategory>
        <XMLSignatureAlgCategory>XMLSignature</XMLSignatureAl
        gCategory>
      </PartSignatureAlgCategory>
    - <PartEncryptionAlgCategory>
        <XMLEncryptionAlgCategory>XMLEncryption</XMLEncryptio
        nAlgCategory>
      </PartEncryptionAlgCategory>
    </SAMsgPart>
  - <SAMsgPart PartName="Image" isOptional="false">
    - <PartEncryptionAlgCategory>
        <XMLEncryptionAlgCategory>XMLEncryption</XMLEncryptio
        nAlgCategory>
      </PartEncryptionAlgCategory>
    </SAMsgPart>
  </SAMsgSecurityPolicy>
- <PublicKeys>
```

```
<sicd:PartyID>x-
  ccns:commerceone.com:CollaborationParty::sellParty</sicd:Party
  ID>
- <sicd:EncryptionKeyInfo KeyOwner="x-
  ccns:commerceone.com:CollaborationParty::sellParty">
  <sicd:PublicKeyID>DefaultTestCert</sicd:PublicKeyID>
- <sicd:X509Data>
      <sicd:X509Certificate>LS0tLS1CRUdJTiBDRVJUSUZJQ0FU
      RS0tLS0tTUlJREZEQ0NBZnlnQXdJQkFnSUVQT0ZQSVRBT
      kJn
      a3Foa2lHOXcwQkFRVUZBREI2TVFzd0NRWURWUVFHRX
      dKVlV6RVZNQk1HQTFVRUNoTU1RMjl0YldW
      eVkyVWdUMjVsTVMwd0t3WURWUVFMRXlSVWFHbHpJR
      U5CSUdseklHWnZjaUIwWlhOMGFXNW5JSEIx
      Y25CdmMyVnpJRzl1YkhreEpUQWpCZ05WQkFNVUhFTnZ
      iVzFsY21OObElFOXVaU0JVWlhOMElFTkJJ
      Rkp2YjNRZ0I6RXdIaGNOTURJd05URTBNVGMxTXpNM1d
      oY05NRE13TlRFME1UWTFNek0zV2pCb01S
      Z3dGZ1lEVlFRREV3OUVZWFpwWkNCVVpYTjBJREl3TURJ
      eEVqQVFCZ05WQkFjVENNVTjFjR1Z5ZEds
      dWJ6RVVNQklHQTFVRUN4TUxSVzVuYVc1bFpYSnBibWN
      4RlRBVEJnTlZCQW9UREVOdmJXMWxjbU5s
      SUU5dVpURUxNQWtHQTFVRUJoTUNWVk13Z1o4d0RRW
      UpLb1pJaHZjTkFRRUJCUUFEZ1kwQU1JR0pB
      b0dCQU5nc2pTQkxjcFp2QnVDQ2lTTHR3RGFkaFZEMGNL
      RXJuQ3M2azg5UEhSUGJSMFdYOHBDUzBy
      ZWxIMkcyaDMxNU5vNGkzQVNidHZhYmdHellRVFNiR2Ez
      cWtNYmVLNDZTSGxtTkJOTUp2YUkvMmZV
      QlBxdkkzejlLTVJSTGh3eUhCMEdFNmUvSzdnVGZkSUo0M
      UJobTZzSmcwYzJqZ041cWt1d3FZQkV4
      eWN1MUFnTUJBQUdqT0RBMk1DY0dBMVVkRVFRZ01CN
      kJIRzE1VkdWemRFVnRZV2xzUUdOdmJXMWpa
      WEpqWlc5dVpTNWpiMjB3Q3dZRFZSMFBCQVFEQWdYZ0
      1BMEdDU3FHU0liM0RRRUJCUVVBQTRJQkFR
      Q0UrNEVaUWZYZWpmVnBsbXEzZnFtUjJZSGZhczErcXA0
      MUg4UWRmNmRESXBiYkZ2OUxocnorYkc2
      c2hWQlptMVpYVXphaHl6N2Q3Z2U3V0MxR2FZVjFHYldF
      TXJMUkZkeXM2c1VlQkZNbHZuNkZPRjNq
      OHdMY3JuN2FFN3pRMEMwa2U5LzVVNVBHTnlaWVaUG
      NLNTlKM0hPdWpzbXUvaENPVW1OOXZVM2M3
      MHVjMmhRaE96aExJQ0VlQ2VTRDFCd2hEMXNkdXZmNn
      VOanAzUGp2eUpCaklTeDVxY2UwS25oQmxp
      cDR3ejRNTWxpdEtTdkFXSElqR1Bvb0w0N01ac3I4N3RLa
      mJHaTgxcWJrQ3hiYlZ1dEloYmkzZDRn
      aW1Ockc1RXJ0dUUxNmwvRW9GUkJLU2VRTXd2cFdGUll
      iN2YreWtKVGE5ZVRLaWF4R2hOcDR4dnc5
      LS0tLS1FTkQgQ0VSVElGSUNBVEUtLS0tLQ==</sicd:X509
      Certificate>
  </sicd:X509Data>
  </sicd:EncryptionKeyInfo>
  <sicd:KeyTypeUsage>ENCRYPTION</sicd:KeyTypeUsage>
  <sicd:KeyTypeUsage>SIGNATURE</sicd:KeyTypeUsage>
  <sicd:KeyTypeUsage>AUTHENTICATION</sicd:KeyTypeUsage>
  </PublicKeys>
- <PublicKeys>
```

<sicd:PartyID>**PartyBSeller**</sicd:PartyID>
<sicd:EncryptionKeyInfo KeyOwner="**PartyBSeller**">
<sicd:PublicKeyID>**RKeyA**</sicd:PublicKeyID>
<sicd:X509Data>
<sicd:X509Certificate>**LS0tLS1CRUdJTiBDRVJUSUZJQ0FU**
**RS0tLS0tTUlJREZEQ0NBZnlnQXdJQkFnSUVQT0ZQSVRBT**
**kJna3Foa2lHOXcwQkFRVUZBREI2TVFzd0NRWURWUVFH**
**RXdKVlV6RVZNQk1HQTFVRUNoTU1RMjl0YldWeVkyVWd**
**UMjVsTVMwd0t3WURWUVFMRXlSVWFHbHpJRU5CSUdse**
**klHWnZjaUIwWlhOMGFXNW5JSEIxY25CdmMyVnpJRzl1**
**YkhreEpUQWpCZ05WQkFNVUhFTnZiVzFsY21ObElFOXVa**
**U0JVWlhOMElFTkJJRkp2YjNRZ0l6RXdIaGNOTURJd05UR**
**TBNVGMxTXpNM1doY05NRE13TlRFME1UWTFNek0zV2p**
**Cb01SZ3dGZ1lEVlFRREV3OUVZWFpwWkNCVVpYTjBJREl**
**3TURJeEVqQVFCZ05WQkFjVENVTjFjR1Z5ZEdsdWJ6RVV**
**NQklHQTFVRUN4TUxSVzVuYVc1bFpYSnBibWN4RlRBVEJ**
**nTlZCQW9UREVOdmJXMWxjbU5sSUU5dVpURUxNQWtH**
**QTFVRUJoTUNWVk13Z1o4d0RRWUpLb1pJaHZjTkFRUJ**
**CUUFEZ1kwQU1JR0pBb0dCQU5nc2pTQkxjcFp2QnVDQ2l**
**TTHR3RGFkaFZEMGNLRXJuQ3M2azg5UEhSUGJSMFdYOH**
**BDUzByZWxIMkcyaDMxNU5vNGkzQVNidHZhYmdHellRV**
**FNiR2EzcWtNYmVLNDZTSGxtTkJOTUp2YUkvMmZVQlBx**
**dkkzejlLTVJSTGh3eUhCMEdFNmUvSzdnVGZkSUo0MUJo**
**bTZzSmcwYzJqZ041cWt1d3FZQkV4eWN1MUFnTUJBQUd**
**qT0RBMk1DY0dBMVVkRVFRZ01CNkJIRzE1VkdWemRFV**
**nRZV2xzUUdOdmJXMWpaWEpqWlc5dVpTNWpiMjB3Q3d**
**ZRFZSMFBCQVFEQWdYZ01BMEdDU3FHU0liM0RRRUJCU**
**VVBQTRJQkFRQ0UrNEVaUWZYZWpmVnBsbXEzZnFtUjJZ**
**SGZhczErcXA0MUg4UWRmNmRESXBiYkZ2OUxocnorYkc**
**2c2hWQlptMVpYVXphaHl6N2Q3Z2U3V0MxR2FZVjFHYld**
**FTXJMUkZkeXM2c1VlQkZNbHZuNkZPRjNqOHdMY3JuN2**
**FFN3pRMEMwa2U5LzVVNVBHTnlaZWVaUGNLNTlKM0hP**
**dWpzbXUvaENPVW1OOXZVM2M3MHVjMmhRaE96aExJQ**
**0VlQ2VTRDFCd2hEMXNkdXZmNnVOanAzUGp2eUpCaklT**
**eDVxY2UwS25oQmxpcDR3ejRNTWxpdEtTdkFXSElqR1Bv**
**b0w0N01ac3I4N3RLamJHaTgxcWJrQ3hiYlZ1dEloYmkzZ**
**DRnaW1Ockc1RXJ0dUUxNmwvRW9GUkJLU2VRTXd2cFd**
**GUlliN2YreWtKVGE5ZVRLaWF4R2hOcDR4dnc5LS0tLS1F**
**TkQgQ0VSVElGSUNBVEUtLS0tLQ==**</sicd:X509Certificate>
</sicd:X509Data>
</sicd:EncryptionKeyInfo>
<sicd:KeyTypeUsage>**ENCRYPTION**</sicd:KeyTypeUsage>
<sicd:KeyAlgorithm>**RSA**</sicd:KeyAlgorithm>
<sicd:Description>**String**</sicd:Description>
<sicd:Location>**String**</sicd:Location>
</PublicKeys>
<PublicKeys>
<sicd:PartyID>**ConnectorB**</sicd:PartyID>
<sicd:EncryptionKeyInfo KeyOwner="**BOwner**">
<sicd:PublicKeyID>**RKeyB**</sicd:PublicKeyID>
<sicd:X509Data>
<sicd:X509Certificate>**LS0tLS1CRUdJTiBDRVJUSUZJQ0FU**
**RS0tLS0tTUlJREZEQ0NBZnlnQXdJQkFnSUVQT0ZQSVRBT**

kJna3Foa2lHOXcwQkFRVUZBREI2TVFzd0NRWURWUVFH
RXdKVlV6RVZNQk1HQTFVRUNoTU1RMjl0YldWeVkyVWd
UMjVsTVMwd0t3WURWUVFMRXlSVWFHbHpJRU5CSUdse
klHWnZjaUIwWlhOMGFXNW5JSEIxY25CdmMyVnpJRzl1
YkhreEpUQWpCZ05WQkFNVUhFTnZiVzFsY21ObElFOXVa
U0JVWlhOMElFTkJJRkp2YjNRZ0l6RXdIaGNOTURJd05UR
TBNVGMxTXpNM1doY05NRE13TlRFME1UWTFNek0zV2p
Cb01SZ3dGZ1lEVlFRREV3OUVZWFpwWkNCVVpYTjBJREl
3TURJeEVqQVFCZ05WQkFjVENVTjFjR1Z5ZEdsdWJ6RVV
NQklHQTFVRUN4TUxSVzVuYVc1bFpYSnBibWN4RlRBVEJ
nTlZCQW9UREVOdmJXMWxjbU5sSUU5dVpURUxNQWtH
QTFVRUJoTUNWVk13Z1o4d0RRWUpLb1pJaHZjTkFRUJ
CUUFEZ1kwQU1JR0pBb0dCQU5nc2pTQkxjcFp2QnVDQ2l
TTHR3RGFkaFZEMGNLRXJuQ3M2azg5UEhSUGJSMFdYOH
BDUzByZWxIMkcyaDMxNU5vNGkzQVNidHZhYmdHellRV
FNiR2EzcWtNYmVLNDZTSGxtTkJOTUp2YUkvMmZVQlBx
dkkzejlLTVJSTGh3eUhCMEdFNmUvSzdnVGZkSUo0MUJo
bTZzSmcwYzJqZ041cWt1d3FZQkV4eWN1MUFnTUJBQUd
qT0RBMk1DY0dBMVVkRVFRZ01CNkJIRzE1VkdWemRFV
nRZV2xzUUdOdmJXMWpaWEpqWlc5dVpTNWpiMjB3Q3d
ZRFZSMFBCQVFEQWdYZ01BMEdDU3FHU0liM0RRRUJCU
VVBQTRJQkFRQ0UrNEVaUWZYZWpmVnBsbXEzZnFtUjJZ
SGZhczErcXA0MUg4UWRmNmRESXBiYkZ2OUxocnorYkc
2c2hWQIptMVpYVXphaHl6N2Q3Z2U3V0MxR2FZVjFHYld
FTXJMUkZkeXM2c1VlQkZNbHZuNkZPRjNqOHdMY3JuN2
FFN3pRMEMwa2U5LzVVNVBHTnlaZWVaUGNLNTlKM0hP
dWpzbXUvaENPVW1OOXZVM2M3MHVjMmhRaE96aExJQ
0VlQ2VTRDFCd2hEMXNkdXZmNnVOanAzUGp2eUpCaklT
eDVxY2UwS25oQmxpcDR3ejRNTWxpdEtTdkFXSElqR1Bv
b0w0N01ac3I4N3RLamJHaTgxcWJrQ3hiYlZ1dEloYmkzZ
DRnaW1Ockc1RXJ0dUUxNmwvRW9GUkJLU2VRTXd2cFd
GUlliN2YreWtKVGE5ZVRLaWF4R2hOcDR4dnc5LS0tLS1F
TkQgQ0VSVElGSUNBVEUtLS0tLQ==
</sicd:X509Data>
</sicd:EncryptionKeyInfo>
<sicd:KeyTypeUsage>**SIGNATURE**</sicd:KeyTypeUsage>
<sicd:KeyTypeUsage>**ENCRYPTION**</sicd:KeyTypeUsage>
<sicd:KeyAlgorithm>**RSA**</sicd:KeyAlgorithm>
<sicd:Description>**String**</sicd:Description>
<sicd:Location>**String**</sicd:Location>
</PublicKeys>
- <ConnectorCapability ConnectorName="**x-ccns:cup.commerceone.com:connector::centerSell**">
<EncryptionCapability>**true**</EncryptionCapability>
<SignatureCapability>**true**</SignatureCapability>
</ConnectorCapability>
- <ConnectorCapability ConnectorName="**x-ccns:cup.commerceone.com:connector::sell**">
<EncryptionCapability>**false**</EncryptionCapability>
<SignatureCapability>**true**</SignatureCapability>
</ConnectorCapability>
- <CPRecvServicesSecurityPolicy>
- <SecurityPolicyTemplatePreference Name="**3DES-RSA-2048**">
<Category>**XMLEncryption**</Category>
<Preference>**1**</Preference>

```
        </SecurityPolicyTemplatePreference>
      - <SecurityPolicyTemplatePreference Name="RSA-MD5-C14N">
          <Category>XMLSignature</Category>
          <Preference>2</Preference>
        </SecurityPolicyTemplatePreference>
      - <SecurityPolicyTemplatePreference Name="RSA-SHA1-C14N">
          <Category>XMLSignature</Category>
          <Preference>6</Preference>
        </SecurityPolicyTemplatePreference>
      - <SecurityPolicyTemplatePreference Name="AES-128-RSA-2048">
          <Category>XMLEncryption</Category>
          <Preference>5</Preference>
        </SecurityPolicyTemplatePreference>
      - <ServiceAuthentication>
          <AcceptedCredentials>X509</AcceptedCredentials>
          <AcceptedCredentials>BASIC</AcceptedCredentials>
          <sicd:AuthenticateMode>SOURCE</sicd:AuthenticateMode>
        </ServiceAuthentication>
      </CPRecvServicesSecurityPolicy>
    </SecurityReceiverInfo>
```

# ComputeSecurityContract.XML

```
<?xml version="1.0" ?>
- <prefix_0:SecurityContractICD
    xmlns:prefix_0="publicid:com.commerceone:schemas/soapextension/contract/security/v1_0/SecurityContract.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  - <prefix_0:SecurityPolicies>
    - <prefix_0:AuthenticationPolicies>
      - <prefix_0:X509CredentialPolicy PolicyId="P-AuthenX.509Source">
          <prefix_0:CredentialPolicyAlgorithm>X.509v3</prefix_0:CredentialPolicyAlgorithm>
          <prefix_0:AuthenticateMode>SOURCE</prefix_0:AuthenticateMode>
        </prefix_0:X509CredentialPolicy>
      </prefix_0:AuthenticationPolicies>
    - <prefix_0:SignaturePolicies>
      - <prefix_0:XMLDsigPolicy PolicyId="P-XMLSignatureRSA-MD5-C14N">
          <prefix_0:SignaturePolicyAlgorithm>http://www.w3.org/2000/09/xmldsig#</prefix_0:SignaturePolicyAlgorithm>
          <prefix_0:SignatureAlgorithm>MD5withRSA</prefix_0:SignatureAlgorithm>
          <prefix_0:HashFunction>MD5</prefix_0:HashFunction>
          <prefix_0:CanonicalizationMethod>http://www.w3.org/TR/2000/CR-xml-c14n-20001026</prefix_0:CanonicalizationMethod>
          <prefix_0:Transform>http://msdn.microsoft.com/ws/2002/01/Security#RoutingSignatureTransform</prefix_0:Transform>
        </prefix_0:XMLDsigPolicy>
      </prefix_0:SignaturePolicies>
    - <prefix_0:EncryptionPolicies>
      - <prefix_0:XMLEncryptionPolicy PolicyId="P-XMLEncrypt3DES-RSA-2048">
          <prefix_0:EncryptionPolicyAlgorithm>http://www.w3.org/2001/04/xmlenc#</prefix_0:EncryptionPolicyAlgorithm>
          <prefix_0:EncryptionMethod>http://www.w3.org/2001/04/xmlenc#3des-cbc</prefix_0:EncryptionMethod>
          <prefix_0:KeySize>2048</prefix_0:KeySize>
          <prefix_0:KeyEncryptionMethod>http://www.w3.org/2001/04/xmlenc#rsa-1_5</prefix_0:KeyEncryptionMethod>
        </prefix_0:XMLEncryptionPolicy>
      </prefix_0:EncryptionPolicies>
    - <prefix_0:EncryptionKeyInfo KeyOwner="x-ccns:commerceone.com:CollaborationParty::sellParty">
        <prefix_0:PublicKeyID>DefaultTestCert</prefix_0:PublicKeyID>
```

```
- <prefix_0:X509Data>
    <prefix_0:X509Certificate>LS0tLS1CRUdJTiBDRVJUSUZJQ
    0FURS0tLS0tTUlJREZEQ0NBZnlnQXdJQkFnSUVQT0ZQSV
    RBTkJn
    a3Foa2lHOXcwQkFRVUZBREI2TVFzd0NRWURWUVFHRX
    dKVlV6RVZNQk1HQTFVRUNoTU1RMjl0YldW
    eVkyVWdUMjVsTVMwd0t3WURWUVFMRXlSVWFHbHpJR
    U5CSUdsekIHWnZjaUIwWlhOMGFXNW5JSEIx
    Y25CdmMyVnpJRzl1YkhreEpUQWpCZ05WQkFNVUhFTnZ
    iVzFsY21ObElFOXVaU0JVWlhOMElFTkJJ
    Rkp2YjNRZ0l6RXdIaGNOTURJd05URTBNVGMxTXpNM1d
    oY05NRE13TlRFME1UWTFNek0zV2pCb01S
    Z3dGZ1lEVlFRREV3OUVZWFpwWkNCVVpYTjBJREl3TURJ
    eEVqQVFCZ05WQkFjVENVTjFjR1Z5ZEds
    dWJ6RVVNQklHQTFVRUN4TUxSVzVuYVc1bFpYSnBibWN
    4RlRBVEJnTlZCQW9UREVOdmJXMWxjbU5s
    SUU5dVpURUxNQWtHQTFVRUJoTUNWVk13Z1o4d0RRW
    UpLb1pJaHZjTkFRRUJCUUFEZ1kwQU1JR0pB
    b0dCQU5nc2pTQkxjcFp2QnVDQ2lTTHR3RGFkaFZEMGNL
    RXJuQ3M2azg5UEhSUGJSMFdYOHBDUzBy
    ZWxIMkcyaDMxNU5vNGkzQVNidHZhYmdHellRVFNiR2Ez
    cWtNYmVLNDZTSGxtTkJOTUp2YUkvMmZV
    QlBxdkkzejlLTVJSTGh3eUhCMEdFNmUvSzdnVGZkSUo0M
    UJobTZzSmcwYzJqZ041cWt1d3FZQkV4
    eWN1MUFnTUJBQUdqT0RBMk1DY0dBMVVkRVFRZ01CN
    kJIRzE1VkdWemRFVnRZV2xzUUdOdmJXMWpa
    WEpqWlc5dVpTNWpiMjB3Q3dZRFZSMFBCQVFEQWdYZ0
    1BMEdDU3FHU0liM0RRRUJCUVVBQTRJQkFR
    Q0UrNEVaUWZYZWpmVnBsbXEzZnFtUjJZSGZhczErcXA0
    MUg4UWRmNmRESXBiYkZ2OUxocnorYkc2
    c2hWQlptMVpYVXphaHl6N2Q3Z2U3V0MxR2FZVjFHYldF
    TXJMUkZkeXM2c1VlQkZNbHZuNkZPRjNq
    OHdMY3JuN2FFN3pRMEMwa2U5LzVVNVBHTnlaZWVaUG
    NLNTlKM0hPdWpzbXUvaENPVW1OOXZVM2M3
    MHVjMmhRaE96aExJQ0VlQ2VTRDFCd2hEMXNkdXZmNn
    VOanAzUGp2eUpCaklTeDVxY2UwS25oQmxp
    cDR3ejRNTWxpdEtTdkFXSElqR1Bvb0w0N01ac3I4N3RLa
    mJHaTgxcWJrQ3hiYlZ1dEloYmkzZDRn
    aW1Ockc1RXJ0dUUxNmwvRW9GUkJLU2VRTXd2cFdGUll
    iN2YreWtKVGE5ZVRLaWF4R2hOcDR4dnc5
    LS0tLS1FTkQgQ0VSVElGSUNBVEUtLS0tLQ==</prefix_0:
    X509Certificate>
  </prefix_0:X509Data>
 </prefix_0:EncryptionKeyInfo>
</prefix_0:SecurityPolicies>
- <prefix_0:SecurityChannel channelId="CHANNEL1" sourceConnector="x-
  ccns:cup.commerceone.com:connector::buy" targetConnector="x-
  ccns:cup.commerceone.com:connector::sell">
 - <prefix_0:Credential AlgorithmId="P-AuthenX.509Source"
    SequenceID="4" DelegationFlag="false">
    <prefix_0:PublicKeyName>BuyerPublicKey</prefix_0:PublicKeyN
    ame>
  </prefix_0:Credential>
 - <prefix_0:Integrity AlgorithmId="P-XMLSignatureRSA-MD5-C14N">
```

```
        <prefix_0:PublicKeyName
          KeyOwner="OwnerA">BuyerPublicKey</prefix_0:PublicKeyNam
          e>
        <prefix_0:MessagePart PartName="Order" isOptional="false" />
      </prefix_0:Integrity>
    </prefix_0:SecurityChannel>
  - <prefix_0:SecurityChannel channelId="CHANNEL2" sourceConnector="x-
      ccns:cup.commerceone.com:connector::centerSell"
      targetConnector="x-
      ccns:cup.commerceone.com:connector::centerSell">
    - <prefix_0:Confidential AlgorithmId="P-XMLEncrypt3DES-RSA-2048">
        <prefix_0:PublicKeyName KeyOwner="x-
          ccns:commerceone.com:CollaborationParty::sellParty">Defa
          ultTestCert</prefix_0:PublicKeyName>
        <prefix_0:MessagePart PartName="Order" isOptional="false" />
        <prefix_0:MessagePart PartName="Image" isOptional="false" />
      </prefix_0:Confidential>
    </prefix_0:SecurityChannel>
</prefix_0:SecurityContractICD>
```

## CLAIMS

1. A method of dynamically determining security options for exchange of at least one message between services, the message having one or more parts, the method including:

providing machine readable security profiles for first and second services, wherein the security profiles identify a plurality of security elements that are acceptable to the respective services and the security elements include:

requirements to sign one or more parts of the message;

requirements to encrypt one or more parts of the message;

one or more signing option subsets for the signing algorithm including a signing algorithm and to be applied to one or more part of the message;

one or more encryption option subsets for the encryption algorithm including an encryption algorithm to be applied to one or more parts of the message;

one or more signature keys to use with the signature algorithm;

one or more encryption keys to use with the encryption algorithm;

at least one authentication algorithm to be applied to one or more parts of the message;

accessing the security profiles and selecting a particular set of the security elements for the message that is acceptable to the respective services; and

communicating the message between the respective services compliant with the particular option set.

2. The method of claim 1, wherein the security profiles are maintained in a registry accessible to security logic of the first and second services.

3. The method of claim 1, wherein one or more of the security elements are specified by a default in a machine readable default security profile.

4. The method of claim 1, wherein the requirements to sign are applied to individual parts of the message.

5. The method of claim 1, wherein the requirements to sign are applied to the message as a whole.

6. The method of claim 1, wherein the requirements to encrypt are applied to individual parts of the message.

7. The method of claim 1, wherein the requirements to encrypt are applied to the message as a whole.

8. The method of claim 1, wherein the signature algorithm is applied to the message as a whole.

9. The method of claim 1, wherein the encryption algorithm is applied to the message as a whole.

10. The method of claim 1, wherein the signature and encryption keys are asymmetrical.

11. The method of claim 1, wherein the encryption keys are symmetrical.

12. The method of claim 1, wherein the authentication algorithm is carried out by a trusted agent before communicating the message and evidenced by an authentication assertion.

13. The method of claim 1, wherein the authentication algorithm includes submitting credentials accompanying the message for examination by the service receiving the message.

14. The method of claim 1, wherein the security elements further include identification of at least one authorization algorithm to establish a sending service's privileges.

15. The method of claim 14, wherein the authorization algorithm is carried out by a trusted agent before communicating the message and evidenced by an authorization assertion.

16. The method of claim 14, wherein the authentication algorithm includes submitting credentials accompanying the message for examination by the service receiving the message.

17. The method of claim 1, wherein the security profiles further include statements of preferences among the signing and encryption security elements and selecting the particular option subset takes into account the preferences of at least one of the services.

18. The method of claim 17, wherein selecting the particular option subset corresponds to the option subset that is acceptable to the respective services and most preferred by the service receiving the message.

19. The method of claim 17, wherein selecting the particular option subset corresponds to the option subset that is acceptable to the respective services and most preferred by the service sending the message.

20. The method of claim 17, wherein selecting the particular option subset takes into account the preferences of both of the services.

21. The method of claim 17, wherein selecting the particular option subset takes into account a highest security level among the security elements that are acceptable to the respective services.

22. The method of claim 17, wherein selecting the particular option subset takes into account a lowest security level among the security elements that are acceptable to the respective services.

23. The method of claim 17, wherein selecting between requirements to sign or encrypt one or more parts of the message takes into account the preferences of at least one of the services.

24. The method of claim 17, wherein selecting between requirements to sign or encrypt one or more parts of the message corresponds to the option subset that is acceptable to the respective services and most preferred by the service receiving the message.

25. The method of claim 17, wherein selecting between requirements to sign or encrypt one or more parts of the message corresponds to the option subset that is acceptable to the respective services and most preferred by the service sending the message.

26. The method of claim 17, wherein selecting between requirements to sign or encrypt one or more parts of the message takes into account the preferences of both of the services.

27. The method of claim 17, wherein selecting between requirements to sign or encrypt one or more parts of the message takes into account a highest security level among the security elements that are acceptable to the respective services.

28. The method of claim 17, wherein selecting between requirements to sign or encrypt one or more parts of the message takes into account a lowest security level among the security elements that are acceptable to the respective services.

29. The method of claim 1, wherein the security profiles further include one or more resources used by the respective services to implement signatures and encryption.

30. The method of claim 17, wherein the security profiles further include one or more resources used by the respective services to implement signatures and encryption.

31. The method of claim 1, wherein the security profiles further include one or more resources used to authenticate the service sending the message.