SECURE REMOTE MANAGEMENT IN THE ESPRIT MIDAS PROJECT

Graham Knight[a], Saleem Bhatti[a] and Luca Deri[b]

[a]Computer Science Dept., University College London, Gower Street, London WC1E 6BT

[b]IBM Zurich Research Laboratory, Saumerstrasse 4, CH-8803 Rueschlikon, Switzerland

ABSTRACT

This paper describes work carried out in the ESPRIT "MIDAS" project to provide for secure management in the context of the ISO standards for network and system management. The intention of the MIDAS work is to make use of security mechanisms which have already been standardised (for example X.509 authentication) and to make these available through a conventional implementation of the CMIP protocol.

The principle application for MIDAS is the management of large X.400 systems. The paper presents an analysis of the security requirements for this application and describes the details of the mechanisms which are being implemented.

Keyword Codes: C.2.1, C.2.3, C.2.4
Keywords: Network operations; Distributed Systems, Application Layer Communication Services, Security aspects; Message Handling Systems; Network Architecture and Design

1. INTRODUCTION

MIDAS (Management in a Distributed Application and Service Environment)[+] is a two-and-a-half year project funded under the EU ESPRIT programme. The objective of the project is to demonstrate remote management of distributed applications in a service environment. Two environments have been chosen for the demonstration; a large public administration network run by CSI-Piemonte in Northern Italy and networks of teleworkers operating over ISDN in Berlin and London.

The intention is that the management systems should be used to *control* the applications not merely to monitor them. This, together with the fact that the applications in question are providing services to real users, means that security considerations are paramount.

This paper discusses the requirements for secure remote management in such an environment and describes the way in which the required security facilities are being realised.

[+] The MIDAS partners are: System Wizards Srl., CSI-Piemonte (Italy), GMD-FOKUS, Cellware GmBH (Germany), University College London (UK)

2. APPLICATIONS AND SECURITY

MIDAS is concerned with providing management facilities for distributed applications and services. Such applications may, in fact, operate across several networks run by several administrations; this could be the case, for example, in a geographically-spread X.400 network operating across a mixture of public and private networks. In general, we must assume that such underlying networks are insecure; that management information may be destroyed or stolen, that malicious third-parties may be able to gain access to the networks and disrupt management activities in a variety of ways. In such cases, the management and security facilities we require must be placed in the parts of the system we can trust - in the applications themselves.

Secure management is especially important for MIDAS since we are keen to demonstrate *control* of applications and services through the use of the OSI Common Management Information Service (CMIS)[1] and not simply the *monitoring* of activity which characterises most SNMP-based management. The applications we wish to control often have native security facilities; for example X.400 offers a range of services including confidentiality, non-repudiation etc. If such an application is to be controlled remotely then there is a danger that these security services may be undermined by a third party through subversion of the management exchange. In the particular case of X.400 for example, subversion of this control could allow a third-party to cause mail for one user to be delivered to another. In fact, the native security services will be only as strong as the security services that are provided for management.

MIDAS, like most ESPRIT projects, is keen to develop technology which can be deployed commercially in the short to medium term. Consequently we have sought security solutions which can readily be introduced into today’s OSI environment.

3. BACKGROUND WORK

Issues of security in distributed applications have been studied extensively and a number of aspects have now reached the stage of standardisation by various bodies. Given our contexts of Open Systems and Management we have drawn heavily on the following;

- ISO/ITU work on authentication through the use of public key encryption and trusted "certification authorities" [2]. This approach to security has been adopted by several OSI applications so there is a strong motivation to use it for management.
- ISO/ITU work on access control both in general [3] and specifically in the case of management [4].
- Internet work on security in SNMPv2 [5].
- The US "OSI Implementors Workshop" (OIW) "Stable Agreements" which attempt to make workable selections from amongst the efforts of the standards bodies [6].
- The OMNIPoint/NM Forum security work [7] which draws heavily on the OIW work.

Unfortunately, at the present time, none of this work completely satisfies our requirements. This is especially true when one comes to the detail of implementation where one frequently discovers gaps in specification or unresolved inconsistencies. When this has occurred we have had to make our own choices. Many of the problems we have encountered will be resolved

by the introduction of the Generic Upper Layers Security standards[8], however, these were not considered sufficiently stable.

4. THREATS TO MANAGEMENT SECURITY

The range of threats to which communication may be subject and the security services through which it may be defended have been described extensively [5][7]. A brief summary is given here in the context of remote management.

When a management interaction takes place between two systems across a network one can usually identify "managing" and "managed" roles. A third-party system may attempt to subvert the management interaction in a variety of ways:

T1) by *masquerading* as a legitimate Managed or Managing system and then performing unauthorised management operations;
T2) by *modifying* information which is in transit between P and Q;
T3) by *re-ordering* or *re-playing* messages in transit between P and Q;
T4) by *capturing* confidential management information in transit between P and Q;
T5) by so *disrupting* traffic between P and Q that management becomes impossible.

5. MIDAS PRIORITIES

Threats T1 and T2 are identified as "Primary Threats" in [7] and these are certainly the most important threats for MIDAS. If these are not tackled than it is possible for a managed system to be fooled into acting on bogus management commands. Although the dangers in this are most apparent for the *managed* system they exist also for the *managing* system since, if such a system receives false reports about the status of the system it is managing, it may be fooled into taking inappropriate management actions.

Threat T3 is perhaps the most easily perpetrated given the ease with which PDUs may be recorded and re-transmitted on a broadcast LAN; it is important for MIDAS to tackle this.

The defeat of threat T4 can, in fact, be achieved quite simply by encryption of all management PDUs. However, this is quite expensive computationally and we are sensitive to the general requirement that management operations should not noticeably affect the performance of the systems they are managing. We are aware that failure to defend against T4 does impose some restrictions on what we can do via remote management. For example, we must not carry in management PDUs information which the applications we are managing would not carry in clear. This includes secret keys, message contents etc. However, we see no need for carrying such information in management PDUs at the present time. In some environments the requirement for confidentiality may be more stringent, there may be a need to protect mail routing and traffic information for example, however we see no need for this in the MIDAS demonstrator. Hence we will not defend against threat T4.

Threat T5 is very difficult to defend against. In fact, an attack which flooded an intermediate network with junk traffic is indistinguishable from normal network congestion as far as the end-systems are concerned. An attack which subverted (say) the ROS[9] or Transport services by injecting disruptive PDUs into the stream cannot be countered by a secure management mechanism but would certainly deny service.

6. SECURITY SERVICES

The following security services are identified in [7] as being suitable to counter the "primary threats".

S1) *Peer entity authentication* which establishes unambiguously the identity of the initiator of an operation and is part of the counter to threat T1 above. Further, having a properly authenticated identity may be an important input to an access control decision.
S2) *Data origin authentication* which provides an assurance that data really does come from where it seems to. This forms part of the counter to threat T1.
S3) *Connectionless integrity* which aims to ensure that management PDUs cannot be modified without detection and so counters threat T2.
S4) *Stream integrity* which guards against mis-ordering of PDUs in a stream (including re-plays) and so counters threat T3.
S5) *Access Control* which enables a system to discriminate between operations it will allow and those it will not.

In the next section we explain the choices we have made of security mechanisms to support these services.

7. 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 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 NM-Forum and the US OSI Implementors' Workshop[6][7].

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:

- 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 ready for use - this is what happens in SNMPv2. With an association it is natural to negotiate a new secret value when an association is established thus eliminating the need for secure storage.
- The protocols employed to maintain the association guarantee sequenced delivery with very high probability. Further, each PDU has an "invoke id" field - an integer which we can insist must take values from a known sequence.This greatly simplifies the design of a stream integrity mechanism. To achieve the same with SNMPv2 requires a rather complex shared clock mechanism.
- 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 have chosen:

M1) To *authenticate associations* through the use of *public key encryption* using the RSA algorithm. This is the mechanism described in X.509 [3, 7] and provides service S1.
M2) To add *cryptographic checksums* to all management PDUs calculated according to the MD5 algorithm [10]. This provides services S2 and S3.
M3) To use a "well-known" invokeID sequence in ROS PDUs and so provide service S4
M4) To implement access control in the managed system as per [4] (service S5).

The sections below discuss the implementation of these mechanisms in more detail.

7.1. MIDAS constraints

Although MIDAS is constructing its own management platform, which includes full CMIP support, we would like, if possible, to ensure that the mechanisms we choose can be used with other CMIP implementations. This means that, not only must we use standard CMIP PDUs, but also we must try to ensure that the various security tokens we use can be passed across "industry standard" APIs like XMP[11].

7.2. Secure associations

M1 is a comparatively expensive operation since the calculations required for the RSA algorithm are complex; therefore we perform this authentication just once as association set-up time. Mechanism M2 - which is relatively cheap - is then applied to all subsequent PDUs sent on the association. In this way we obtain a strong assurance that all operations on the association originate from the entity whose identity we have authenticated. This gives us protection against threats T1 and T2 and provides the basis for our access control decisions.

The entity which is authenticated can represent a variety of real-world objects; humans, hosts, processes etc. as determined by the security policy in force. Which it is is of little importance to the recipient of the communication since this cares only whether the authenticated identity represents an entity with the necessary authorisation. In fact, MIDAS restricts the identity to be a "Directory Name" (DN); ie. a distinguished name of an entry in the global X.500 Directory. This guarantees that identities are globally unique and is well-suited to M1.

The algorithm used by M1 is based on an entity demonstrating its possession of the RSA secret key for its own identity. Before any authentication can take place, the entity must establish its right to assume a particular identity and obtain the corresponding secret key. How this is done is a purely local matter; two ways which are likely to be used in MIDAS are:

- *Use of smart-card*. "Identity" (DN) plus secret are held on the card. The human user establishes his/her right to possess the card by typing a "PIN" when the card is read;
- *Use of the filestore*. DN and secret are tied to "userids" on the client system. The userid is verified through the normal password checking mechanism and is mapped to the name of a secure file which contains the DN and secret.

Mechanism M2 requires there to be a shared secret and it is desirable that this be generated anew for each association and exchanged (confidentially) at association set-up time. Confidentiality is achieved by encryption using the recipient's public key. Figure 1 shows the complete association set-up exchange.

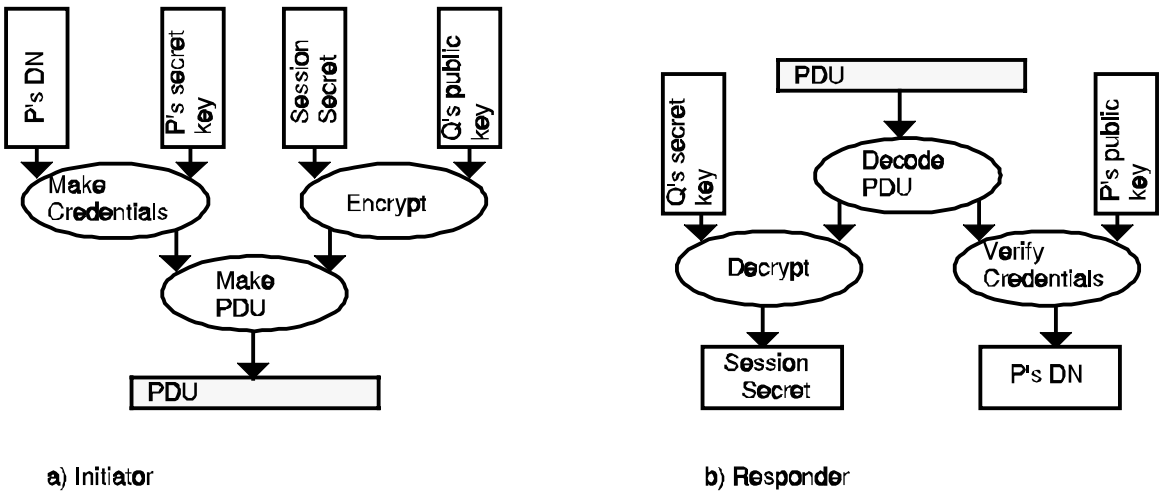

Figure 1 Association set-up

In Figure 1 the "credentials" are formed essentially in accordance with X.511[12]. There is no explicit place in the X.511 "credentials" syntax for carrying a session secret. One possibility is to make use of the "random" field; however we are concerned that this field may be used for other purposes in some security packages. At present, a modified syntax is used consisting of a SEQUENCE of X.511 StrongCredential plus an OCTET STRING for the secret.

```
-- MIDAS Credentials

SessionSecret ::= SIGNED ENCRYPTED SEQUENCE {       -- as per X.509
        secretId        OBJECT IDENTIFIER,
        secret          OCTET STRING (SIZE (8))}

MidasCredentials ::= SEQUENCE {
        credential      StrongCredential,           -- as per X.511
        sessionSecret   SessionSecret OPTIONAL }
```

This syntax is carried in the *user-information* field of the AARQ PDU and so may easily be passed across the XMP API.

We have also to decide how to carry the integrity checksum in the CMIP PDUs - no field is reserved for this. Initially we planned to use the CMIP *access control* field as we did not intend to use it for our access control mechanism. However, the field is not present in all CMIP PDUs and we were not happy with our attempts to overload other fields to remedy this. Instead, we overload the *invokeID* field in the ROS PDU which now carries the real *invokeID* value together with the checksum. The algorithm we use at present is as follows:

P parameters of the CMIP PDU encoded with DER
S the session secret
I_{VAL} the "real" value of the *invokeID* which must be carried to the peer entity
I_{TX} the value actually placed in the *invokeID* field for transmission
X the MD5 checksum

$$X = MD5(P, S), \qquad I_{TX} = I_{VAL} \oplus X \quad \text{(where } \oplus \text{ means "exclusive or")}$$

We further require that the I_{VAL} values belong to a known sequence. Since the receiver knows this sequence, and CMIP preserves PDU order, the receiver is able to repeat the calculation, verify integrity and extract I_{VAL}. If a CMIP user has several asynchronous requests outstanding then the calculation of I_{TX} must be done for each of the outstanding I_{VAL} values - however, this calculation is not costly.

One possible weakness of this scheme is that two identical PDUs will generate the same value of X. An attacker in possession of two such PDUs would not know X directly but would possess values of $I_{VAL} \oplus X$ for two values of I_{VAL} a known distance apart in the sequence. If the I_{VAL} values belonged to a completely determined sequence (increasing integers starting at zero for example) then a replay attack would be simple. Instead, the values are generated from a recurrence relation based on a random sequence generator with initial seed S. Thus, an intruder must know S both to calculate the checksum and to produce the next valid *invokeID*. At present we believe that the nature of the random sequence generator used makes an attack based on capture of two identical PDUs infeasible. Should this not prove to be the case the scheme can be modified so that "identical" PDUs do not generate the same MD5 checksum. One way of doing this is to calculate:

$$X = MD5(P, S, I_{VAL}), \qquad I_{TX} = X$$

The disadvantage of this is that it is more costly in the case where there are multiple asynchronous requests outstanding since the requestor must now calculate the MD5 checksum for each of the possible values of I_{VAL}.

It is possible to use the *invokeID* field in this way in any implementation which allows the CMIP user to allocate *invokeID* values. In fact this mechanism can be used to carry integrity checks for *any* application layer protocol based on ROS. We now have an experimental implementation of the algorithm and we are studying it in terms of its efficiency and strength. It is possible that the details will change in the light of this.

7.3. ACCESS CONTROL

The basic building block of OSI management is the "Managed Object" (MO) - an abstract representation of a resource which is to be managed. OSI management essentially boils down to the remote manipulation of the attributes, actions and notifications of MOs. Secure OSI management additionally ensures that a particular manipulation of a particular attribute (or whatever) can only be invoked by an entity which is authorised to do so. Access rights are generally expressed as a set of access control *rules* which is information passed to each access control decision.

The two main issues to decide are the granularity of the access control and the means whereby the initiator of the operation is identified to the access control function.

7.3.1. Granularity

The coarsest level of granularity applies access control solely to the management association. If the association is accepted then the initiator has unrestricted access, otherwise it has none. The finest level applies access control on a per-attribute, per-operation basis.

The "*item rule*" mechanism defined for OSI management in the committee draft which was available to us[4] does indeed allow this very fine granularity; further, it specifies a very powerful mechanism to identify the "*targets*" (MOs, attributes etc.) to which an access control rule should apply. A manager may create an arbitrary number of item rules each of which specifies the access rights of a set of *initiators* (See below)with respect to a set of targets. The target of an item rule is specified through CMIP scoping and filtering expressions. It appears, therefore, that the determination of which item rules are applicable to a particular operation requires the evaluation of scoping and filtering expressions for each item rule present in the system - a very heavy run-time load. Note that it is not possible statically to mark the MOs and attributes to which a particular item rule applies since this set is dependent on the evaluation of filtering expressions and so varies as attribute values vary. We do not see a reasonable way to implement the full function of the item rules as specified in [4] and our initial implementation will not include them in any form. We believe, in any case, that the next draft CD may change the item rule mechanism. We *have* implemented the *global* and *default* rules defined in [4] which work as follows:

- Global rules apply in the absence of item rules. They apply to every MO in the Managed System but only with respect to a designated list of "initiators". Global rules may either grant or deny access;
- Default rules apply in the absence of item and global rules. They apply to every MO in the Managed System and for all initiators.

One can use these rules to partition the initiators of management operations into three sets; the "privileged", the "ordinary" and the "excluded". The "privileged" initiators are identified in global "grant" rules and can perform whatever operations these rules grant - probably this would allow everything. "Ordinary" initiators can perform only the operations allowed by the "default" rules - perhaps "GET" only. The "excluded" initiators are identified in global "deny" rules and are denied all access. The precise set of operations allowed in each case is configurable through OSI management.

We feel that the granularity provided by the global and default rules is sufficient for our present needs. In the future we do see a need for partitioning the MO containment tree so that some initiators have privileged access to certain sub-trees but not to others. It seems this can be achieved by implementing the item rules without the filtering expression.

7.3.2. Initiators

Here we must distinguish between the "initiator" of a management operation and the "initiator information" which is the input to an access control decision. For our purposes, the former is the entity whose identity has been authenticated for the association and is identified by a DN, the latter may or may not be the same thing.

It is possible to use the DN directly as a parameter of an access control decision through the use of access control lists (ACL) associated with each access control rule. For example, the ACLs might contain the DNs of human users and so allow access rights to be allocated

on a per-user basis - similar in fact to what pertains in a typical multi-user file system. However, it can become difficult to maintain consistency in ACL-based schemes when large numbers of objects and systems are being managed by large numbers of users. For example, downgrading a user's privileges can mean that large numbers of ACLs must be searched and the user's identity be removed from each of them.

Alternatively, the DN can be mapped onto a capability or security label for use in capability or label-based scheme. Given that our first implementation will use only the global and default rules with their "privileged"-"ordinary"-"excluded" semantics, we favour a label-based scheme which reflects these semantics.

At some point there has to be a mapping from DN to label; the standards leave open the issue of where this should happen. We do the mapping in the Managed System. This allows us to base all access decisions on the DN which was authenticated at association set-up time. The authenticated DN is mapped onto a label which is then used for all subsequent operations on the association (we insist that all associations are authenticated as described in Section 7.2) This scheme avoids the managing system having anything to do with labels and the consequent need to secure labels in transit.

8. CONCLUSIONS AND CURRENT STATUS

MIDAS requires the following security services in the management platform:

i) *Peer entity authentication*
ii) *Data origin authentication*
iii) *Connectionless integrity*
iv) *Access Control (label based)*

In order to provide these services MIDAS will use the following security mechanisms

i) *Authenticated associations* based on Directory Names and public key encryption using the RSA algorithm. .
ii) *Cryptographic checksums* in PDUs calculated according to the MD5 algorithm and carried in the ROS *invokeID* field.
iii) *Access control*.

The mechanisms described in this paper are being incorporated into the Managed System Platform which is being developed by UCL within the MIDAS project; the UCL "OSISEC"[13] package is being used to provide cryptographic support. The release of a secure version of this platform to the other partners is scheduled for the end of May 1994 and this software will be incorporated into the next full release of UCL's OSIMIS[14] software at about the same time. No systematic analysis has yet been completed on the impact the security mechanisms have on the performance of the platform. Subjective experience suggests that the use of the integrity check has no noticeable effect on performance whilst the authentication dialogue has a larger but still acceptable impact.

REFERENCES

[1] ISO/IEC 9695: "Information Technology - Open Systems Interconnection - Common Management Information Service Definition", 1991

[2] CCITT Recommendation X.509 (ISO DIS 9594-8) "The Directory - Authentication Framework", Gloucester, November 1987

[3] ISO/IEC JTC1 SC21 DIS 10181-3.2 "Information Technology - Open Systems Interconnection - Security Frameworks in Open Systems - Part 3:Access Control Framework"

[4] ISO/IEC JTC 1/SC 21/N 7661 CD 10164-9.3 "Information Technology - Open Systems Interconnection - Systems Management - Part 9:Objects and Attributes for Access Control"

[5] J. Galvin, K. McCloghrie, "Security Protocols for version 2 of the Simple Network Management Protocol (SNMPv2)", 05/03/1993.

[6] "OIW Stable Implementation Agreements for Open Systems Interconnection Protocols: Part 12 - OS Security", OIW, June 1993

[7] OMNIpoint/NM-Forum 016 "Application Services: Security of Management", NM-Forum, Bernardsville, NJ, 1992

[8] ISO/IEC CD 11586-1, "Generic Upper Layers Security - Part 1: Overview, Models and Notation

[9] ISO/IEC 9072, "Information Processing Systems - Text Communication - Remote Operations", 1989

[10] R. Rivest, Internet RFC1321, "The MD-5 Message Digest Algorithm", April 1992

[11] "Systems Management: Management Protocols API (XMP), X/Open Preliminary Specification, X/Open Co. Ltd, July 1992

[12] CCITT Recommendation X.511 (ISO 9594-3) "Information Processing Systems - Open Systems Interconnection - The Directory - Abstract Service Definition", March 1988

[13] "The OSISEC Security Package - OSISEC User's Manual (Release 1.0, v0.5)", OSI Security Team, University College London, May 1993

[14] "The OSI Management Information Service User's Manual (Version 1.0, for system version 3.0)", The UCL Network and System Management Team, Feb 1993