# Secure document management and distribution in an open network environment

Antonio Lioy, Fabio Maino, Marco Mezzalama

Politecnico di Torino,
Dip. di Automatica e Informatica,
Torino, Italy

**Abstract.** This paper analyzes the problem of secure document management and distribution in an open network environment. Reader and author authentication, document integrity, origin, and privacy are addressed by a public-key based solution which exploits a combination of the PEM format with SSL-enhanced FTP and HTTP servers and clients. The solution is being implemented as part of a project to provide network security to the Italian public administration.

## 1 Introduction

The work presented in this paper origins from the need of the Italian public administration to securely exchange documents externally with the citizens and internally between various departments.

Within this general framework, several initiatives are being carried out; many of them share the common architecture of a central repository to store and distribute documents with different security levels. The documents must be accessible to several people but the security levels and the distribution lists must be individually controlled by the author of the document. In other words, the central repository is used only as a distribution point but the minimum level of trust compatible with its functionality is put in its operation. This is usually due to the problem that a central repository is useful for the users of the information but it is normally external to many organizations, which therefore don't trust it very much as they don't have full control over it.

The analysis of the users' needs has led to the identification of several system requirements that can be summarized by the following terms:

**document integrity** the document must contain code to prove that it has not been modified in any way since it was generated and sealed

**document origin authentication** the document must contain information to prove the (electronic) identity of its author and must be useful for non-repudiation

**document privacy** if required, the document must be readable only by authorized users; this must be true even in case the document is stolen from the repository that distributes it

**document destination authentication** the system must keep track of the identity of people who have received a copy of a document

**secure remote document management** it must be possible to remotely manage the repository in a secure way, including document addition/removal and authorization granting/revoking

On the implementation ground, the system has to be an open one: people that interact are not supposed to belong to the same organization and not even to the same country. It was therefore a natural choice to base security on public-key techniques and X.509 certificates. This allows each actor to be identified in his own country (with his own language and procedures).

Additionally we wanted the system to work equally well in an Intranet, Extranet, or Internet environment (i.e. LAN or private/public WAN), with non-proprietary components as much as possible. Therefore the system had to work in a TCP/IP network with standard protocols.

Before developing our own solution, we surveyed various products that could possibly satisfy the requirements. We have found that existing proposals for cooperative systems don't provide a strong solution to reader and author authentication, document integrity and privacy. As stated in [1] the reason can be twofold: first, access control models for groupware tend to be complex and require a deep analysis of relations between principals and objects involved in the cooperative system; second, the need for a running prototype normally postpones the security issues to a "future" release of the product.

To see how security is managed in cooperative systems, we will shortly analyze three very different approaches to groupware:

- BSCW *(Basic Support for Cooperative Work)* [2], a shared workspace written in the Python programming language, freely available for non-commercial use from GMD, the Germany's national Research Center for Information Technology
- DOMINO, the Lotus Development Corporation product that, integrated with the Lotus Notes suite of products, offers an integrated set of services for groupware
- a "self-made" solution for cooperative work based on the filesystem workgroup sharing of a Windows-NT WWW server.

BSCW is widely open to different platforms and operating systems: emphasis is placed on the sharing of documents and security issues have been addressed in the latest version only. The HTTP basic authentication method is used to authenticate both readers and writers, but this is a very weak kind of authentication that clearly exposes the readers and author's password. A small improvement is obtained by using an SSL protected channel, but nothing is done in order to grant document integrity or protection of reserved documents when stored on the server.

The Lotus solution acts as a front-end to the World Wide Web for documents stored in a Lotus Notes database: using private Lotus client authors can spread documents over the Internet through DOMINO. Author authentication is done using standard X.509 certificates, that allows documents to be signed or encrypted even when stored in the database. On the WWW side, the use of SSL

allows channel-oriented security and reader authentication but document security is restricted only to those users equipped with Notes clients. This, of course, forbids the diffusion of the products to companies that have an heterogeneous set of applications. Moreover the restrictions on exporting cryptographic products from USA allow the use of short-key encryption only, significantly reducing the security level for cooperative systems outside the United States.

The wide diffusion of corporate networks based on Windows-NT suggests the use of its workgroup sharing capabilities, in order to allow access to the filesystem of a NT WWW server. This *naive* approach to cooperative working is widely used in small Intranets thanks to its easy implementation and management. By using the NT audit capabilities it is possible to set up a simple logging of writer's access to shared documents, but this solution doesn't seem to provide enough protection to documents inserted from different authors of the same workgroup. No kind of encryption, integrity, or authentication is available for stored documents that are unprotected from unauthorized access via the WWW server.

The analysis of these solutions shows the need for a cooperative system really open to different platforms and operating systems, based on simple WWW interfaces and able to ensure document integrity, origin, destination authentication, and privacy, additionally the system must be securely manageable from a remote site.

Since, for different reasons, none of the various products examined satisfied all our requirements we moved on to designing our own solution, which is described in the sequel of the paper.

## 2 System architecture

Three different roles can be identified in the system. The *author* is the one who generated a document and he is the only one that controls its security level and distribution scope. The *document master* is the system manager in charge of properly setting up and managing the server used as document repository. As this role and this server are often external to the author's organization (e.g. because they are outsourced), we want to place the minimum level of trust in their actions. Finally, the *reader* is any user wishing to get an electronic copy of a document.

There is no restriction about the basic document formats supported: any standard electronic format (Postscript, PDF, GIF, JPEG, MS-Word, ...) can be managed because the document itself is treated as an opaque field as long as the security functions are concerned.

The system is designed to handle two security levels: *open* and *reserved* documents. In both cases, the document is sealed in an envelope which carries an integrity and origin authentication code. We find useless in a formal environment to provide electronic documents without ensuring at least these two basic security properties. Additionally, a reserved document is also encrypted to prevent its disclosure to unauthorized parties.

### 2.1 Secure document distribution

Open documents have unlimited distribution and are immediately delivered by the document server upon request, with no need for reader's authentication or authorization checks (see Fig. 1).

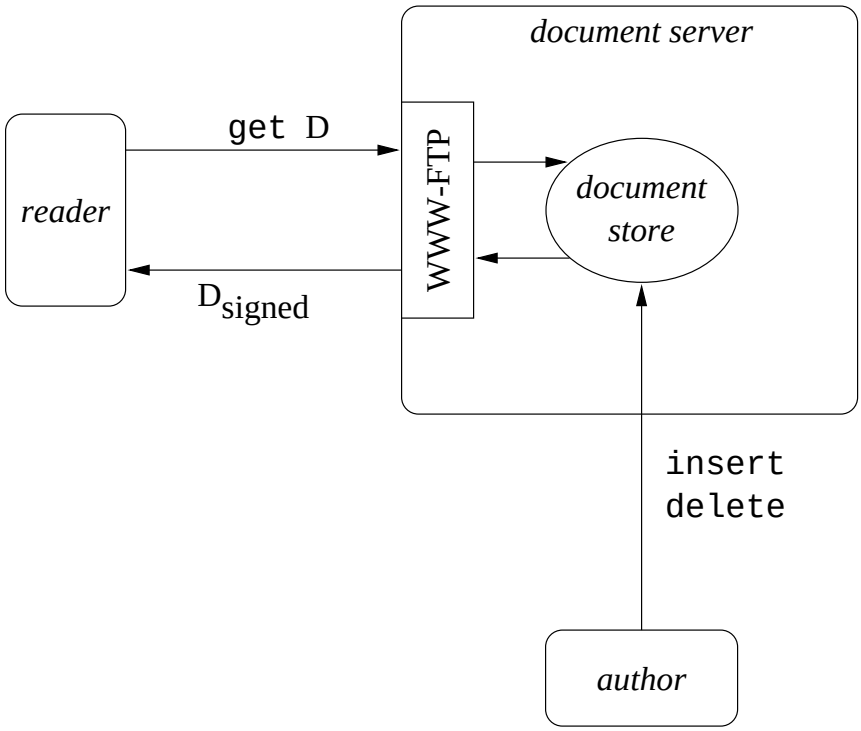


**Fig. 1.** System architecture for documents of the open type

On the contrary, a reserved document is not only encrypted but has limited distribution too (see Fig. 2). A pre-requisite to access a reserved document is that the reader be authenticated. We chose to use public-key authentication based on the RSA algorithm and X.509v3 public-key certificates. In turn this requires the existence of a public-key infrastructure (PKI) with certification authorities (CA) that emit the public-key certificates to formally bind public-keys to personal attributes, such as an identity. Since there is not yet a world-wide accepted PKI, our system is able to trust several roots simultaneously. However, our preference (and default root) is that of the ICE-TEL European project [3] which has set up an infrastructure which is increasingly being used in several countries of the European Union.

Limited distribution is enforced by means of an *access control list* (ACL): each document in the repository has an associated ACL which contains the list of all

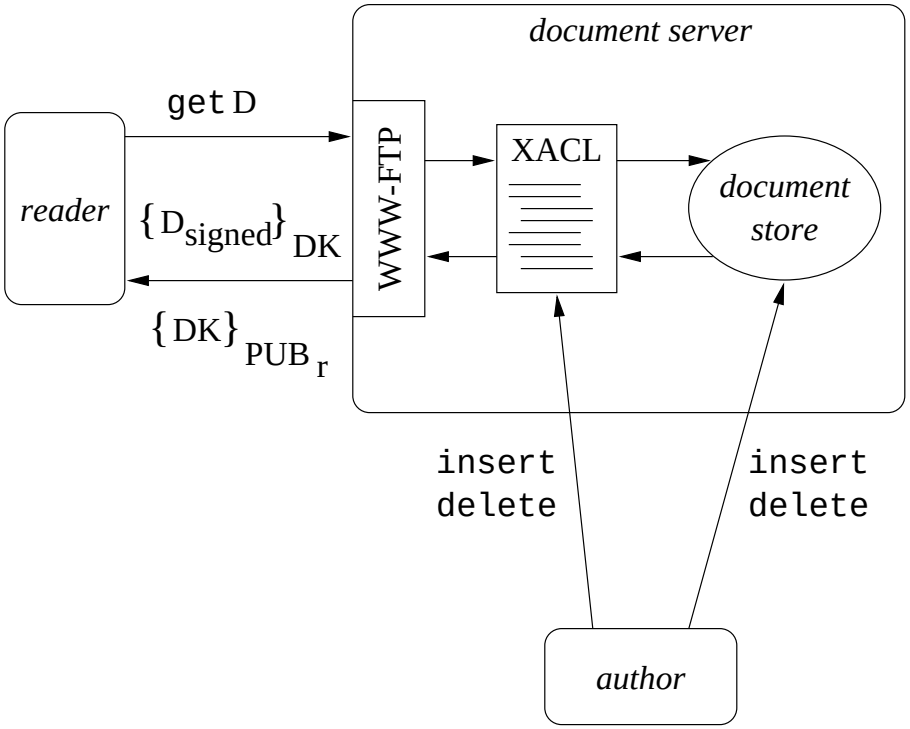


**Fig. 2.** System architecture for documents of the reserved type

the distinguished names (DN) authorized to retrieve a copy of the document.

Privacy is guaranteed by storing documents on the server only in encrypted form. This shields the documents from direct attacks by the document master as well as from illegal distribution. To allow the authorized recipients of a document to decrypt it we use a modified ACL, named an *extended ACL* or XACL. To describe its operation, let DK be the secret key used to encrypt the document D. The extended ACL for D is the set of pairs that associate the DN of an authorized reader to the document key encrypted with the public-key of the reader. This can be formally expressed as[1]:

$$\mathrm{XACL(D)} = \{\ \langle \mathrm{DN}_i, \{\mathrm{DK}\}\mathrm{PUB}_i \rangle : \forall\, i \in \{\textit{authorized users for document } D\}\ \}$$

In this way, when a generic user X requests a reserved document the following actions take place:

1. X authenticates himself via the SSL protocol, either to the WWW server or to the FTP one

[1] The notation $\{X\}K$ represents X encrypted with the key K.

2. the server extracts the user's identity and public key from the X.509 certificate received from the client and checks them against those contained in the the XACL components:

$$\text{authorized(X)} \ \textit{iff} \ \exists\, i : \text{DN}_X = \text{DN}_i \ \wedge \ \langle DN_i, \{\text{DK}\}\text{PUB}_i \rangle \in \text{XACL(D)}$$

3. if in the previous step authorization is granted, then the document is delivered to the user along with the key to decrypt it:

$$\text{X} \longleftarrow \{\text{D}\}\text{DK}, \{\text{DK}\}\text{PUB}_X$$

4. when the reader receives the document, an helper application is available to check its integrity and authentication and to decrypt it to permit viewing and/or printing.

It should be noticed that on the reader's side documents are always saved in their protected form. In particular reserved documents are kept encrypted to prevent unauthorized disclosure. The helper application permits to save the document in an unprotected form too, but this is strongly discouraged: the user is warned that his action lowers the security of the system and he is legally responsible in case that the unprotected copy is illegally disclosed. (As an aside, this issue has stirred interest in investigating methods to insert into the distributed documents deliberate modifications to permit tracing documents even in their unprotected form. However, this is a completely different research.)

As long as the implementation is concerned, we chose the PEM message format [4] to seal and encrypt the document, mainly due to its simplicity and flexibility. This is in contrast with the current industrial emphasis on S/MIME as preferred format for securing e-mail messages. However, we feel that S/MIME has several disadvantages if applied in our context:

- it is more complex than PEM
- it is less flexible than PEM as long as the specification of the encryption, digest, and authentication algorithms is concerned
- its standardization status is still uncertain, as it has not yet attained RFC status and several of its components are under control of RSADSI only

For all these reasons our first implementation uses PEM with triple-DES, MD5, and 1024-bit RSA. We are already considering to move towards RC5, IDEA, and SHA. Nonetheless, there is no technical reason which prevents the use of S/MIME as an envelope format for the document and we will closely monitor the evolution of this format - especially with respect to standardization issues - ready to adopt it when it will eventually become widely adopted.

We have also paid attention to the DSIG initiative [5] of the W3C but it was found unsuitable to our context as it provides digital signatures only (i.e. integrity and authentication) and doesn't address document privacy. Moreover DSIG is strictly bound to HTML pages, which makes it difficult to be used as an encapsulation format for objects that can live outside of the WWW too.

The PEM-encapsulated documents are distributed by an SSL-enabled WWW or FTP server that has been modified to properly handle XACLs. In this context, SSL [6] is used to provide server authentication when distributing open documents, and mutual authentication (i.e. both client and server authentication) for the reserved documents.

### 2.2 Secure remote document management

One of the system requirements was to put the author in complete control of the privacy and distribution of the documents. This target has been achieved by developing a simple document management interface that allows an author to perform on his local workstation several tasks:

- to choose a document from the local storage to be uploaded to the repository
- to seal a document with the proper integrity and authentication codes (i.e. to digitally sign the document)
- to encrypt the document with a random key, which is maintained in a local personal security environment for future use
- to upload a document (sealed and possibly encrypted) to the repository
- to manage the XACL for a document D, by adding, deleting, or modifying entries
- to upload the XACL to the repository and to compare the local XACL with the one on the server, for verification purposes

The interface has been developed by using the Tcl/Tk language for the GUI, and our own version of SSL-ftp to provide channel security in the form of mutual authentication between the author's workstation and the repository. In particular, SSL-ftp has been chosen because the SSL protocol provides adjustable channel security (single or mutual authentication and integrity with or without encryption) while the FTP protocol allows remote management of a file system in a way very similar to that possible with a local one.

Since availability for several platforms is an issue in our project, we developed our own version of SSL-ftp by starting from the WU-ftpd package and the SSLeay crypto library [7]. The resulting package implements an SSL-ftp client for WIN32 and UNIX machines, while the server side is currently available for UNIX servers only (a Windows-NT porting is in progress). We are looking into porting the client to the Macintosh environment too.

## 3 Open issues and future work

The system has been designed to resist to a certain number of attacks but others are still possible. For example, the system is vulnerable to denial-of-service attacks leaded by the document manager who can refuse documents to authorized users or provide them with the wrong key. While this is annoying from the reader's point of view, it doesn't lower the security of the documents in any way: reserved data can never be exposed to unauthorized people. In the same

fashion, if logging of the people who received a document is wanted this can be altered by the document manager as well.

However the biggest problem we have still to solve is related to ACLs: they work fine for a limited number of users but are impractical for large sets. A solution can be searched in using some form of group authorization. However it was an explicit design choice to avoid group authentication to enforce the principle of personal responsibility. We are therefore looking into using groups only for the DN-based part of the authorization procedure, while maintaining disclosure of the document key strictly connected to personal keys. For example, in this case a group entry in an XACL could look like this:

$$\langle \texttt{(C=IT, O="Politecnico di Torino", CN=*)} , \textit{empty} \rangle$$

With this syntax we would authorize all the people from the Politecnico di Torino to access a document; however, in this case the component $\{DK\}PUB_i$ cannot be pre-computed and must be generated on the fly as needed. This has the obvious disadvantage of requiring the author to be on-line when a request arrives but, at the same time, offers a solution for secure logging and for key revocation. We will continue to explore this issue and look for alternative solutions too.

We are continuing the development and the deployment of the system to gain valuable feedback from the user community, ready to improve the architecture according to the user requests.

## Acknowledgment

This project is being carried out with the support of CNR (the Italian national research council) under grant "Progetto strategico Informatica nella Pubblica Amministrazione - sottoprogetto DEMOSTENE".

## References

1. K. Sikkel, "A Group-based Authorization Model for Cooperative Systems", *Proceedings European Conference on Computer-Supported Cooperative Work (EC-SCW'97)*, Lancaster (UK), September 1997.
2. R. Bentley, W. Appelt, U. Busbach, E. Hinrichs, D. Kerr, S. Sikkel, J. Trevor, G. Woetzel, "Basic Support for Cooperative Work on the World Wide Web", *International Journal of Human-Computer Studies: Special issue on Innovative Applications of the World Wide Web*, Spring 1997
3. "ICE-TEL: Interworking Public-Key Certification Infrastructure for Europe", project RE 1005 of the Telematics programme of the European Commission `http://www.darmstadt.gmd.de/ice-tel/`
4. J. Linn, "Privacy Enhancement for Internet Electronic Mail, Part I: Message Encryption and Authentication Procedure", RFC-1421, February 1993
5. "Digital Signature Initiative", World Wide Web Consortium `http://www.w3.org/Security/DSig/Overview.html`

6. A.O. Freier, P.L. Karlton, P.C. Kocher, "The SSL Protocol (version 3.0)", Netscape Communications Corporation
`http://home.netscape.com/eng/ssl3`
7. E.A. Young, T .Hudson, "SSLeay and SSLapps FAQ",
`http://psych.psy.uq.oz.au/~ftp/Crypto/`

This article was processed using the LaTeX macro package with LLNCS style