THE ROLE OF ENCIPHERMENT SERVICES IN DISTRIBUTED SYSTEMS

R.W. Jones and M.S.J. Baxter
ICL Defence Systems, Defence Technology Centre,
Lovelace Road, Bracknell, Berks RG12 4SN, England

The Open System Interconnection standard (ref 1) describes a model for communication among application processes at different computer installations (formalised as "open systems"). Possible ancilliary services provide security such as: user authentication, data privacy, data authentication, access control, protection against forgery and repudiation. Work is going on at present in standards committees to produce a security appendix to the OSI model. OSI security services and protocols should then follow.

This paper describes how an encipherment service and key distribution service may be incorporated into an end system and discusses possible key distribution protocols.

In figure 1, A, B and KDC are open systems in the OSI sense. Each operates as far as the outside world is concerned as if it had an entity for each of the OSI communication layers. In each case the section named 'communication services' represents those entities.

At each open system there is a "key distribution service" and an "enciphering service". These two together provide the encipherment services which are the subject of this paper. The functions which the key distribution service should provide are discussed in ref. 2. They may be summarised as key generation, key delivery and key acceptance. The functions provided by the enciphering service are encipherment and decipherment and the associated manipulation of keys. In the design we are considering here the enciphering service recognises keys of different types. The type of a key is shown by tag bits which are enciphered with it, using the ideas described in ref. 3.

The enciphering service and the key distribution service are usable by

F. Pichler (Ed.): Advances in Cryptology - EUROCRYPT '85, LNCS 219, pp. 214-220, 1986.

the communication services. In practice the intention is that they be directly usable by a communication entity at least as low (in terms of layers) as that whose data will be enciphered. Higher layer entities wishing to use them address them as if they were remote services and the service which uses them directly routes the messages appropriately, removing protocol headers as necessary. The services, their users, and the functions they perform are summarised in figure 2 which lists the following cases.

Application entities use the communication services (the normal use to pass messages to remote entities).

The key distribution service uses the communication services (when requested to distribute a key).

The communication services use the enciphering service to encipher and decipher messages which are sent and received on behalf of applications.

Application entities use the enciphering service to encipher and decipher data held locally.

The key distribution service uses the enciphering service to encipher and decipher keys and associated data during key distribution. The encipherment needs of key distribution protocols have caused us to modify the ideas described in ref. 3. During distribution keys are accompanied by data. To simplify matters we allow the key and the data to be enciphered by the same "key encryption key". Such data is accompanied by a tag to distinguish it from a key and to signal to the enciphering service that it may be output in plain text form. Data enciphered by a "data encryption key" has no appended tag in its enciphered form.

The communication services use the key distribution service to generate and distribute a key when asked to establish a secure connection.

Application entities may use the key distribution service in order to generate a key for local use or to establish a common key with remote entities.

A key at an open system, except when it is in the local store of the enciphering service, is itself enciphered. A hierarchy of key enciphering keys is possible but there must be some key or keys enciphered directly by a master key for that open system. The master key is held in plain text form in the store of the enciphering service. When a key is delivered to another open system it must be re-enciphered by the master key of the receiving open system. In large networks it may be impractical for every open system to know the master key of every other open system with which it needs to communicate. A special open system is then created which has been called (among other names) a Key Distribution Centre (KDC)(see ref 4). A KDC can send and receive keys securely to and from each open system either because it knows its master key or because it shares with it another key enciphering key (KEK) for this purpose. Thus when the key distribution service at A in figure 1 wishes to send a key to its counterpart at B it does so with the help of the KDC.

Key generation, being a sensitive operation, may be another function of the KDC. In a very large community it is likely that there will be more than one KDC and they themselves will form either a network or a hierarchy in their ability to pass keys to each other, using shared KEK's.

We now discuss a number of key distribution protocols which we have considered. At the time of writing we have not yet picked a definite protocol.

Figure 3 shows a protocol which is, in essence, that of ref. 4 with improvements (the time stamp) suggested by several commentors. A wishes to establish a common key with B. To do so A sends a message to KDC enciphered by KA, a key known only to A and KDC. The message contains B's identity and a time stamp, DT. KDC generates a key and returns to A the key, a new time stamp, B's identity and a package, P, for A to send on to B, all enciphered by KA. The package consists of the same key and time stamp, and A's identity, all enciphered by KB (known to B but not A). A sends this message to B, who is assured that the key has been generated by the KDC at time DT, and is to be used also only by A. Messages 4 and 5 assure B that the correspondent knows KS and therefore is A. The time stamp eliminates the threat that the correspondent is a false A who has discovered the plain text form of an old KS. However, if B is unable to go ahead with the

connection within the agreed timing window, the timestamp will render the key unusable, and A must go back to phase 1.

Figure 4 shows a protocol we have devised which differs from the previous one for three reasons.

First, it seems worthwhile to eliminate the threat of a malefactor replaying an old KS whose plain text form he has discovered and to do so without the need for synchronised clocks for time stamping.

Second, it is better (on grounds of security and efficiency), to apply to the KDC for a session key only when both A and B are ready to proceed.

For these two reasons A does not ask the KDC to generate KS. Instead A generates a random number, R, sends it to B and is convinced when he receives it back, encrypted by KA in message 4, that the KS which accompanies it has been generated by the KDC in reply to B's request, triggered by the original message from A. R is sent with each message as a transaction code.

Finally, the messages contain extra fields caused by the environment in which they are exchanged. In figure 1 we can see that the key distribution services which exchange the keys are not the eventual users. Thus : "Buser" is the identity of the eventual user (often the communication service at B); "Kref" is a reference number invented by B which is to identify the key and which will be told to Buser and related by Buser to its opposite number at A; "tag" tells B the kind of key (e.g. KEK or DEK) which A wants. In message 2, B asks the KDC for a key of type 'tag', to be sent to A. R and Kref are sent to the KDC so that it may include them in the package to be sent on to A, enciphered by KA. The identity A tells KDC to use KA which it holds. I is invented by KDC and included in the package sent on to A. The fact that A can send back I's plain text form in message 5 assures B of A's identity (although B knows that an impersonator without knowledge of KA cannot understand subsequent messages). S and S+1 are sequence numbers to preserve the integrity of the chain of messages between B and the KDC.

Figure 5 illustrates the messages which are exchanged when attempting to establish a transport connection. Assuming that encryption is done

in that layer we are interested in integrating the key distribution protocol with the connection protocol. Figure 5 shows that A and B each contribute a value (A ref and B ref) to identify themselves and the particular connection. They may also be used to identify the distributed key.

With this in mind figure 6 illustrates a protocol which combines the connection and key distribution protocols. An additional difference from figure 4 is that the KDC does not pass A's version of KS back to B so that B may send it on to A. It keeps it for A to retrieve as shown in messages 5 and 6, another way of eliminating the "Packaged Key replay" threat. If the KDC has functions extra to the generation and distribution of keys, such as recording who used which keys and when, this method becomes more attractive since the KDC has the information needed in any case. In figure 6 as compared with figure 4, there are other small points of difference which may be adjusted in deciding on a definite protocol. It is assumed that the tag of the key is implicit in figure 6. Aref and Bref in figure 6 together correspond to both R and Kref in figure 4. If they are too easy to guess an extra randomising value may be needed. Similarly R might serve as a reference to the key in figure 4.

References

1. International Standard ISO/IS 7498. Information processing systems - open systems interconnection - basic reference model.

2. Jones, R.W.: "User functions for the generation and distribution of encipherment keys", ICL Tech. J, 1984, 4(2), 146-158.

3. Jones, R.W.: "Some techniques for handling encipherment keys, ICL Tech. J, 1982, 3(2), 175-188.

4. Needham, R.M. and Schroeder, M.D.: "Using encryption for authentication in large networks of computers". Communications of the ACM, December 1978.

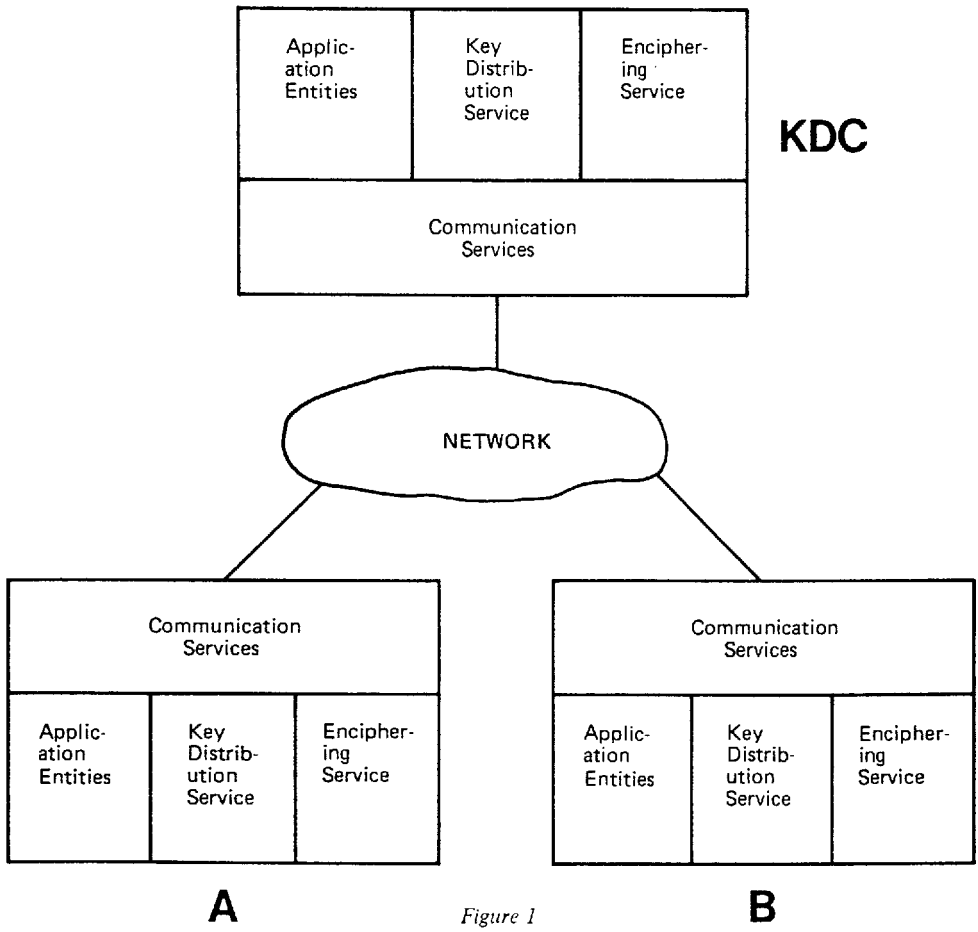

Figure 1

Service	User	Functions
Communication	Applications. Key Distn.	Secure communications.
Encipherment	Comms. Service. Applications. Key Distn. Service.	Key generation. Encipher/decipher data and keys.
Key distribution	Comms. Service. Applications.	Distribution and provision of tagged keys.

Figure 2

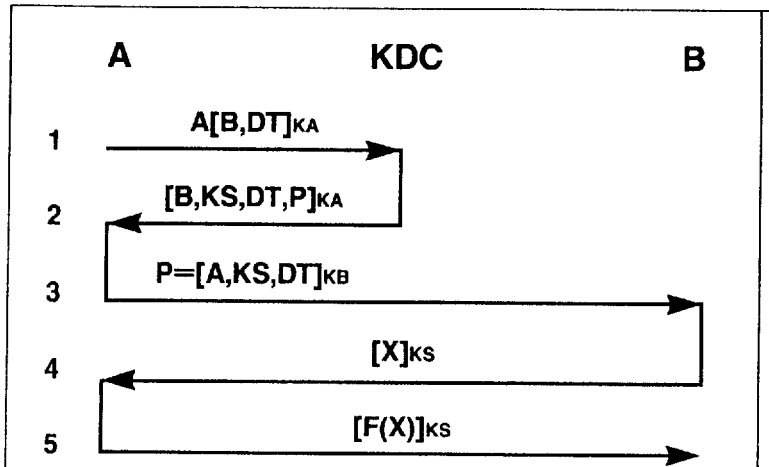

Fig 3

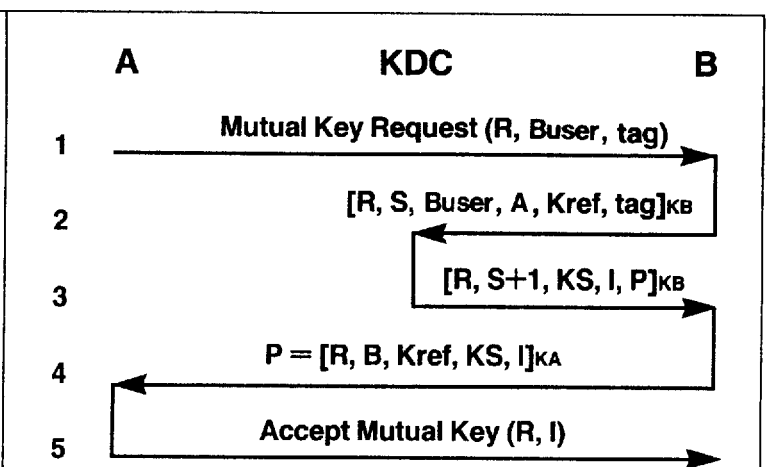

Fig 4

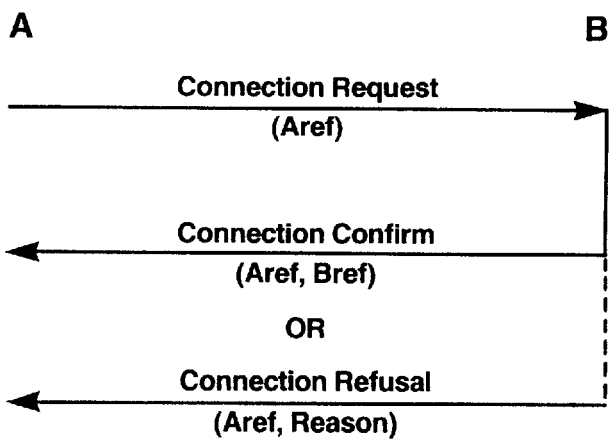

Fig 5

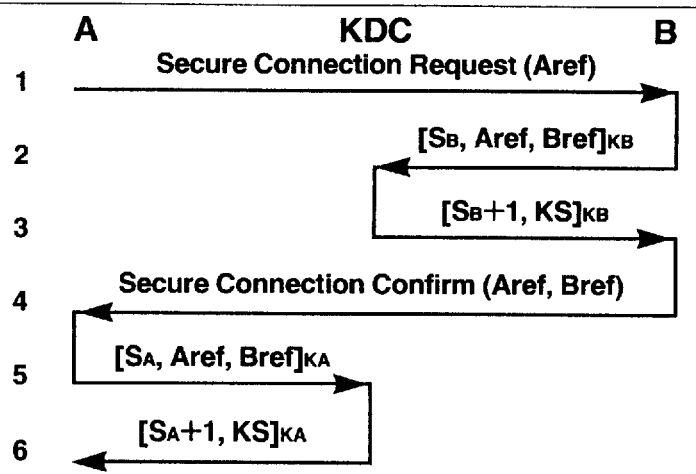

Fig 6