SMMM – A SECURE MULTIMEDIA MAIL SYSTEM

Marcel Stanley A. de Moura ① Guido Lemos de Souza Filho ② Thaís Vasconcelos Batista ② Luiz Fernando G. Soares ①

① Lab. Telemídia – DI, PUC-Rio
R. Marquês de São Vicente, 225
22453-900 - Rio de Janeiro, Brazil

② DIMAp – UFRN
Campus Universitário, Lagoa Nova
59072-970 - Natal-RN, Brazil

E-mail: marcel@telemidia.puc-rio.br, {guido, thais}@dimap.ufrn.br, lfgs@telemidia.puc-rio.br.

ABSTRACT

This paper presents Secure MultiMedia Mail System (SMMM), a secure multimedia electronic mail system that deals with issues related to security, network traffic reduction and storage management of multimedia messages. The composition of M/H (multimedia/hypermedia) messages in SMMM is based on NCM (Nested Context Model), a conceptual model for hypermedia documents. SMMM provides support for multimedia message exchange with security services that can be independently defined for messages and for their contents. SMMM is a platform independent mail system implemented in Java™ that uses widespread cryptographic mechanisms and is built on top of existing mail transport protocols for interoperability reasons.

Keywords: electronic mail, hypermedia mail, document composition, security services and secure e-mail messages.

1 INTRODUCTION

Most of existing tools and current standards for multimedia mail, as for example the Multipurpose Internet Mail Extensions (MIME) [2], do not include enough support for hypertext facilities. The integration of hypermedia systems with e-mail transport and storage systems can give the need answer to support hypermedia document exchanges.

Hypermedia documents should be based on a clearly defined conceptual model. In this paper, a conceptual model for hyperdocuments, called NCM (Nested Contexts Model) [9], is used. As in all models for hypertext documents, NCM is based on the concept of nodes and links. However, links are not embedded in the content of a node, as they are in HTML (as will be seen above, compositions are used to structure documents). Moreover, NCM distinguishes two basic classes of nodes, called content nodes and composite nodes. A *content node* contains data whose internal structure is application dependent, modeling a traditional hypermedia node. Content nodes may be specialized in other classes (text, graphic, audio, video, etc.) as required by applications. The logical structuring of a document is done using the composition concept (*composite node*) as a group of document's components (that is, nodes, including composite nodes, recursively) and their relationships (represented by links).

Although NCM has been chosen as the conceptual document model for the proposed mail system, it should be stressed that any other hypermedia model with composite nodes could be used without any loss of generality.

As e-mail is one of the most used applications, it is naturally the target of several attacks. To deal with this problem some security models were developed, such as *Pretty Good Privacy* (PGP) and *Privacy Enhanced Mail* (PEM) [8]. Nevertheless, these systems do not support multimedia facilities.

There are multimedia e-mail systems that deal with multimedia and security issues. However, they fail to handle structured messages they exchange (the components are simply added to the message). They are not flexible on applying security services to message components. For example, a user must always select one level of security to all components. Different levels of security for different parts of a message are not allowed.[11]

This paper presents the Secure MultiMedia Mail System (SMMM). The main contribution of the proposed approach is the support provided for multimedia message exchange with security services that can be independently defined for messages and for their contents.

The requirements that guided the SMMM development are:

- The media types used should be extensible, although the use of some media may seem awkward on now existing hardware.
- Existing mail transport agents should be used. As, in general, these agents only support ASCII transfer, the messages should consist of an ASCII representation of referenced hypermedia documents.
- The security mechanisms used should not be patent or export restricted.
- The security algorithms used should be well known and heavily tested.
- The security mechanisms should be applied only to the sensitive parts of the messages.
- The storage space required should be minimized. This is one of the main challenges regarding multimedia e-mail, since media such as audio and video may use a huge amount of memory. If the same media content (represented by a node) pertains to more than one message, or to more than one part of the same message, it should be referenced and not replicated.
- The network bandwidth required should be minimized. In the case of the previous item, the node that already exists in the destination should not be transmitted. Moreover, only nodes that the mail receiver agent is able to handle should be transmitted. In the case that some media, such as video, audio and image, could not be supported, the system should work as a formatted text mail, notifying the missing parts.

This paper is organized as follows. Section 2 describes the basic concepts of SMMM. Section 3 presents the implementation issues of SMMM. Section 4 gives the conclusions.

2 SMMM

SMMM is based on SMHMS [10] and MMM [1], multimedia mail systems created as part of the HyperProp project [9]. SMHMS uses NCM (Nested Context Model) [9] to compose M/H (multimedia/hypermedia) messages. MMM is a version of SMHMS integrated with the WWW.

This section starts defining the data format used in the system. Finally, it describes the system architecture.

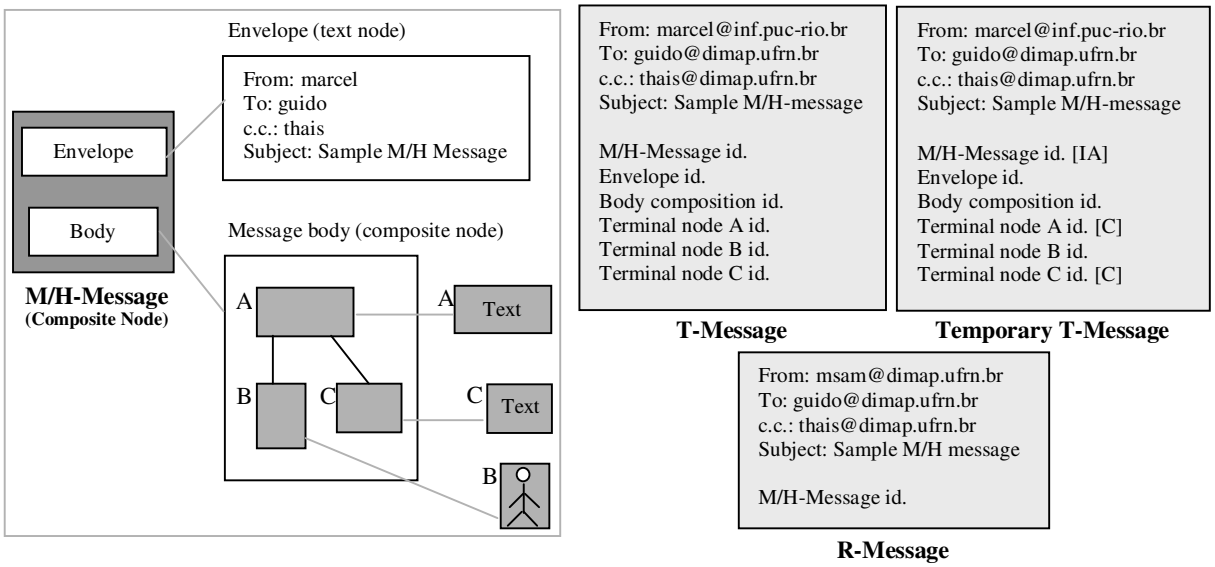

Figure 1 - Messages in SMMM

2.1 The Interchanged Data Format

In SMMM, a composite or a content node is named *M/H-document*. An SMMM message is described by an M/H-message. An M/H-message is a composite node that contains a text node, named **envelope** (whose content is the usual e-mail header, defined in RFC 822 [3]) and an M/H-document, named **message body** (see Figure 1).

The creation of an M/H-message begins with the selection of its components. After that, security services are defined and applied to some components. Finally, information to execute the security services is provided to the system, which can then send the message.

As mentioned, one of the system requirements is to use standard mail facilities. As standards for mail transport agents deal only with ASCII messages, SMMM derives an ASCII message from the identifiers of the recursively contained M/H-message components, as shown in Figure 1 This derived ASCII message, called *temporary transfer message (temporary T-message)*, is then forwarded to the SMMM local domain server (discussed in Section 2.2) where it is transformed in a *final-form transfer message* (or, simply, *T-message*). The T-message is also in conformance with the standard defined by RFC 822. To identify an SMMM message, the field **Content-Type** is set to **text/x-smmm** in the message header. To transfer the T-message, SMMM uses the Internet conventional electronic mail transfer system, based on SMTP (Simple Mail Transfer Protocol) [7].

Therefore, the transferred message is short (some systems constrain the size of the messages interchanged by them) and the destination system will be able to retrieve only the components it is able to handle. Thus, useless network traffic can be avoided [1].

The M/H-message data model provides a high degree of flexibility. As any node can be contained in different composite nodes, messages in the server, regardless their destination users, can share the same M/H-message body, or any of its components, according to the access rights defined by their authors. In addition, the message body (M/H-document) can be created outside SMMM and included in the M/H-message later on.

At the destination network, when the system recognizes an SMMM message, it checks for the availability of each M/H-message component in the M/H server of its domain. Unavailable components are then retrieved, if they can be presented with the hardware and software resources available at the destination.

An *R-message* is then created and stored in the user's mailbox. The difference between an R-message and its peer T-message is that the first one only has the M/H-message identification (see Figure 1).

Additional information about the M/H-message sending and receiving processes can be found in Section 3.

2.2 System Architecture

Figure 2 shows, as an example, a scenery with two networks, A and B, with the same configuration: an SMTP server station, an M/H server station and two (or more) user stations.

The SMTP server station runs a conventional mail server, using the SMTP protocol. This server sends messages via TCP/IP and stores them in conventional user mailboxes. It redirects received M/H-messages (indeed, represented by T-messages) to the SMMM server that processes them.

The M/H server station has three components: an M/H-document server (M/H server), an SMMM server (Message Manager) and a Key Manager (KM). The SMMM server receives messages (temporary T-messages), at the source network, from the source user station, and apply security services defined by the user, generating new T-messages. At the destination network, it identifies the message components and then asks the M/H server to retrieve the remote ones which are not already stored in the local M/H server and that can be handled by the hardware and software resources of the destination site. The M/H server stores and retrieves M/H-documents by accessing the repositories of M/H-documents and secure documents. When the retrieval finishes, the destination SMMM server generates an R-message (corresponding to the received T-message) and stores it in the conventional user mailbox using the SMTP server. The KM manages, stores and retrieves the user's keys, necessary to the execution of the security services.

Each user station runs an e-mail user interface used to compose conventional mail messages. This interface is also used to define the security services that will be applied on documents, by the SMMM server. Additionally, the user station must run an M/H-document interface for creating and presenting M/H-messages (it accesses the M/H server via an

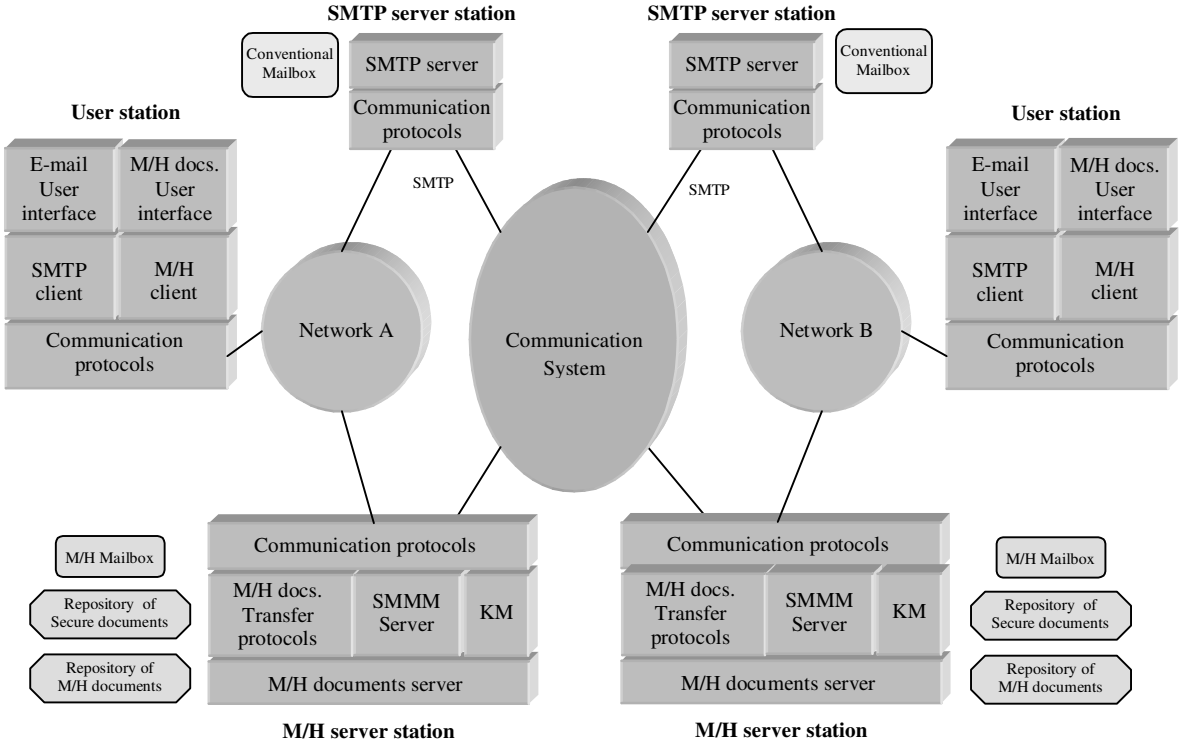

Figure 2 - SMMM Architecture

M/H client). The user station runs an SMTP client to transfer T-messages and receive R-messages.

3 SMMM IMPLEMENTATION

SMMM was implemented in Java™ because of its inherent portability.

This section describes the composition of M/H-messages. Next, it discusses the delivering and reception of M/H-messages and, finally, the message visualization process.

3.1 M/H-message Composition

The composition of an M/H-message starts by filling the envelope information. After, the user may edit an ASCII message or select the components of an M/H-message.

The message composition continues with the security service settings. The user may choose the message nodes where the security services have to be executed. These nodes are called **sensitive nodes** of a message.

The security services[1] SMMM provides are: Confidentiality, Integrity and Sender authentication.

When the user asks to send a message, the temporary T-message is generated and automatically sent to the SMMM server of its domain. This message contains a set of identifiers (ID) which describes all documents contained in the message and the security services to be applied on them, as showed in Figure 1. The first ID is a reference to the M/H-message.

3.2 M/H-message Delivery

The SMMM server has a component that runs the security services configured by the message author. The security services are implemented using security mechanisms, such as: hash functions (MD5), public (RSA) and private-key cryptographic algorithms (DES) [8]. Keys required by cryptographic algorithms are stored in a key database (keystore). All system users have a **keystore**, where private keys, public keys and partners' public keys are stored[2]. Each key in the keystore is identified by an alias and has a password. The **key manager (KM)**, which is performed by the KeyTool™ (provided by Java2™ platform) in the present version of SMMM, accesses the keystore, retrieving the keys the SMMM server needs to execute the security services configured by the user.

To provide confidentiality to a message, the SMMM server uses public and private-key cryptographic algorithms and a hash function. The receiver public-key must be provided to the SMMM server.

The integrity service is implemented with a public-key cryptographic algorithm and a hash function. Again, the SMMM server needs the receiver's public-key.

Finally, to implement sender authentication, SMMM uses the same mechanisms used to provide confidentiality. However, in this case, it is necessary to retrieve the sender's private-key.

Usually, various security services are applied simultaneously. In this case, the following order must be obeyed: confidentiality, integrity and source authentication.

Note that, whenever sensitive data is interchanged by internal system components, it is done via secure channels, implemented with public-key cryptographic mechanisms.

When the encryption process finishes, the secure components are moved to the repository of secure documents. Simultaneously to the application of the security services, the SMMM server generates the final T-message. The content of this T-message must reflect the changes made in the components (the identifiers of the temporary T-message are updated to reflect the new locations). Finally, the T-message is sent to the SMTP server, which is then responsible for its

[1] The security services provided by SMMM follow the guidelines given by ISO 7498-2 document, regarding TCP/IP characteristics. [6]

[2] All keys stored in a *keystore* must be certified.

delivery.

3.3 M/H-message Reception

The reception of an M/H-message is divided in three parts: reception of the T-message, retrieval of its remote components and generation of the corresponding R-message.

Messages are received by the conventional e-mail server through the **sendmail** program, which redirects all them to the **procmail** program that is responsible for detecting incoming T-messages. If the message Content-Type field is set to **text/x-smmm**, procmail redirects it to the SMMM server. Otherwise, the message is stored in the user's conventional mailbox.

When the SMMM server receives a T-message, it parses the message body. For each ID found, it verifies if the referenced document is locally stored or not. If not, the document is retrieved by HTTP (if the hardware and software resources of the destination site support its media type).

The M/H-documents server maintains a **Document Table (DT)** to administer all local documents. This table associates each local file with its original location. The search for a component in the server is done using its original name. If it is not locally stored, a new row *<original name, local name, date of the last modification, occurrences>* is added to the table. Even when a document is locally stored, its date is compared with that one in DT. If the dates are different (different versions of a same document), the document referred in the message body is retrieved and stored conveniently.

As mentioned, the SMMM server considers hardware and software constraints when retrieving the remote components of the messages. To accomplish this, a configuration file (profile) lists the media handled by the system. Thus, components with media that cannot be handled are never retrieved.

After all T-message components have been retrieved (or detected as locally available), the SMMM server generates the R-message and stores it in the recipient's mailbox.

3.4 M/H-message Visualization

Received messages listed by the mail interface are played properly. Proper players are started to show the messages' nodes, depending on their MIME types.

When visualizing secure components of an M/H-message, different information is needed according to the security services applied. For instance, if the service applied to a component is confidentiality, the user has to inform his private key, giving the alias and corresponding password, available in the user's keystore. Such information is safely sent to the SMMM server that, with the proper keys retrieved from the keystore by the KM, decrypts the document's nodes and safely deliver them to the user interface.

4 CONCLUSIONS

The Secure MultiMedia Mail is a platform independent mail system implemented in Java™. It uses a hypermedia model to structure M/H-messages that benefits the storage space and network traffic minimization and also facilitates the interchange of M/H-documents.

The system introduces new concepts, such as the specification of sensitive nodes. When composing a message, the user can select message components that need security services, since it is not usually necessary to employ security mechanisms to all components. It is also possible to specify different security services to each sensitive component. Non-repudiation is the only service not provided by SMMM and will be added in future versions.

The mechanisms used to provide security services in the present SMMM version are DES (private-key cryptography), RSA (public-key cryptography), and MD5 (hash function). However, the open architecture of SMMM allows that other cryptography and hash algorithms may be easily added in future versions in order to deal more efficiently with video and audio components.

Due to security restrictions, in the present SMMM implementation some desirable multimedia requirements — such as storage space saving and network traffic reduction — were not deeply explored. When interchanging secure multimedia messages it was necessary to generate several identical files to different message receiver, when they had different keys. To solve this problem, an approach based on access levels was developed. This approach considers user, group and domain keys. When sending a message to several receivers belonging to a group with a unique key, it is not necessary to generate a copy to each one, thus saving space on the source and destination servers and saving network bandwidth. This feature, however, is not implemented yet.

5 REFERENCES

[1] Batista, T.; Rodriguez, N. L. R.; Soares, L. F. G. & Resende, M. C. "Hypermedia Mail over WWW" Proceedings of the 3rd International Workshop on Community Networking (CN3). Antwerp – Belgium - May 1996.

[2] Borestein, N & Freed, N. "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", *RFC 2045*. November 1996.

[3] Crocker, D.H. "Standard for the Format of ARPA Internet Text Messages". RFC 822. August 1982

[4] Elkins, M. "MIME Security with Pretty Good Privacy (PGP)", RFC 2015, October 1996.

[5] Halasz, F.G. "Reflections on NoteCards: Seven Issues for the Next Generation of Hypermedia Systems". *Communications of the ACM*. Vol. 31, No. 7, July 1988.

[6] Lynch, D. C. & Rose, M. T. Internet System Handbook. Addison-Wesley Publishing Company, Inc., 1993.

[7] Postel, J.B. "Simple Mail Transfer Protocol", RFC 821, August 1982.

[8] Schneier, B. E-Mail Security — How to Keep your Electronic Messages Private. John Wiley & Sons. 1995.

[9] Soares, L.F.G.; Casanova, M.A. & Rodriguez, N.R. "Nested Composite Nodes and Version Control in an Open Hypermedia System". International Journal on Information Systems; Special issue on Multimedia Information Systems, vol. 20, no. 6. Elsevier Science Ltd. England. September 1995. pp. 501-520.

[10] Soares, R.A.M. & Soares, L.F.G. "SMHMS: A Multimedia/Hypermedia E-mail System", Proceedings of the 12th Computer Networks Brazilian Symposium (SBRC), Curitiba – Brazil, May 1994.

[11] Wiegel, Burkhard. "Secure External References in Multimedia Mail Messages", at 3rd ACM Conference on Computer and System Security, 14-16. March 96, New Delhi, India.