Creating A Single Global Electronic Market

ebXML Technical Architecture Risk Assessment v1.0

ebXML Security Team

May 10, 2001

1 Status of this Document

There are three categories of ebXML deliverables:

- *Technical Specifications* conform to the ebXML Requirements document.
- *Technical Reports* are either guidelines or catalogues.
- *White Papers* constitute a snapshot of on-going work within a Project Team.

This Technical Report has been approved by the ebXML Technical Architecture Security Team and has been accepted by the ebXML Plenary.

This document contains information to guide in the interpretation or implementation of ebXML.

Distribution of this document is unlimited.

Note: Implementers should consult the ebXML web site for current status and revisions to all specifications (http://www.ebxml.org) .

This version:

www.ebxml.org/specs/secRISK.pdf

***Latest version*:**

www.ebxml.org/specs/secRISK.pdf

2 ebXML Participants

The authors would like to acknowledge the support of the Security Team who contributed ideas to this document by the group’s discussion email list, on conference calls and during the face-to-face meetings.

Zahid Ahmed, CommerceOne
Igor Balabine, NetFish
Ralph Berwanger, bTrade
Al Boseman, ATPCO
Allen Brown, Microsoft
Paul Bussey, Cyclone Commerce
Eva Cheng, Chinatrust Commercial Bank
Gary Crough, Cyclone Commerce
Hatem ElSebaaly, IPNet
Chris Ferris, Sun
Maryann Hondo, IBM
Eric Klein, Softshare
Dale Moberg, Sterling
Per Myrseth, PKI Consulting Services
Farrukh Najmi, Sun
Rich Salz, Zolera Systems
Krishna Sankar, Cisco
Amlan Sengupta, Sun
Mark Scherling, RSA Securities
Jeff Turpin, Cyclone Commerce
Jenny Xu, Great Wall Technology LLC

3 Table of Contents

4 Executive Overview

We live in interesting times. The further we move toward opening our borders both in a social sense and a business sense, the more we expose ourselves to risk. E-Business technology, like any new technology reflects this environment, and risk is inevitable. But, while there may still be much security work to be done, we should recall the words of one keynote speaker at a recent security conference:

> *The reason not to panic is that we have to accept the poor state of security and work to mitigate the risk of attacks rather than try to prevent attacks altogether -- an impossible task. Technology is not the enemy of security. It's only a tool, one that hasn't been used very well.*

ebXML is an attempt to open borders to global business. Given the limited time frame it faced, the security team decided early on that the most productive role to take would be two-fold:

- First, work with liaisons from the different working groups to discuss and identify security issues within the working group context; and
- Second, provide an initial risk assessment of the technical architecture to identify security issues that exist across groups or totally outside the existing group structure.

This document is the result of that work. The effort has exposed some risks within ebXML, exactly as was the intent of the exercise. While it would have been nice to have found that ebXML is risk-free, we know this would be naive: all real systems have risks associated with them. The risks that have been identified are risks that exist in the broader internet business environment today and should be viewed in this context. To get to the point of having secure e-business, means you have to start somewhere[1]. Classic advice in the security field is to start by securing the weakest link, then address the next link, and so on. This is the first step for ebXML: knowing how things stand. A valuable next step would be to integrate the information from the risk assessment as requirements into any ongoing activities for the respective working groups.

There are well-known security technologies that can be used by implementers of the ebXML specifications to provide a base level of security between any two ebXML partners. SSL and S/MIME are the primary candidates for providing confidentiality and authentication of endpoints. XML Digital Signatures can provide data integrity on messages, and existing authentication and authorization schemes are available to registry providers to enforce access control over data kept in the repository. Aside from XML Digital Signatures, these are the same mechanisms that are found in most web based service models today.

The bulk of the risks exist in the area of:

[1] Figure 1. in [BS7799-2], step 3 undertake a risk assessment.

- Dynamic business process definition
- Service discovery
- Negotiation.

This can be attributed to the immaturity of the technology.

Knowing where you are is often half the problem, and that's what this document tries to show.

5 Introduction

This document describes security issues present in the ebXML technical architecture as defined by the ebXML specifications listed in Section 5.3. It provides a high level overview of the security issues in the relationships, interactions, and basic functionality of the ebXML architectural components.

5.1 Audience

Security architects and implementers should use it as a roadmap to learn:

1. What risks are present in the ebXML architecture

2. What problems the ebXML security recommendations and profiles can help solve; and

3. Perhaps most importantly, what security issues are yet to be addressed.

5.2 Scope

The security issues raised here should be considered when reviewing the design or implementation of an ebXML application. This document alone does not provide all the details required to build a secure ebXML application. Please refer to each of the ebXML component specifications listed in Section 5.3 Related Documents and the related reference specifications listed in the References for more details.

One of the difficulties in integrating security into a set of specifications that are being developed in parallel is that it potentially results in additional concepts needing to be addressed in a future iteration of the architecture or one of its components. In this document components of the architecture are reviewed and recommendations to address unresolved issues from a security perspective are identified and summarized in Section 15 .

5.3 Related Documents

This risk analysis considered the following ebXML Specifications on the following topics:

EbXML Collaboration Protocol Profile and Agreement Specification v0.91 [ebCPP]
EbXML Message Service Interface Specification v 0.93[ebMS]
EbXML Registry and Repository Specification v0.84[ebRS]
EbXML Technical Architecture [ebTA]
EbXML Business Process Spesification Schema [ebBPSS]

6 Design Objectives

6.1 Problem Description & Goals for ebXML Security

Implicit in business exchanges is the notion of trust. Two entities engage in a business relationship with the expectation that each party will fulfill their part of their business agreement. Without this fundamental understanding there could be no exchange.

The companies that have implemented *Electronic Data Interchange (EDI)* agreed to implement common middleware that requires a significant investment to provide the assurance of secure transactions. Within the overall the business world, only a small percentage of companies are using EDI; consequently, *Common Business Processes* are dominated by paper transactions. Alternative standards in this area are emerging, but at this time it is not possible to provide a complete security architecture for electronic commerce based on open standards.

Network and system manufacturers are currently moving towards policy-based management. This is driven partly by the influence of large organizations such as ISPs and ASPs and partly by their own need to facilitate the management of large implementations of networks and systems. In providing a complete risk assessment it is important to consider this trend.

The left side of the picture below, Figure 1, attempts to illustrate how individual applications today are developed in isolation and the information and security for each is left within the application domain. This means that security decisions are closely tied to the application and it is difficult to grow or change the security infrastructure without requiring a rewrite of the application itself.

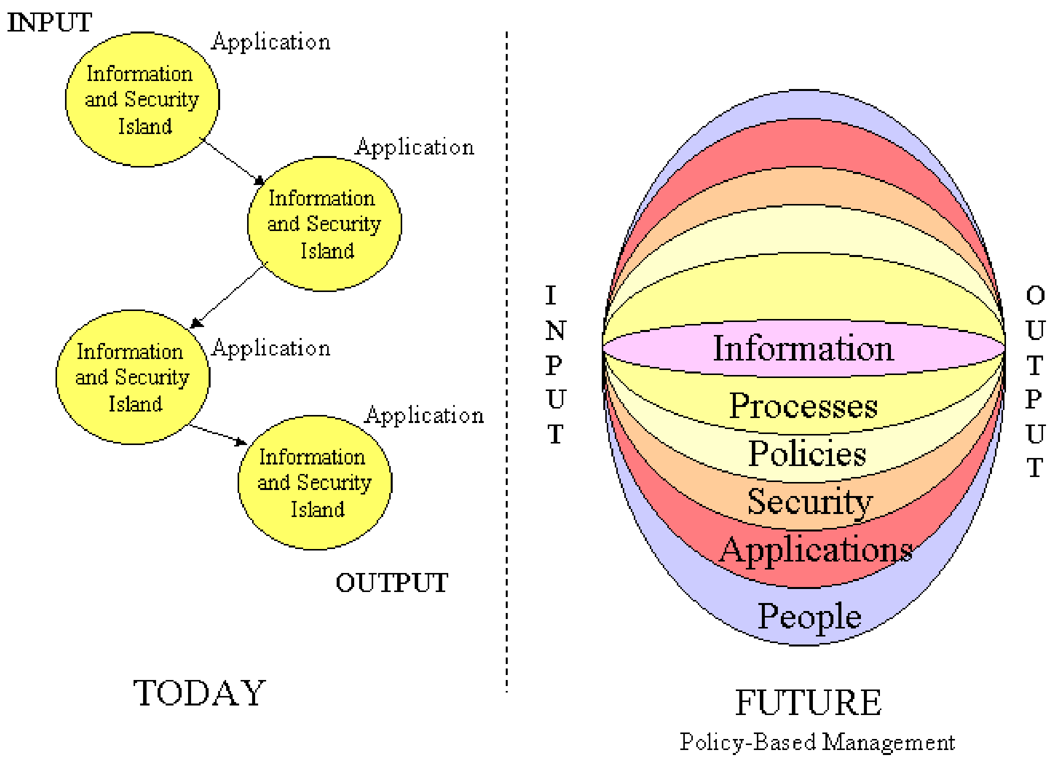

Figure 1. Future for Policy-driven Security

The right side of the picture illustrates a more modular approach. In a Policy-Based Management scheme, the emphasis is on building a layered infrastructure so that the application can specify security requirements in terms of the business need. The entities responsible for the infrastructure and management can then make the appropriate decisions for mapping the application requirements into the environments security capabilities and mechanisms.

This document attempts to begin a conceptual layering of ebXML applications. It translates the business need for trust captured by the *Business Process and Information Meta Model* into a set of risk assertions that can be addressed using standard security technologies. The document also identifies emerging standards that offer the potential for additional levels of security in the future.

This document describes security for ebXML in two dimensions. First, there are security technologies available that have been identified in some of the ebXML project specifications (Business Process, Trading Partners, Registry & Repository, and Transport Routing & Packaging). This process is similar to the isolation model. Each project is addressing security within a narrow scope and demonstrating their individual piece of ebXML. Second, there are security risks that need to be addressed across layers of ebXML architectural components in any implementation of the ebXML architecture. In the process of performing this risk assessment, we introduce the notion of layering security.

A set of security risks have been documented in the following Section 7, ebXML Risks. Implementers should use the references cited to provide a complete risk assessment of their implementation.

7 ebXML Risks

Within any organization there exist vulnerabilities or risks that must be mitigated or reduced to an acceptable level in order for the organization to perform business functions. The following list identifies key risks for ebXML:

- Unauthorized transactions and fraud – The benefit of human experience in identification of unusual or inconsistent transactions is reduced with e-transactions. This automation of transactions may present more risk to businesses by increasing the number of opportunities to change an entity's computer records and/or those of the entity's trading partners which could cause or allow fraud to be perpetrated. In the automated payment generation area, the manipulation or diversion of payments, payment generation in error or the inappropriate timing of payments (funds not in place or payment delivered too early) are an increasing risk to business.
- Loss of confidentiality – Sensitive information may be inadvertently or deliberately disclosed on the network. External parties might gain information about transactions or specific entity knowledge without the primary party's knowledge.
- Error detection (application, network/transport, platform) – Errors in processing and communications systems may result in the transmission of incorrect trading information or inaccurate reporting. Application errors can result in significant losses to trading partners and potential business losses.
- Potential loss of management and audit – There is the potential for the loss of data if proper controls are not implemented. Policies for retention of data are also an issue. EDI transaction data are normally maintained for long periods of time and without consideration of legal and audit issues the parties may not be able to provide adequate or appropriate evidence.
- Potential legal liability – the legislation for the legality of electronic transactions and records are still being created. Although legal precedence has been set for the use of digital signatures in the US and other countries, there are still a number of countries that do not have any legislation in place for dealing with electronic information . Without proven audit and control, the presentation and admissibility of electronic evidence is still immature and inconsistent between jurisdictions.

The major categories of security risks and some countermeasures for ebXML are briefly defined and then categorized in the matrix below.

A more complete view of information security management which is covered in [BS-7799/ISO-17799] including all the aspect of risks need to be measured and controlled to establish a security management framework.

Risk Category	Risk element	Currently Availabel Conter measure	Emerging Technology for Counter measures
Unauthorized transactions and fraud	Identification	Biometrics (physical); electronic (userid and password, token, certificate; notarized documents	SAML[SAML]
	Authentication	Userid and password; PKI; token; biometrics;	SAML
	Authorization	RBAC; delegated;	SAML
	Non-repudiation of origin	XML-DSIG; PKI; paper; policies and procedures including audit and control	
	Non-repudiation of receipt	AS1, AS2, MDN[EDI] ebXML TRP persistent signed receipt plus policies and procedures	
	Secure timestamp	Notary; signed audit logs;	

Risk Category	Risk element	Currently Availabel Conter measure	Emerging Technology for Counter measures
Loss of Confidentiality	Application	SMIME/PGP policies and procedures including audit and control	
	Message	SMIME/PGP policies and procedures including audit and control	XML Encryption [XMLENC]
	Transport	SSL; TLS	
		VPN	
		policies and procedures including audit and control	

[EDI] http://www.ietf.org/internet-drafts/draft-ietf-ediint-as1-12.txt , http://www.ietf.org/internet-drafts/draft-ietf-ediint-as2-09.txt

Risk Categry		Risk element	Currently Availabel Conter measure	Emerging Technology for Counter measures
Error Detection	Application	Virus	Anti-virus software plus policies and procedures	
		Improper configuration	Configuration management; policies and procedures including audit and control	
		Improper use	Testing and code reviews	
	Network/ MessageLevel	Virus	Anti-virus software plus policies and procedures	
		Denial of Service		
		Intrusion detection	Intrusion detection software	
		Subversion		
		Protocol-level attacks		
	Network/ Transport Level	Improper configuration	Configuration management; policies and procedures including audit and control	
		Denial of Service	policies and procedures including audit and control	
	Platform	Virus	Anti-virus software plus policies and procedures	
		Improper configuration	policies and procedures including audit and File Access Control; Server Security; Backup and archive; CERT based safe operating practices[2]	

[2] CERT® Coordination Center (CERT/CC), www.cert.org

Risk Category	Risk element	Currently Availabel Conter measure	Emerging Technology for Counter measures
Potential loss of Management and Audit	Electronic evidence	policies and procedures including audit and control; backup and archival; demonstrable secure processing	WebTrust Principles and criteria for Certificate Authorities AICPA/CICA; PKI Assessment Guidelines (PAG) ABA (two guidelines for assessing and facilitating interoperability of PKIs)
	Key management	policies and procedures including audit and control; CA	XKMS[XKMS]

Risk Category	Risk element	Currently Availabel Conter measure	Emerging Technology for Counter measures
Potential Legal Liability		policies and procedures including audit and control	

Figure 2. Risk Matrix

8 ebXML Security Overview

The *Business Process* is ultimately what defines a need for security. The security process often becomes a morass of details and technical discussion. At the root of it all is some business requirement for security, often expressed as a desire to lessen a particular risk or exposure. The current discussions on security revolve mostly around separate security mechanisms such as encryption and signing. Questions arise such as: is it necessary for confidentiality to encrypt the manifest as well as the payload? There are many such questions, and it is difficult to determine what the business process requires based on a simple desire to apply or not apply a particular security mechanism.

The pictures and text below attempt to capture the relationship between the security elements and the ebXML Technical Architecture components: Business Process, Trading Partners, Registry & Repository, and Transport Routing & Packaging.

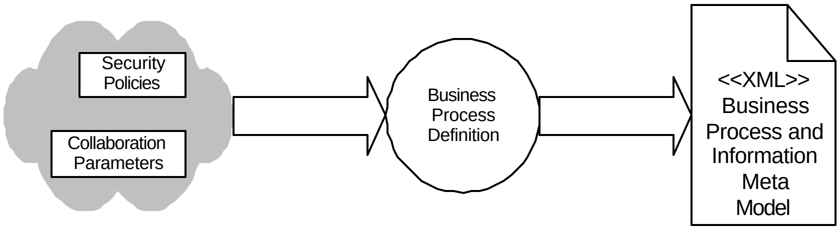
Security Policies
Collaboration Parameters
Business Process Definition
<<XML>> Business Process and Information Meta Model

Figure 3. BP defines security characteristics

The Business Process (BP) definition phase attempts to capture security characteristics of business process collaboration at a relatively high level (Figure3). In the current ebXML flow, the information model is then translated into an XML representation and combined with other environmental information.

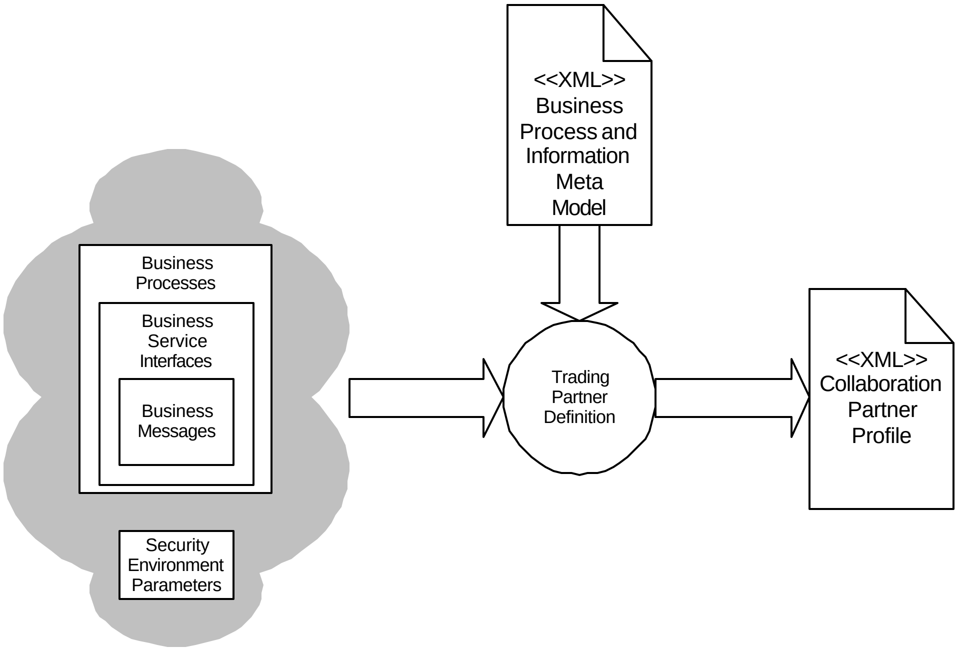
<<XML>> Business Process and Information Meta Model
Business Processes
Business Service Interfaces
Business Messages
Security Environment Parameters
Trading Partner Definition
<<XML>> Collaboration Partner Profile

Figure 4. CPP is crafted from different inputs

The generation of the *Collaboration Protocol Profile* (CPP) is driven by the *Business Process Information Meta Model* (and contains a reference to the model in its structure) but is not completely an automatic process. Figure 4 attempts to capture this by identifying a step called the "trading partner definition". For the ebXML architecture to move towards supporting policy-based management, it will require further work in this area to model security practices and services as well as applications. In the CPP, the business requirement for providing secure transport becomes an XML element called **`secureTransport,`** and the business requirement for security characteristics becomes an XML attribute called **`Characteristics`** under the **`DeliveryChannel`** element as indicated in the XML fragment below.

```
<DeliveryChannel >
      <Characteristics
```

```
            nonrepudiationOfOrigin=''false''
            nonrepudiationOfReceipt=''false''
            secureTransport=''true''
            confidentiality=''false''
            authenticated=''false''
            authorized=''false''
      />

</DeliveryChannel>
```

This sub-element of a **DeliveryChannel** then indicates that certain additional elements within the CPP must be defined to provide the details on how secure transport is to be provided. Following the example, if the security attribute **secureTransport** is indicated in the CPP, then the **Transport** element of the CPP might contain details like the following fragment:

```
<Transport transportId="N12">
       <Protocol version="1.1">HTTP</Protocol>
            <Endpointuri=https://www.ebxmlregisterservices.org/asynch
            type="request"/>
      <TransportSecurity>
            <Protocol version="1.0">TLS</Protocol>
            <CertificateRef certId="N05"/>
      </TransportSecurity>
<Transport>
```

The CPP can also define different levels at which security may be present. For example, the Document Exchange Section of the CPP might include tags for an *ebXML binding* [ebCPP]. An ebXML binding contains elements for describing reliable messaging and non-repudiation that contains a reference to a **Certificate** structure that references the key used to sign an ebXML document [XMLDSIG][3]. Security can also be defined at the transport level (e.g. SSL via TLS). These patterns can be combined within the CPP document.

Once a CPP has been defined, it may be stored in the ebXML compliant Registry & Repository (See Figure 5). When business partner A wishes to collaborate with business partner B, it locates the CPP for partner B and the two parties engage in a process of negotiating an agreement based on matching complimentary items in the two profiles. The end result of this negotiation is a *Collaboration Protocol Agreement* (CPA) document. Currently this is a manual process.

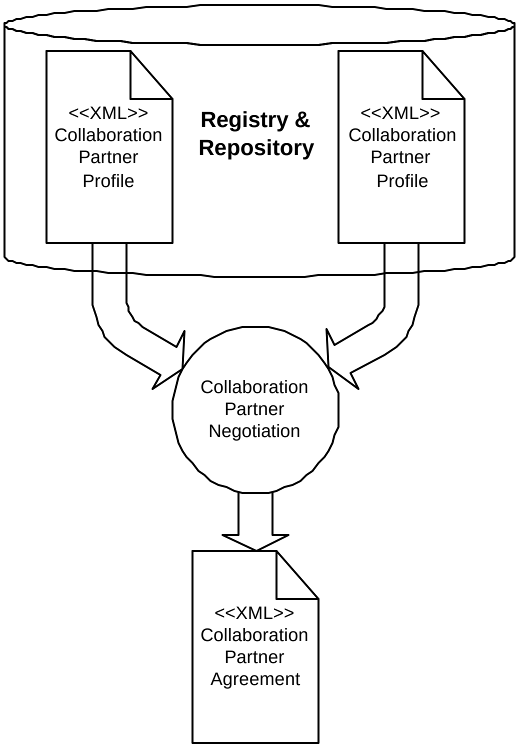
Registry & Repository
<<XML>> Collaboration Partner Profile
<<XML>> Collaboration Partner Profile
Collaboration Partner Negotiation
<<XML>> Collaboration Partner Agreement

Figure 5 Storing a CPP and generating a CPA

The CPA is then used to configure the runtime for the ebXML components so that the business collaboration can execute the secure business process (Figure 6).

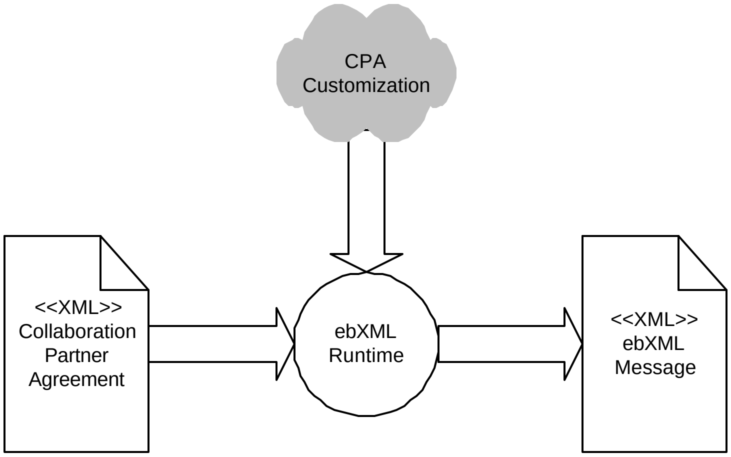

Figure 6 Configuring the runtime

9 ebXML Business Process Specification Layer

The security model for ebXML relies on an assumption that the modelling of security attributes at the *Business Operational View* (see the text below) is mapped appropriately to the *Functional Service View* (expanded tags in the CPP).

The security model only addresses those security attributes that have been represented in XML as a result of the conversion of business process and information models into an XML representation. The current set of security characteristics that the business process [ebBPSS] has chosen to represent in XML is as follows:

```
nonrepudiationOfOrigin
nonrepudiationOfReceipt
secureTransport
confidentiality
authenticated
authorized
```

Currently the *Business Process* asserts security characteristics at a very coarse level. An example of this coarse granularity is given in the paragraphs below in the description of the issues surrounding **non-repudiation**.

To provide end-to-end security it must be possible to assert security requirements at a finer level of granularity in the business information model. For example, there are a number of things within the business model to which security characteristics can be applied; documents, delivery channels, or business processes as a whole.

This cannot be done with the current level of detail. The coarser the granularity of the security characteristics, the simpler but more limited the options are. In the beginning of any such effort, it is natural to start with the simple, coarse-grained security characteristics. However, eventually the business process will require finer granularity to the security characteristics despite the challenging nature of such added detail.

For example, it is difficult with the current set of security characteristics to indicate whether **non-repudiation** is handled by the application or by the message service layer. It is also difficult to see how this is represented by the CPP. To assert that non-repudiation of receipt is addressed means that some pieces of the message header and payload are being asserted as evidence. In addition, a hash has been generated over this information and evidence that the receiver is able to verify that same hash value is returned in the acknowledgement of receipt to the sender. The sender then needs to archive this information as evidence.

Currently each party defining a BP must choose to apply or not apply each security mechanism at each level separately. This leads to a complex representation within a CPP and a potential problem with an increased risk of improper configuration at the packaging stage where it must be decided which parts of the message security should be applied to.

To bootstrap the ebXML process, a set of profiles that represent typical business requirements must be established. If additional scenarios are identified, new profiles could be created/documented and added to the choices for parties defining business processes. Sample profiles could address particular business needs, and define those security services necessary to meet those needs. A good example profile would be one for non-repudiation of receipt (NRR). The business process might require that the sending party receive solid proof that the receiving party received the *payloads* unaltered. If NRR is desired, signing will almost always be required as well. In addition it is most likely only necessary to sign the *payloads*, and generate the NRR response over the *payloads*. A profile could be created for this scenario, and the party generating the BP could simply choose to apply this profile rather than having to choose a more complex and obtuse set of security settings. In Appendix B Packaging Profiles, there are four sample profiles for secure packaging of the application payload:

- Application encryption over payload using PGP [PGP]
- Application encryption over payload using S/MIME [SMIMEV2][SMIMEV3]
- Application signing over payload using PGP]
- Application signing over payload using S/MIME

10 Trading Partner Information

In order to reduce risk to an acceptable level, potential trading partners must be able to authenticate each other's identity, verify the integrity of the messages they exchange, and ensure the confidentiality of those messages as they transit the network (known collectively as an ebXML security policy). The degree to which they will want to do these things will vary greatly depending on the situation.

There are many factors that can affect the ability to accomplish the desired level of trust. These include the following:

- Some nations regulate the export, import, or use of cryptographic software. The only means to address this is to ensure that algorithms, key sizes etc are always identified

- Most cryptographic protocols actually support a suite of algorithms and data structures (known collectively as mechanisms). So, even if both parties use XMLDSIG, partners will not be able to validate and verify a signature if one uses X.509[PKIX] [] mechanisms while the other only uses PGP. A potential way to address this is by defining some base-level profiles that all implementations support to identify which mechanisms a party uses so that “common operating dialects” can be found.

- Even when using common mechanisms, proper interpretation of authentication data can be very difficult and error-prone. For example, even after years of standardization, correct specification of how to validate X.509 certificate paths proves elusive. Given the current state of PKIX[PKIX]development, deferring to the manual evaluation step in CPP/CPA negotiation may be the only appropriate action for agreeing to a certificate validation scheme.

- Important pieces of a complete on-line solution are not widely deployed or even specified. For example, determining if a partner's certificate has been revoked, or if they are authorized to make purchases, can only be solved –if at all—through a series of ad hoc methods. This technology will evolve but again, manual evaluation is the only practical option for establishing revocation policies at this time.

    - This document proposes that a trust anchor element be created within the CPP and that it be represented as an XML Digital Signature [XMLDSIG] KeyInfo element. It is an endpoint for a set of credentials used by the party. It is important to recognize that a single policy will probably have multiple anchors. For example, a small enterprise might have an SSL certificate from a DNS registrar, yet use PGP [PGP] keys signed by a particular staff member for all purchasing agents.

In spite of these factors, it is still possible to create a secure association between trading partners, and automate a large portion of the establishment of that association by defining a `SecurityPolicy` element in the CPP. This element would advertise the set of security mechanisms a party understands, the profiles for those mechanisms, and the trust anchors that will be issuing the credentials used within that policy. The policies can be asymmetric, allowing separate identification of what it can accept from what it will, itself, generate. For example, a party might accept SSL-protected messages, but will itself, only generate [XMLDSIG] signed acknowledgements.

In order to encourage maximum interoperability, the following standard mechanisms are identified and vendors are encouraged to implement them:
-

- When exchanging identity information, use X.509v3 Certificates that follow the IETF profile (RFC2459 and its successors). [PKIX]
- When symmetric-key encryption is needed, use 3DES or the AES.
- When asymmetric encryption is needed, use RSA encryption with the OAEP encryption scheme and a key size of 1024 or 2048 bits.
- When hashing (or digesting) is needed, use SHA-1.
- When transport-level security is required, use SSLv3 or TLS with RSA keys and the RC4 (or ARC4) stream cipher.

The intent of this document is to initially establish the profile above as a text reference and identify it by the URN *urn:security.ebxml.org/profiles/baseline*. Future versions of the ebXML standards may provide detailed profiles as the correct format for this information and its relationship to the CPP elements are further refined.

10.1 PKI Interoperability Issues

A Public Key Infrastructure is more than just technology. In fact, technical interoperability accounts for about 20% of the issues when organizations want to cross certify or otherwise trust each other's certificates. There are a number of business, policy, procedure, audit and control issues that must be addressed prior to cross certification. This type of information should be covered in the CPA. Some of the key issues are covered below:

- Legal issues – for dispute resolution there may be a requirement to resolve the dispute in court and it should be determined up front what laws apply and in what jurisdiction
- Liability issues – who accepts liability, when and how much should be determined (usually per transaction but could be daily or some other means that meets both parties' needs)
- Level of assurance – in determining the limit of liability, the level of assurance (the level of assurance is based on the level of risk associated with identification, authentication, authorization and security of a certificate) must be determined for each organization and the proof of compliance to that level (compliance audit performed)
- Cultural and political issues – when dealing with entities external to an entity's borders there may be different cultural or political issues that must be addressed
- Policies and procedures (see level of assurance) there is a need to determine how certificates are managed such as revocation and timely posting to CRLs and/or OCSP responder, what applications are enabled, how they are enabled, key escrow (NOTE private signing keys should NOT be escrowed) etc.
- Technical – key size, certificate extensions, algorithms used, physical controls, key usage periods, private key protection, etc.

Appendix C documents a sample XML fragment for defining CPP elements related to public key policies.

10.2 CPP/CPA Security Elements

In the current version of the CPP/CPA, the specification of security elements is limited. It is recommended that XML schema be considered to more effectively express security attributes. For example, the security characteristic is a single element that contains attributes with Boolean values indicating whether or not a security attribute has been addressed. It would be useful to have the security characteristics have a type and be able to have a reference id to include on lower elements (like the transport element), which contain the details like the protocol.

In addition, it is entirely feasible to develop a super schema that would combine a description of the CPP with description of the CPA and correlate the relevant components of the two using the key/keyref mechanism of XML schema. This would allow a contract validator to match the correlated components to make sure that the contract is actually met.

The current CPP/CPA does not contain all the details needed to express both the policy and the operational details for specifying security. It is important that any ebXML follow on activity consider creating a group of participants from Business Process, Trading Partners, Security and TR& P to evolve the security attributes currently specified in the CPP.

It is unclear from the current analysis, where new elements should be attached within the CPP. Two options considered are to attach them to a delivery channel or to attach them to the service binding element of the CPP. If the details are attached to a delivery channel the entire document must be parsed in order to look for matching security attributes. If the details are attached to the service binding, it is easier to relate the security attributes with the packaging elements currently specified in the service binding. Grouping Trust Anchor elements like Certificate elements and allowing the channel specifications to reference the id of a trust anchor subset should be considered. Below is sample text for expressing Trust Anchors.

```
    <SecurityPolicy>
      <TrustAnchors>
          <!-a set of <ds:KeyInfo> elements. -->
          <ds:KeyInfo ID='foo'>...</ds:KeyInfo>
          <ds:KeyInfo ID='bar'>...</ds:KeyInfo>
          <ds:KeyInfo ID='chumley'>...</ds:KeyInfo>
      </TrustAnchors>
      <Profiles>
          <!-- A set of "Profile" elements.  Each profile
             identifies a profile, and then the anchors
             used in that profile.  -->
          <Profile ID="pf1" URN="urn" ANCHORS="foo bar"/>
      </Profiles>
      <WillUse>
          <--  A set of profiles the party  will use. -->
          <ProfileRef>pf1</ProfileRef>
      </WillUse>
      <WillAccept>
          <--  A set of profiles the party  will accept. -->
          <ProfileRef>pf1</ProfileRef>
      </WillAccept>
    </SecurityPolicy>
```

To address the secure packaging part of the Transport Routing & Packaging configuration in the CPP, the CPP should also document the packaging of the message header, payload and attachments so that S/MIME or XMLDSIG can be used to protect the appropriate elements of the message. If the packaging is well defined, it will allow the security tags within the CPP to specify the appropriate certificate data (X.509, PGP, etc.) to be applied to securely sign/encrypt the elements of the Message. This new `Packaging` Element in the CPP has been proposed, but it needs to be reviewed and an assessment made of whether it addresses this requirement

11 Registry and Repository

From a security perspective, the *Registry Service* of ebXML can be seen as a specific case of an ebXML transaction. It is possible to model its operations according to the ebXML Specification Schema and generate an appropriate CPP in the same way any other application would.

11.1 Registry

A security proposal for the Registry and Repository is documented in [REGSEC].

The following scenario illustrates how security for Registry processes ***might*** be specified. Note the following paragraphs and Appendix D Registry Sample documents an exercise to explore how an application might define its Business processes and messages as a way of illustrating the process of defining security for any ebXML application. The Registry group is encouraged to engage in such an exercise upon completion of their specification and to add to the profiles defined by the security group.

For the purposes of this exercise, the parties identified are the Registry Guest, the *Content owner of Submitting Organization* and the *Registry Service*. The *Content owner of Submitting Organization* wishes to register its business information in the ebXML Registry and Repository. The Content Owner evaluates the CPP in the Registry, which describes how a document can be submitted. It then creates and signs an ebXML document containing this business information and constructs a message (`RegistrySubmitManagedObject`) to send to the Registry Service.

The *Registry Authority* receives the registration request (via an XML document in a TRP message envelope)

Any Registry Guest is able to read all business entries.

Appendix D contains a skeletal CPP. In the CPP, the role of "content owner" is defined and a reference is made to an external document, which contains the Process Specification Document for ebXML Registry & Repository. A content owner who wants to add a CPP document to the Registry, creates a CPP document, signs it and sends it to the Registry. The Registry needs to know who is responsible for the document and the connection to the registry must be authenticated.

A second CPP is included which identifies the role of "registry guest". Requests for information from a registry are public requests. There is no security required for the connection to the registry in this instance.

11.2 Repository

Security for the repository is currently the responsibility of the implementer. This is an appropriate security choice, but it may have implications for authorization of access to the registry. It is suggested that recommendations for implementers of a repository include performing a risk assessment for the interface between the registry and the repository.

12 Messaging Service Functionality

The initial assessment of the *Message Service* was done on the December 2000 version of the document. Within the TRP document security issues are well documented and addressed primarily in Section 12. The latest TRP specification V0.99includes a merging of ebXML messaging and the SOAP messaging model, and an initial assessment has been made of this new model. There are several topics some of which are not specifically related to security mechanisms that are identified here as topics to consider in future ebXML activity related to secure reliable messaging.

12.1 SOAP-SEC extensions and Signatures in ebXML Messages

Given that an ebXML message is carried within a SOAP message, there are currently two ways of signing messages. This may cause some confusion or runtime failures due to misinterpretation. There has been a note posted to the W3C, which identifies one possible set of processing instructions for signing SOAP messages. Below are some "similarities and differences" that may help people wade through the notations. In addition, there is a good reminder in the concluding section of the XMLDSIG note about digital signature not itself preventing replay attacks. The "no-dupes" of reliable messaging can be used to address this type of attack.

1. SOAP-SEC[SOAP-SEC] uses its own namespace and has a schema that wraps around the XMLDSIG namespace, unlike the ebXML example.

2. SOAP-SEC and ebXML Digital Signatures both have the signature under the SOAP-ENV:Header.

3. The SOAP-SEC schema allows just one signature

4. SOAP-SEC uses the SOAP-ENV:actor and SOAP-ENV:mustUnderstand elements, whereas the ebXML example does not.

5. The actual W3C XMLDSIG machinery is shared. Of course, the ebXML example illustrates using an XPATH transform to cut out the TraceHeaderList (though the S1 value for the id attribute doesn't point to anything in the ebxml example)

6. The ebXML-Sig Reference [ebMS] mechanism uses cid: style URIs, but these are also acceptable in SOAP-SEC (section 3.2).

7. SOAP-SEC uses the soap protocol conventions of the mustUnderstand and actor constructs. It is not certain whether this is an advantage or just overhead. It might be a disadvantage if SOAP processing and ebXML MSH processing are "walled-off". In that

case, no defined lines of communication to the MSH from the SOAP layer exist so that MSH won't have access to the outcomes of checking. In general, it is difficult to assess the impact on implementations, but using SOAP-SEC within ebXML would tend to promote writing a SOAP processing layer as part of the MSH to facilitate communication.

12.2 Lack of Processing Rules

The TRP document addresses wire format only. Given the complex nature of composing a message that adequately reflects both security and reliability in addition to the correct business process data, there is a good deal of the processing of a business message through the MSH to the SOAP process that is left as an exercise for the reader. While the TRP specification makes a recommendation on how signatures should be applied to a *Message Envelope*, there are still areas of overlap between the SOAP envelope and the ebXML envelope that probably need further definition. As is mentioned in Section 12.1 item 7, there is no defined line of communication to the MSH from the SOAP layer. There are several areas in which the specification of the sequence of processing of a message would be helpful.

Intermediaries and the processing of “via” elements in TRP and SOAP actors with mustUnderstand attributes is one area in which there is a risk of runtime failures if the message flow from both the SOAP processor and the ebXML processing agent is not well understood by all parties.

There are several other areas of processing that are just general areas of caution due to the relative immaturity of XML technology. Transformations are one such area of concern. TRP signing identifies style sheet transforms (as does the XMLDSIG specification) as of particular concern due to the inconsistency of output from different implementations. In particular caution should be used when data from a signed message is parsed and validated and then the data is to be included in another signed message. The data should be re-signed rather than attempting to pickup a signed piece of information within one message and appending it to another message. The technology to perform consistent transformations is something that will evolve over time. The addition of XML encryption in combination with XML Digital signatures will possibly make this even more complex before it becomes more consistent.

12.3 Manifests

Independently and collectively, SOAP (with and without attachments), XML digital signatures (and, prospectively, XML encryption) and ebXML offer multiple mechanisms for component reference. Most notable among these is the "manifest". These reference mechanisms allow the composition of macroscopic message structures from microscopic message components. Similarly, SOAP and ebXML each offers a way of routing messages through intermediaries: the "actor" attribute in the case of SOAP and "via" element in the case of ebXML. These routing mechanisms can be thought of as a way of constructing processes on messages and this can be done dynamically.

Any design environment offering multiple ways of accomplishing the same end challenges the application developer with choices that often seem unmotivated, hence difficult to explain. (The existence of the largely interchangeable attribute and element constructions in XML itself are a good example.) This greatly increases the likelihood of error. The deeper concern, however, is how these compositional mechanisms interact. As there are neither syntactic nor semantic constraints on the interleaving of these functionally similar features, it is probably wise to anticipate that there will be unpleasant system surprises, especially when independent developers make use of composability. While our concern is a generic one, it comes vividly into focus when combining security with messaging.

A case in point is a scenario in which a SOAP-encoded ebXML message mentions “vias” V1 and V2. Suppose further that the SOAP envelope mentions “actors” A1 and A2. The designers' intention is that V1 signs the ebXML message and V2 does signature validation. On the other hand the SOAP server has been configured to direct all traffic through, A1which encrypts while A2 decrypts. This means that A2 needs to process the decryption before V2 is readable. In this case, what if A2 does not know about V2? The “ebXML” process thought the message would go from V1 to V2 and was unaware of the outer routing. And this is a simple case. On the face of it, there seems to be nothing to prevent routing episodes in which attempted signing, encryption, validation and decryption may fail.

12.4 Key Management

Key management is a major issue that needs to be addressed with respect to the capabilities of the TR& P Message Service Handler. In particular, if the MSH will be called upon to apply digital signatures, the appropriate private keys must be available to the MSH. Private keys must be managed very carefully and deliberately. Thus, some configuration will be necessary to establish the key management mechanisms to be used by the MSH.

Another major issue of key management is the distributing and registering of public keys or certificates used in Public Key Infrastructure (PKI), which is broadly adopted by many applications now for signing or encrypting information.

Currently a XML Key Management Specification [XKMS] proposed by VeriSign, Microsoft and webMethods has been submitted to W3C for consideration. It is intended to complement the emerging W3C standards activities in the XML Digital Signature and XML Encryption Working Group. There are two subparts in XKMS: the XML Key Information Service Specification (X-KISS) and the XML Key Registration Service Specification (X-KRSS).

13 Conformance

13.1 Overview

Conformance will be based on adhering to the specific conformance requirements delineated in the ebTA, ebRS, ebMS, ebBPSS and ebCPP specifications.

13.2 Conformance Requirements

Types of conformance requirements can be classified as:

a) Mandatory requirements: these are to be observed in all cases;

b) Conditional requirements: these are to be observed if certain conditions set out in the specification apply;

c) Optional requirements: these can be selected to suit the implementation, provided that any requirement applicable to the option is observed.

Furthermore, conformance requirements in a specification can be stated:

- Positively: they state what shall be done;
- Negatively (prohibitions): they state what shall not be done.

14 Future Requirements

14.1 Multi-hop and third party security services

The ability to simultaneously support multi-hop traceability and message integrity validation is an issue that must be addressed. For message integrity validation, it is desirable to apply a digital signature to of as much of the message as possible. To support multi-hop traceability, each intermediary must add a new section of signed traceability information. Care must be taken to establish message structuring and processing that allows the traceability information to be added without disturbing any pre-existing integrity or traceability components. With this in mind, it is constructive to consider the proposed ebXML message structure (shown below) in conjunction with potential security mechanisms.

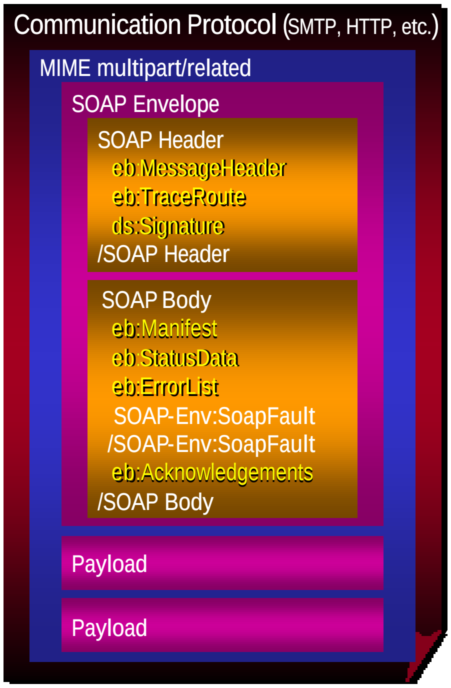

Figure 7 ebXML message structure

There have been discussions of applying S/MIME security mechanisms to the entire message (in the previous figure, this would include the elements grouped under the MIME multipart/related label).

The move to using an underlying SOAP message envelope may require the restructuring of the current CPP definition of the "nonrepudiation" element and its sub elements. The current tag specifies a protocol and hash algorithm but does not adequately express how this can be applied to an ebXML message (either parts or the complete message) to provide evidence that the receiver has adequately verified the receipt of a signed message and replied with a receipt acknowledging the same hash value over the signed message.

14.2 Archiving

The mechanisms for storing Business Process Information Models, Collaborative Partner Profiles and other related business information should supply assurances that the information stored and retrieved has not been modified by an unauthorized entity. The requirements state that the information should be able to be reconstructed at some point in the future, and at present it is difficult to know if this requirement has been met by the registry security proposal.

14.3 Minimum Security

It is currently assumed that the collaboration agreement (CPA) reached between two Trading Partners adequately reflects the ordering and priority of security policies stated in the CPP, but there is no mechanism for establishing minimum security requirements. The current CPP DTD does not allow the tagging of security configuration at a level that indicates what is required, what is optional, or what is preferred. There is not sufficient detail regarding properties like geography or liability (financial as well as legal) that might affect the choice of security mechanisms in an automated negotiation process.

Describing business' capabilities may misrepresent the intent of the CPP.

14.4 Automated CPA Generation

Within the Trading Partner group there is discussion about the dynamic generation of a CPA. The resolution of the CPA generation may require an additional version of this document to address the security issues in CPA negotiation, but it is currently out of scope.

14.5 Issues for non-repudiation of receipt (NRR)

(NOTE: This discussion focuses on message level NRR. Application level responses are out of the scope of this discussion).

From a top level (business level) perspective, the most important issue is to determine exactly what parts of the message are subject to NRR. For example, should NRR be applied to the payload items and/or the header? One suggested solution would be to apply NRR to only those parts of the message that were signed by the originator.

Another issue concerns how the NRR response should be sent back to the message originator. Should the message be sent back as part of another ebXML message, or should a separate mechanism be used (such as AS1 and/or AS2)?

The third and final issue is determining what format the NRR response should take. If it is chosen to use an externally defined transport and format such as AS1 or AS2, then this decision is already made. If, however, ebXML is the chosen transport, it needs to be decided where the NRR response should reside (in the SOAP header, or body, etc.). Additionally, the content of the NRR needs to be decided. It has been proposed within the TRP group that a NRR response should simply be the acknowledgements element which has been signed, but that neglects to include a hash of the parts of the original document for which the NRR is being generated. At a minimum, the hash of the original message parts and a reference to those parts (such as the acknowledgements element) must be signed to supply NRR. As part of the format used, there much be a decision made about what algorithms and transformations will be used to sign the NRR response.

Once all of those issues have been decided, there must be some mechanism within the CPP for any optional information (such as the scope of the desired NRR) to be supplied.

14.6 Registry and Repository Authentication

In selecting distinguished names as the binding mechanism to a key, the risk is run that other nonX.509 key binding schemes are ignored. A more generic alternative mechanism is recommended for mapping from keying material to a unique identifier within the registry. A registration process to associate the keying material with the implementation identity would allow supporting alternative key binding schemes. (For further reading please see section 9.1 first paragraph of the [ebRS]).

14.7 Messaging without a CPA

There has been discussion on the TRP mailing list including participants from TP and Security around the topic of CPPs and CPAs and whether they are required for Messaging. The risk analysis provided in the overview of this document is dependent upon an agreement between two trading partners being reflected in the creation of a CPA document. It is recommended that a CPA be signed by both parties to indicate their commitment to the agreement.

The TRP spec [ebMS] currently requires a CPAId element (a string that identifies the parameters that control the exchange of messages between the parties) in a message exchange. Businesses who engage in transactions without documenting their agreement should be aware that all assurance that the business process was adhered to is outside of the ebXML architecture and must be agreed upon and substantiated by some other means.

15 Additional Requirements and Recommendations

Registry & Repository

- A more generic alternative mechanism is recommended for mapping from keying material to a unique identifier within the registry.
- It is recommended that implementers of a repository perform a risk assessment for the interface between the registry and the repository.

CPP/CPA

- Additional policy-based elements need to be added to the CPP and several suggestions are included in this document.
- A stronger use of schema to type security could aid in the automatic generation of CPAs.
- Defining a set of common profiles would greatly improve chances for interoperability.
- The coarse grained nature of the security characteristics element may increase the risk of improper security configuration. Manual review of the CPA is therefore recommended.

Business Process

- Modeling of the business process should include a finer grained expression of security characteristics. The current set greatly limits the ability to represent security throughout the creation and transport of the business content.

Transport Routing and Packaging

- The absence of processing rules for message composition in particular, with regard to security in messages, may increase the risk of runtime failure due to misunderstanding of the ordering of actions to successfully decompose the message.
- The absence of a clearly defined handoff between SOAP and ebXML and the existence of “intermediaries” at both the SOAP and ebXML level may increase the risk of runtime failures.

16 Reference

[BS-7799/ISO-17799] Information security management part 1 and 2.

[PGP] IETF RFC 2440 OpenPGP

[PKIX] IETF RFC 2459 PKIX Certificate & CRL Profile

[REGSEC] ebXML RegRep Security-003.doc, Farrukh Najmi, Krishna Sankar, December 9, 2000

[SAML] Security Assertion Markup Language, http://www.oasis-open.org/committees/security/docs/draft-sstc-use-strawman-03.html

[SOAP-SEC] W3C Note, Applying Digital Signatures to SOAP, Hiroshi Maruyama, Blair Dillaway

[SMIMEV2] IETF RFC2311-2315, 2268

[SMIMEV3] IETF RFC2630-2634

[XKMS] draft version 1.0, Nov 27th, 2000, http://www.verisign.com

[XMLENC] W3C XML Encryption Syntax and Processing, http://www.w3c.org/Encryption/2001/03/12-proposal.html

XMLDSIG W3C XML Digital Signatures, http://www.w3.org/TR/2000/CR-xmldsig-core-20001031/

17 Disclaimer

The views and speculations expressed in this document are those of the authors and are not necessarily those of their employers. The authors and their employers specifically disclaim responsibility for any problems arising from correct or incorrect implementation or use of this design.

18 Contact Information

Team Leader

Maryann Hondo
IBM
1Rogers St
Cambridge, Ma 02142
US

Phone: 617-693-4299
Email: mhondo@us.ibm.com

Editor:

Paul Bussey
Cyclone Commerce

Email: pbussey@cyclonecommerce.com

Appendix A. Security Assertion Markup Language (SAML) ebXML use case

The Oasis Security Services Technical Committee is in the process of developing a set of requirements and use cases to develop a language for security assertions. The following use case has been submitted as a generalized use case for ebXML applications that require authentication and authorization. It is based on the work done by the security and registry groups in an exercise to develop a POC example for a business process that required authorization. The use case was submitted to the SAML group so that some ebXML application requirements would be considered in the specification that the SAML group will produce.

When the specification is issued, its use within ebXML will need to be explored and documented. Additional elements might be required in the CPP to provide the appropriate information about authorization and authentication authorities and parameters of the assertions.

The submitted ebXML use case was grouped with others in the “business to business” scenario.

Scenario 1: General Use cases for ebXML authorization

1) Party A wishes to engage with Party B in a business transaction. To do this, Party A accesses information stored in an ebXML CPP about Party B’s requirements for doing business. Some of this information might include:
    a. Party B requires authorization credentials from AuthorizationServiceXyz
    b. Party B requires that Party A be authorized by XYZ in the BuyerQ role.
2) Party A then must be able to determine:
    a. How to get these authorization credentials
    b. Where/how to insert these credentials in an ebXML message (need to define ebXML bindings)
3) Party B has received a digitally signed ebXML message from party A and wishes to obtain authorization information about party A
    a. Authorization data must be retrievable based on the DN in the certificate used to sign the ebXML message
4) Party A has enrolled with AuthorizationServiceXYZ. Party A engages in ebXML business transactions and wants to restrict what entities are able to retrieve its authorization data.

Appendix B. Packaging Profiles

PGP profile for application encryption of payload

```
<?xml version="1.0"?>
<!-- Simple ebXML PGP profile for application encryption of payload. No
signature supplied by application. -->
<Packaging>
      <ProcessingCapabilities generate="Yes" parse="Yes"/>
      <SimplePart id="header" mimetype="application/vnd.eb+xml" >
      </SimplePart>
      <SimplePart id="pgpversion"
            mimetype="application/pgp-encrypted" >
      </SimplePart>
      <SimplePart id="payload" mimetype="application/xml" >
      </SimplePart>
      <CompositeList>
            <Encapsulation id="encryptedpayload"
                              mimetype="application/octet-stream" >
                <Constituent idref="payload" />
            </Encapsulation>
            <Composite
                  id="envelopedpayload”mimetype="multipart/encrypted"
                  mimeparameters=
                  "protocol=&quot;application/pgpencrypted&quot;" >
                  <Constituent idref="pgpversion" >
                  <Constituent idref="encryptedpayload" />
            </Composite>
            <Composite id="ebxmlmessage" mimetype="multipart/related"
              mimeparameters="type=&quot;application/vnd.eb+xml&quot;;
                  version=&quot;1.0&quot;">
                  <Constituent idref="header" />
                  <Constituent idref="envelopedpayload" />
            </Composite>
      </CompositeList>
</Packaging>
```

PGP profile for application signing of payload

```
<?xml version="1.0" ?>
<!--  Simple ebXML PGP profile with application signing of the
  payload. Confidentiality if needed can be supplied at the
  network or transport layers.   ->
<Packaging>
    <ProcessingCapabilities generate="Yes" parse="Yes" />
    <SimplePart id="header" mimetype="application/vnd.eb+xml" />
    <SimplePart id="payload" mimetype="application/xml" />
   <CompositeList>
     <Encapsulation id="pgpsig" mimetype="application/pgp-
        signature">
        <Constituent idref="payload" />
      </Encapsulation>
     <Composite id="signedpayload" mimetype="multipart/signed"
        mimeparameters="protocol="application/pgp-
        signature";"micalg="pgp-md5"">
        <Constituent idref="payload" />
        <Constituent idref="pgpsig" />
      </Composite>
     <Composite id="ebxmlmessage"
        mimetype="multipart/related">
        <Constituent idref="header" />
        <Constituent idref="signedpayload" />
      </Composite>
    </CompositeList>
</Packaging>
```

S/MIME profile for application encryption of payload

```
<?xml version="1.0" ?>
<!--
Simple ebXML S/MIME for application-based payload encryption. No
authentication supplied.
-->
<Packaging>
    <ProcessingCapabilities generate="Yes" parse="Yes" />
    <SimplePart id="I001" mimetype="application/vnd.eb+xml" />
    <SimplePart id="I002" mimetype="application/xml" />
   <CompositeList>
     <Encapsulation id="I003" mimetype="application/pkcs7-
        mime" mimeparameters="smime-type="enveloped-data"">
        <Constituent idref="payload" />
      </Encapsulation>
    -<Composite id="I004" mimetype="multipart/related"
        mimeparameters="type="application/vnd.eb+xml";version
        "1.0"">
        <Constituent idref="I001" />
        <Constituent idref="I003" />
      </Composite>
    </CompositeList>
</Packaging>
```

S/MIME profile for application signing of payload

```
<?xml version="1.0" ?>
<!-- Simple ebXML S/MIME profile for application-based,
  clear/detached signing of payload. Confidentiality can be
  supplied at the network or transport layers. -->
 <Packaging>
    <ProcessingCapabilities generate="Yes" parse="Yes" />
    <SimplePart id="I001" mimetype="application/vnd.eb+xml" />
    <SimplePart id="I002" mimetype="application/xml" />
   <CompositeList>
     <Encapsulation id="I003" mimetype="application/pkcs7-
        signature">
        <Constituent idref="I002" />
      </Encapsulation>
    <Composite id="I004" mimetype="multipart/signed"
        mimeparameters="protocol="application/pkcs7-
        signature";micalg="rsa-sha1"">
        <Constituent idref="I002" />
        <Constituent idref="I003" />
      </Composite>
    <Composite id="I005" mimetype="multipart/related"
        mimeparameters="type="application/vnd.eb+xml";version=
        "1.0"">
        <Constituent idref="I001" />
        <Constituent idref="I004" />
      </Composite>
    </CompositeList>
</Packaging>
```

Appendix C. Sample Certificate Policy Element

```
<?xml version="1.0" encoding="UTF-8" ?>
<CertificatePolicies
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <CertificateProfile id="C06" version="X.509 Version 3">
    <ds:KeyInfo>
      <ds:X509Data>
        <!--
         two pointers to certificate-A
        -->
        <ds:X509IssuerSerial>
          <ds:X509IssuerName>CN=John Doe, OU=TRL,
              O=ebXML,L=location, ST=state/province,
              C=country</ds:X509IssuerName>
          <ds:X509SerialNumber>12345678</ds:X509SerialNu
              mber>
        </ds:X509IssuerSerial>
        <ds:X509SKI>31d97bd7</ds:X509SKI>
      </ds:X509Data>
      <ds:X509Data>
        <!--
         single pointer to certificate-B
        -->
        <ds:X509SubjectName>Subject of Certificate
            B</ds:X509SubjectName>
      </ds:X509Data>
      <!--
       certificate chain
      -->
      <ds:X509Data>
        <!--
        Signer cert, issuer CN=arbolCA,OU=FVT,O=IBM,C=US,
          serial 4
        -->
        <ds:X509Certificate>MIICXTCCA..</ds:X509Certificat
            e>
        <!--
         Intermediate cert subject
          CN=arbolCA,OU=FVTO=IBM,C=US
          issuer,CN=tootiseCA,OU=FVT,O=Bridgepoint,C=US
        -->
        <ds:X509Certificate>MIICPzCCA...</ds:X509Certifica
            te>
        <!--
         Root cert subject
          CN=tootiseCA,OU=FVT,O=Bridgepoint,C=US
        -->
        <ds:X509Certificate>MIICSTCCA...</ds:X509Certifica
            te>
      </ds:X509Data>
    </ds:KeyInfo>
    <PolicyInformation oid="">
      <PolicyConstraints>
```

```
        <!--
         Liability contraints, etc.
        -->
        <Constraint>
          <ConstraintProcessing />
        </Constraint>
      </PolicyConstraints>
      <PolicyQualifiers>
        <Qualifier />
      </PolicyQualifiers>
      <CertificateExtensions>
        <Extension />
      </CertificateExtensions>
      <CRLProfile version="">
        <CRLDistributionPoints>
          <DistributionPoint />
        </CRLDistributionPoints>
        <CRLExtensions>
          <Extension support="mandatory" />
          <Extension support="optional" />
        </CRLExtensions>
      </CRLProfile>
    </PolicyInformation>
  </CertificateProfile>
</CertificatePolicies>
```

Appendix D. Registry Sample

```
<?xml version ="1.0"?>

<CollaborationProtocolProfile>
<PartyInfo>
            <PartyId type =
            "urn:DUNS:nineplusfour">9876543211234</PartyId>
            <PartyRef xlink:type = "simple"
                  xlink:href =
            "http://www.collaborationparticipant.com/myid.html"/>
      <CollaborationRole roleId = "I1001">
      <CollaborationProtocol version = "1.0"
            name ="RegistrySubmitManagedObject"
            "locator"
            xlink:href =
            "http://www.ebxml.org/namespaces/RegistrySubmitManagedObjec
            t.xsd"/>
      <Role name = "RegistryServer"
            xlink:href =
            "http://www.ebxml.org/namespaces/RegistrySubmitManagedObjec
            t.xsd"
            xlink:type = "simple">RegistryServer
      </Role>
      <CertificateRef certId = "I10002">
            CN=CollaborationsRUs;O=CollaborationParticipant;C=US
      </CertificateRef>
      <ServiceBinding channelId = "I1010" name = "RegistryServices">
            <Packaging id="I1003" parse = "yes" generate = "yes">
            <SimplePart id = "I1004" mimetype = "application/eb+xml"/>
            <SimplePart id = "I1005" mimetype = "application/xml"/>

            <CompositeList>
                  <Encapsulation mimetype = "application/pkcs-signed"
                   id ="I1006"
                        mimeparameters = "smime-type=signed">
                        <Constituent idref = "I1005"/>
                  </Encapsulation>
                  <Composite  mimetype = "multipart/signed"
                        id = "I1007" mimeparameters = "">
                        <Constituent idref = "I1005"/>
                        <Constituent idref = "I1006"/>
                  </Composite>
                  <Composite mimetype = "multipart/related"
                        id = "I1008"
                        mimeparameters = "type=application/eb+xml">
                        <Constituent idref = "I1004"/>
                        <Constituent idref = "I1007"/>
                  </Composite>
            </CompositeList>
            </Packaging>
                  <Characteristics
                  nonrepudiationOfOrigin = "true"
                  nonrepudiationOfReceipt = "false"
                  secureTransport = "true"
                  confidentiality = "true"
                  authenticated = "true" />
      </ServiceBinding>
      </CollaborationRole>
      <Certificate certId = "I1002">
            <KeyInfo>
            <KeyValue>
                  <RSAKeyValue>
                        <Modulus>
                        zO7xXoKl4jPRpcUzLdPD3XJjdwop2LsU2sd1Dr3kb0bRO4z
                        X8SnAl3ov93eVGhylSRPrTpjTpOw3uUmPYgXolk639GYqmn
                        VAuffAlTz6BTrMN2OScjq2VLi5i6YxAMP0eXzKw+NXa9KI5
                        MfM2zV/IouSeo3M6t60/dG4IiBe6N8=
                        </Modulus>
                        <Exponent>AQAB</Exponent>
                  </RSAKeyValue>
            </KeyValue>
            <X509Data>
                  <X509SubjectName>C=US, O=CollaborationParticipant,
                  CN=CollaborationsRUs</X509SubjectName>
                  <X509Certificate>
                        IICWjCCAcOgAwIBAgIBAjANBgkqhkiG9w0BAQQFADBMMRow
                        GAYDVQQDExFDb2xsYWJvcmF0aW9u1JVczEhMB8GA1UEChMY
                        Q29sbGFib3JhdGlvblBhcnRpY2lwYW50MQswCQYDVQQGEwJ
                        VUzAeFw0wTAzMTYwMTAwMzJaFw0wMjAzMTYwMTAwMzJaMEw
                        xGjAYBgNVBAMTEUNvbGxhYm9yYXRpb25zUlVzSEwHwYDVQQ
                        KExhDb2xsYWJvcmF0aW9uUGFydGljaXBhbnQxCzAJBgNVBA
                        YTAlVTMIGfMA0GCSqGIb3DQEBAQUAA4GNADCBiQKBgQDM7v
                        FegqXiM9GlxTMt08PdcmN3CinYuxTax3UOveRvRtE7jNfxc
                        CXei/3d5UaHKVJE+tOmNOk7De5SY9iBeiWTrf0ZiqadUC59
                        8CVPPoFOsw3Y5JyOrZUuLmLpjEA/R5fMrD41dr0ojkx8zbN
                        X8ii5J6jczq3rT90bgiIF7o3wIDAQABo0wwSjAMBgNVHRMB
                        Af8EAjAADoGA1UdEQQzMDGBL2NvbGxhYm9yYXRpb25zUlVz
                        QHNtdHAuY29sbGFib3JhdGlvbnBhcnRuZXIu29tMA0GCSqG
                        SIb3DQEBBAUAA4GBAMv/9o/rc2sVmxRB/D/3o2/k2HHlkN8
                        AHx3fD9unqlDjKvhLt1JtqYwkHK897o3MwmE+yWKEWMAQsO
                        l0bVCmT1q4QrXcU6mAcB/QxPnObri5vRRVQ1AoZ1Jn2JqMj
                        xheLZWCfOQoxtpOph84HQGHnyn89lALw6JHOzogXFRNR0
                  </X509Certificate>
            </X509Data>
      </KeyInfo>
      </Certificate>
            <Certificate certId = "I1050">
            <KeyInfo>
            <KeyValue>
                  <RSAKeyValue>
                        <Modulus>
                              zO7xXoKl4jPRpcUzLdPD3XJjdwop2LsU2sd1Dr3kb
                              0bRO4zX8SnAl3ov93eVGhylSRPrTpjTpOw3uUmPYg
                              Xolk639GYqmnVAuffAlTz6BTrMN2OScjq2VLi5i6Y
                              xAMP0eXzKw+NXa9KI5MfM2zV/IouSeo3M6t60/dG4
                              IiBe6N8=
                        </Modulus>
                        <Exponent>AQAB</Exponent>
                  </RSAKeyValue>
            </KeyValue>
```

```
            <X509Data>
                  <X509SubjectName>C=US, O=CollaborationParticipant,
                  CN=CollaborationsRUs</X509SubjectName>
            <X509Certificate>
                  IICWjCCAcOgAwIBAgIBAjANBgbkqhkiG9w0BAQQFADBMMRowGAYDV
                  QQDExFDb2xsYWJvcmF0aW9u1JVczEhMB8GA1UEChMYQ29sbGFib3J
                  hdGlvblBhcnRpY2lwYW50MQswCQYDVQQGEwJVUzAeFw0wTAzMTYwM
                  TAwMzJaFw0wMjAzMTYwMTAwMzJaMEwxGjAYBgNVBAMTEUNvbGxhYm
                  9yYXRpb25zUlVzSEwHwYDVQQKExhDb2xsYWJvcmF0aW9uUGFydGlj
                  aXBhbnQxCzAJBgNVBAYTAlVTMIGfMA0GCSqGIb3DQEBAQUAA4GNAD
                  CBiQKBgQDM7vFegqXiM9GlxTMt08PdcmN3CinYuxTax3UOveRvRtE
                  7jNfxcCXei/3d5UaHKVJE+tOmNOk7De5SY9iBeiWTrf0ZiqadUC59
                  8CVPPoFOsw3Y5JyOrZUuLmLpjEA/R5fMrD41dr0ojkx8zbNX8ii5J
                  6jczq3rT90bgiIF7o3wIDAQABo0wwSjAMBgNVHRMBAf8EAjAADoGA
                  1UdEQQzMDGBL2NvbGxhYm9yYXRpb25zUlVzQHNtdHAuY29sbGFib3
                  JhdGlvbnBhcnRuZXIu29tMA0GCSqGSIb3DQEBBAUAA4GBAMv/9o/r
                  c2sVmxRB/D/3o2/k2HHlkN8AHx3fD9unqlDjKvhLt1JtqYwkHK897
                  o3MwmE+yWKEWMAQsOl0bVCmT1q4QrXcU6mAcB/QxPnObri5vRRVQ1
                  AoZ1Jn2JqMjxheLZWCfOQoxtpOph84HQGHnyn89lALw6JHOzogXFR
                  NR0
            </X509Certificate>
      </X509Data>
      </KeyInfo>
      </Certificate>
      <DeliveryChannel
            channelId = "I1010" transportId = "I1011"
            docExchangeId = "I1012">
      </DeliveryChannel>
      <Transport transportId = "I1011">
            <SendingProtocol>HTTP-Synch</SendingProtocol>
            <ReceivingProtocol>
                  <Endpoint uri =
                  "https://www.collaborationpartner.com/RegistryRespons
                  eSink" type = "allPurpose"/>
            </ReceivingProtocol>
            <TransportSecurity>
                  <Protocol version = "1.0">TLS</Protocol>
                  <Protocol version = "3.0">SSL</Protocol>
                  <CertificateRef certId = "I1002">
                        CN=CollaborationsRUs;O=CollaborationParticipant
                  ;C=US
                  </CertificateRef>
            </TransportSecurity>
      </Transport>
      <DocExchange docExchangeId = "I1012">
            <ebXMLBinding version = "1.0">
            <ReliableMessaging
                  deliverySemantics = "BestEffort"
                  idempotency = "true">
                  <Timeout>10000</Timeout>
                  <Retries>5</Retries>
                  <RetryInterval>1000</RetryInterval>
            </ReliableMessaging>
            <NonRepudiation>
                  <Protocol version = "1.0">S/MIME</Protocol>
                  <HashFunction>SHA-1</HashFunction>
                  <SignatureAlgorithm>RSA</SignatureAlgorithm>
                  <CertificateRef
                         certId = "I1050">string
                  </CertificateRef>
            </NonRepudiation>
            <NamespaceSupported
                  schemaLocation =
            "http://www.ebxml.com/namespace/RegistryServices.xsd"
                  version = "1.0">
            </NamespaceSupported>
            <NamespaceSupported
                  schemaLocation ="http://www.w3.org/2000/09/xmldsig#"
                  version = "1.0">
            </NamespaceSupported>
            </ebXMLBinding>
      </DocExchange>
</PartyInfo>
<ds:Signature/>
            <Comment>This sample includes packaging and role element
            changes, v32 or so. It is not at 1.0!!</Comment>
</CollaborationProtocolProfile>
```

Copyright Statement

This document and the information contained herein is provided on an "AS IS" basis and ebXML DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.