Experimental Implementation of Emerging e-Business Technologies: ebXML and PKI

Marko Topolnik, Damir Pintar, and Mihaela Sokić
Faculty of Electrical Engineering and Computing
Unska 3, Zagreb, Croatia

Abstract-- **As the technology of the Internet is maturing, it is becoming a reliable and secure infrastructure to globally exchange sensitive data. This makes it a feasible medium for conducting business and building a global electronic marketplace. New Internet technologies are emerging which deal with various aspects of this vision. In this paper, experimental installations of two of these technologies are described. One, *ebXML*, defines an architecture of a global marketplace. The other, *Entrust PKI*, provides a public key infrastructure used to ensure privacy, authenticity, and security in e-business communications. The PKI services can be used within the ebXML framework.**

I. Introduction

The benefits that a global public information network would bring to the business world were speculated upon even before the emergence of the Internet. An obvious one is an increase in the efficiency of peer-to-peer communication by replacing the paper documents with their electronic counterparts. Before the Internet, this was achieved using leased lines and custom documents and software. It was also rather expensive and therefore available only to larger businesses, and primarily for communication between dislocated outposts of a single company.

The Internet has brought the ability to achieve an affordable digital connection between any two businesses in the world. But, due to its academic origin, it lacks the foundations for establishing a secure and private communication channel between authenticated parties. This infrastructure has to be built on top of the basic Internet protocols. In this paper, one commercially available solution of this problem, *Entrust PKI*, is described.

Also, if any two companies are to successfully conduct business over the Internet, there is an obvious need for a globally accepted rich standard which specifies the collaboration protocols, documents, security standards, message formats, and everything else needed to form a universal information model of business-to-business communication.

Another potential of the Internet, one that could turn out to be even more important, is to provide a service for efficient business partner discovery. Traditionally, businesses had to rely on paid advertisements. More recently, presence on the Web, in combination with general Web search services, gave a hint of the Internet's potential in this respect. What is needed to fully exploit this potential is a single, globally accessible repository where companies can store all the information about their profile, supported business models, implemented information services, etc. This repository could then be browsed or searched by any other company needing a business partner.

The emerging e-business standard, *ebXML*, addresses the points mentioned in the previous two paragraphs. It is actually a set of standards specifying a broad spectrum of aspects of the e-business vision it proposes. The standards range from the physical structure of a single peer-to-peer message to high-level metamodels for describing a company's business model. In this paper, a small experimental ebXML environment is described.

II. The ebXML vision

The ebXML standard specifies the following aspects of the e-business environment:

- the information model of the *ebXML Repository* (storage place for the data of each company participating in the ebXML marketplace) and *Registry* (a searchable index of the Repository);
- structure, type and level of detail of business data that describes a company (contained in the CPP – *Collaboration Protocol Profile*). The company submits this data to the ebXML Repository, where it is publicly accessible;
- the procedure for partner discovery and subsequent negotiation of the collaboration protocol (forming of the CPA – *Collaboration Protocol Agreement*);
- the required information infrastructure that each company has to implement at its site in order to join the ebXML marketplace (*Business Service Interface*, *ebXML Message Service*).

As can be seen from the list above, the standard is concerned with the inter-company and marketplace aspects of e-business. Its reach inside the company is up to the company's public business interface. It is company's own responsibility to adapt its internal business and information model to cooperate with the ebXML business interface.

The overall business scheme envisioned by ebXML is presented in Fig. 1.

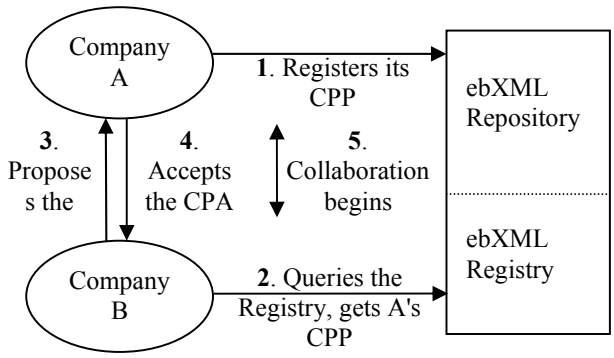

Fig. 1: Basic steps to ebXML collaboration

The goal of this scheme is to achieve a global electronic marketplace where companies can:

- discover each other electronically via the ebXML Repository/Registry;
- negotiate a business collaboration agreement (CPA) based on each company's CPP, or use a pre-made standard CPA designed for a specific industry branch;
- conduct business by exchanging standardized XML messages, according to standardized business collaboration schemes and using off-the-shelf software.

III. The ebXML Message Service

The first step of our experimental ebXML implementation was the installation of the *ebXML Message Service*. This service will be described in more detail.

The ebXML Message Service defines robust, yet basic, functionality to transfer messages between trading parties using various existing communication protocols [1]. It is structured to allow for messaging reliability, persistence, security and extensibility.

A. The ebXML Message

The basic building block of ebXML communication is the *ebXML Message*. It is structured in compliance with the *SOAP Messages with Attachments* specification. It is transport protocol-independent and can be exchanged using any of the transport protocols, such as HTTP, FTP or SMTP, mentioned above. SOAP follows the peer-to-peer communication paradigm, so there is no inherent role distinction between the communicating parties.

A SOAP Message with Attachments is a MIME/Multipart *Message Package* containing the following MIME parts:

- the *Header Container*, containing a SOAP 1.1 compliant message, referred to as the *SOAP Message*. It is the central element of an ebXML Message;
- zero or more *Payload Containers*, containing application-level payloads. These are the SOAP Message attachments.

The *SOAP Message* is an XML document that consists of the SOAP `Envelope` `root` element. This element consists of the following sub-elements:

- a `Header` element, specifying the nature of the ebXML message (such as whether the message is a request or an answer; the type of request/answer, etc.);
- a `Body` element, containing message service handler data and information related to the payload parts (attachments) of the message.

The structure of the ebXML Message is shown in Fig. 2.

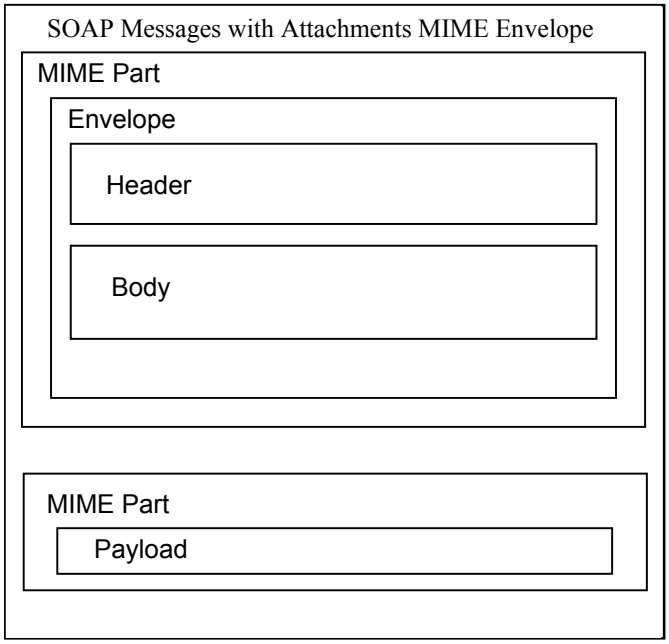

Fig. 2**:** ebXML Message structure

B. Experimental Installation of the Message Service

The ebXML Message Service can be functionally broken down into the following layers:

- *Message Service Interface* (MSI), an abstracting interface between the MSH (see 2. below) and the ebXML application;
- functions provided by the *Message Service Handler* (MSH): SOAP Processing, Header Processing and Parsing, Message Packaging, Security Services, Reliable Messaging Services.;
- mapping to the underlying transport service (HTTP, FTP, SMTP and other).

In the experimental installation presented in this paper, the Message Service was implemented in *Java Servlet* technology. This technology introduces the distinction between a server and a client. Since all ebXML parties are

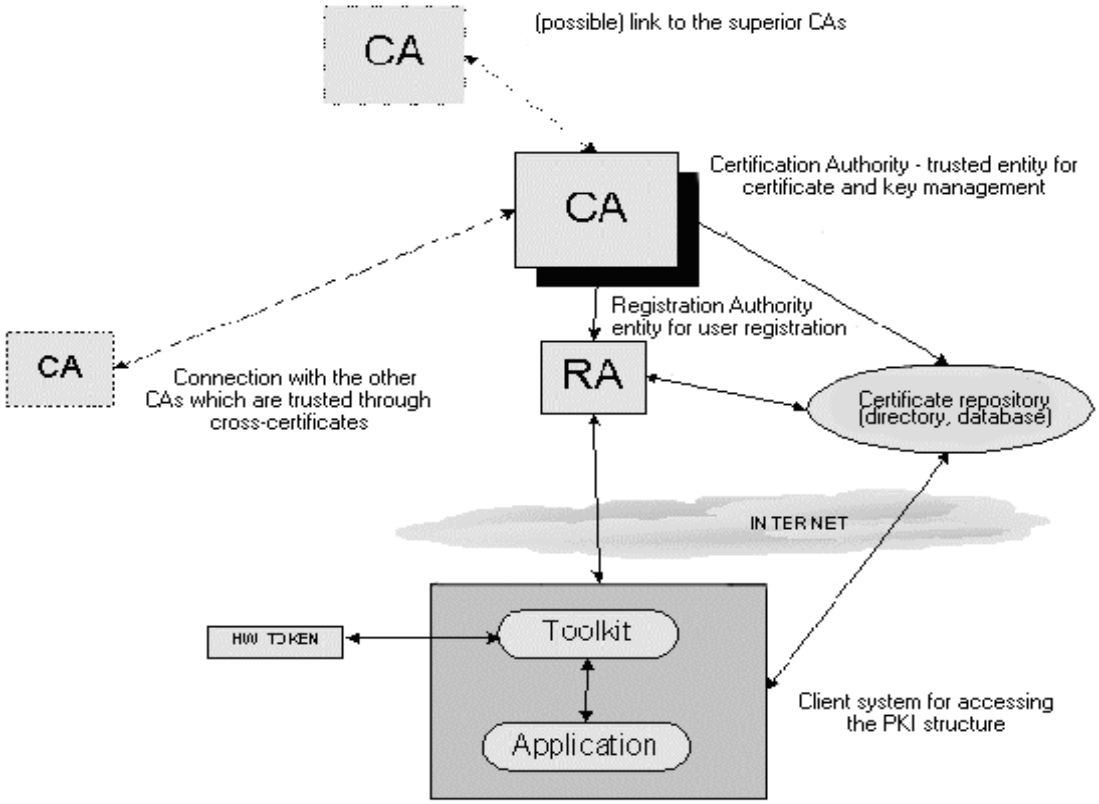

Fig. 3: Basic PKI architecture

peers, each of them implements both the server and the client aspect of communication using servlets. A peer is acting as a server when it receives and automatically acts upon an incoming request, and as a client when it is generating such requests. Because of this, the functional components mentioned above (MSI, MSH, transport protocol mapping) are not integral services, being split between the server and the client component.

The server component implements the MSH through several interacting servlets. The publicly accessible servlets are the `RpcRouter` and the `MessageRouter`. These servlets accept incoming requests from a remote peer (acting as a client). Each of them is responsible for one of two technologically different requests:

- the `RpcRouter` processes requests for a Remote Procedure Call (explicit request for the execution of a code module which produces a return value sent back as a response);
- the `MessageRouter` accepts ebXML messages and forwards them for processing according to their type. The response is typically an ebXML message containing the requested data, an acknowledgment of the received purchase order, etc.

The client component is implemented through executable Java classes, each capable of generating a specific type of request. The response is mediated through the standard output and available for further processing.

For the bottom layer, transport protocol mapping, the Java TCP/IP Sockets technology is used. The communication channel can be secured using Secure Sockets Layer (SSL).

IV. Messaging Security

The ebXML standard deals with the messaging security issues by defining the security-related categories and specifying approaches to achieving the desired level and type of security.

Within the CPP specification, there is a defined method of declaring what type of security a company supports/requires for its transactions. Also, there is a provision within the Message Service specification to implement secure messaging.

For the experimental ebXML installation, the use of the Public Key Infrastructure is considered for achieving security, privacy, authenticity and non-repudiation of ebXML messages.

A. Cryptography, encryption and digital signature keys

Conventional cryptography uses the concept of the "key" - only the person possessing the correct key can "unlock" the

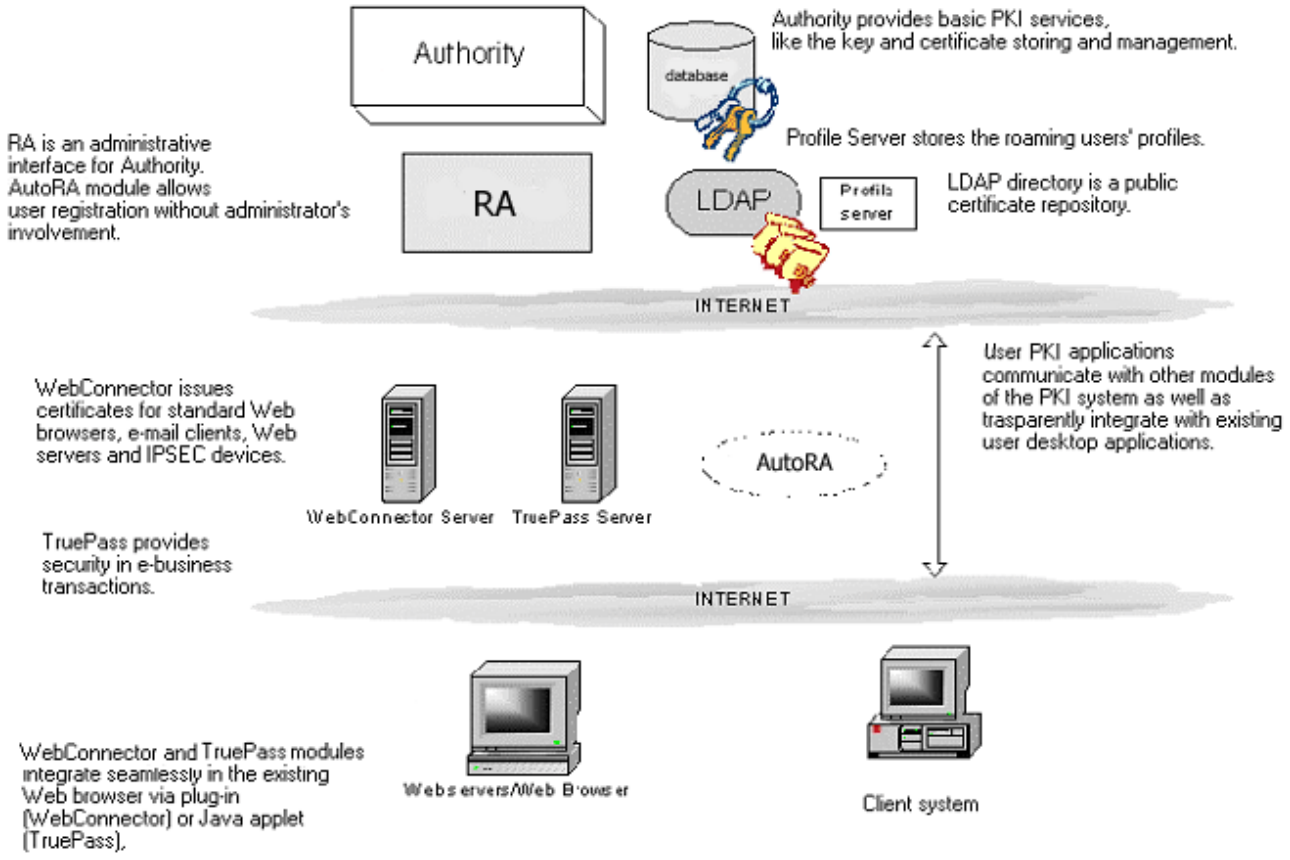

Fig. 4: An overview of the experimental PKI installation

document and read the data inside.

There are two ways of encrypting data: asymmetric and symmetric coding. With symmetric coding, the same key is used for encrypting and decrypting the data. This can be impractical, especially when many parties are involved in a secure communication. Asymmetric coding, however, uses two keys: public and private. Public keys are used for encrypting, private for decrypting. The private key cannot be deduced from the public one.

Encryption solves two problems:

- *confidentiality* of the sensitive data, and
- *access control* (only the intended person can see the data).

There are a few problems which are still unresolved: assurance that the data has not been tampered with, and the matter of resolving the sender's identity (since the encrypting key is publicly available).

For these reasons the concept of digital signature is introduced. Mathematical algorithms are used to make a unique summary of the data, in a way that changing a single letter in the original data will result in a complete different summarized block of data. This block is then encrypted with the sender's private key and represents the "digital signature". There is also a public key that is used for decrypting the signature. After decrypting the signature, the recipient uses the same algorithm mentioned above on the received data and compares it with the block of data, which was the result of decrypting. If there are the same, he can be sure of three things:

- *authenticity* of the message - the sender's identity is guaranteed if the blocks are the same
- *data integrity* - whether the data has been tampered with, and
- *non-repudiation* - the sender cannot deny the sending of the message (since nobody else has the private key he signed it with)

B. Public key infrastructure (PKI)

The industrial standard for e-business security is the *Public Key Infrastructure* (PKI). It is a combination of hardware and software products, guidelines and procedure, and it is based on public and private keys, digital signature and certificate system.

PKI system is responsible for creating and managing the key pairs for each user, as well as publishing the public keys in forms of "certificates" in publicly available directories. It provides the modules that perform the following functions:

- Certification Authority
- Certificate Repository
- Certificate Revocation System
- Key Backup and Recovery System
- Support for Non-Repudiation
- Automatic Key Update
- Management of Key Histories
- Cross-certification
- Timestamping
- client software support for communication with the above components

It is important to emphasize that while the description of the PKI system given below is based on the experimental installation of the specific software we used, the concepts and basic modules are the same or similar for the other solutions.

The *Certification Authority* (CA) is a fully automatic software module for key and certificate management. After installation and setting the basic parameters this managing is fully transparent for both users and administrators.

The *Registration Authority* (RA) is an administrative interface to the PKI system. It is used for adding new users, modifying their rights and permissions etc.

The *Certificate repository* (often implemented as an LDAP directory) is a publicly available source of certificates which contain the users' public keys. Besides the certificates it can contain other information, like the revoked certificates list, login information for roaming users etc.

The *Client system* has its own software support for accessing the system. It is possible, even recommended that hardware tokens are used in the process of user authentication (unique profiles saved on disks or smartcards).

CA's can be organized in hierarchical structure (for instance, government CA could be top-level, root CA which will guarantee the liability of subordinate CA's, for instance the CA's in various companies or public services).

Different CA's can trust each other through cross-certificates, which are exchanged through prior agreement.

Since the parties involved in a secure communication often cannot establish a direct trust relationship, we have a concept of "third-party trust". All the parties trust the chosen CA, which in turn provides the solutions for all the proposed security issues.

C. Experimental PKI installation

There are several commercially available solutions that implement the PKI concepts. In the experimental installation presented in this paper, the software package from Entrust was used.

There is important issue before installation that must be mentioned and seriously considered: the actual physical placement of the software modules. It is not acceptable for a failure of a single machine to result in a breakdown of the entire security system.

It is therefore highly recommended that not just the machines storing the core components of the security system be placed in a physically very secure environment, but that the different components be on different machines which are physically apart as much as possible and feasible.

Since our experimental installation was not intended for actual use in a working environment (except for testing purposes), we have decided for the economical and practical reasons to put all the components on a single machine. Being careful not to contradict with the whole concept however, we have made a separate virtual machine for each of the components, which in turn simulates the actual machine that should be physically apart from the others.

Upon installation and basic configuration, one of the important things that is inevitably noticed was that the maintenance personnel for the whole system should not be numerous (which conforms nicely to the whole security idea). Of course, the sole responsibility of maintaining the system should also not be assigned to a single individual – it is actually recommended that all the major decisions concerning modification of the system require approval from two or more responsible persons.

The whole process of managing certificates and keys, once it is configured, is fully automatic and transparent. Basic maintenance is not required, apart from the routine check-ups, upgrades or changes to some of the parameters, or perhaps solving potential unexpected problems usually dealing with some sort of physical component failure.

Administrators of the PKI system are simply "power users" who deal with administrative issues and are not concerned with any of the hardware ore software-specific issues. They do not need any special training or tutorials for the use of the system, due to the mentioned automatic nature of the whole process. For example, they do not need to know the details of the key and certificate storage and management, encryption etc. They just use the interface for adding users, modifying their rights, permissions and privileges, revoking and reissuing certificates etc.

The most important aspect to consider are the users of the system. If the whole concept is not implemented in way to make it easy to use in everyday business environment, there is no chance of its global acceptance. It was shown that the user support is as transparent to the underlying security mechanisms as it was possible to be. The clients' interface for security matters is integrated in their usual desktop environment, providing extra security options like "encrypt," "sign" etc. How they work, where is the public encryption key found, getting the certificate and checking its

verification, is all hidden from view. This is especially important if we take into account that the clients sometimes are not expert computer users, but just use the computer for everyday business purposes.

D. Practical use of PKI solutions

In the end, the way of using the whole concept in practical e-business situations should be discussed.

When looking at internal affairs of the companies, PKI makes additional security functions available. The employees can exchange secure e-mails, and be sure that the e-mails they get are sent from authenticated people. Furthermore, sensitive documents can be encrypted for entitled people and then made publicly available on the corporate Web site – there is no need for any additional security measures since nobody else cannot decrypt the document.

It has already been mentioned that the companies can trust each other with the use of cross-certificates. This of course means that all the security options are made possible between these two companies transparently. If their business communication is based on exchanging ebXML messages, each message can be encrypted, signed and sent over a secure communication channel, making e-business a very plausible and easy solution.

Furthermore, commercially available PKI solutions are usually shipped with toolkits used by software developers for easy integration of PKI security mechanisms into their own applications. This will definitely result in an easier acceptance of the PKI standards since the security mechanisms can be added to the underlying structure of an application giving extra security options with minimal implementation effort from the developers' side.

Companies that do their business over the Web can also profit through the use of PKI solutions. Sending sensitive information over the Internet is considered very risky nowadays, and the assurance that the transaction is completely safe and secure is a crucial requirement. As yet, conducting financial transactions over the Internet has not been widely accepted, especially by big companies who deal with large transactions and are put off by inadequate existing security mechanisms. The new security options proposed by PKI are a solid assurance that those financial transactions will be completely safe.

It should also be mentioned that in January, Croatia has established a digital signature law which approves digital signature as a legal way of identifying and authenticating companies and individuals. This can certainly be seen as a step forward towards implementing PKI on a national level which should furthermore result in installing top-level government CA's. They would in turn correspond with other government CA's, as well as guarantee the liability of subordinate company CA's. This will de facto make PKI a standard for e-business security solutions.

E. Intra-company business software

There are several software components with different characteristics considering PKI infrastructure that companies use:

- Web applications,
- User desktop applications,
- Legacy and enterprise resource planning systems (ERP) where core business processes are implemented.

E-commerce gives companies improved efficiency and reliability of business processes through transaction automation. We aim to extend our experimental environment by integrating a common system in which business processes are implemented.

Since we are part of Oracle Academic Initiative (OAI), Oracle Enterprise Resource Planning (ERP) system (software) is installed. We expect to integrate described PKI infrastructure with ERP software modules operating in Electronic Market and implement safe electronic communication between sell and buy side of Electronic Market.

V. Conclusion

In this paper, experimental implementations of ebXML Message Service and Public Key Infrastructure are described. The ebXML initiative promises a global electronic marketplace where companies can conduct business transactions over the Internet. A key aspect of these transactions is messaging security. Approaches are being studied to utilize the PKI to implement this security in the ebXML Message Service.

References

[1] Transport, Routing & Packaging Team, *Message Service Specification v1.0*, http://www.ebxml.org/specs/ebMS.pdf, 11 May 2001

[2] Mihaela Vrhoci, Dubravka Đelmiš, Jarmila Maksimović, *Collaboration-Protocol Profile (CPP) and Collaboration-Protocol Agreement (CPA) in B2B transactions*, unpublished, 2002.

[3] Ivan Magdalenić, Ivo Pejaković, Dražen Pranjić, *Business documents presentation and interchange using SOAP*, unpublished, 2002.