Secure Automated Document Delivery

Tom Casey *Michael Roe* *Bill Tuck* *Steve Wilbur*

Department of Computer Science
University College London

Abstract

A document retrieval system is a specific type of distributed processing which has its own particular security requirements. This paper proposes a model for secure delivery of documents and describes a prototype system, based on earlier work on secure electronic mail and automated document delivery systems, being developed at University College London.

In the proposed architecture, security protection is provided for document requests and the actual documents delivered. E-mail protocols are used for document requests and delivery, although file transfer protocols could be used in some circumstances. The paper begins with a discussion of the document delivery system background and then sets out the client–server model for the secure system. The security philosophy, requirements, policy, and techniques are dealt with next. The criterion for validation is analyzed; the relationship to OSI is shown; implementation issues are discussed and the direction of future efforts is pointed out.

1 Background

The Secure Automated Document Delivery System (SADDS) described in this paper is an application program which provides security for an automated document delivery system. The system merges several technologies, prototypes of which are currently under development at University College London; privacy enhanced e-mail [4], directory support for secure messaging [17], and automated document delivery systems [18].

A prototype Automated Document Delivery System (ADDS) is the fundamental model to which security features are to be added. This ADDS prototype was developed under Quartet, a British Library research project. It provides an automated system for searching bibliographic databases, initiating requests to the document filestore, and the mechanism for document delivery to the end user. In some distributed environments, where security is not a concern, the ADDS fulfills many of the requirements for accessing and retrieving documents from a central depository. Yet in other circumstances security is necessary.

The principles underlying a SADDS are applicable in many data processing environments where users are provided with read only access to a document database. Particularly relevant are applications where document requests originate at remote sites, and clients, for security reasons, are denied access privileges to the host server. The SADDS effectively allows the client to perform a restricted set of operations on a collection of objects. Client-host session establishment and the concomitant retention of system state information is not an operational requirement. The ability to service document requests in an accountable manner with limited access rights to the underlying host system has clear advantages over other types of security services. Furthermore, the system makes no assumptions about the security

[0] This work was supported by grants from the British Library Research and Development Department (project Quartet), the Alvey programme (project LOCATOR) and the DELTA programme (project CAPTIVE).

of the underlying e-mail facility or the communications links over which it operates.

A prime application could be a multipurpose or multinational space station where communications links may need to be shared. Diversity of political and economic interest present difficult problems in a shared environment; integrity, confidentiality, and authentication features of a document delivery system may well address some of these problems.

2 The Model

The model consists of one or more client entities (service requestors), and one server entity (service provider).

For the purposes of clarity, the protocol is explained in terms of a single client. The transactions are identical if there is more than one client. Multiple servers are not considered at this time, but could be included in future extensions.

It should be emphasized that clients do not establish session connections with the server. Transactions are carried out by messages (datagrams) which may be carried by any e-mail protocol. Our implementation of the prototype will use the internet SMTP protocol.

As illustrated in Fig.1, both client and server have a trusted area, their Trusted Computing Base (TCB), in which transactions are processed. All messages are cryptographically protected before exiting the TCB. No assumptions are made about the safety and security of the communications channel between client and server.

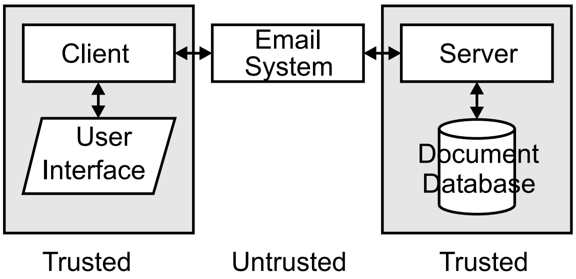

Figure 1: The SADDS System

2.1 The Security Protocol

All messages between a client and a server are protected by the security protocol shown in Fig.2.

$$\text{C} \rightarrow \text{S} : \{K\}S^{(P)}, \{\{C, d, n\}C^{(S)}\}K$$
$$\text{S} \rightarrow \text{C} : \{K\}C^{(P)}, \{\{S, d, n, \text{text}\}S^{(S)}\}K$$

Where:

$C^{(P)}$	client's public key
$S^{(S)}$	server's secret key
K	randomly generated transaction key
d	document id
n	request id
text	the document or error report
S	the name of the server
C	the name of the client

Figure 2: The Protocol

Unique transaction keys are generated for each client–server transaction. An encrypted version of the key is included with each message.

Requests contain a sequence number so that the server is able to reject replayed messages.

Authentication of the error reports is used to ensure that denial of service attacks are always detected. Each request by a client to the server generates a response either in the form of a secure document with guaranteed integrity, or a secure error report. The client records the message ids that it generates and checks them against the replies from the server. If a request is not replied to eventually, this is detected by the client and a warning given to the user.

Additionally, error reports are encrypted as they contain client id, message id, and current sequence number, all of which might be of use to an attacker.

2.2 Client

The client processes run on a physically secure workstation with access to e-mail. A physically protected environment means that the user assumes responsibility for the installed processes, the encryption key, the audit log, any plaintext files, and the processes which create secure requests and handle incoming documents.

Fig.3 shows the structure of the client. To request a document, the client would normally search the local database index and create a well-formed inquiry. This consists of a request id, document id, and the user id. These are written to the transaction log. The client's inquiry is signed and encrypted as in Fig.2. It is then encoded and encapsulated in an e-mail envelope and sent to the e-mail process.

Replies from the server are decoded, decrypted and authenticated. The received documents or error reports are checked against the request id which is held in the client's transaction log.

At this stage the client has access to the document as plaintext. The confidentiality of the document is now the responsibility of the client.

2.3 Server

Fig.4. shows the structure of the server. The processes for security enhancements, error reporting and database access are held in the TCB. The input queue, output queue, and queue manager may reside outside the TCB, as documents, requests, and error reports are in an encrypted form while in transit between client and server.

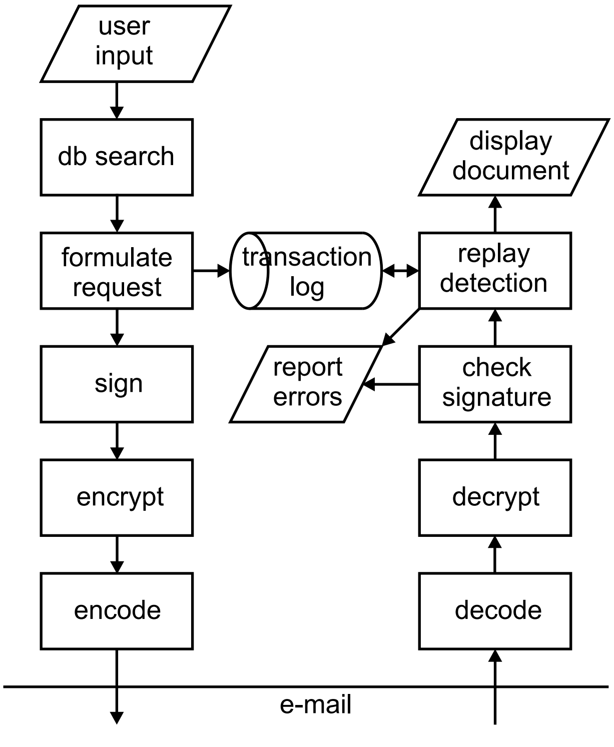

Figure 3: The Client

Client requests are passed from the input queue to the authentication process; the message is decrypted, the digital signature is computed and checked against the user's public key, and the sequence number of the message is checked. Errors at this stage generate error reports and entries into the audit log. Error reports, when generated, are signed, encrypted and passed to the output queue for transmission to the client.

Valid requests are passed to the access control check where user id and document id are compared against a database to determine whether the request is permitted. Errors are treated as above; permitted requests cause a document to be fetched, then signed and encrypted as shown in Fig.2.

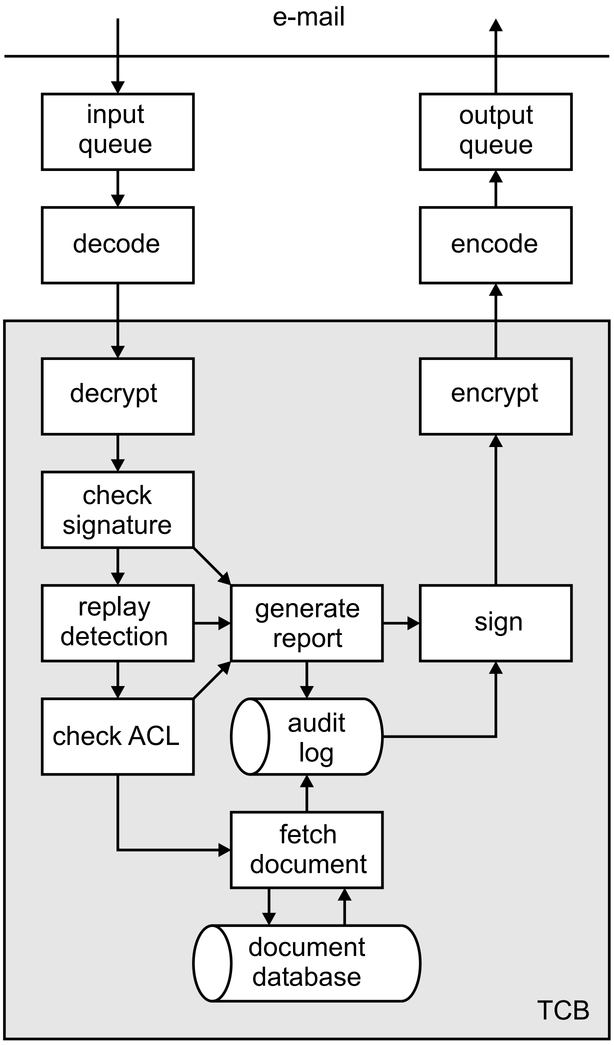

Figure 4: The Server

The document is placed on the output queue and the transaction is recorded in the log. The client's digital signature is recorded so that the client cannot deny having sent the request.

3 Security Philosophy

The underlying security philosophy of SADDS is similar to a doctrine which seems to be evolving among the community of practitioners in the commercial sector. Namely, that security requirements are related to a user's perceived threats, and his assessment of the cost of a security breach. Security, like any other feature of information systems, must be analyzed in terms of costs and quality of service. The value of a potential loss is probably the most a firm would pay for security; and the estimated value of the gain by an intruder is probably the most the intruder would pay to breach the system. These two perceived values may not be the same.

The model we propose makes the cost of protection, i.e. of implementing a SADDS, unequivocally known. How that cost relates to the value of the information being protected can only be determined by the user or potential intruder. Technological change over time affects security costs and requires continuous re-evaluations of the risk and protection measures in use.

4 Security Requirements

The security policy of the SADDS provides the rules by which documents will be managed, protected, and distributed within the domain of clients and servers. The SADDS recognizes the following general security requirements.

- Authentication of clients and the server.
- Integrity of documents, error reports, and requests.
- Non-repudiation of client actions.
- Confidentiality of documents and requests.
- Access control.

The specific prototype model with which we are working is well defined and restrictive. These restrictive characteristics can be revised where greater complexity is necessary. Some of the pertinent features of and assumptions made in the development of this model are as follows: the model deals only with document delivery between a secure server and clients; documents are held by a central server.

The server must be implemented over a trusted computing base which prevents unauthorised access to files and processes. For commercial applications the TCB need not be validated by Orange/Red book criteria. Any secure system would be an acceptable architecture over which the SADDS could be implemented.

The protocol ensures that the copy of a document received by the client is the same as that held by the server. The encryption performed by the SADDS provides data integrity for all transactions between the TCB and the output server; no assumptions are made about the security or reliability of the e-mail system or its communication channel.

At present the client does not have ability to query the database directly, and there is no direct access to the server processes. Client transactions are objects presented to the server in an e-mail message.

5 Security Techniques

There are many ways of preventing data from being read by unauthorised persons — the most obvious being to physically prevent access. However, it is often the case that the data must be transmitted over some public network (for example, the telephone network), which is difficult or impossible to make physically secure. In these cases, protection can be provided by encrypting the data before transmission.

The earliest forms of cryptosystems were *symmetric* — that is, they use the same key for both enciphering and deciphering a message. They have the major disadvantage that the key must be exchanged between the communicating partners (so that the intended recipient can decrypt the message), while at the same time the key must be kept secret from everyone else.

Typically, this is achieved by using a physically secure (but low bandwidth) channel to transmit the key. For example, the system operator is given a piece of paper with the key written on, which she then types in.

Symmetric key cryptography reached its apotheosis in 1987 with the Data Encryption Standard [12]. Since then, most applications requiring a symmetric cryptosystem have used DES.

Asymmetric (*public-key*) cryptosystems [9, 16] simplify the key distribution process by removing the requirement that encryption keys be kept secret. This makes it possible for there to be a publicly available list of which key to use for communication with each user.

Although the list of keys does not have to be kept secret, it must be kept correct. This requirement led to the concept of a *Certification Authority* [19] — an organisation which is trusted by the users of a secure communication system to provide reliable information as to who has been issued which encryption key.

The main disadvantage of asymmetric encryption algorithms is that they are considerably slower than symmetric ones. However, a SADDS does not provide interactive access, and so the extra few seconds added to the turn around time by using an asymmetric algorithm will not be noticed by the user. (Some manual document delivery systems provide a turn around time of 48 hours or more).

Accordingly, we have chosen to employ asymmetric cryptography in the design of the SADDS. Our pilot implementation will use RSA [16], because this is the most widely known public-key scheme that can provide confidentiality as well as authentication. Although no proof of the intractability of breaking RSA is available, the algorithm has withstood many attempts to attack it over the last few years.

6 Validating the SADDS

There are a variety of methodologies available for validating the security of systems; for example that of the Orange Book [13]. An essential feature of all such approaches is that the security policy is precisely defined, making it explicit which threats are being protected against.

Formal (mathematical) techniques may be used in the validation. To do this the system's behaviour is algebraically described and a proof given that the security policy is never violated. The algebraic descriptions can then be used to generate test cases for the implementation.

Several mathematical techniques are available; the Bell–LaPadula model [1] is favoured (although not mandated) by the U.S Department of Defense [13], although some authors doubt that it is always appropriate [11, 5]. Other techniques, such as that of Burrows, Abadi and Needham [2] may prove more useful for validating distributed systems.

6.1 Orange Book

The Orange Book [13] imposes (depending on classification) the following constraints on the architecture of a secure system:

1. An access control policy is defined and enforced.
2. Users are adequately identified.
3. An audit trail is kept.

Our model of a SADDS allows identification and audit to be as thorough as is desired, as clients are always authenticated to the server and all transactions are recorded.

Access controls are divided into two types, *discretionary* and *mandatory.* In order to be certified at any class above C2, a system must be able to enforce both types.

Mandatory access controls are intended to prevent data crossing security boundaries. They are statements about the state of the whole system, rather than local interactions, which means that they must be enforced by some process with authority over the whole system (a *reference monitor*) and cannot be changed by individual users.

Discretionary access controls restrict who may interact with particular parts of the system, and may be changed by the notional owners of the objects to which they refer.

The main access controls used by a SADDS are discretionary. Each document has an associated *access control list* naming the users or groups of users which are allowed to retrieve it. These access control lists are set by the system administrator when documents are added to the database, and may not be changed by remote users. This prevents the propagation of access rights and eliminates the security problems that may result from it.

Mandatory (information flow) controls can also be provided within the server, by assigning security labels to documents and to clients. It is much easier to add these to a SADDS than, for example, a real-time operating system. This is because new, untrusted 'active' objects cannot be created within the server's TCB (remote users cannot run new programs on the central machine), and so the reference monitor never has the problem of recalculating the security label assigned to a user process.

At higher classifications, any covert channels should be identified. As client s cannot alter documents or access control lists, these cannot be used to transfer information. The server's response time could be used for signalling, but the high variance of the transmission delay in the e-mail system makes this impracticable.

6.2 Red Book

The Red Book [14] extends the Orange Book to distributed applications. The additional evaluation criteria include:

1. Communications Integrity
2. Data Confidentiality
3. Denial of Service
4. Effect of combining TCB's

Integrity and confidentiality depend on the strength of the cryptographic algorithms used, and the correctness of the way that they are used. The Burrows–Abadi–Needham logic [2, 3] can be used to check that encryption is being used correctly; we have done this for the protocol of Fig.2. Our reasons for choosing RSA were explained in the previous section.

The SADDS does not attempt to *prevent* denial of service, as this can only be done by the message transfer system. However, the protocol is designed so that denial of service is always detected.

Even though the server is capable of enforcing information flow controls, the Network Trusted Computer Base consisting of the server and its clients combined may not be; since the client machines are outside the server's trusted computing base, it is impossible for the SADDS server to prevent a client using a document in a way that violates the information flow controls. The best that the server can do is to only give documents to clients that it trusts not to misuse them (as determined by labels, for example). This probably means that the SADDS system can achieve at most a 'C2+' rating (as defined by Red Book) rather than B3.

6.3 Clark–Wilson Model

The Clark–Wilson model [5] sets out four criteria for the evaluation of a secure commercial system:

1. Each user is identified and authenticated.
2. The system maintains an audit log.
3. Data items are manipulated by a restricted set of programs which adhere to well-formed transaction rules.
4. Users are associated with a set of programs they can run that meets the separation of duty rule.

We have already dealt with the first two of these requirements. The third is satisfied because documents are only manipulated by the document fetcher. The separation of duty requirement is met because the access control lists and security labels can only be set by the system man ager, not by clients.

7 Relationship to OSI

Our pilot implementation is based upon the Internet protocol suite, because production implementations of these protocols are available. However, the SADDS model could equally well be implemented over OSI protocols.

The OSI Office Document Architecture standard [21] defines a format for the exchange of 'multi-media' documents. (That is, documents containing pictures or diagrams in addition to text). Upgrading the

SADDS to use ODA is simple, as the SADDS makes no assumptions about the *content* of the documents that it is delivering. The only change needed to the remote clients is to replace the document display program with an ODA multi-media editor. The server is either upgraded to hold documents in ODA format, or continues to hold them in G4 fax format and transmits them with an ODA header indicating that the body is G4 fax.

Similarly, X.400 [20] mail could be used instead of RFC822 [6] to carry messages, with only minimal changes to client and server. Secure X.400(88), when it is available, will provide encryption, decryption and authentication of messages, replacing the RFC1040 [10] component. However, the application-specific functions such as checking the access-control lists of documents, and providing audit trails cannot be taken over by X.400; there will still be a need for a SADDS server to provide the specific service of secure batch access to documents.

8 Implementation

Implementation of the SADDS begins with a protoype model of an automated document delivery system (ADDS) without security features. Such a model was developed for the British Library. That model used a database of sample pages of published papers that were scanned and stored as G3 facsimile images on a PC hard disk. Modems were used to submit request for documents. PC fax cards at the output server and the client PC workstation provided the document delivery functions, with an ethernet LAN providing communications between the PC output server and the Unix machine.

Several of the ADDS features are inappropriate for this pilot. For example, the modem interface on the request side and the PC output server on the server side. Basically that model is being redesigned with an e-mail interface for both the client and server processes. The prototype will run on a secure Unix operating system at the server site, and a physically protected system at each client site.

PEM, a UCL implementation of RFC1040, will be used to provide many of the security features of the model. The encryption/decryption routines of PEM are being updated to use asymmetric cryptography in line with the draft successor to RFC1040. The routines for access control, audit trails, transaction logs, and sequence numbering will have to be written.

G4 Facsimile communications standards already exist, and a large database of G4 Fax format files are accessible for research at UCL. There is a 10 channel broadband analogue video system, "Live-net", that connects eight geographically dispersed colleges of the University of London in the UK. By next year broadband digital data channels will connect these sites. Despite the high bandwidth demands of G4 fax, it is anticipated that the SADDS pilot can be tested using G4 format documents.

9 Summary

This paper specifies a model for a secure document delivery system which is being implemented at UCL. It is a straightforward model which provides a high degree of security in a cost effective manner, and has many desirable features. It is a multiple client, one server model designed to provide authentication, data integrity and confidentiality for a document delivery system.

The system makes no assumptions about the reliability of the underlying network. It can be run over a public packet switch network and is compatible with most public or private e-mail facilities.

An implementation of the model could meet TCSEC class B3 security requirements, but would most likely evaluate to the C2+ category of the red book security criterion for the reasons cited in the validation section of the paper.

The system provides well designed access points into the trusted environment and protects against intrusion through covert or signal channels. Furthermore once the user authentication is complete, all security features of the document delivery system will be transparent to the user.

At present the system is under development in the Department of Computer Science at UCL. Many parts of the system are currently being used by other research projects. It is expected that the system will be operational and tested by the time this paper is presented and some implementation experience can be reported.

References

[1] D. E. Bell and L. J. LaPadula, “Secure Computer System: Unified Exposition and MULTICS interpretation”, *MITRE MTR–2997* (March 1976).

[2] M. Burrows, M. Abadi and R.M. Needham, “Authentication: A Practical Study in Belief and Action”, *Proceedings of the Second Conference on Theoretical Aspects of Reasoning about Knowledge* (1987) pp. 325–342.

[3] M. Burrows, M. Abadi and R.M. Needham, “A Logic of Authentication”, *DEC SRC Research Report 39* (February 1989).

[4] T. J. Casey and S. R. Wilbur “Privacy Enhanced Electronic Mail”, *Proceedings of the Fourth Aerospace Conference on Secure Computer Applications*, IEEE Computer Society, 1988.

[5] D. D. Clark and D. R. Wilson, “A Comparison of Commercial and Military Computer Security Policies”, *Proc. 1987 IEEE Symposium on Security and Privacy.*

[6] D. H. Crocker, “Standard for the Format of ARPA-Internet Text Messages”, *RFC822*, Arpanet Working Group Requests for Comments, DDN Network Information Center, SRI International, Menlo Park, Ca., 1982.

[7] D. Davies and W. L. Price, *Security for Computer Networks*, John Wiley and Sons, 1984.

[8] D. Denning, *Cryptography and Data Security*, Addison-Wesley, 1982.

[9] W. Diffie and M. Hellman, “New Directions in Cryptography”, *IEEE Trans. on Information Theory*, Vol 22(6) pp 644—654 (Nov. 1976).

[10] J. Linn, “Privacy Enhancement for Internet Electronic Mail: Part 1: Message Encipherment and Authentication Procedures”. *RFC1040*, Arpanet Network Working Group IAB Privacy Task Force Request for Comments. BBNCC. (Jan. 1988).

[11] J. McLean, “Reasoning about Security Models”, *Proc. 1987 IEEE Symposium on Security and Privacy.*

[12] National Bureau of Standards (US), “Data Encryption Standard”, *Fed. Info. Process. Stand. Publ. (FIPS PUB) 46* (1977).

[13] National Computer Security Center, *Department of Defense Trusted Computer System Evaluation Criteria*, DoD 5200.28—STD (December 1985).

[14] National Computer Security Center, *Trusted Network Interpretation*, NCSC—TG—005 (July 1987).

[15] R. M. Needham and M. D. Schroeder, “Using Encryption for Authentication in Large Networks of Computers”, *Communications of the ACM* Vol 21(12) pp 993—999 (December 1978).

[16] Rivest, Shamir and Adleman, “A Method for Obtaining Digital Signatures and Public-key Cryptosystems”, *Communications of the ACM 21,2* (February 1978) 120–126.

[17] M. Roe, “Directory Services as Support for Secure Messaging”, *Electronic Messaging Systems ’88*, Blenheim-Online, 1988.

[18] W. Tuck, “Using ISDN for Document Delivery Services”, *Program* Vol. 22, no. 4 (October 1988).

[19] “The Directory — Authentication Framework”, *ISO/IEC International Standard 9594—8.*

[20] “Message Transfer System: Abstract Service Definition and Procedures”, *ISO/IEC Draft International Standard 10021—4.*

[21] “Office Document Architecture (ODA) and Interchange Format: Part 5 — Office Document Interchange Format (ODIF)”, *ISO/IEC International Standard 8613—5.*