THE SECURITY ARCHITECTURE OF THE SECURE MOBILE MESSAGING DEMONSTRATOR

Chris Mitchell, Dave Rush and Michael Walker

Issue c2
13th April 1988

1. INTRODUCTION

Locator is part of the Mobile Information Systems project, a major demonstrator within the Alvey project; the partners within Locator are Hewlett-Packard Ltd., Racal Imaging Systems Ltd., Racal-Milgo Ltd., Racal Research Ltd. and University College, University of London. Locator has as its goal the demonstration of a research prototype secure messaging system with mobile access. The messaging system to be used in the demonstration conforms to the CCITT X.400 Recommendations, (2), (3), (4), (5).

The Locator demonstrator involves the use of three different types of message handling entity, namely User Agents (UAs), Message Stores (MSs) and Message Transfer Agents (MTAs). In the mobile scenario, a mobile UA will communicate with a fixed MS over a cellular radio link. These mobile UAs will be implemented on Hewlett-Packard Portable+ personal computers. In this scenario the use of an MS is essential, because a UA will be unavailable for receiving mail for a large percentage of the time.

The purpose of this paper is to describe the security architecture that has been designed for use in the Locator demonstrator. This architecture is based on the use of certain security features incorporated into the latest drafts of the 1988 versions of the CCITT X.400 and X.500 Recommendations, (2), (3), (4), (5), (6). A companion paper, (7), describes the demonstrator architecture and its physical implementation in greater detail.

The security architecture supports a number of security services which are described in Section 2. In Section 4 the provision of these services is considered in detail, preceded by a discussion of key management issues in Section 3. The paper concludes in Section 5 with a discussion of certain problems that have been encountered in using the security features in the draft X.400 Recommendations.

2. SECURITY SERVICES

The security architecture has been designed to support a number of security services, all bar one of which are 'end-to-end' in nature. The selected services were identified during the Locator Project Definition Study as being those most significant to end users of commercial messaging systems. The end-to-end services are: content confidentiality, message origin authentication, content integrity, non-repudiation of origin, replay detection and non-repudiation of delivery. The other service is access control which, within Locator, is only provided on the UA-MS link, although Draft Recommendations X.411 and X.413, (4), (5), allow it to be provided on all links between message handling entities. It should be observed that the service names we use here are those used in the draft X.400 Recommendations; they do not correspond precisely to the names given in the OSI security architecture, (9).

Within the Locator demonstrator these services may not all be requested independently. For instance, the message origin authentication, content integrity and non-repudiation of origin services are grouped together under a single user 'authentication service'. There are two reasons for this. First, they may be provided using essentially one and the same mechanism, so that there are good practical reasons for grouping them. Second, it is considered unlikely that users of a mail system would want one of these services without the others, or indeed could really distinguish between them.

A second, and more significant, restriction on the provision of services is that a user may not simultaneously provide content confidentiality and request non-repudiation of delivery. The reason for this restriction stems from the need for the Message Transfer System to deliver messages to an MS rather than to the intended recipient UA. This rather undesirable restriction on the provision of security services in the mobile environment is discussed in detail in Section 5. It serves to illustrate the problems that arise when attempting to use the X.400 security features to provide a comprehensive set of security services, and to pinpoint where amendments to the standards are needed.

3. KEY MANAGEMENT

The management of the cryptographic keys required for the provision of the Locator demonstrator security services is achieved by using security features built into the directory service specified in the X.500 series of Draft Recommendations. Of particular importance is the authentication framework specified in Draft Recommendation X.509, (6).

The key management system is based on the use of public key (asymmetric) cryptosystems (pkcs). They are used within the Locator demonstrator for digital signatures and encryption. In a pkc, keys are produced in pairs, one of which is made public whilst the other is known only to its owner, (1). The X.509 authentication framework allows a user's public key to be stored in its directory entry. One user wishing to exchange secure messages with another obtains the other user's public key from the appropriate directory entry, and then uses this key to provide the required security services, as described in Section 4.

3.1 Digital Signatures

The X.509 authentication framework does not specify any particular pkc, although it does require the use of a pkc satisfying a special property. This property (listed in clause 6.1 of X.509, (6)) is that 'both keys in the key pair can be used for encipherment', i.e. both the public key and the secret key can be used to operate on arbitrary data. The reason for insisting on this special property is that the framework specifies how the pkc is to be used to provide digital signatures, and this method requires that the pkc has the specified property.

The method specified for digital signatures is simple. First the data to be signed is 'hashed' using a collision-free hash function, (8). Within Locator the hash function used is that suggested in Annex D of X.509, and is based on the repeated use of modular squaring. The end result of this hash function (the 'hash value') must be sufficiently small to be processed by a single encryption operation of the selected pkc. The hash value is then encrypted using the selected pkc under the control of the secret key of the signer. Since the hash function is public, the signature can then be checked by anyone who has access to the public key of the signer. As in X.509, (6), we denote the signing of data block I using the secret key of X by

X{I}

where X{I} is defined to consist of a copy of I followed by the value obtained from hashing and enciphering I.

Unfortunately, there are very few pkcs known which have the specified property, and the only well-established one is RSA, (1); consequently this is the pkc used within Locator. There is no need to specify so precisely how digital signatures are produced, and the fact that this is done within X.509 is a shortcoming of the authentication framework. Indeed, a small change to the framework would allow arbitrary pkcs to be used with arbitrary (and possibly unrelated) digital signature algorithms.

3.2 Certificates and Certification Authorities

Since the directory is not a secure or trusted service, means need to be provided for users to verify public keys read from the directory. This is achieved by using off-line trusted entities known as Certification Authorities (CAs) who provide 'certificates' for users' public keys.

In order to store a copy of a public key in the directory, a user must choose a CA; this CA must be trusted by the user, since a fraudulent CA has the power to mislead the users for whom it acts. Like the users, every CA must also have its own pkc key pair.

The user and the CA exchange their public keys in such a way that each trusts the validity of the received key and the identity of the other party. The CA then computes a digital signature on the following set of data: the CA's name, the user's name, the user's public key and the period of validity for the user's public key. This signature is computed with the CA's secret key, using the technique described above. The set of data, together with the signature itself, forms the user certificate, which is stored in the user's directory entry. Using the notation introduced earlier, if user A has certification authority X, then A's certificate has the form

```
  X{ X, A, Ap, TA }
```

where X and A are the names of X and A, Ap is A's public key and TA indicates the period of validity of the certificate. We denote such a certificate by

```
  X<<A>>.
```

Any other user which has a trusted copy of X's public key can then check the signature on the certificate, and hence obtain a verified copy of A's public key.

The scheme described so far no longer works when two users are served by different CAs. To cover this possibility, one CA may generate a certificate for another CA's public key; such certificates are called 'cross-certificates'.

As an example of the use of cross-certificates, suppose users A and B are served by CAs X and Y respectively. Then X and Y generate (and store in the directory) the following certificates:

```
  X<<A>>, Y<<B>>.
```

Additionally suppose that CAs X and Y are able to exchange public keys in a verifiable way, and thence generate the following two cross-certificates:

```
  X<<Y>>, Y<<X>>.
```

Then user B may use the sequence of certificates

```
  Y<<X>>, X<<A>>
```

(in combination with a trusted copy of Y's public key) to first obtain a verified copy of X's public key and then obtain a verified copy of A's public key. Such a sequence of certificates is called a certification path. Note that, in order to trust a public key checked using such a path, it is necessary to trust all the CAs in the path.

Within the Locator demonstrator, there will only be three distinct CAs, each of which will cross-certify each other. Therefore certification paths will only ever contain two certificates. However, the concept of certification path may be generalised to paths containing many certificates.

4. SECURITY SERVICE PROVISION

The majority of the security features incorporated into the draft X.400-1988 Recommendations make use of a cryptographic construct called a token. In fact, tokens will be used in the provision of all the security services in the Locator demonstrator. We begin this section by describing the general form of a token, and then consider how such a construct is used in the provision of the various security services.

4.1 Tokens

A token consists of a series of data fields with a digital signature appended, exactly like a certificate. Unlike certificates, tokens are always generated by a user for transmission to a single other user. The precise form of a token sent by user B to user A is

```
  B{ tB, A, sgnData, Ap[encData] }
```

where tB is a timestamp, A is the name of the recipient user A and sgnData and encData are collections of security-related parameters; the contents of sgnData and encData vary depending on the security services being provided. The notation Ap[encData] means that the data field encData is sent encrypted under A's public key, i.e. the contents of encData are available only to the intended recipient. Within Locator encData is only ever used for the transmission of secret key information as part of the content confidentiality procedure described below. It is important to note that, whatever the contents of the sgnData and encData fields, the signature on the token prevents them being changed in an undetectable way.

It should be observed that the token signature is applied after the encData has been encrypted under the recipient's public key. This is the construction that is specified in X.411 and X.509, (4), (6), but it is arguable that encryption should be applied after the signature, either to the entire token or to selected 'secret' fields of the token. This question of order of the operations may well lead to a debate, which could ultimately result in a revision of the construction of tokens within X.509, (6).

Within the Locator demonstrator all the end-to-end services are provided using a special type of token called a message-token. Some of these services may be provided in other ways, but we concentrate here only on those techniques used by Locator. Message-tokens are sent with individual messages, on a per-recipient basis, i.e. a distinct token is sent for each recipient of the message for whom security services are being provided.

4.2 Content Confidentiality

If content confidentiality is required for a particular message, then, unlike other security services, this must be provided either to all or none of the recipients. This service is provided by encrypting the message content. The encryption algorithm used for this process is not specified by the X.400 Recommendations, and may be either conventional (symmetric) or public key (asymmetric) in type. In the Locator demonstrator the DES algorithm is used, (1). The key used for the encryption process is selected at random by the message originator, a new key being selected every time a confidential message is sent. A different message-token is sent with the message for every recipient, and the key used to encrypt the message content is included in the encData field of each token.

4.3 Authentication Services

In Locator, the three services: message origin authentication, content integrity and non-repudiation of origin, are all provided together and may not be requested independently. In fact, the mere existence of a message-token for a recipient provides message origin authentication for that recipient, although the service is of dubious value on its own since there is no guarantee that the message content has not been altered. This is one reason why all three services are combined in the Locator demonstrator, resulting in a meaningful and useful set of options being offered to users.

To provide these three services a 'Content Integrity Check' (CIC) is generated and included in the sgnData of the message-tokens for all recipients for whom the services are to be provided. This CIC must be computed as a 'one-way function' of the message-content. The function to be used to compute the CIC is not specified within Draft Recommendation X.411, (4). However, if it is to provide the non-repudiation of origin service, then it must satisfy the same properties as are required of a hash function used in the computation of digital signatures. In Locator the CIC is computed using precisely the same function as that used for hashing data in certificates and tokens, i.e. the modular-squaring function described in Annex D of Draft Recommendation X.509, (6). The presence of the CIC in a token enables a recipient to verify the integrity and authenticity of the message-content.

The replay detection service is provided to a recipient by including a message sequence number within the sgnData of the message-token for that recipient. This service forms part of the message sequence integrity service described in Draft Recommendations X.400 and X.402, (2), (3). The way in which the sequence number is used is not completely specified within the X.400 Recommendations, and so we describe how it is used within Locator. Every user keeps a list of all other users for whom this service is to be provided, and a number is associated with each entry in the list. This number represents the message sequence number assigned to the last message sent to that user. The next time a message is sent to that user it is assigned a sequence number one larger than the stored value, and the stored value is updated. The receiver of this message will also keep a list of numbers (one for each other user from whom messages are received that incorporate replay detection). This enables the message sequence number in the message token to be checked for its 'freshness'. The inclusion of the sequence number in the token ensures its integrity, and therefore provides a secure replay detection service.

4.4 Non-repudiation of Delivery

The final end-to-end service, namely non-repudiation of delivery, is rather different in nature from other services, in that it is provided in two stages: request and provision. The originator of a message does not provide the service, but rather requests its provision. The service is actually provided by the message recipient through the return of a 'receipt' for the message called a 'proof-of-delivery'.

The request for this service is achieved by including a 'proof-of-delivery-request' flag in the message token for the recipient(s) concerned. When such a message is received, the proof-of delivery is computed as a digital signature on the (unencrypted) message-content and various delivery-related parameters. The signature is evaluated using the recipient's secret key, and the function used is precisely the same as that used for signatures on certificates and tokens. The proof-of-delivery is then returned to the message originator within the delivery report, and can be used by the originator to give the desired non-repudiation of delivery service.

4.5 Secure Access Control

We conclude this discussion of security services by describing how the only security service which is not end-to-end, namely secure access control, is provided. As has already been mentioned, within Locator this service is only provided on the link between a UA and its associated MS. The service is based on the exchange of another special type of token, called a bind-token, at the time a connection is set up between a UA and its MS. The service is restricted to access control, and does not provide connection integrity or confidentiality. However, the bind-tokens could be used to exchange keys for the provision of such connection-based services, although no means of providing them are defined within the draft X.400 Recommendations.

In more detail, the initiator of the connection between a UA and its MS, which must be the UA, includes a bind-token in its initial communication; for details of the connection initiation see Draft Recommendation X.413, (5). Prior to generating the token, the connecting UA selects a random number specifically for this connection. The size and form of this random number is not specified within Draft Recommendation X.413; however, within Locator, it will be a 64-bit value. This number is then included in the sgnData field of a bind-token sent from the UA to its MS; this token is called an 'initiator-bind-token'. The encData field of the token will be empty, although, as mentioned earlier, this could be used to convey keys for providing connection-based security services.

On receipt of an initiator-bind-token, the MS checks the signature on the token and recovers the random number from the sgnData field. The MS also checks the time value within the token in order to check that it is 'fresh'. Given that the received token is deemed acceptable, the MS generates another bind-token, called a 'responder-bind-token', and returns it to its UA. The random number taken from the sgnData field of the initiator-bind-token is reproduced in the sgnData field of the responder-bind-token. On receipt of the responder-bind-token, the originating UA checks the token signature and the random number in the sgnData field, and if these tests pass, the connection is allowed to proceed.

5. PROBLEMS WITH USE OF THE X.400 MECHANISMS

We now consider some problems that have been encountered with using the security features within the draft X.400 Recommendations to provide the required security services in the mobile environment. The major problems have centred around provision of the non-repudiation of delivery service, and we start by considering this service in a little more detail.

In the description of the generation of the proof-of-delivery by a message recipient, the precise identity of the key used to compute the signature was deliberately not discussed, and was merely referred to as the recipient's secret key. Problems arise because of the fact that Draft Recommendation X.411, (4), requires that the proof-of-delivery be generated and returned to the delivering MTA at the time the message is delivered. If the message is delivered to an MS, then the MS must generate the proof-of-delivery rather than the intended recipient UA. This requires the MS to have access to an RSA key pair.

One solution to this problem would be to give every MS access to the secret RSA key belonging to its associated UA. This solution has a major drawback in that many MSs may be implemented on the same remote machine, which is not trusted to the same degree as the portable UA machine. The only other solution, and the solution adopted for the Locator demonstrator, is to equip every MS with its own RSA key pair, distinct from the key pair belonging to its associated UA. The secret key from this MS key pair is then also used to sign the responder-bind-token used in the provision of secure UA-MS access control.

With this solution, problems arise with generating certificates for the MS public keys. Such a certificate must be distinguishable from a UA certificate, or else possessors of MS keys will be able to masquerade as UAs. Within the current (November 1987) draft of the X.509 authentication framework, (6), there is no provision for doing this, since a UA and its corresponding MS share the same O/R name. Within Locator we have therefore been forced to use non-standard means to achive our aims. One possibility under consideration is to commandeer a different field of the certificate as follows; note that the scheme described below violates the spirit of the ISO naming and addressing conventions.

In addition to names, dates and a public key value, every certificate contains an 'algorithm-identifier' (within the public key parameter), intended to specify which algorithm the public key is intended to be used with (e.g., RSA). In Locator, we use this data field to include an additional field indicating the 'scope of use' of the key, i.e., whether the key belongs to a UA, an MS or to a CA. This is admissible since there are, as yet, no standards for algorithm-identifiers. This is the only part of the Locator security implementation in which we are considering using non-standard security facilities.

A further problem arises when an MS is required to compute a proof-of-delivery. We stated above that the proof-of-delivery is computed using the unencrypted message-content. This is a problem for an MS, since the key required to decrypt the message-content is within the encData field of the message token, encrypted using the recipient UA's public key. It is therefore not possible for the MS to recover the unencrypted message-content unless it has access to the UA's secret RSA key. As has already been stated, this is undesirable, and not allowed within Locator. The 'solution' adopted within Locator is to prohibit an MS from providing non-repudiation of delivery if the message content is encrypted. This is clearly unsatisfactory, but there is no obvious way to improve the situation.

In conclusion, we have identified two significant problems with the current versions of the draft X.400 and X.500 Recommendations. Although these problems are not catastrophic, in that it is still possible to build a secure electronic mail service, it is nevertheless of vital importance that they be addressed within the next study period of the CCITT.

6. ACKNOWLEDGEMENTS

The work described in this paper was carried out as part of the UK DTI Alvey Programme in the Mobile Information Systems Demonstrator. The authors would like to acknowledge the help and support of their colleagues on the Locator project who have collaborated in the design of the security architecture described in this paper.

REFERENCES

1. Beker, H., and Piper, F., 1982, "Cipher systems", van Nostrand, U.K.

2. C.C.I.T.T., 1987, "Draft Recommendation X.400. Message Handling: System and Service Overview", Version 4 (Gloucester, November 1987).

3. C.C.I.T.T., 1987, "Draft Recommendation X.402. Message Handling Systems: Overall Architecture", Version 5 (Gloucester, November 1987).

4. C.C.I.T.T., 1987, "Draft Recommendation X.411. Message Handling Systems: Message Transfer System: Abstract Service Definition and Procedures", Version 5 (Gloucester, November 1987).

5. C.C.I.T.T., 1987, "Draft Recommendation X.413. Message Handling Systems: Message Store: Abstract Service Definition", Version 5 (Gloucester, November 1987).

6. C.C.I.T.T., 1987, "Draft Recommendation X.509. The Directory - Authentication Framework", Version 7 (Gloucester, November 1987).

7. Cole, R., Hall, C., Hassall, M., Pell, A., and Walker, J., 1988, "Demonstrating the mobile office", these proceedings.

8. Damgard, I.B., 1988, "Collision free hash functions and public key signature schemes", Proceedings of Eurocrypt 87, Springer-Verlag, Berlin (to appear).

9. I.S.O., 1987, "DIS 7498-2. Information processing systems - Open Systems Interconnection - Basic Reference Model. Part 2: Security Architecture", International Organisation for Standardisation.