A WEB SERVICE FOR CERTIFIED EMAIL

Vincenzo Auletta, Carlo Blundo, Stelvio Cimato and Guerriero Raimato
Dipartimento di Informatica ed Applicazioni
Università di Salerno
84100 Baronissi (Salerno), Italy
{auletta, carblu, cimato, raimato}@dia.unisa.it

Keywords: Security Protocol, Certified Email.

Abstract Web service technology allows the completion of complex tasks through the definition of simpler collaborating services. As soon as Web services become available a challenging problem is the provision of means by which the data exchange and the service usage can be proofed. An interesting test case comes from the development of Web based certified email systems. In such systems users exchanging email messages would like to get some additional guarantees on the security of the communication during the interaction. In this work we describe the design and the implementation of a certified email application realized through the definition of interacting servers. Users access the service through a Web based application and are able to get, at the end of the transaction, a valid receipt which can be used in case of dispute.

1. Introduction

Web service technology is changing the way complex distributed applications are being developed. Indeed, such an approach offers a comprehensive set of platform-independent technologies and open standards which have been released in order to ease the delivery of new services over the Internet. Many companies are investing large amount of money in research and development projects related to the exploitation of such kind of technologies, expecting a number of technical and economical benefits [CCS03].

One of the key feature of Web service technology is the "loose coupling" paradigm which favors the integrations of interacting applications within a company or between different companies. By adopting such new development model, time and cost savings can be obtained by the use of standard interfaces and the reuse of existing applications. In addition, complex tasks can be achieved by the composition of simpler collaborating services. Instead of developing static applications, new business services can be defined out of available services, providing the flexibility needed to respond to rapidly changing customer and market requirements.

In order to support the creation of higher level and cross-organizational business processes, several proposals have been made to define extensions to the Web services standard framework. The terms *orchestration* and *choreography* of Web services usually refer to the set of emerging specifications dealing with the combination and collaboration of Web services for the creation of composite business processes. Microsoft's XLANG [Tha01], IBM's WSFL [Ley01], BPEL4WS [CGKL$^+$02] (by a consortium grouping Microsoft, IBM, BEA; SAP and other companies) are some of the currently available initiatives addressing some of the orchestration issues.

As soon as companies provide business services and release new Web services enabled products and tools, security becomes an increasing concern. Indeed, Web services applications let the information flow overcome the traditional boundaries between different companies, exposing data and internal structures to external potentially malicious users. Furthermore, Web service transactions span multiple applications and organizations, complicating the way access control policies can be setup or the tracking and the response to an attack can be done. In such scenario, it is important to provide security mechanisms which can be used during the data exchange in order to guarantee some basic security properties, stating for example the origin of a message (authentication), the destination of the message (authorization), the protection of the content of the message from modifications (integrity) and unauthorized access (confidentiality). [Bro03]

An interesting application useful to illustrate how some of the above security properties can be achieved in the development of a Web service is a certified email system. A certified email service tries to provide users with additional guarantees on the content and the delivery of the email messages, with respect to the ordinary email service. Indeed the standard e-mail service is based on the Simple Mail Transfer Protocol which offers no guarantees on the delivery and the authenticity of the messages, neither gives to the sender any evidence on the sending as well as any return receipt. A transmitted message could be eavesdropped over its path from the origin to the destination, and its content could be manipulated or corrupted by any malicious adversary. In literature, several distributed protocols for certified email have been proposed, relying on a trusted third party (TTP) to ensure the fairness of the protocol.

In this paper we describe a certified mail service based on the protocol described in [AHGP02]. The system is realized through the interaction of three main entities: the Webmail server for the sender, the Webmail server for the receiver and a Trusted Third Party. At the end of the transaction, the sender is able to get a receipt which can be shown in case of dispute. We discuss the details of the protocol in Section 3, and the basic security properties guaranteed by the proposed system in Section 6.

2. Certified Email Protocols

Certified email protocols ensure that a participant exchanges a message for a receipt, which the receiver should release at the end of the transaction. Indeed, the aim of such protocols is to provide a procedure for the secure exchange of messages which is resistant to possible attempts of cheating by the different participants. Certified email protocols can be seen as instances of fair exchange protocols [ASW98] which guarantee the fair exchange of objects, i.e., at the end of the exchange, both participants get what they expect or nobody gets any valuable information.

Recently a lot of research has been dedicated to the problem of designing certified email protocols that satisfy the above properties. Several protocols involve a trusted third party (TTP for short) which is delegated by the participants to control the behavior of the parties, assist them during the exchange of messages, and resolve any dispute if necessary. According to the role played by the TTP, protocols have been classified as *inline* or *optimistic*. In inline protocols [BT94, GZ96, DGLW96, RS98], the TTP is actively involved in each message exchange: both the parties send their messages to the TTP , which checks for their integrity and forwards them to the intended receiver. In optimistic protocols [ASW97, ASW98], the sender and the receiver first try to exchange the message by themselves, without the intervention of the TTP and rely on the TTP only for the cases where a dispute arises (maybe because one of the party is trying to cheat). Protocols that do not require a trusted third party have also been proposed [Blu83, EGL85]. The main drawback of these protocols is that they require a large number of exchanges between the parties.

3. The Framework

The scenario we consider consists of a number of users which are willing to exchange e-mail messages using a certified mail service. The service provides them with some additional guarantees on the delivery of the messages and on the security of the communication. To such purpose, the protocol relies on a Trusted Third Party (TTP) actively involved in each message exchange.

In the following we describe the cryptographic primitives used in the protocol: $Sig_A(m)$ denotes the digital signature of the message m using the private key of user A under a public-key signature algorithm; $h(m)$ indicates the hash of message m using some collision resistant hashing scheme. A collision resistant hash function maps arbitrary length messages to constant size messages such that it is computationally infeasible to find any two distinct messages hashing to the same value; $PK_B(m)$ denotes the encryption of message m

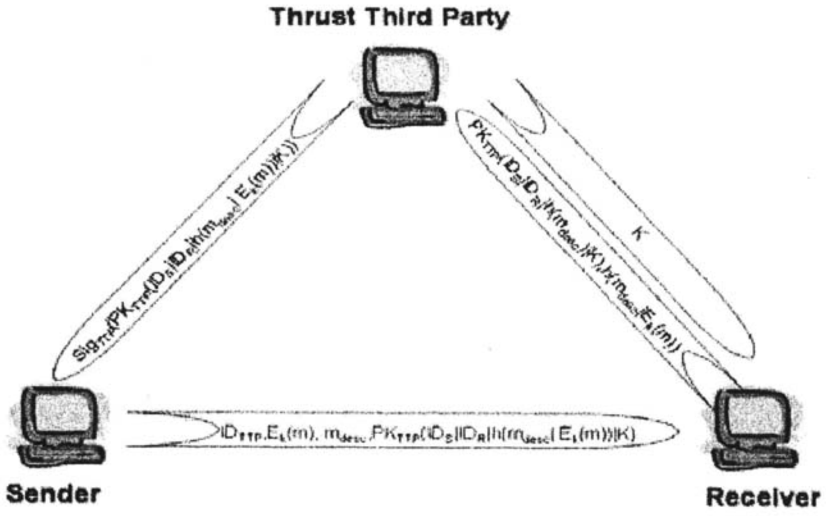
Thrust Third Party
Sig_TTP(PK_TTP(ID_S||ID_R||h(m_desc|E_k(m))||K))
PK_TTP(ID_S||ID_R||h(m_desc||K)),h(m_desc|E_k(m))
K
Sender
ID_TTP,E_k(m), m_desc,PK_TTP(ID_S||ID_R||h(m_desc|E_k(m))||K)
Receiver

Figure 1. The protocol

using the public key of user B in some public-key encryption algorithm. The algorithm should provide non-malleability, i.e., given a ciphertext it is impossible to generate another ciphertext such that the respective plaintexts are related; $E_k(m)$ denotes the encryption of message m using the key k under some symmetric encryption algorithm.

Description of the Protocol

In this section we briefly recall the protocol presented by Abadi et al in [AHGP02], whose sketch is presented in Figure 1. The goal of the protocol is to allow a sender S to send an email message to a receiver R, in such a way that R reads the message if and only if S receives a valid return receipt. Other properties discussed in Section 2 are satisfied. Some of these properties derive from the use of cryptographic primitives, others derive directly from the protocol.

1 When S wants to send an email message m with subject s to R, he chooses a session key k and encrypts the message m using k obtaining $E_k(m)$. Afterwards, he adds a brief description m_{desc} of the message m and computes the hash $h(m_{desc}||E_k(m))$. S should also compute and store the $S2TTP$, which is the encryption under the public key of the TTPof the receiver identifier, the sender identifier, the key k associated with the message and the hash previously computed, i.e.,

$S2TTP = PK_{TTP}(ID_R, ID_S, k, h(m_{desc}||E_k(m)))$. The message sent to R consists of the TTP identifier, which R can contact to read the email message, the encrypted message $E_k(m)$, the message description in cleartext m_{desc}, and the string $S2TTP$.

2 On the reception of the message, R can read the description of the message and decide whether he wants to get the content of the email message from S. If he does, he forwards to the TTP (whose address is contained in the received message) the $S2TTP$ together with the hash $h(m_{desc}||E_k(m))$ containing the message description and the encrypted message. Otherwise, if R is not interested in the message, he can simply ignore the message and abort the protocol.

3 Whenever the TTP receives the message from R, he can decrypt $S2TTP$ and retrieve the key k and the hash $h'(m_{desc}|E_k(m))$. He then verifies that $h'(m_{desc}|E_k(m))$ matches with the hash received from R and in this case forwards the key k to R in order to let him read the mail, and sends S the signed receipt $Sig_{TTP}(S2TTP)$, which S can show as proof of the email sending.

4. The Architecture of the Certified Email Service

In this section we describe an architecture to implement the Web service based certified email system. Our system relies on three main entities interacting with the client and the receiver applications. Such components expose the services needed to realize all the phases of the protocol:

- **Sender's Webmail Server (WMS_S):** The WMS_S exposes two services: $Send$ and $Store\_Receipt$. The former is invoked by S whenever he wants to forward a certified email message. The latter is invoked by the TTP at the moment the receiver accepts and reads the message. The receipt is verified and stored and made available to S.

- **Receiver's Webmail Server (WMS_R):** The WMS_R exposes the $Store\_Message$ service which is called by the WMS_S in order to store a delivered email message. Receiver server invokes also a service on TTP to obtain data to generate the decryption key. Furthermore, this server provides to store cipher and plain mail.

- **TTP Server:** The TTP exposes the $Key\_Request$ service which is invoked by the WMS_R during the reading of a certified email message in order to retrieve the decryption key associated with the messageû

Both sender's and receiver's Webmail servers allow users to access one's mailbox through web pages using a standard Internet browser. In addition to

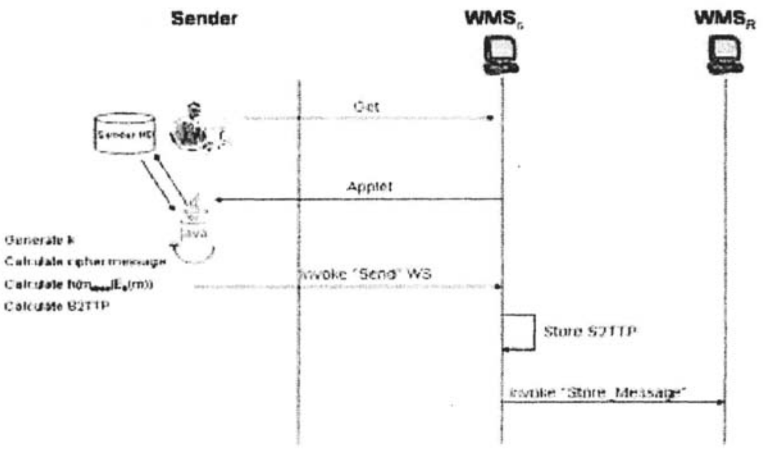

Figure 2. Sending Email

the standard functions of an e-mail client, such servers offer the functionalities of composing and receiving certified email messages, in a (as much as possible) transparent way for the users, which are requested to provide the data necessary to execute the phases of the protocol. We assume that the communication channel among the participants are secured using standard techniques, like SSL. To this purpose it is important that both S and R know the public key of the TTP and are able to establish a secured connection with their respective Webmail servers by authenticating themselves (e.g., via login and password). The communication among the Webmail servers and the TTP should be secured using strong authentication mechanisms (via certificate exchange).

Let us show in more detail the interactions occurring during the sending and the reading of the mail. As shown in figure 2, in order to send a certified email message, the sender invokes the *Send* service provided by WMS_S. To this purposes he constructs a SOAP message which encloses all the data needed to follow the protocol above described. On the server side, the message is processed such that the data needed to construct the SOAP message containing the request to the WMS_R *Store_Message* service are retrieved. The WMS_R stores the message such that R is displayed the notification of a received certified email message.

If R is interested in reading the certified email message, a SOAP message containing the *Key_Request* service invocation is constructed and forwarded to the TTPserver. Such request contains the $S2TTP$ which the TTP can de-

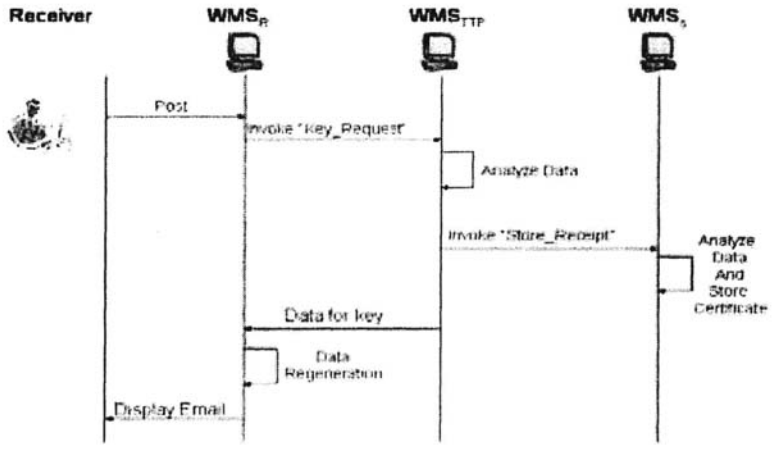

Figure 3. Reading Email

crypt in order to recovery the key associated with the message together with the hash of the subject of the email and the encrypted message. According to the protocol, on a successful checking of the validity of the received data, the TTP invokes the *Store_Receipt* service on the WMS_S, which enables the sender to get a valid receipt, and constructs the response message for the WMS_R, which enables R to decrypt and read the mail message.

5. Implementation

The implementation and the deployment of Web services applications involve several critical aspects which must be taken into account in order to satisfy the security requirements. We developed a prototype implementation of the proposed certified email system using Java (version 1.4.2) as development language. Java offers also a comprehensive set of cryptographic tools and libraries which have been used in order to execute the operations needed to execute the protocol. Some of the needed cryptographic functions have been implemented using the BouncyCastle Cryptographic Service Provider.

As application server we used Tomcat (version 4.1), usually referred as the official implementation for the Java Servlet and JavaServer Pages technologies. The SOAP engine is provided by Apache Axis (version 1.1) which offers a framework for constructing SOAP processors, a server which plugs into servlet engines (such as Tomcat), extensive support for the Web Service Description Language (WSDL), together with a set of other useful tools for the development phase.

It is important to notice that during the sending of a certified email message, some computation must be done on the client machine. To this aim an applet has been developed which performs all the cryptographic operations necessary to the generation of the key associated with a new certified email message, its encryption (using 3-DES), its hash (using MD5 digest function), the computation of the signed string S2TTP (using RSA-SHA1) and finally the invocation of the *Send* service on the WMS_S. To access the data stored on the local disk, some access-rules contained in the java-policy file for the sender must be modified.

Whenever a certified email message is sent to the receiver, i.e., the *Store_Message* service is invoked on the WMS_R, the WMS_S stores the *S2TTP* string computed for the message, the receiver's address, the URL of the TTP, and the arrival time of the request in a file. Whenever the TTP forwards a receipt, the WMS_S is able to control the validity of the data contained in the receipt with respect to the data stored in the file associate with the message. If the data match, the signed receipt is stored and can be used by S in case of dispute.

On the receiver's side, if R decides to read a certified email message, at the end of the interaction between the WMS_R and the TTP, the notification is substituted with the decrypted copy of the message.

6. Discussion

As discussed in [AHGP02], some of the security properties presented in section 2 are satisfied by the protocol and the service implementation we proposed. In particular, the implementation provides confidentiality and integrity of the mail messages. Indeed, confidentiality with respect to external users and the TTP is guaranteed by the assumption that all the communication channel are secured, and the message is encrypted with the session key k. Cleartext message is not known to the WMS_S and the TTP, but only to the WMS_R after the completion of the protocol (since the decrypted message is stored in the mailbox of R like ordinary email messages). To increase the confidentiality of the exchange, and avoid that the WMS_R knows the decrypted message, it is possible to execute the decryption operation directly on R's local machine.

Fairness and non repudiation of receipt properties are also satisfied. If both the sender and the receiver behave as expected and messages are delivered, at the end of the protocol each party gets the desired information. The protocol implementation relies on the TTP server in order to ensure that after a message exchange, R is able to read the email if and only if S receives the corresponding return receipt. Indeed, at the end of the protocol execution, S can show in case of dispute the message $S2TTP$ signed by the TTP, which constitutes a non repudiable certification of the reception of the message by R.

7. Conclusions

The emerging Web services standards provides a new technology supporting the release of Web based applications. Such new paradigm of computing allows the development of distributed applications, realized through loosely coupled collaborating services which implement business processes and are available through the Internet to end users [CCS03].

In this work we experimented the design and the implementation of a certified email service. Usually other approaches [AHGP02] require the users to install some additional software, such as a plug-in tailored to a particular email client or Internet browser. Such an approach limits the number of target users, since they are usually reluctant to modify or install software on the local machines. Other approaches rely on the development of a mail proxy, which filters incoming and outcoming messages, executing transparently the certified email protocol. But also in this case users should trust and use an additional piece of software. The proposed Web system does not require any additional software apart from a Java enabled browser (and some modification to the Java local policy file). We plan to extend the approach to the implementation of other email protocols, satisfying a larger number of security properties achieved through the combination of simpler available services.

References

[ASW97] Asokan, N., Schunter, M., and Waidner, M., 1997, Optimistic Protocols for Fair Exchange, In *ACM Conference on Computer and Communications Security*, pp. 7–17.

[ASW98] Asokan, N., Shoup, V., and Waidner, M., 1998, Asynchronous Protocols for Optimistic Fair Exchange, In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pp 86–99.

[BT94] Bahreman, A., and Tygar, J. D., 1994. Certified Electronic Mail. In Dan Nesset (General Chair) and Robj Shirey (Program Chair), editors, *Proceedings of the Symposium on Network and Distributed Systems Security*, California, Internet Society, pp 3–19.

[Blu83] Blum, M., 1983, How to Exchange (Secret) Keys, In *STOC*, pp 440–447.

[Bro03] Brose, G., 2003, A Gateway to Web Services Security – Securing SOAP with Proxies, In *Proceedings of International Conference on Web Services-Europe 2003 (ICWS-Europe 2003)*, Lecture Notes in Computer Science, vol. 2853, pp. 101-108.

[CCS03] Chen, M., Chen, A. N. K., and Shao. B., 2003, The Implications and Impact of Web Services to Electronic Commerce Research and Practices, *Journal Electronic Commerce Research*, Vol. 4, N. 4.

[CGKL$^+$02] Curbera, F., Goland, Y., Klein, J., Leymann, F., Roller, D., Thatte, D.. and Weerawarana, S., 2002, Business Process Execution Language for Web Services (BPEL4WS 1.0).

[DGLW96] Deng, R. H., Gong, L., Lazar, A. A., and Wang, W., 1996, Practical protocols for Certified Electronic Mail, *Journal of Network and System Management*, 4(3).

[AHGP02] Abadi, M., Horne, B., Glew, N., and Pinkas, B., 2002, Certified Email with a Light On-Line Trusted Third Party: Design and Implementation, In *Proceedings of eleventh International World Wide Web Conference*, ACM Press, New York.

[EGL85] Even, S., Goldreich, O., and Lempel, A., 1985, A Randomized Protocol for Signing Contracts, *Communications of the ACM*, 28(6).

[Ley01] Leymann, F., 2001, Web Services Flow Language (WSFL 1.0) IBM Software Group.

[RS98] Riordan, J., and Schneier, B., 1998, A certified E-mail protocol with No Trusted Third Party, In *Proceedings of the 13th Annual Computer Security Applications Conference.*

[Tha01] Thatte, S., 2001, XLANG: Web Services for Business Process Design, Microsoft Corporation.

[GZ96] Gollmann, D., and Zhou, J., 1996, A Fair Non-Repudiation Protocol, In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, CA. IEEE Computer Society Press, Oakland, pp 55–61.