Some Experiences with Secure Management

Graham Knight <g.knight@cs.ucl.ac.uk>

Saleem N. Bhatti <s.bhatti@cs.ucl.ac.uk>

Abstract

This paper describes work carried out in the ESPRIT MIDAS project to provide secure management facilities. The work is based on extensions to the OSI CMIP management protocol which provide for mutual authentication at association set-up and integrity checks in all PDUs. The first version of this mechanism has been implemented and tested; in practice it has been found to be rather slow in operation. This paper proposes a series of measures designed to streamline operation and so improve performance.

A major goal of the work has been to provide secure access to management information. This implies the existence of a flexible, yet implementable, access control model. The limitations of the existing standards in this area are discussed.

I. Introduction

This paper describes work carried out in the ESPRIT MIDAS (Management In a Distributed Application and Service Environment) project to provide secure management. MIDAS, as its name suggests, is concerned with the management of applications and services rather than of networks. This has some impact on the choice of security services to be implemented. Section II of this paper outlines the kind of environment at which MIDAS management is aimed by describing one of the demonstrators which is taking place during the last six months of the project. Section III explains the choices of security services the project has made for this environment. Section IV describes the mechanisms which have been included in the initial implementation in the OSIMIS[1] software.

Now that this implementation is complete it is possible to draw some conclusions about what has been done and to make suggestions for improvements. Section V considers the performance of the authentication and integrity mechanisms and suggests improvements. Section VI considers access control and suggests a model which is more flexible than the one implemented but which can still be implemented efficiently.

II. The MIDAS WAN Demonstrator

It has always been intended that the MIDAS demonstrators should involve “real networks” with “real users” so that a realistic evaluation of the work can be achieved. The WAN demonstrator is shown in Figure 1. It is centred on a public administration network for the Provincia di Piemonte managed by one of the MIDAS partners - CSI-Piemonte. The application to be managed consists of a database of documents of interest to administrators and to members of the public. The size of the documents and the nature of some of the networks involved mean that interactive access is

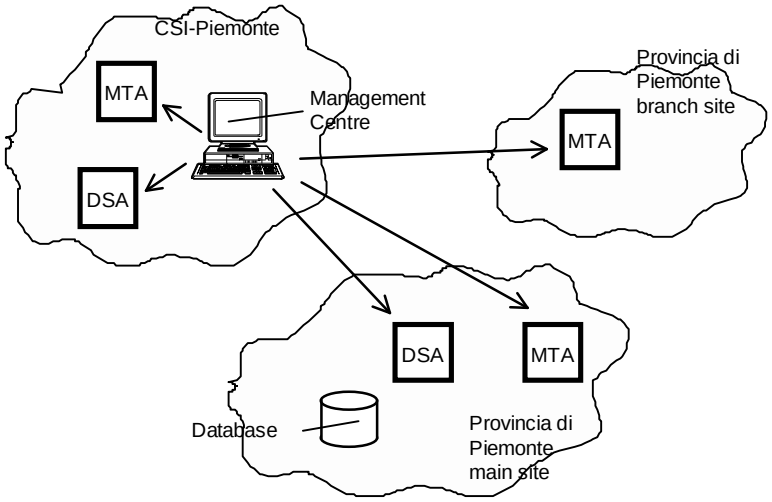

Figure 1. MIDAS WAN Demonstrator

not appropriate. Instead, access to the documents is via X.400 mail. An index to the database is provided in an X.500 directory. The X.400 MTAs use the ISODE PP software whilst the X.500 DSAs are the DirWiz implementations from another MIDAS partner, System Wizards[1]. Both have been enhanced with management support during the project. A sophisticated GUI has been built by CSI using a toolkit provided by GMD. This enables the management of X.400 and X.500 to take place in an integrated way.

"Management support" provided for the DSAs and MTAs in fact implies transforming these into "Managed Systems" (MS) according to the ISO/OSI management model[2]. This has been done using the OSIMIS management platform which has been developed at UCL in MIDAS and in several other EU-funded projects. An OSI MS presents management information to the outside world as a tree-structured object-oriented database. Each MS includes a database engine of considerable power which allows a remote manager to locate and manipulate information in a flexible way. In addition to supporting database search, read and write operations a MS may emit "event reports" when a condition of interest to a manager occurs. Precisely what management information is exported by a MS is specified in a "Management Information Base" (MIB). The MIBs implemented for the MTAs and DSAs support three kinds of management activity:

i. Configuration Management

   MTA routing tables can be manipulated as can the parameters of interactions with adjacent MTAs. Information on users and their mailboxes can be adjusted. There are similar facilities on DSAs; knowledge - cross-references, subordinate references etc., may be accessed and modified remotely. Remote DSAs and MTAs may be brought into and out of service.

ii. Performance Management

   The statistics of MTA and DSA operations are collected and may be accessed remotely. Thresholds may be set from which event reports may be generated. In particular "monitor metric" objects may be created. These allow a broad range of statistical functions to be calculated with the precise details being determined at run-time.

iii. Fault and diagnostic management

   Fault conditions are notified by event reports. The level of event reporting may be adjusted by the remote manager.

It is evident from the list above that extensive damage could be done to the system through malicious or ill-judged usage - hence the emphasis in the MIDAS project on security features. Since some parts of the network are accessible to organisations other than the Provincia di Piemonte, including two Universities[2], the risk is a real one. The principal requirement is to be able to prevent unauthorised access to management information and control functions. Different classes of user require different access privileges; some CSI-Piemonte staff will require unrestricted access, local management staff may only be allowed access to statistics and user information, users themselves may only have access to information on their own usage and on the distribution lists they manage.

Note that there are also important security issues for the applications themselves - some, at least, of the documents retrieved may be confidential. However these issues are not considered in this paper.

III. Summary of MIDAS secure management

III.A. Security Services

Our analysis [3] followed that of [5] and revealed a need for five security services:

S1) *Peer entity authentication* which establishes unambiguously the identity of the initiator of an operation;

S2) *Data origin authentication* which provides an assurance that data really comes from where it seems to;

S3) *Connectionless integrity* which ensures that management PDUs cannot be modified without detection;

S4) *Stream integrity* which guards against mis-ordering of PDUs in a stream (including re-plays);

S5) *Access Control* which enables a system to determine which operations should be allowed.

[1] The complete set of MIDAS partners is; Cellware GmbH (D), CSI-Piemonte (I), GMD-FOKUS (D), System Wizards (I) and University College London (GB).

[2] Notorious security risks!

III.B. Security Mechanisms

Many of the mechanisms which implement the security services we require are based on encryption techniques. In choosing mechanisms for MIDAS we have tried to follow the pattern which prevails in the OSI world but, at the same time, to borrow from the SNMPv2 work [4] which is geared particularly to the needs of management. This has also been the policy of other groups who are attempting to define security mechanisms for management, for example the US OSI Implementors' Workshop and the NM-Forum [5][6].

The principle difference between OSI and SNMPv2 management services is that the OSI one establishes a long-term, reliable association whilst SNMPv2 does not. This has some impact when security mechanisms are chosen:

i. Confidentiality and integrity mechanisms typically require the two communicating parties to have shared knowledge of a secret value. Without an association it is usual to expect this secret to be known to the two parties a priori and it must be stored securely by each of them. With an association it is natural to negotiate a new secret value each time an association is established.

ii. The protocols employed to maintain the association guarantee sequenced delivery with very high probability. Each PDU has an invokeID field - which takes values from a known sequence. This simplifies the design of a stream integrity mechanism.

iii. Once an association has been established it will normally be held for a comparatively long period. This makes it reasonable to implement quite complex security mechanisms in the association establishment phase in the knowledge that they will be used only rarely. It is feasible, for example, to use public key encryption in association establishment.

With this in mind we chose:

M1) To *authenticate associations* via public key encryption using the RSA algorithm[7]. This provides service S1. In addition a “session conversation secret” is exchanged during the association set-up. This is used in M2.

M2) To add *cryptographic checksums* to all management PDUs calculated according to the MD5 algorithm[8]. This provides services S2 and S3. Note that we assume that all operations on an association are invoked on behalf of the entity which was authenticated at the start.

M3) To use a "well-known" invokeID sequence in ROS[9] PDUs and so provide service S4

M4) To implement access control in the managed system as per [3] (service S5).

III.C. Implementation

In designing the detail of the mechanisms we aimed for generality, maximum re-use of existing work and minimal modifications to existing standards. We believe we have achieved generality in that the precise details of the cryptographic algorithms to be used are not pre-determined. Instead, we have specified transfer syntaxes which include identifiers for the particular algorithms in use. Re-use has entailed employing syntaxes already used in other applications, for example the “credentials” syntax used in the ISO Directory standards. Finally we have managed to make use of existing fields in standard PDUs to carry our security-related information. The full details of the mechanisms used are given in [3] only a brief summary is given here.

Suppose an association is to be set up between **A** and **B**. **A** forms a message with two components (roughly) as follows (see Table 1 for notation);

A->B	is a certification path from **A** to **B** - essentially a guarantee from one or more trusted authority that **A**’s public key is what **A** claims it is.
A{x}	is a message **x** digitally signed by **A**. Construction of this involves applying a hash function then RSA encryption with **A**’s public key.
$\mathbf{B_p(x)}$	is a message **x** encrypted with **B**’s public key. Again, RSA encryption is involved.
MD5 (x)	is a hash value calculated over a message **x**. **MD5** is a “one-way” function.
t	is a timestamp
r	is a random number needed to defeat replay attacks
S	is a “session conversation secret” which will be used during the association

Table 1. Notation used in the algorithm descriptions

a) **A->B, A{t, r, B}** b) **A{B_p(S)}**

The first component uses the "credentials" syntax from X.500. The second uses a simple syntax of our own devising. **S** is a value chosen at random for this particular association. In the current implementation it is used to provide the integrity checks. It could, equally, be used as a key for DES encryption to provide a confidentiality service. **S** is generated from a pseudo-random process seeded by the time which makes it virtually impossible for an intruder to guess it. Note that **A** needs to verify a certificate for **B** before constructing **A{B_p(S)}**. The components a) and b) are sent from **A** to **B** in the association request. **B** must then:

i. Use the certification path to verify **A**'s public key[3]

ii. Use RSA decryption with **A**'s public key to check the signatures on a) and b)

iii. Use RSA decryption with **B**'s private key to obtain **S**.

This exchange is illustrated in Figure 2 below.

At this point **B** can be certain; that **A** sent the message, that **A** really did choose **S,** and that the message has neither been tampered with nor replayed. However, **A** cannot be certain that **B** is not an impostor. In order to remove that possibility we need mutual authentication which we achieve by a similar message sent from **B** to **A**.

S is used in the calculation of an integrity checksum which is applied to all CMIP PDUs sent on the association. The calculation for a PDU **P** simply involves appending **S** and applying a hash function, i.e.:

$$\mathbf{x = MD5(P, S)}$$

Because of the "one-way" property it is impossible for an intruder to forge a **P** which will produce **x** without knowing **S**. The remaining problem is how to transport **x** in the CMIP PDU. This has been solved by overloading the ROS invokeID field as follows:

$$\mathbf{I_T = I_{REAL} \oplus x}$$

where ⊕ means "exclusive or", $\mathbf{I_{REAL}}$ is the "real" value of the invokeID which must be carried to the peer entity and $\mathbf{I_T}$ is the value actually placed in the invokeID field for transmission. The $\mathbf{I_{REAL}}$ values are generated from a recurrence relation based on a random sequence generator with initial seed derived from **S**. Thus, an intruder must know **S** both to calculate the checksum and to produce the next valid invokeID. We believe that the nature of the random sequence generator used makes an attack based on capture of two identical PDUs infeasible.

The OSISEC[10] package we are using supports the management of certificates by the Directory. It is possible that two Directory accesses will take place whilst the association is authenticated - more if a complex certification path has to be established. These accesses may be avoided, if desired, by keeping certificates in local storage.

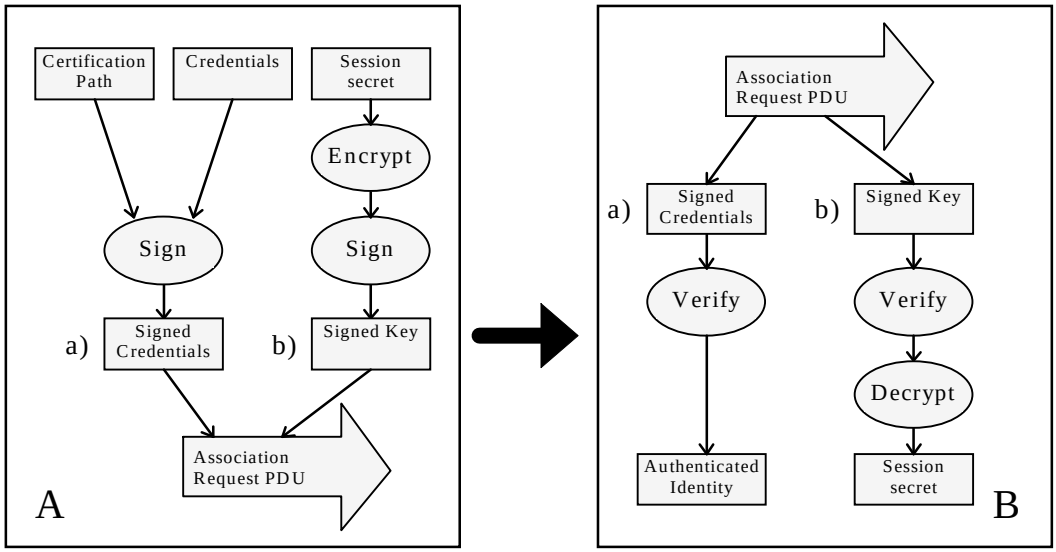

Figure 2. Authentication of A to B

[3] Actually the certificate is optional in the credentials. If is is not present then B must obtain it from elsewhere - from the Directory for example.

At present the access control which has been implemented is very limited and does not really satisfy the requirements. Essentially we allow three classes of users who are assigned to three privilege levels. Normally these levels would grant zero, read, or read/write access to the MIB. The principle reason for this limited functionality was the instability in the relevant standards and the perception that what the standards seemed to be proposing was unimplementable (see [3]). Our thoughts on how to improve the access control are in Section V.

IV. Tuning the Mechanisms

The scheme above has been implemented in the UCL OSIMIS management platform with the security algorithms being drawn from the UCL OSISEC security package. The software certainly works. We have been able to demonstrate secure management in MIDAS and have been able to measure the performance of the software and draw some conclusions:

i. The integrity checks on the CMIP PDUs work well. MD5 is quick and the other operations involved are cheap. We estimate that the integrity check imposes an overhead of about 15% on average;

ii. The mutual authentication is slow. Setting up an authenticated association can take as much as 20 secs. on SUN4 IPX machines with 16MB of RAM (average 12-15 secs).

The slow authentication did not come as a surprise. Construction of an authenticated ACSE[11] request involves the following RSA operations:

At the client	**At the agent**
Verification of B's certificate	Verification of A's certificate
RSA encryption of the session secret	Checking the signature on the secret
Signing of the secret	Checking the signature on the credentials
Signing of the credentials	Decrypting the session secret

Then we repeat the whole thing in the opposite direction - a total of 16 RSA operations.

This heavy processing is partly a consequence of our re-use of existing syntaxes and procedures - this makes it necessary to sign two separate messages for example. No doubt the encryption software could be speeded up - there are certainly faster RSA implementations than OSISEC's (for example hardware versions are available). However the biggest savings will undoubtedly result from changing the algorithm to reduce the RSA load and this is what we plan to do.

IV.A. The bind token

We propose a modified form of the Directory bind token which includes a session secret. A slightly simplified version of this syntax is given below:

```
MidasToken ::= SIGNED SEQUENCE{
  algorithm    AlgorithmIdentifier,
  name         DistinguishedName,
  time         UTCTime,
  random       BIT STRING,
  sessionToken MidasSessioToken
}
MidasSessionToken ::= ENCRYPTED SEQUENCE{
  algorithm    AlgorithmIdentifier,
  sessionKey   BIT STRING,
}
```

Using this in the credentials eliminates the need for separate signing of the secret and so saves one RSA operation at each end. This is illustrated in Figure 3 (the processing at B is an exact inverse).

IV.B. Mutual Authentication

As we insist that the encrypted session secret **must** be present in the message from **A** to **B** we can eliminate all of the RSA from the authentication of **B** to **A**. This is because the secret is encrypted by **A** with **B**'s (verified) public key - hence we can be confident that only **B** can decrypt the key. The second part of the authentication set-up can, therefore, be treated in the same way as the subsequent CMIP PDUs, i.e.. we construct:

$$\mathbf{x = MD5(R, S)}$$

where **R** is derived from the ACSE PDU being sent from **B** to **A**. It is impossible for **B** to calculate **x** without having **S** - and only **A** and **B** know **S**. The checksum, **x** can be carried in the user information field of the PDU. **A** performs the same MD5 calculation and compares the result with **x**. This exchange is illustrated in Figure 4.

IV.C. Management of Certificates

In our system it is Directory Names (DN) which are authenticated. Eventually the authenticated DN will be part of the input to an access control decision. We can assume, therefore, that the agent

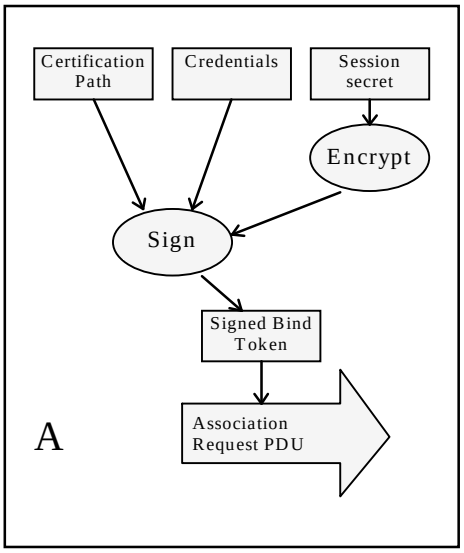

Figure 4. Modified scheme for Authentication of A to B

has access to a table containing, *inter alia*, the DNs of the entities with which it is prepared to associate. There is no reason why the agent should not extend the scope of this table and use it to cache certificates for the DNs - there are no issues of secrecy here since our certificates merely confirm signatures, they do not imply privileges. Certificates are, of course, designed to be unforgeable. If a certificate is present in the cache then the agent can avoid going to the Directory during the authentication phase. If it must go to the Directory then the certificate(s) retrieved can be added to the cache. If the table entries are implemented as Managed Objects then a manager could place certificates in the cache (or delete them) using CMIP operations.

A further step would be to cache the results of certificate verification so that a subsequent association request from the same entity is passed "on the nod" so to speak. We would want some control over the duration of this cacheing so we propose adding attributes to the MO which will allow a time limit to be set. Obviously the time limit present in the certificate would also need to be checked.

In principle, verification of a certificate should be accompanied by checking whether the certificate has been revoked by its issuer. Normally this would require a Directory operation. Clearly we cannot ignore this; especially as a certificate may have been revoked because the key it certifies has been compromised. If this were the case it would be very dangerous to trust any party which authenticated itself with that key. We propose that the "cached verification" scheme should be extended to cover the checking for revocation. Thus the result of a revocation check will be cached for a period which is configurable by management. A Directory operation will only be required when the cached result becomes "stale".

Obviously there is a loss of security inherent in these cacheing strategies. However, the extent of this is under management control and is selectable on a per-DN basis. DNs for entities with high privilege could have cache timeout periods set to zero - effectively disabling cacheing.

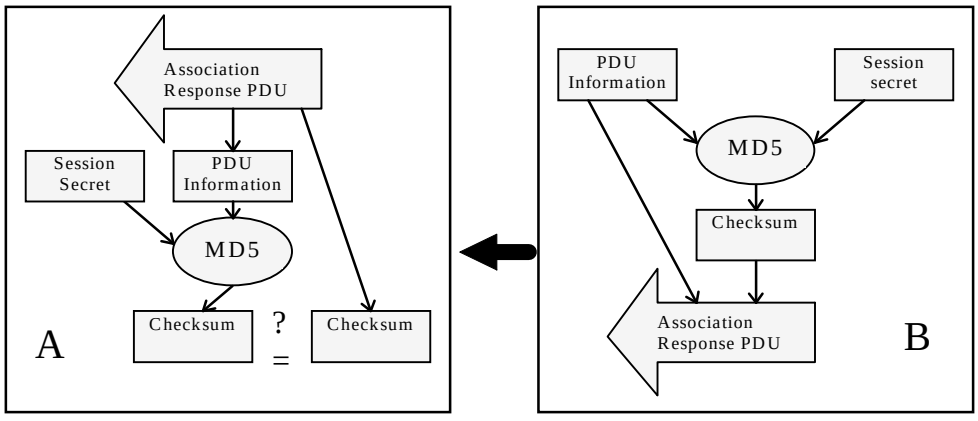

Figure 3. Authentication of B to A using MD5

One final point on revocation; we maintain a clear distinction between authentication of identities and access rights. The certificates we are

using at present merely confirm a signature, they do not imply any particular of privilege or security clearance. It is quite possible that several certificates, from several CAs, may be available for a single DN. For example, the authors have certificates issued by two CAs: the main UCL-CS one and a special MIDAS one. If the MS is prepared to trust both these CAs, then a certificate from either will do for authentication. The privileges granted will be the same in each case, however, since these are dependent solely on the DN. Note that revocation of (say) the MIDAS certificates would make no difference so long as the MS continued to trust the UCL-CS authority. If a manager **does** want to revoke an entity's privileges it must do so by manipulation of the access control objects in the MIB (see Section V).

Assuming these cacheing schemes are in use then, in the best case, another RSA operation and maybe a Directory operation also are eliminated.

IV.D. Summary

In the best case - with A's public key already verified and cached the RSA operations for mutual authentication are now:

At the client	At the agent
Verification of B's certificate	Checking the signature on the credentials
RSA encryption of the session secret	Decrypting the session secret
Signing of the credentials	

This is a total of 5 RSA operations for mutual authentication, a reduction of 11 operations.

V. Access Control - a Discussion

For maximum flexibility, we would like our managers to be able to specify access control rules which control access on a per user, per operation and per MO basis. It is easy to invent schemes which do this. It seems to be less easy to invent schemes which do this with acceptable performance penalties.

The ISO standards propose a scheme which includes "item rules" [12]. These allow access control rules to be applied to a set of objects, attributes and operations in a highly flexible way. The rules allow very fine-grain access control - down to the level of individual attributes. It seemed to us that this fine granularity would be very costly at run-time and was probably unnecessary. If the MIB was designed in a reasonable way then all the attributes of a MO should be closely related. This would make it unlikely that one would wish to grant privileged access to some attributes but not to others. Granularity at the level of complete MOs seems to us to be the best compromise.

The key point, however, is that the target objects to which a rule should apply are designated through a "scoping and filtering" expression. Recall that the ISO standards structure management information as a tree of MOs. "Scoping" selects a sub-tree of the MO tree, "filtering" refines this selection through the application of a logical expression to the values of chosen attributes. Scoping and filtering are part of the normal mechanism for selecting objects to which management operations should apply. Extending the concept to access control is very powerful. For example it allows the definition of a rule which denies access to "all MOs of class Transport Connection if the remote transport address is X". Most importantly, such a rule can be defined even though there are currently no MOs satisfying the filtering expression. This means that, as soon as an object appears which **does** satisfy the filtering expression, it is protected.

Unfortunately this scheme can be extremely expensive at run-time. If the filter expression can be completely general, then the set of objects to which it applies may change with time. Imagine a rule which allows "Knight" to access "all Transport Connection MOs which have error counts greater than X". If such a rule is allowed, then every time a CMIP operation is performed, filtering and scoping have to be carried out at least twice; once to determine the set of objects to which access is requested and once for each access control rule (to determine to which MOs that rule currently applies)

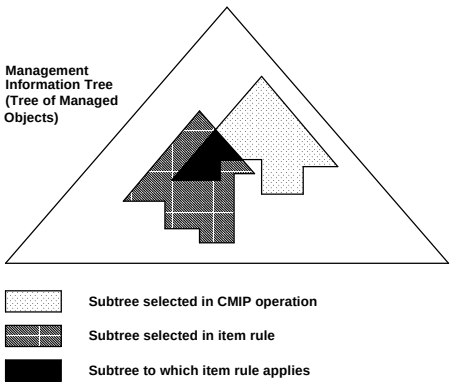

Figure 5. Interaction of scoping and filtering with item rules

- this is illustrated in Figure 5. Furthermore, you cannot cache the results of the rule determination as

these may be dependent on attributes which vary in time. We believe that such a scheme is unworkable. As a consequence of this we do not have the item rules in our initial implementation.

We feel there are a number of requirements (some conflicting) for an access control scheme:

i. For efficient operation the access control information needs to be kept close to the MOs to which it applies. Ideally it is stored in the MOs themselves.

ii. It is useful to be able to define access control rules for MOs even though no MOs satisfying those rules currently exist. This is clearly in conflict with *i*.

iii. It is important that access control rules should be applied consistently. This argues for a certain amount of centralisation of access control information. Again, there is a conflict with *i*.

Currently we are considering the following scheme:

- We specify "policy" objects which store sets of rules of the form {Identity, Permissions} such as

```
{c=GB@o=UCL@ou=CS@cn=Knight, GET:SET}
```

These rules apply at the granularity of the MO.

- Within each MO we store a set of references to the policy objects which apply to it. (The indirection implied here is necessary in order to satisfy *iii* above.). These references are stored as attributes and so are accessible through normal management operations (themselves subject to access control of course).

We believe such a scheme can be implemented efficiently. However, we have apparently lost one of the most powerful features of the "item rules" - the ability to specify protection for MO instances which have not yet been created. We can restore this functionality by exploiting the tree-structure of the management information. When a MO is created it has a unique superior in the tree. We propose that, in addition to the references to policy objects described above, each MO can have a set of policy references specifying protection for its potential direct subordinates. This is similar to the access control mechanism used in the "Quipu" implementation of the X.500 Directory[13].

A refinement of this scheme would be to allow the subordinate protection to have a filtering expression. Thus, in a MO of class "Transport Entity" there could be a filter attribute which says that a particular policy reference should appear in a subordinate if:

```
{subordinate class == Transport
    Connection && Remote Address != X}
```

We believe this preserves the most valuable parts of the functionality offered by the item rules which confining the evaluation of filters to the comparatively rare event of MO creation. Incidentally, the Quipu implementation includes another useful optimisation since it allows attributes which are duplicated across many objects to be shared. This approach is likely to reduce our memory requirements considerably since access control attributes frequently are shared.

VI. Conclusions

The MIDAS project has demonstrated both the need for secure management and the implementation of it. The security functions were originally introduced with special attention being paid to generality, re-use of existing syntaxes etc. A consequence of this has been high functionality but low performance. We have tried to address the performance issues in this paper by suggesting a new authentication mechanism which can eliminate more than half the RSA operations. This mechanism relies on the introduction of a modified bind token and what is effectively authentication by "Challenge". We have also looked at working methods which can reduce the number of Directory accesses and certificate verifications.

Once the initiator of an association has been authenticated , any operations that are launched must be subject to access control rules. The authors recognise the power of the so-called item rules but do not believe that theses are implementable in a reasonable way. Instead we are studying a scheme which exploits the tree structure of the MIB in which references to access control information is stored with the MOs themselves with initial values being stored in the superior MOs.

VII. Acknowledgements

We would like to thank the reviewers of the paper for their helpful suggestions which have enabled us to improve the paper in a number of important areas.

VIII. References

1 UCL Department of Computer Science, The OSI Management Information Service User's Manual, Version 1.0 for system version 3.0, Feb.1993.

2 ISO/IEC 9595, Information technology -- Open Systems Interconnection -- Common

management information service definition}, May1990.

3 G. Knight, S. Bhatti, L. Deri, Secure Remote Management the ESPRIT MIDAS project, Proceedings of IFIP WG 6.5 International working Conference on Upper Layer Protocols, Architectures and Applications, Barcelona, Jun. 1994

4 J. Galvin, K. McCloghrie, Security Protocols for version 2 of the Simple Network Management Protocol (SNMPv2), Internet RFC1446, Apr. 1993.

5 OSI Interoperability Workshop, OIW Stable Implementation Agreements for Open Systems Interconnection Protocols: Part 12 -- OS Security, Jun. 1993.

6 NM-Forum, Applicatio`n Services: Security of Management, OMNIPoint/NM-Forum 016, Bernardsville, NJ, Aug. 1992.

7 R. L. Rivest, A. Shamir, L. A. Adleman, A Method for Obtaining Digital Signatures and Public Key Cryptosystems, Communications of the ACM, number 21, volume 2, pages 120-126. Feb. 1978

8 R. Rivest, The MD5 Message-Digest Algorithm, Internet RFC1321, 16 Mar. 1992.

9 ISO/IEC 9072, Information processing systems -- Text Communication -- Remote Operations, 1989.

10 UCL Department of Computer Science, The OSI Security Package OSISEC User's Manual, May 1993.

11 CCITT Recommendation X.227, Connection Oriented Protocol Specification for the Association Control Service Element, Sep. 1992.

12 ISO/IEC CD 10164-9.3, Information Technology -- Open Systems Interconnection -- Systems Management -- Part 9: Objects and attributes for Access Control, Borehamwood, UK, Dec. 1992.

13 UCL Department of Computer Science, The ISODE User's Manual, Version 7.0, Jul. 1991.

Author Information

Graham Knight obtained a BSc in Mathematics from the University of Southampton in 1970 and an MSc in Computer Science from UCL in 1980. He has taught and researched in the field of data communications for 15 years. He has led UCL teams in EU-funded collaborative projects since 1986. His research has focused on network and systems management, teleworking and ISDN. Most recently he has been involved in the ESPRIT MIDAS project which has studied: the security of network management systems, interworking multi-media applications between LANs and ISDN, and teleworking.

Saleem N. Bhatti received a B.Eng.(Hons) in Electronic and Electrical Engineering in 1990 and a M.Sc. in Data Communication Networks and Distributed Systems in 1991, both from University College London. Since October 1991 he has been a member of the Research Staff in the Department of Computer Science involved and research in various communications related projects, and also in teaching. He has worked particularly on projects involving ISDN and Network & Distributed Systems management.