16

Security in group applications: Lotus Notes as case study

Andrew Hutchison

Department of Computer Science, University of Zürich,
Winterthurerstrasse 190, CH-8057 Zürich, Switzerland

Group applications provide ways for computer users to work together on common tasks. They are distinguished by their support for co-operative work and the provision of techniques and mechanisms for sharing and collaboration. The proliferation of group applications, which treat information as a shared entity, leads to a re-assessment of the way in which security is provided. In this paper the provision of *security in group applications* is addressed, and Lotus Notes – a commercial groupware product in wide use – is examined as a case study in terms of proposed security requirements for group applications.

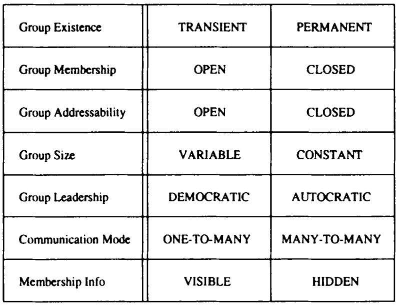
1. INTRODUCTION

The proliferation of group applications and the growing desire to employ 'groupware' in businesses presents new security challenges for developers and users. Traditional security policies have focused on individuals, whereas the advent of group applications suggests security policies and mechanisms pertinent to group interaction. As with the provision of security in other areas of computing, security for group applications has been something of an afterthought. It is inevitable that the development of new technologies – such as groupware – precedes their refinement with features such as those for security, but failure to consider security requirements early enough can also make it difficult to integrate them later.

In this paper we try to assist in the groupware design process by highlighting security considerations which should be addressed. By identifying *security requirements for group applications* these can best be satisfied with consideration right from the conception of a groupware application. Lotus Notes represents one of the first commercial groupware products and, in light of this, it is useful to examine the security philosophy and realisation in this product as a *case study* from which to go forward.

The purpose of the paper is thus to introduce the concept of security for group applications; to propose requirements for achieving secure group applications; and to present the security implementation of Lotus Notes as a detailed case study, assessing how well Lotus Notes satisfies the requirements. It is hoped that the insights given will assist developers in providing secure groupware, and industry users in evaluating how successfully security has been realised in a groupware product.

J. H. P. Eloff et al. (eds.), *Information Security — the Next Decade*



Group Existence	TRANSIENT	PERMANENT
Group Membership	OPEN	CLOSED
Group Addressability	OPEN	CLOSED
Group Size	VARIABLE	CONSTANT
Group Leadership	DEMOCRATIC	AUTOCRATIC
Communication Mode	ONE-TO-MANY	MANY-TO-MANY
Membership Info	VISIBLE	HIDDEN

Figure 1. Table of Group Attribute Options

2. GROUP APPLICATIONS: THE SECURITY CHALLENGE

Groupware systems have as their goal 'support' of a co-operative process. As part of this support, it is necessary to provide security features which are adequate in the context of shared procedures and information.

Figure 1 shows seven group attributes according to which groups can be classified. It is important to be aware of the different possibilities for each group attribute, and to try to determine the nature of a group. The group nature will determine what security mechanisms are required to support a particular group.

In terms of **group existence** this can be either *transient* (such as in a one-time group established to conduct a teleconference) or *permanent* (such as in an operating system, statically defined user group). Having considered the lifetime of the group we can then consider if **group membership** is *open* (meaning that anyone can join the group) or *closed* (indicating that membership is restricted to parties with some credential). Even if group membership is closed, it is sometimes the case that **group addressability** is *open* (in that non group members can still send messages to the group) or addressability may be *closed* (where only group members can send to the group). The **group size** may be *variable* (meaning members may join and leave the group) or *constant* (meaning that membership remains static for the duration of the group's existence). In the case where group actions or decisions are carried out (for example admission of a new member to an open, variable group) these can be based on *democratic* (some quorum of group members in consensus) or *autocratic* (a single actor) decision making. This group attribute has more to do with the ongoing functioning of the group than its actual structure. The **communication mode** of the group is another characterising feature and this could be *one-to-many* (such as in the case of a conference where there is one sender transmitting to many recipients) or *many-to-many* (like in a teleconference where any party can act as a sender). The visibility of group **membership information** can be characterised as *visible* (if this information is obtainable or discoverable by non group members) or *hidden* (where either group membership cannot be determined, or group membership is known but it is not desirable that knowledge of group membership be available to those outside the group).

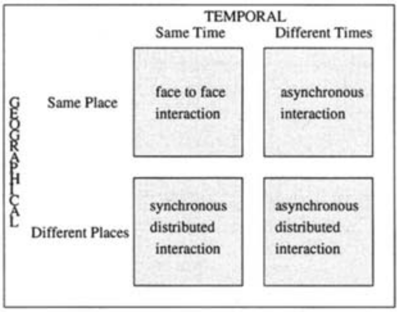
2.1. Security requirements for group applications

Group security requirements are based on the nature of the group to which they apply and this can vary – as described in the previous section. The security requirements are thus presented in a general manner and require interpretation according to the particular group manifestation. In general however the following security features could be required of a group application:

- Members of a group should be able to trust the purported identities of other members of the group.
- Non-members of the group should not be party to any group interactions or be able to undermine the integrity of intra-group communications.
- Group membership should be co-ordinated in some well-defined manner according to the membership policy.
  - A group policy should exist for admitting new members to the group. This could mean, for example, the appointment of a gatekeeper, or a vote amongst existing members or even that there is a policy of open admission.
  - A group policy should exist for the departure of members and revocation of any group rights which they may have inherited through group membership. This may typically include revocation of group keys.
- The extent to which group members should be protected from each other requires consideration. It should not be possible for one group member to intercept or modify a communication within the group, or to fake another group member's identity.
- Group activities should be traceable to an individual, for example for billing and logging.
- Non-repudiation of *sending* by the group (or by an individual in the group) and of *receiving* by the group (or by all members).
- Anonymity of membership or of sending to a group is a possible requirement in certain contexts.

3. LOTUS NOTES: A GROUPWARE SECURITY STUDY

Lotus Notes is a software system for networked PCs and workstations which interconnects *workgroups*. A Notes workgroup comprises a group of people who share information and work together regularly. The group members can be in the same location or geographically distributed. The Notes application consists of a Notes *server* and Notes *workstations*. The server provides *shared database* and *mail routing* services to Notes workstations, and electronic mail and bulletin-board type conferencing are integral parts of the Notes environment. Through the exchange of messages (either directly or via a



GEOGRAPHICAL	TEMPORAL Same Time	Different Times
Same Place	face to face interaction	asynchronous interaction
Different Places	synchronous distributed interaction	asynchronous distributed interaction

Figure 2. Groupware Time–Space Matrix

forum) and the sharing of databases, computer support for groups is provided by Notes. Other specialised services are also provided, either by a single *general* server or by a number of task specific servers.

In order to be precise with regard to the strategic vision for Notes, it has been reported that Lotus aims "to position it as a general-purpose platform for applications that collect, organise and distribute 'semi-structured' information, route documents and manage work flow" [1]. In this light, Notes can be seen as more of a workflow and process oriented application rather than a groupware system for real-time interaction. In terms of the groupware taxonomy of Johansen, classifying the geographical and temporal aspects of groupware [2], Lotus Notes currently falls into the category of *asynchronous distributed interaction* (Figure 2). It could be argued in this light that Lotus Notes is not representative of all categories of groupware since it does not (currently) offer synchronous distributed interaction. The reason for choosing Lotus Notes as a security case study was largely because of its strong customer base and wide commercial acceptance. Additionally, as one of the first commercial groupware products, it is informative to examine its security philosophy and realisation.

3.1. Notes' server topology

There are six different types of server which can be configured for Lotus Notes. Depending on the size of the installation it is possible that a single (general) server will provide all functions. The purpose of nominating and explaining each type in this section is to show a) what the server functions are and b) how these could be distributed amongst different machines in a large installation. From a security study angle this gives an idea of the possible complexity of a Notes topology, while also introducing some of the fundamental Notes concepts.

- **General** Server: A general server typically provides the functionality of some or all of the other types of server. This type of server holds Name & Address Books[1] as

[1] A Name & Address Book is used to manage a Notes domain – that is, a grouping of all Notes users who will send and receive mail. A Name & Address Book contains all user and server names, as well as

well as users' mail databases and any shared databases.

- **Mail** Server: A mail server is dedicated to Notes mail functions and as such would hold public Name & Address Books and users' mail databases, in addition to performing mail routing.

- **Database** Server: A database server stores other, non-mail, shared Notes databases and replicates databases if required.

- **Dial–In** Server: A dial-in server supports remote users, allowing them to dial in for the purposes of sending and receiving mail, and working on shared databases. Such a server can be used to provide Notes capabilities to remote users without the cost of installing a server at every location.

- **Hub** Server: A hub server is dedicated to performing mail routing and database replication between servers on different network topologies, or connected by a WAN. Users do not access this server directly. The use of a hub server eliminates the need to maintain many connection documents[2] in each server's Name & Address Book. A hub server can also be used to connect dissimilar networks.

- **Gateway** Server: A gateway server connects Notes to another electronic mail system, or can be used to send faxes through Notes. Its purpose is to convert Notes mail and documents to the format of 'foreign' mail applications and vice-versa. The gateway server is generally not a dedicated machine, but consists of software run on another Notes server.

If a General server is used, fewer servers are needed. The disadvantage of this approach is that long access times may occur if the server is heavily utilised. Another problem with a *single* General server, is that availability becomes an important issue since the system becomes unusable if the server is down.

Figure 3 illustrates a hypothetical Notes topology. Two LANs are shown, and each demonstrates a different Notes configuration. LAN A exhibits an arrangement where separate Database, Mail, Dial-In and Hub servers exist. LAN B, on the other hand, shows a configuration where the database, mail and inter-connection functions are all handled by a single General server. In each case there are other Notes workstations on the LAN which access the server(s).

The accessing of the network from remote workstations on LAN A, has warranted the inclusion of a dedicated Dial-In server with modems attached to it. A Gateway server also runs on the Notes Mail server on LAN A, and converts mail to a format in which it can be communicated with the DEC VAX on LAN A.

The Hub server on LAN A facilitates mail routing and database replication between LAN A and LAN B. On LAN B, the General server has to perform this function, in

routing information. Name & Address Books can be *public* or *private* depending on whether they belong to a server or an individual.

[2] A connection document must exist in order to make a connection from one server to another. Where a database is replicated, the connection record determines how often and when the database will be replicated between the servers it connects. Up to about 400 connection documents can be stored in the public Name & Address Book.

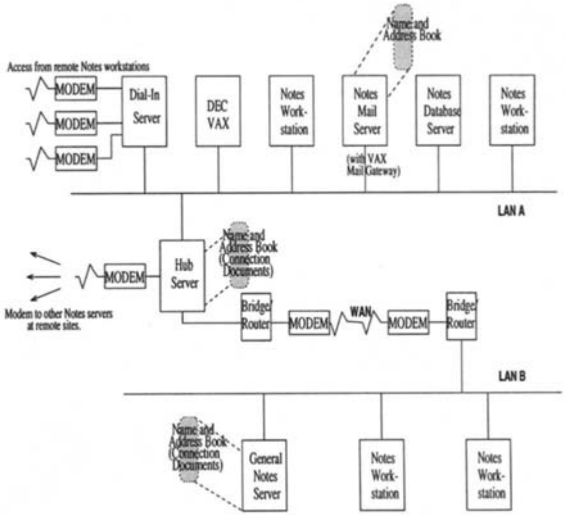


Figure 3. Example Notes Topology

addition to its other tasks. LAN A is connected to a number of other servers and this motivated a Hub server with 'spokes' going to those servers. The bridge/router component represents the need to transfer information between two networks, independently of the network protocols (bridge) or to transfer packets of information between two networks at the network layer (router), the latter case requiring compatibility between the network protocols.

The diagram also illustrates how in LAN B the Name & Address Book and connection documents are both held by the General Notes server, whereas in LAN A the connection documents need only be kept on the Hub server and the Name & Address Book is offloaded to the mail server.

3.2. Security overview

The Notes implementors identify five *primary security issues.* These 'issues' can be recognised as being based on the OSI network security services [3]. *User authentication* addresses the establishment of a level of confidence about purported user identity. In Notes, public key techniques are used for authentication. *Access Control* follows user authentication, and deals with the problem of controlling what the authenticated user may and may not do. Notes maintains *server access lists* (SALs), controlling access to servers, and makes subsequent use of *access control lists* (ACLs) to control user and server access to databases. *Confidentiality* requirements dictate that information should be kept safe during storage and transport. Notes addresses this requirement with facilities for the encryption of data both on disk and as it travels over a communication link. Encryption and decryption are carried out on the workstation (as opposed to the server). When these operations are carried out, encryption keys never travel across communication links thus eliminating potential vulnerability. *Source verification* addresses the need to establish the authenticity of some piece of information. This provides certainty regarding the origin of the information. In Notes, source verification is provided with an *electronic signature* that the originator and receiver of the information can trust. *Data integrity* is the final security issue. In the process of establishing that some information came from the person purporting to have sent it, a signed message digest also guarantees that the message has not been tampered with during its transmission or storage. The public key encryption technique employed by Notes facilitates source verification and data integrity via a message digest. A 'fingerprint' of the message is encrypted with the sender's private key and appended to the message being transmitted. The sender's public key can subsequently be used by the receiver to validate the signature.

The Notes implementors, in integrating the primary security considerations into the product, organise this into three main areas: a) physical security of Notes servers and workstations, b) a user certification process that controls user access to appropriate servers and c) data security, which is implemented within the application through ACLs, the definition of roles for using forms and views, and document encryption to fulfill the needs mentioned above.

3.3. Server security issues

A number of security issues are raised in the presentation of the Notes servers. The presence of a Notes server and a network file server on a same machine is advised against since users could access the file server and circumvent ACL security on Notes. Similarly, in OS/2 it is feasible for a Notes server to also be used as a workstation. This is again discouraged, since a user with access to the server can access all unencrypted databases.

Special precautions are also required in the area of communication with servers external to the organisation, for the purposes of routing mail or replicating databases. The Notes documentation suggests identifying and increasing security on *contact servers*, these being any servers which establish external connections. A system of certification is used to control access between servers and enable authentication. Certification is the topic of consideration in the next section.

An administrator can create Server Access Lists (SALs) which refine access to a server. Even if access to a server is allowed in terms of the certification procedure (to be discussed), the SALs enable the system administrator to specify DENY_ACCESS and ALLOW_ACCESS entries in the server's configuration file. CREATE_REPLICA_ACCESS can be used to allow or prohibit replication of the databases on the contact server by certain users or groups of users. CREATE_FILE_ACCESS can similarly be used to control whether or not a user is entitled to create databases on the server. Access via particular ports of the server can also be controlled using SAL entries in the configuration file.

3.4. Certification issues

In Notes user certification is viewed as a "process that controls user access to appropriate servers" [4] (p. 9-1). This "process" is satisfied by the features described in this section, Server Access Lists (introduced previously) and also the Access Control Lists used to ensure data security (the consideration of Section 3.6).

A User ID file identifies a legitimate Notes user. All IDs – whether pertaining to a

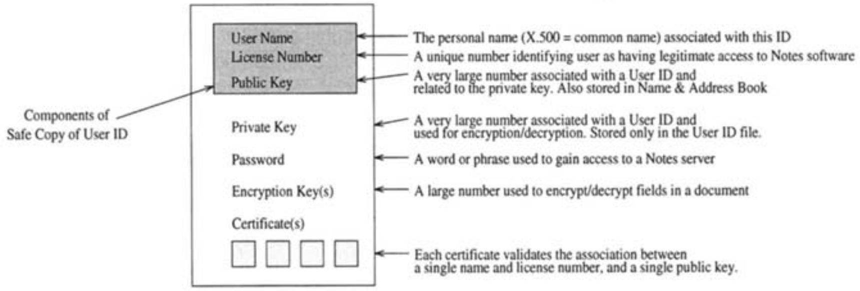

User Name
License Number
Public Key
Components of Safe Copy of User ID
Private Key
Password
Encryption Key(s)
Certificate(s)
The personal name (X.500 = common name) associated with this ID
A unique number identifying user as having legitimate access to Notes software
A very large number associated with a User ID and related to the private key. Also stored in Name & Address Book
A very large number associated with a User ID and used for encryption/decryption. Stored only in the User ID file.
A word or phrase used to gain access to a Notes server
A large number used to encrypt/decrypt fields in a document
Each certificate validates the association between a single name and license number, and a single public key.

Figure 4. Lotus Notes User ID File

user or a server[3] – have the format shown in Figure 4. The User ID File contains fields indicating the user name, license number, public and private keys, the user's password, one or more encryption keys and one or more certificates.

A so-called 'safe copy' of a User ID contains only the user's name, license number and public key fields. These fields are shaded in Figure 4. During the certification process it is necessary to communicate user information. By introducing safe copies of the User ID file, the private key, encryption keys and password of the user are safely excluded from communication.

The User ID file is stored on the workstation, and the Notes developers 'recommend' that it is password protected. The documentation also proposes that instead of workstation users storing their User IDs on the hard disk, a removable disk – which is inserted and read during the establishment of server connections – is used [5]. If a User ID is stolen, and not password protected, it can produce a forged signature – falsely assuring a mail recipient that the received mail came from the user name associated with the stolen User ID.

Each certificate in the User ID is an electronic 'stamp'. A certificate is created using the private key of a certifier, held in a Certifier ID. A Certifier ID is a special ID file, which is used by the certifier to create a unique certificate attesting to the fact that *the name on a User ID is correctly associated with the given license number and public key of that User ID.* The certificate is stored in the User ID and makes up a proof of identity, sanctioned by the certifier. Figure 5 shows this procedure. Multiple certificates are held when certification is given by more than one certifier.

Proof of identity is required when a user connects to a Notes server. In order for any Notes user (person or server) to access a Notes server, the User ID of the accessing party must be presented. When a user sends signed mail, all the sender's certificates accompany the document. At the recipient's workstation, Notes verifies that the sender's name, license and public key are correct by checking that it has a certification path to one of the certifiers of the certificates presented.

There are three types of certificate used in Notes, and we now describe each of these in

[3] Servers are considered as a special case of user.

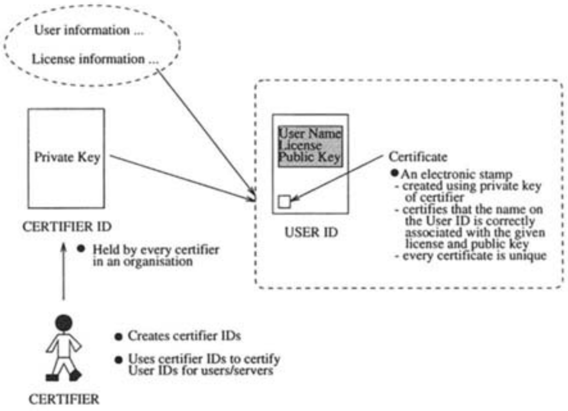

User information ...
License information ...
Private Key
CERTIFIER ID
● Held by every certifier in an organisation
User Name
License
Public Key
USER ID
Certificate
●An electronic stamp
- created using private key of certifier
- certifies that the name on the User ID is correctly associated with the given license and public key
- every certificate is unique
● Creates certifier IDs
● Uses certifier IDs to certify User IDs for users/servers
CERTIFIER

Figure 5. The Certification Procedure

turn:

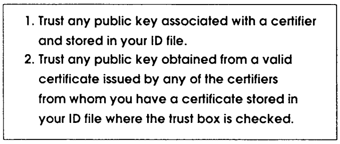
- Flat certificates: These certificates certify a 'flat name' (or common name portion of a hierarchical name). Such a name has no intrinsic meaning in a hierarchy, but represents some level in the hierarchy. When a certifier at a particular level of the hierarchy certifies a User ID, a new certificate is added to that user's list of certificates.

- Hierarchical certificates: These certificates work with X.500 distinguished name structures. This enables creation and management of an X.500 tree-like namespace reflecting a company's organisational structure. Instead of having a sequence of certificates certifying each level of a hierarchy (as in the use of *flat* certificates), the entire sequence of hierarchical certificates can be replaced by a certificate at some ceiling level of the hierarchy.

- Cross certificates: These are hierarchical certificates issued to a hierarchical name, where no parent–child relationship exists between the certifier and the ID being certified. Such certificates are often used when two independent organisations need to communicate. The first organisation sends a safe copy of its contact server's ID to the second organisation. The second organisation's certifier issues a cross-certificate to that ID and places this in the second organisation's public Name & Address Book. The process also takes place in the other direction, with the second organisation sending the first a safe copy of its contact server's ID to be certified.

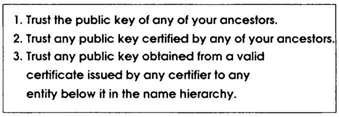
3.5. Authentication

When establishing a connection each party needs to verify the claimed identity of the other, and *authentication* is the process by which mutual trust of identity is established. In Notes the certificates held by users form the basis on which trust is established. A



1. Trust any public key associated with a certifier and stored in your ID file.
2. Trust any public key obtained from a valid certificate issued by any of the certifiers from whom you have a certificate stored in your ID file where the trust box is checked.

Figure 6. Rules for Establishing Trust: Flat Certificates



1. Trust the public key of any of your ancestors.
2. Trust any public key certified by any of your ancestors.
3. Trust any public key obtained from a valid certificate issued by any certifier to any entity below it in the name hierarchy.

Figure 7. Rules for Establishing Trust: Hierarchical Certificates

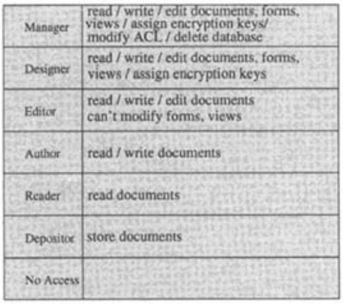
challenge–response dialog takes place between a client and server establishing communication. The basis of the authentication process in Notes is the trust of the public key being presented. A client wishing to establish a connection with a server forwards a User ID safe copy together with its list of certificates. The server will then use either the Rules for Establishing Trust: Flat Certificates (Figure 6), or the Rules for Establishing Trust: Hierarchical Certificates (Figure 7) to determine whether or not it can trust this client [5].

Once trust of the public key has been established, the server checks that the client is not masquerading, and does in fact know the private key associated with the public key. A random number (nonce) is sent from the server to the client. The client encrypts this number with its private key and sends the number back to the server. The server uses the public key which the client presented (and is now trusted as belonging to the client) to decrypt the number, and if the number matches the original then the server assumes that the client is authentic. This process also takes place in the other direction, with the client using the same techniques to authenticate the server.

3.6. Data security issues

The central organising principle for Notes data security is the Access Control List (ACL). ACLs are used to determine whether or not a user has access to a database. If a user has access to a database, the ACL defines precisely what the user can do to the database. An ACL defines what database privileges users have, including what forms and views they may use.

The implementors state that the "ACL defines the information access hierarchy of the group(s) using the database"[4](p. 9-1). Figure 8 shows the access classes provided in Notes. ACL entries can be *group* or *individual*. In Release 2 *privileges* were also provided as a way to give or deny a subset of users access to certain forms and views of the database. Privileges could be specified for individuals or named groups. The problem



Manager	read / write / edit documents, forms, views / assign encryption keys/ modify ACL / delete database
Designer	read / write / edit documents, forms, views / assign encryption keys
Editor	read / write / edit documents can't modify forms, views
Author	read / write documents
Reader	read documents
Depositor	store documents
No Access	

Figure 8. Database Access Categories

with privileges was that only a user who had been explicitly assigned a specific privilege could access the relevant object in the privileged manner. In Release 3 this mechanism has been superseded by *roles*. Roles are effectively groups local to a specific database. Roles are defined, and users, servers and groups are given roles through the ACL. In this way users, servers and groups can be assigned to any number of roles implicitly and users having that role can automatically access 'privileged' fields.

Use of roles could prevent salary information from being visible to personnel employees who require access to other employee information. In Release 2 an explicit user privilege such as *Salary* was a requirement for accessing the salary form. In Release 3 a personnel employee may be given, for example, the [Personnel Employee] role without access to the salary form, while the department manager might have the role of [Personnel Manager] with salary access by default. In this way users, groups and servers can be mapped to specific roles (with associated access levels) rather than giving individual privileges for accessing documents, views and forms.

Document *encryption* can be used to secure sensitive information. The Notes documentation explains that "if one 'field' in a form is encrypted, users without the encryption key won't be able to compose and save documents with that form, no matter what their access level" [4] (p. 9-5). This raises some questions over the flexibility of the encryption mechanism.

Notes uses both the public key RSA Cryptosystem [6] and the shared secret key RC2[4] algorithms for encryption. When a mail message is sent, the body is encrypted with a random RC2 key, and then the RC2 key encrypted with the recipient's public key is appended. Upon receipt the receiver(s) recover the RC2 key using their private key after which the message body can be decrypted.

When a document field is encrypted by a user, the encryption key (created using an encryption key creation dialog) is stored in the user's User ID file (see Figure 4). The encryption key can be distributed to other users, and this occurs by encrypting the en-

[4] RC2 is an unpublished, proprietary algorithm of RSA Data Security, Inc. It is a variable-key-size symmetric block cipher.

cryption key with the public key of the recipient (retrieved from a Name & Address Book). Upon receiving and 'accepting' an encryption key, the key becomes part of the accepting user's User ID file.

The issue of international encryption is raised in the documentation, whereby one encryption technique can be employed in the USA but another one needs to be employed internationally. A North American license will invoke the use of a more sophisticated encryption scheme than an international license. If a database is to be replicated internationally (outside North America) then the international encryption scheme must be employed. A user with a North American license cannot send encrypted mail to a user with an international license but users with an international license can send encrypted mail to a North American licensed user.

4. HOW WELL DOES LOTUS NOTES SATISFY SECURITY REQUIREMENTS IN GENERAL?

As noted previously, the whole Notes system of security is premised on the physical security of the servers. The documentation warns that "[it] is critical that the issue of physical security be resolved first. Otherwise, anyone who has access to the server can circumvent other methods of Notes security such as Access Control Lists and obtain access to databases on the server" [4](p. 4-16). A Notes system is thus only as secure as its servers. This could be said to be true of other computer systems too, although some operating systems provide more protection than others.

It is apparent that the security concepts of Notes have developed with successive releases of the product, and the developers have learnt from experience as evidenced by the introduction of cross-certificates and hierarchical certificates and the introduction of roles. The use of hierarchical certificates and X.500-like naming is effective, and simplifies the exchange of Notes data within and between organisations. An exchange of identifications must precede any exchange of data though, and this can make the granting of temporary access, or establishment of one-time data transfers, cumbersome.

Extensive though the security features may be, security in Lotus Notes is largely reliant on the system administrator. If the system administrator makes some mistakes, or does not carry out all procedures correctly, the system could be potentially vulnerable.

The vulnerability of User IDs, and the potential for faking based on their use, is a problem. An underlying problem is that Notes workstations are often based on operating systems without adequate protection mechanisms, so there is potential for malice. If workstations cannot be locked, unattended stations could facilitate masquerading. Local databases (stored on a workstation) are also potentially vulnerable if the workstation operating system cannot administer access controls on directories and files of the local operating system.

After undertaking this task of understanding and describing Notes' security, it is evident that the documentation provided by Lotus, and particularly the Notes Internals book on Security aimed at "MIS and ISS managers, system administrators, application designers and database managers" is vague and elusive. Some of the terminology applied is confusing, with subtle differences which are not always explained. The examples provided are often too elementary and do not help clarify more complex and interesting scenarios. Since the effectiveness of the security measures in a Notes installation can be greatly influenced by what system administrators do or do not do, it is essential that such key personnel can gain an in-depth understanding of the security mechanisms which are present and how they should be applied. The security policies of a site (such as requiring users to password protect their User IDs) can stem from this understanding.

5. HOW WELL DOES LOTUS NOTES SATISFY THE PROPOSED GROUP SECURITY REQUIREMENTS ?

Having presented a fairly extensive description of Lotus Notes security, we are now in a position to evaluate this security realisation in terms of the group security requirements identified in Section 2.1.

Before doing so we should refine our understanding of the specific type of group for which security is provided in Lotus Notes. In terms of the group attributes identified at the start of this paper, groups in Lotus Notes can be viewed as *permanent* (in that they are statically defined in a Name & Address Book) and *closed* (in that membership is restricted to those users explicitly given membership). The size of a group is *variable*, but membership does not change dynamically and it is likely that group sizes remain constant for long durations. Decisions concerning group membership could be said to be *autocratic* in that an administrator manages these, but these activities fall within the larger realm of the organisation. No intrinsic consent of existing group members is required to add a new member to a group in Notes, but this could happen extrinsically according to organisational policy. The communication mode in Lotus Notes can be *many-to-many* (with any party acting as a sender), but this is in an asynchronous sense and there is no specific multicast support for group receipt. Group membership information can be characterised as *visible*, since this is available in Name & Address Books.

The requirement that *members of a group be able to trust the identities of other group members* is satisfied in Lotus Notes by the certification and authentication processes described. The public key scheme which is the basis of this security feature enables other problems of security to be solved within this framework. As an example, data integrity and non-repudiation of sending by an individual are provided by signing messages with the private key contained in the User ID file. In this way the sender of a message can be verified. There is no concept of group signing (where for example a group public and private key (or Group ID) exists) though, and the origin of a message is always associated with an individual. Sections (or fields) of documents can also be made signable. In a group sense multiple signatures may be required (when more than one signer is necessary) but Notes has a policy of replacing previous signatures and only storing a single signature. This can be seen in terms of the workflow type model, where information flows sequentially. A more sophisticated group model should extend this to enable a group of signatures to be visible, and signing to occur in less restrictive sequences.

Access controls and encryption ensure that *non-members of a group are not party to group interactions.* It could be said that in Notes there are no synchronous group interactions as such, and that 'group interaction' in Notes is merely a collection of individual interactions. In this case, the requirement for protection from 'non-members' reduces to protection from other users. It is possible that in future Notes will be extended to include synchronous interaction. In this case it can be said that the security framework established should scale well, since fundamental problems of authentication and certification are solved, and provision and distribution of group keys (for example for secure real-time conferencing) could be done within the existing and established scheme.

The third security requirement identified for group applications proposed that *group membership should be co-ordinated in some well-defined manner.* Having noted that groups in Lotus Notes are static, and explained that an administrator manages group memberships, there is no notion of membership policy, or of how members join and leave groups, within Lotus Notes itself. This consideration can be seen as more appropriate to transient, open groups which are variable in terms of members joining and leaving. The area of secure group joins and key revocation upon member departure is a topic of ongoing investigation. Aspects of this have been published in [7] and [8].

Group members are *protected from one another in group communications*, to the extent that individual users are in Lotus Notes. The conception of groups is currently such that no group member can intercept or modify an intra-group communication, since all multiple addressing reduces to an individual sending to other individuals. Since there is no means of simultaneously sending a message to n group members, a group send maps to n individual transmissions. In the case of encrypted mail, for example, a message is encrypted with a random RC2 key and this key is then encrypted with the recipient's public key. Sending to a group of recipients would imply n encryptions of the RC2 key (using n public keys).

The requirement that *group activities be traceable to an individual* is largely satisfied in Lotus Notes – since it is as an individual (with a user, as opposed to group, identity) that a user acts. Group membership is pre-defined, and known. It is feasible that within a group there are multiple holders of an encryption key though, so in this case any key holder could modify the encrypted field without a record remaining of who made the change.

The issue of *non-repudiation of sending* was discussed together with the trust of identities. *Non-repudiation of receipt* is not provided for in Notes.

The requirements of *anonymity of membership* and of *anonymous sending to a group* are not supported in Notes, and specifically with concern to visibility of membership, group affiliations are revealed in Name & Address Books. It is necessary to add groups to a Name & Address Book in order that they become valid ACL entries (in terms of which access controls can be specified).

6. CONCLUSION

In this document the concept of security for group applications has been introduced. Requirements for secure group applications were proposed, and Lotus Notes was introduced as an example of a commercial *groupware* product. Lotus Notes has gained a fairly widespread base of user acceptance, as one of the first commercial applications to offer support for co-operative processes. With this status we reviewed the features and security aspects of Lotus Notes, examining it against the proposed requirements for security in group applications.

The group conception in Lotus Notes is for static groups, and there is no dynamic notion of a group at all. Groups must be pre-defined and the security mechanisms rely on the attributes given to these groups. It is not easy to make ad hoc connections, or to have temporary groups so in this regard better support for group work could be added. In terms of the interaction and inter-communication which it does allow, Lotus Notes provides a comprehensive and sophisticated security solution.

As a general comment, it has been suggested that increased security can decrease ease of use and hinder users in working with systems [9]. As we move forward to other groupware applications and consider security requirements this should be the primary trade-off in security for groupware: ease and flexibility of access, versus necessary and sufficient protection. The intention of group applications is to enable and facilitate co-operation, and so the provision of security features should not hinder the ease of use of the system nor restrict the interactions amongst parties to any significant extent.

ACKNOWLEDGEMENTS

This work has been informed by Lotus documentation and particularly the on-line "Notes Internals: Security" [5] document. In addition the insights of Edi Lanker and Johannes Doerler of the *University of St Gallen, Switzerland* were very helpful in understanding the practical application of Notes security in a large installation. I am also grateful to Liba Svobodova, Matthias Kaiserswerth and Kurt Bauknecht for their comments on drafts of this paper.

REFERENCES

1. J. Udell. One Thumb Up, One Thumb Down. *BYTE*, pages 161–168, July 1993.
2. R. Johansen. *Leading Business Teams.* Addison-Wesley, 1991.
3. ISO7498-2. *Information Processing Systems, OSI Reference Model Part 2: Security Architecture.* ISO7498-2, 1988.
4. Lotus Development Corporation. *Lotus Notes: Site Planning Guide.* Lotus, 1991.
5. Lotus Development Corporation. *Notes Internals: Security (on-line documentation).* Lotus, 1993.
6. R.L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, February 1978.
7. M. Reiter, K. Birman, and L. Gong. Integrating security in a group oriented distributed system. In *1992 IEEE Computer Society Symposium on Research in Security and Privacy*, May 1992.
8. M.K. Reiter. *A Security Architecture for Fault-Tolerant Systems.* Ph.D Thesis, Cornell University (TR93-1367), 1993.
9. O. Behrens and M. Yin. Konzepte fur das security management in groupware umgebungen. In *Proceedings Workgroup Computing '92 Computer Supported Cooperative Work*, 1992.