Poseidon House
Castle Park
Cambridge CB3 0RD
United Kingdom

TELEPHONE: **Cambridge (0223) 323010**
INTERNATIONAL: **+44 223 323010**
FAX: **+44 223 359779**
E-MAIL: **apm@ansa.co.uk**

ANSA Phase III

A Framework for Federating Secure Systems

John Bull, Owen Rees

Abstract

This document proposes a framework for security in ANSA.

A major concern of ANSA is support for systems that are not part of a common management hierarchy, but which nevertheless need to cooperate. The ANSA security framework shows how security may be provided, both within and between such systems.

ANSA has adopted an object-based model to support the requirements of distribution. The ANSA security framework explains how security concerns fit into the model.

APM.1006.00.02 **Draft** 21 October 1993

Architecture Report

Distribution:
Supersedes:
Superseded by:

A Framework for Federating Secure Systems

Architecture Report

A Framework for Federating Secure Systems

John Bull, Owen Rees

APM.1006.00.02

21 October 1993

The material in this Report has been developed as part of the ANSA Architecture for Open Distributed Systems. ANSA is a collaborative initiative, managed by Architecture Projects Management Limited on behalf of the companies sponsoring the ANSA Workprogramme.

The ANSA initiative is open to all companies and organisations. Further information on the ANSA Workprogramme, the material in this report, and on other reports can be obtained from the address below.

The authors acknowledge the help and assistance of their colleagues, in sponsoring companies and the ANSA team in Cambridge in the preparation of this report.

Architecture Projects Management Limited

Poseidon House
Castle Park
CAMBRIDGE
CB3 0RD
United Kingdom

TELEPHONE UK	(0223) 323010
INTERNATIONAL	+44 223 323010
FAX	+44 223 359779
E-MAIL	apm@ansa.co.uk

Contents

1 Overview

This document proposes a framework for security in ANSA.

A major concern of ANSA is support for systems that are not part of a common management hierarchy, but which nevertheless need to cooperate. The ANSA security framework shows how security may be provided, both within and between such systems. A collection of systems which can interact but which are not part of a single hierarchy is called a federation. The ANSA security framework addresses the need to provide security in federations.

ANSA has adopted an object-based model to support the requirements of distribution. The ANSA security framework explains how security concerns fit into that model.

1.1 Purpose of the framework

The purpose of the ANSA security framework is to help designers to understand the issues of security in federations of distributed systems, and to understand how various mechanisms to support security and distribution can be combined.

The framework identifies the concepts and terminology which are used to describe security, and in particular, secure computer systems.

The framework identifies the issues that arise in distributed and federated systems which affect the approach to security.

The framework does not prescribe any particular security policy; it is intended to allow a wide range of policies. Policies will depend upon the organisations to which they are to be applied. The framework can help the administrators to understand the implications of their choice of policy; it cannot make that choice for them.

The framework is intended to allow the use of existing security mechanisms, so it identifies security problems for which security mechanisms may be solutions, rather than dictating that particular security mechanisms be used for particular purposes. The framework is intended to support evolution by allowing improved security mechanisms to be introduced with minimal disturbance.

Since the framework is part of ANSA, it is consistent with other parts of the architecture. This applies both to the style of the descriptions and to the proposed structure for mechanisms to support security.

1.2 Topics to be covered

This document covers a number of topics and must address several audiences.

One audience is those who wish to design, implement or operate distributed systems, and who are aware of the need for security, but who are not familiar with the work that has been done in that field.

Another audience is those whose primary interest is security but have not studied the problems of federated distributed systems in any detail.

There are also other potential readers, including those who are familiar with both security and distributed systems and wish to study the ANSA approach to the requirement for security in distributed systems.

Given the variety of audiences, it is inevitable that readers will find that the treatment of their areas of expertise does not explore issues in depth.

The remainder of the document is structured as a number of chapters; a guide to the content of these chapters is given below.

1.2.1 Security concepts and terminology

2 is mainly for the benefit of those who are not familiar with established work in the field of computer systems security. It is intended as an introduction to security concepts and terminology which will be sufficient to understand the rest of the document. Those who wish to study any of the issues in more detail will find many references to published work which they may use as a starting point.

1.2.2 Architectural implications

3 introduces those aspects of ANSA that affect the approach to security that has been adopted. It includes discussions of assumptions to be avoided, the use of projections, the object-based model, and the naming model.

This chapter sets out to show how security can be integrated into ANSA by pointing out relationships between security concepts and concepts that have been introduced into ANSA for other reasons.

1.2.3 Monitoring and accounting

4 is a brief introduction to the issues of observing what is happening and responding when appropriate. It also considers the issue of how the effectiveness of the security mechanisms can be measured.

1.3 Additional reading

This document gives only a brief introduction to concepts described in detail in other ANSA documents. Of particular importance for this document are the concepts and definitions in The ANSA Naming Model [AR.003], and the interaction model component of The ANSA Computational Model [AR.001].

2 Security concepts and terminology

A great deal of work has been and is being done on computer security, and some aspects of computer security have been studied in detail. This chapter is intended to introduce, to those readers not familiar with the field, the main concepts and terminology that have been developed; and to point readers to the literature, so that they may build an understanding of the issues.

2.1 Background

2.1.1 Introductory material and glossaries

A general introduction to computer security can be found in a variety of textbooks, such as the one by Denning [DENNING83]; more recent work is covered in books such as the one based upon work carried out as part of the CEC COST-11 Ter. Programme [MUFTIC89]. Annex A of the OSI Security Architecture standard [ISO89] gives a useful overview which is more generally applicable than its title “Background Information on Security in OSI” suggests.

Extensive glossaries can be found in the documents produced by various standards organisations [ISO89] [ECMA88] [ECMA89] [DOD85] [NCSC87]. Most terms can be found in at least one of these sources but caution is required as in some cases the definitions differ. In addition to the standards that have been completed and published, work is in progress on further standards which define additional terminology or modify definitions of previously defined terms [ISO92a] [ISO91c] [ISO91d].

2.1.2 A brief survey of existing work

2.1.2.1 The access matrix

Lampson proposed the access matrix [LAMPSON71] as an abstract model for the wide variety of access control or protection systems that had been developed. This model requires the identification of the entities to be protected, and the identification of the entities to which rights may be granted. These are conventionally known as objects and subjects respectively, following the terminology of Graham and Denning in their exposition and reformulation of Lampson’s model [GRAHAM72].

Each entry in the access matrix gives the access privileges of a subject to an object. Graham and Denning describe three strategies for storing the access matrix efficiently: as capability lists, as access control lists, and using a lock and key approach that combines the features of the other two.

Identity-based security policies (definition in [ISO89]) are commonly explained by reference to the access matrix model.

2.1.2.2 The reference monitor concept

Graham and Denning also introduce the notion of a monitor for each type of object through which all access to objects of that type must pass. This idea appears in a slightly modified form in the Anderson Panel report [ANDERSON72] which introduced "a reference monitor which enforces the authorized access relationships between subjects and objects" and the reference validation mechanism as an implementation of the reference monitor concept. The Anderson Panel also adopted the name "security kernel" for a reference validation mechanism.

The three design requirements for a reference validation mechanism were defined by the Anderson Panel (quoted in [DOD85] as follows):

- The reference validation mechanism must be tamperproof.
- The reference validation mechanism must always be invoked.
- The reference validation mechanism must be small enough to be subject to analysis and tests, the completeness of which can be assured.

These requirements assume a single central mechanism which is not appropriate to federated distributed systems. The other aspects of these requirements remain valid.

2.1.2.3 Multilevel Security

An alternative to using an access matrix based approach to represent what is authorised, is to adopt a rule-based security policy (definition in [ISO89]). A great deal of work has been done to establish such rules for the so-called "military" policy for handling classified information (which is also required for diplomatic and other governmental purposes). The Bell and LaPadula model [BELL76] proposes a set of rules intended to capture that policy.

The Trusted Computer System Evaluation Criteria [DOD85] (TCSEC, or, informally, the "Orange Book") sets out to "provide a basis for the evaluation of effectiveness of security controls built into automatic data processing system products." It is specifically aimed at systems that are to process classified information, particularly multilevel secure systems which can handle a range of sensitivities simultaneously according to a rule-based policy.

The TCSEC has since been interpreted for networks [NCSC87] but the main emphasis there is on what it calls a "single trusted system view". In this view, the network, and the systems connected to it, are treated as a single system with "a common level of trust throughout", and "accredited as a single entity". This approach may be satisfactory for closed networks, but it still does not address security in an open federation of systems.

2.1.2.4 Commercial policies

The TCSEC explicitly states that it places "particular emphasis on preventing the disclosure of data". Clark and Wilson [CLARK87] point out that commercial and industrial requirements tend towards integrity as a more important issue. They show that "military" policy, including the model proposed by Bell and LaPadula, is not well matched to commercial requirements and therefore cannot be accepted as a general model of security. Clark and Wilson also discuss the limitations, for commercial purposes, of attempts [BIBA77], [LIPNER82] to apply a Bell and LaPadula style of policy to integrity.

2.1.2.5 Networked and distributed systems

Attention in the area of security for networks of computers has been directed towards the use of encipherment for communications security. A substantial survey is given by Voydock & Kent [VOYDOCK83].

There has also been a lot of work on how to perform authentication in computer networks. The Kerberos authentication system [STEINER88] is based upon the third party authentication server model proposed by Needham and Schroeder [NEEDHAM78], with timestamps as proposed by Denning and Sacco [DENNING81]. A number of limitations and weaknesses of Kerberos have been identified [BELLOVIN90]. Some of the problems are corrected in revised versions of Kerberos, but those considering Kerberos need to be aware of the problems since the earlier version has been widely disseminated.

One of the weaknesses of Kerberos is its susceptibility to replay attacks (where messages may be intercepted, stored, and retransmitted at a later time). The prevention of such attacks had been studied in detail in an earlier design for integrating an authentication protocol into a remote procedure call system [BIRRELL85].

Authentication protocols which do not involve time have been proposed, both as extensions to the Needham and Schroeder protocol [NEEDHAM87], and by adopting a different pattern of interaction [BAUER83], [OTWAY87]. These still use a third party which acts both as an authentication service and a key distribution service.

The problems of designing an authentication service for a very large scale, very long lifetime distributed system have also been explored in some detail [BIRRELL86]. This has been integrated with work on reasoning about authentication [BURROWS88] to produce a theory of authentication that can cope with roles and delegation as well as with hierarchical authority structures in large scale distributed systems [LAMPSON91]. That work also separates the issues of authentication from those of key exchange.

Many of the ideas discussed by Lampson et al. were adopted from the Digital Distributed System Security Architecture (DSSA) [GASSER89]. Many issues discussed in this document are described there, the significant difference being the approach to naming. DSSA security is integrated with a global naming scheme, but federation requires context relative naming [AR.003].

The potential benefits of a distributed system from the point of view of security have also been studied. In particular, work has been done on the exploitation of separation in distributed systems to achieve isolation and enforcement of security policies [RUSHBY83].

Several research groups have experimented with the use of capability-based protection for access control. In Amoeba [MULLENDER86], for example, services are accessed via protected port objects. Access rights can be delegated by including the names of ports in messages and an irreversible encipherment function is used to encode port names so that only authorized processes can access a port.

2.1.2.6 Proxies and delegation

The issue of incomplete trust has been explored in the context of something acting on behalf of something else [SOLLINS88]. The similar issues that arise where the activity initiated by a user passes from one machine to another across a network have also been studied [KARGER86].

These issues are usually described under the headings of proxies and delegation. They are of particular importance in distributed systems as can be seen from more recent work on this topic [GASSER90] [VARADHARAJAN91].

2.2 Security policy

Standard ECMA-138 [ECMA89] defines a security policy as:

- A set of rules which define and constrain the types of security-relevant activities of entities

This definition of security policy has also been adopted in the current draft of the ODP Descriptive Model [ISO92a] where 'policy' in general is defined as "A set of prescriptive requirements imposed upon the behaviour of an object in order to meet certain objectives". In the ODP Prescriptive Model [ISO92b],the definition of policy from the Descriptive model is adopted but policy is also identified as being "a desired future state of affairs i.e. a goal or objective …". These different definitions show that the concepts and terminology need to be clarified.

The current draft of the ISO "Security Frameworks for Open Systems - Overview" [ISO91c] includes a similar definition of security policy as a set of rules, but then goes on to say that "The policy defines what is meant by security within the domain …."

While it is necessary to have a set of rules to follow in an operational system, and the set of rules can be called the security policy, it is important to distinguish between the rules and the concept which the rules are intended to capture. Failure to do so means that there is no scope for determining whether or not the appropriate rules have been chosen.

The Trusted Computer System Evaluation Criteria gives the Bell and LaPadula model as an example of a Formal Security Policy Model. McLean [MCLEAN87] points out that as a formal statement of a particular security policy, the Bell and LaPadula model is faulty. McLean compares the basis of the Bell and LaPadula approach to predicate calculus; the Bell and LaPadula "model" has rules (which they call properties) which are similar in nature to the axioms and inference rules of a formal deduction system. There is, however, nothing corresponding to the idea of a semantically valid formula and so there is no means by which the formal system can be shown to be sound. Soundness is an essential property of any formal deduction system. McLean demonstrates that the Bell and LaPadula rules do not capture the notion of security even in the context in which it was developed. The problem is that the rules are not sound with respect to the implied model of security; i.e. by following the rules from some initial secure state it is possible to reach a state which would not be considered secure by any usual definition of security.

Whatever names may be chosen for these concepts, it is essential that the "desired future state of affairs" be defined independently of whatever rules or constraints are proposed to achieve that goal. This makes it possible to demonstrate, to whatever degree of rigor is deemed appropriate, that the rules or constraints achieve that goal.

2.3 Security properties and objectives

There are a number of concepts, and associated terminology, which are commonly used when discussing security objectives. These are described below.

2.3.1 Integrity

There are many definitions of integrity or data integrity in the literature already referenced. Some are obviously inadequate, such as the one in the TCSEC [DOD85] which refers to “source documents”. That definition cannot be applied to “computerized data” for which there is no such source document. The Trusted Network Interpretation [NCSC87] attempts to remedy this deficiency by adding, as an alternative:

- The property that data has not been exposed to accidental or malicious alteration or destruction.

The OSI Security Architecture includes the definition:

data integrity: The property that data has not been altered or destroyed in an unauthorized manner [ISO89]

This definition is also adopted by Standard ECMA-138 [ECMA89] rather than the earlier ECMA definition:

integrity

A security property of an object that prevents or is used to prevent:
- its condition of existence being changed and/or
- its contents being changed.

This property is relative to some subject population and to some degree of security. [ECMA88]

Clark and Wilson [CLARK87] use the term ‘integrity’ in the sense used with respect to databases. The references to “well formed transactions”, and the double-entry bookkeeping example, illustrate the requirement to avoid “incorrect combinations of database values” [BHARGAVA87].

The OSI Security Architecture definition can be interpreted to cover both “database integrity” and the control of who may alter the data; unfortunately, it may be interpreted as being limited to only one of these concepts. Both need to be considered when establishing security requirements and so a concept of integrity that includes both these aspects is required. Given the proliferation of automated control systems, it is also appropriate to include, under this general heading, protection of the ability to make the system operate physical devices.

For ANSA, integrity is interpreted as: the aspect of a security policy which is concerned with establishing which entities may influence the behaviour of the system and what kind of influence they may have.

Since the past cannot be changed, integrity is the aspect of a security policy which is concerned with protecting the evolution of the system.

This definition of integrity covers both unauthorized modification of data and denial of service, since both can influence the future of the system. Integrity is often divided into these two different aspects which correspond to the ‘safety’[1]

1. ‘Safety’ here means conforming to a specification and is not directly related to the avoidance of death or injury to people or destruction of property.

and 'liveness' properties associated with reasoning about programs [LAMPORT77] [ALPERN87]. This division can be useful since the mechanisms required to achieve these properties are different, as is the reasoning needed to establish that they have been achieved.

2.3.2 Confidentiality

The OSI security architecture includes a widely accepted definition of confidentiality:

confidentiality: The property that information is not made available or disclosed to unauthorized individuals, entities, or processes.

The definition, in terms of information rather than data, means that data which reveals no information may be revealed. This permits the use of cryptography. It also means that the inference problem must be considered.

The implications of the definition are subtle: information may be information about the past, or information that allows predictions to be made about the future. Predictions about the future are made by knowing how the system will respond to a variety of circumstances and knowing which of those circumstances has occurred. The way in which the system will respond depends upon how it was constructed and, if the construction is considered as part of the history of the system, then confidentiality can be viewed as the aspect of a security policy concerned with establishing which parts of the history of the system may be revealed to which entities.

In general the exact history of a system is not recorded, only a state which could have been reached through any of a set of histories. Confidentiality is therefore protection of information about the state of the system, both the part which corresponds to 'application state' and the state related to how the application is supported. This state is recorded both as the states of objects and by the existence of those objects and both of these need to be protected.

2.3.3 Availability

Availability [ISO89] is the property threatened by denial of service [ECMA88]. The definitions and explanations in the referenced documents are satisfactory and no additional clarification or interpretation is necessary.

2.3.4 Non-repudiation

Repudiation is the false denial of sending or receiving of messages [ECMA88]. Measures to protect against repudiation are described under the heading of non-repudiation [ISO89]. The definitions and explanations in the referenced documents are satisfactory and no additional clarification or interpretation is necessary.

2.3.5 Accountability

Accountability is the ability to report the history of the system, and in particular, which actions were performed by which entities [ISO89]. The issues are also discussed under the heading of audit [ECMA88].

Further discussion of accountability issues can be found in 4.

2.4 Summary

This chapter has introduced some of the terminology and identified the documents that contain definitions of other terms that might be encountered. The emphasis has been on the concepts associated with establishing the objectives rather than those, such as authorisation and authentication, associated with how the objectives are to be met. These other concepts will be introduced as required throughout the remainder of the document.

3 Architectural implications

This chapter presents aspects of ANSA that are relevant to security. It relates the concepts introduced in 2 to ANSA concepts and terminology.

3.1 Architectural principles

An important principle of ANSA is to identify and to avoid those assumptions which are commonly made in centralised systems but which are inappropriate for federated, distributed systems. Only those relevant to security are described here, descriptions of other important assumptions to avoid can be found in other documents describing ANSA [ANSA91a] [ANSA91b].

3.1.1 Explicit communication

In a centralised system, there is a single common infrastructure which must be trusted by all parts of the system since it is in direct control of all processing and memory resources. In a distributed system this is not the case; there are many infrastructures with explicit communication between them.

In a distributed system, the requirements for communication security between trusted infrastructures must be identified and mechanisms installed to satisfy those requirements.

The absence of a single common infrastructure also has benefits. Those parts of a system with less stringent security requirements need not pay the cost of using an infrastructure designed to provide a high degree of security.

3.1.2 Federation

Separate organisations interact by negotiating contracts, and appealing to a mutually acceptable arbitrator, or to some other judicial process, if something goes wrong.

Such organisations may require interaction between the information processing systems that they own. Instead of a single system, there is a collection of systems which may have different and conflicting objectives.

Each organisation will wish to protect its own interests and may be required, either by contracts or by legislation, to protect particular interests of other organisations (including governments) or of individuals. The overall approach to security and the security mechanisms must accommodate these issues.

3.1.3 Failure model

In a distributed system, and especially in a large collection of interconnected systems, it will usually be the case that some but not all of the components are operating normally. Those components that are operating will be able to continue despite observing that other components are not operating. Which components are operating and which are not will be changing continually.

Components will fail and be restarted after repair; obsolete components will be removed permanently and new components will be introduced; components will be shut down and restarted for a variety of operational reasons.

Security must not depend upon all of the system being operational since this situation is unlikely ever to be achieved, and, even if it is, it will be impossible to know that it has been achieved. This rules out any idea of a secure initial state for the whole system. Individual components may be started or restarted in a known secure state, but they will then join the interconnected collection of systems, and must be able to protect themselves within that environment.

A different kind of failure occurs when a party to an agreement fails to honour that agreement. As for other kinds of failure, this is a partial failure and does not imply that all agreements have been broken. This issue is particularly important where the agreement was to preserve security properties such as confidentiality or integrity. The design should adhere to the principle of least privilege [DOD85] in order to limit the damage caused by such failures.

3.2 Enterprise and information issues

ANSA provides five projections for analysing distributed systems [ANSA91b]. For each projection, ANSA provides a model for making statements about systems in that projection. The requirements for security arise from issues that are most clearly visible in the enterprise and information projections.

There is a great deal more work to be done on the enterprise and information projections in ANSA. This section shows how the work that has been done can be used, and also identifies some issues to be studied in further work.

3.2.1 Policy

3.2.1.1 Policy elements

In the ANSA enterprise projection, a policy consists of an objective, missions and constraints [LINDEN91b]. Each part of the policy has a security aspect.

The objective is a desired future state of affairs and so the security aspect of the objective defines what future state is desired for security. This is a definition of what security means to an organisation that adopts this policy.

The security aspect of the missions will define the activities associated with security such as granting or revoking access rights.

The security aspect of the constraints defines the rules that must be followed to preserve security when carrying out the activities defined by the missions. This has been called ‘security policy’ elsewhere as explained in §2.2.

3.2.1.2 Soundness and consistency of policy

The problem identified by McLean, described in §2.2, is a confusion of the security aspect of the objective with the security aspect of the constraints.

Separating the objective, the missions and the constraints makes it possible to question whether or not following the missions and constraints achieves the objectives. It is also possible to consider both the case of the organisation as its owners or managers would like it to be, and the case of how it really is, or at least the best estimate that can be made from the available data.

Many organisations start from a less secure position than they should have and could not afford to shut down operations to attain a secure position. The situation could be improved by a policy that would make the organisation become secure and preserve security once it had been achieved. If the system is self-stabilizing [DIJKSTRA74] with respect to its security policy (that is, if incremental changes naturally converge to a system state that is within the policy), then the requirement to recover after failures will be satisfied.

3.2.1.3 Multiple policies

As was mentioned above, ANSA is concerned with the issues that arise when there is interaction between as well as within organisations; the issues that arise in a federation of systems. This means that it must be possible to accommodate a wide range of security policies and also resolve the problems of interworking between organisations with different security policies.

Each organisation is free to choose its own security policy, although this freedom may be limited by legislation. Within an organisation there may be a number of different security policies, and the extent to which they are similar will depend upon the nature of the organisation. Some organisations operate almost as if they were a collection of separate organisations with little or no central control; in other organisations, the rules may be laid down by a head office with little or no freedom of action for branches or departments.

Federation also affects the fundamental elements that are used to describe the policy. Each organisation may have its own naming system [AR.003] and this will affect how the security relevant entities are named. When interworking between organisations with different naming systems, the federation of the naming systems must be addressed before the policies expressed using those naming systems can be reconciled.

3.2.1.4 Obligations

Although each organisation may choose its own policy, there are some common issues that will arise when making that choice. There are three important concepts which should be distinguished, and for which the ANSA enterprise projection adopts the following terminology:.

- *responsibility* - who is blamed if things go wrong
- *accountability* - who has to explain what has gone wrong
- *liability* - who has to compensate for things that have gone wrong.

These obligations have been described in terms of what happens when problems occur. This makes it easier to see the distinctions between these obligations and to reason about the appropriateness of separating them. These distinctions can be particularly important where the policy provides for delegation. It will usually be necessary to transfer rights when transferring responsibility; this does not necessarily imply that accountability or liability have been transferred as well.

3.2.2 Enterprise model and security concepts

The ANSA Enterprise Model defines three sorts of entities: *agents*, *activities* and *resources*; and six kinds of relations between them. Although the enterprise model is at an early stage of development, these concepts and relations can be used to explain some security issues and identify issues which might otherwise be overlooked.

The access matrix corresponds to the agent-resource relation for some given set of agents (subjects) and resources (objects). Introducing the subjects as objects in the access matrix corresponds to defining the agent-agent relation which is why different rights are commonly used as examples in these parts of the access matrix.

Clark and Wilson observe that it is necessary to ensure that a data item can be manipulated only by a specific set of programs [CLARK87]. This corresponds to defining the activity-resource relation.

The agent-activity relation is concerned with who is allowed to do what. This clearly has security implications. Clark and Wilson identify this issue indirectly in their discussion of separation of duty.

The principle of separation of duty is an attempt to ensure a correspondence between data objects within a system and the real world objects they represent. This correspondence cannot normally be verified directly, but is ensured indirectly by separating all operations into distinct subparts, and requiring that each be executed by a separate person. The Clark-Wilson discussion introduces a requirement to have agents that are a compound entity including more than one person. This security issue identifies a need for the enterprise modelling work.

This leaves two other relations and the question of whether or not there are security issues associated with those relations.

Further work on security in the enterprise model is required to determine how the agent, activity and resource concepts can be related to the concepts such as ‘principal’, ‘role’, ‘delegation’, and ‘conjunction’ used in the authentication logic proposed by Lampson et. al. [LAMPSON91].

3.2.3 Authorisation and authority structures

Many security definitions, such as the definition of confidentiality in the OSI security architecture, depend upon the idea of some things being authorised and others unauthorised. Since the concept of being authorised[1] is of such importance, and each policy in a federation will determine what is authorised in some part of the federation, the concepts of authorisation and authority structures must be explored in more detail.

The necessary concepts have been defined in earlier ANSA work [TR.038]:

Administration: a set of members of an organisation subject to a common source of authority.

Administrator: the role of controlling the extent to which service providers and service users can interact.

Authority[2]: the obligations of an administrator, that when recognized by the members of an administration, allows the administrator to affect the ways in which the members interact.

1. The term ‘being authorised’ is used here in preference to ‘authorisation’ since the latter is more commonly used to refer to the process by which ‘being authorised’ is determined rather than the proposition whose truth is to be established.

2. ‘Authority’ is used here in the sense of power to make a decision. It is used in Standard ECMA-138 [ECMA89] both in this sense (1.5.2.3.22 the definition of ‘Security Administration’) and in the sense of an expert whose opinion is accepted (1.5.2.3.8 the definition of ‘Authority’). Both senses are important distinct concepts for security; unfortunately, there is no obvious alternative term for either.

Hierarchy: a structure of administrations that results when an administration passes on some of its authority to one or more sub-administrations (which may themselves be a hierarchy).

Federation: a structure of administrations in which each administration surrenders little or none of its authority; a federation results when two or more administrations achieve a level of cooperation enabling the sharing of services whilst each administration retains its authority.

This terminology can be used to describe how the rules which are to be mechanised come into being.

An organisation appoints an administrator who chooses a policy. This will involve identifying the relevant entities — *agents*, *activities* and *resources* — thereby defining an administration for the organisation. The administrator can then decide what is permitted and what is prohibited for those entities. The administrator uses his authority to impose a policy on an administration.

The administrator may define sub-administrations and appoint sub-administrators to whom the relevant authority is delegated. The administrator may define how sub-administrations are to interact or they may be left to interact as a federation.

Where administrations have no common superior, there is no administrator to impose a policy upon them. In this case, the only option is to interact as a federation and the administrators must negotiate in order to establish the policy for such interaction. Members of the administrations will then follow the rules imposed by their own administrator for interactions with members of the other administration.

A particular feature of this approach is that it does not depend upon anything more than pairwise agreements. It will often be useful to adopt standardised forms of agreement so as to reduce the complexity of inter-administration interaction but this will be an issue of efficiency.

These concepts of administrators, administrations, hierarchies and federations will be reflected in the computational and engineering parts of the security framework.

3.2.4 Proxies

In addition to the permanent transfer of authority from an administrator to a sub-administrator, it is important to be able to transfer a right for a specific and limited purpose.

Passing the ability to use a service is a particular feature of the ANSA computational and engineering models. The associated security issues of transferring the rights to use a service must therefore be addressed. Although most of the discussion of these issues will use the computational and engineering projections, the authority structure discussed above must have been set up in a way that allows the problems to be solved.

The term ‘proxy’ has been adopted from Standard ECMA-138 [ECMA89] which includes some discussion of the requirements and of how proxy relationships may be supported. Standard ECMA-138 includes two definitions of ‘proxy’. The first defines a proxy as an entity that acts on behalf of another, the sense used in this document. The second defines a proxy as the privileges that allow an entity to act on behalf of another; that sense is not used in this document. Other discussions of the issues, and proposals for mechanisms to support proxy relationships, can be found in the literature referenced in §2.1.2.6.

3.2.5 Trust

Using the enterprise concepts introduced above, trust can be described as the belief that an obligation will be fulfilled.

Trust is something that one entity has about a statement, usually one that involves some other entity. A statement about trust must include not only the entities but also the particular obligation; “A trusts B to do X” and not just “A trusts B”. The latter statement implies “for all X, A trusts B to do X” in which case it is very much better to make this explicit.

Requiring that the obligation be specified makes it clear that trust is not transitive. “A trusts B to do X” and “B trusts C to do Y” says nothing about the relationship between A and C. For example, nothing in the logic allows an inference that A trusts C to do Y, nor even if Y were to be replaced by X, that A trusts C to do X, even though such trust relationships might exist. The question of propagation of trust arises when rights are transferred, and it is there that care must be taken to ensure that the trust relationships are valid.

In a federation, it is unreasonable to assume the existence of a universally trusted entity. Approaches based upon implementing the reference monitor concept, in its original form, assume that the reference validation mechanism or security kernel is a single entity that must be trusted by all others.

In practical terms, trust means that the trusting entity is prepared to act on the assumption that the statement is true. Trust is therefore based upon establishing sufficient confidence rather than absolute certainty. Misplaced trust is one of the failures identified in §3.1.3 — the failure to honour an agreement. The degree of confidence chosen as sufficient will depend on the cost of establishing that confidence and the cost of the failures.

All of the mechanisms described below depend upon trust. Given the nature of federated systems, and the potential for failures, it is important to identify the nature of the trust in each case. Trust relationships must be traceable back to the entities identified in the enterprise model. Although trust will be attributed to mechanisms, both as the trusting and the trusted party, there will be people who are ultimately responsible, accountable, or liable for the actions of those mechanisms.

The definition of trust given above, although expressed in different terms, is consistent with the definition given in ECMA TR/46 and the usage in Standard ECMA-138 [ECMA88][ECMA89].

3.2.6 Interpretation, data and information

The ANSA information model shows that an important operation over data is that of interpretation [LINDEN91c].

The distinction between data and information is vital as without it, it is not possible to justify the use of cryptography to support either confidentiality or integrity.

The OSI security architecture defines confidentiality in terms of information but integrity as data integrity [ISO89]. It is not clear whether or not the same distinction is being made there as is made here.

Given the nature of interpretation, confidentiality is hard to achieve since information is not manipulated directly. Since it is hard to know how an unauthorised entity will interpret data, it is hard to know what data may be revealed without allowing unauthorised entities to obtain information.

Despite this difficulty, it should be recognised that the protection of information rather than data is likely to be a security objective.

Protecting the integrity of information is a more plausible security objective than protecting the integrity of data. If the recipient can obtain the correct information despite alterations to the data, it is likely that the real objective will be satisfied.

Availability is also likely to be improved by arranging that data is transmitted with redundancy so that it can be interpreted to obtain the required information despite some tampering. This would preserve information integrity but not data integrity.

3.2.7 Summary of enterprise and information issues

Security is about distinguishing the authorised from the unauthorised and ensuring that only that which is authorised takes place. This raises some questions that must be answered in the enterprise model:

- who decides what is authorised?
- what is to be protected?
- to whom or to what may rights be granted?
- how may information (data plus its interpretation) be protected?

The enterprise model provides the elements that can be used to answer these questions in particular situations.

The enterprise projection also identifies issues to be considered when choosing the answers appropriate to a particular organisation, and concepts to help in making that choice:

- Policy - objective, missions and constraints
- Obligations - responsibility, accountability and liability
- Authority structures - hierarchy and federation
- Trust

The enterprise and information concepts be used not only to identify security issues in a particular system, but also to reason about security issues in different systems that are to be federated.

3.3 Security in the ANSA object model

This section will focus on how the security issues described above are related to the ANSA object model as it is used in the computational and engineering projections. The enterprise projection of an organisation will define what is authorised. Some of these decisions will be explicit, others will be expressed as general rules and some initial decisions so that the answer for a particular case will need to be calculated.

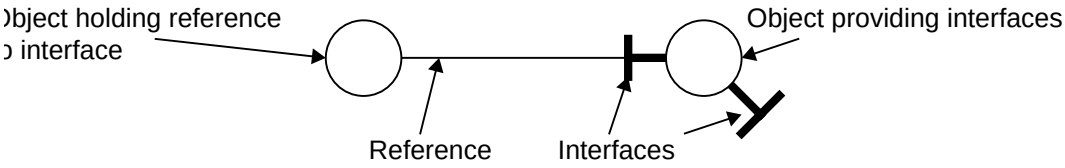
Once the decisions that define the policy have been made, the questions that must be answered in the computational and engineering models are:

- how are the decisions represented for automated enforcement?
- how are the decisions enforced?

3.3.1 Objects - encapsulation and interaction

The essential element in the ANSA object model is that of an object, which consists of a collection of data (state) and a set of operations, defined on that state. The operations are potentially visible to other objects through a number of interfaces. Interaction takes place by means of invocations. An activity in one object invokes an operation in an interface to which it has a reference. This requests that the object providing that interface perform the specified operation with some parameters. When the operation has been performed, the outcome of the evaluation is reported in a response which may also have some parameters. 3.1 is an example of the kind of diagram that will be used to show which objects hold references to which interfaces.

Figure 3.1: Objects and interfaces



A detailed description of the interaction model, including the parameter passing scheme, can be found in the description of the ANSA Computational Model [AR.002].

In the ANSA computational model, objects encapsulate their state. This means that the only way to either observe or to alter the state of an object is to invoke an operation in one of its interfaces.

The ANSA engineering model provides the mechanisms necessary to support the computational model in a heterogeneous, distributed environment.

Adopting this object model means that the problem of supporting security can be divided into two requirements:

1. The encapsulation must be enforced.
2. The interactions at interfaces must be controlled.

The first of these requirements must be addressed entirely within the engineering and technology models. Particular questions that need to be studied are how resources are allocated when an object is created and how they are reclaimed when it is deleted. These are places where the encapsulation mechanism is being put in place or removed; this must be done in such a way that nothing leaks into or out of the object.

The second of these requirements can partially be satisfied by mechanisms defined in terms of the computational model, although engineering model mechanisms are required to support some aspects of parameter passing within the computational model.

3.3.2 Granularity of control

The ANSA object model outlined above must be related to the enterprise concepts of agents, activities and resources. The policy, established in enterprise terms, must then be related to the security concepts of ‘subject’ and ‘object’, and to computational objects, interfaces and activities and their engineering counterparts.

3.3.2.1 Objects

- Objects are the units of encapsulation.

This means that security controls must be relative to objects since there is no separation within an object. Although we shall deal with rights to use part of the service provided by an object, a right cannot be granted to part of an object without it becoming available to all of the object.

- Each object is affiliated to an agent

Any obligation attributed to an object must be an obligation of an agent in the enterprise model. The agent is responsible (perhaps even liable) for the actions of the object. This raises the question of ownership and control of objects. For simplicity of description, the agent to whom an object is affiliated will be called the 'owner' of the object.

The need for compound agents in order to accommodate separation of duty was raised in §3.2.2. The issue of whether it is reasonable for objects to be affiliated to such compound agents is for further study.

3.3.2.2 Units of granting of access

The conventional access matrix uses the term 'object' for the entity that may be accessed, but subdivides that access into a variety of rights. Rights to objects are the units of granting of access in that model. It is usual to assume that the same rights are applicable to a large class of objects and that, although objects may be created and added to the access matrix, the set of rights is predefined.

The ANSA object model provides an alternative approach to defining the unit of granting of access which is not limited by this assumption.

In the ANSA model, the interface is the unit of provision of service. It is also the target of invocation requests and an appropriate place for an access control mechanism.

The principle of least privilege dictates that the interface is also the appropriate unit of granting of access. A potential client should be passed a reference to an interface that contains only those operations necessary for the task to be performed. Granting the right to use the operations in that interface should not require that any other rights be granted. Since objects can have multiple interfaces, there is no difficulty in providing as many interfaces as are necessary for the kinds of access required.

3.3.2.3 Integrity - both kinds

Two distinct kinds of integrity were identified in §2.3.1. The encapsulation of objects and the granting of access on the basis of interfaces allows both kinds of integrity to be addressed.

The 'database integrity' requirement is addressed by the data abstraction (as defined in Wegner's taxonomy [WEGNER87]) provided by the encapsulation of objects and interaction by invoking operations. The state encapsulated by an object can be observed or modified only by an operation in that object. The object designer must ensure that the operations make all the necessary changes to the state. Protection against incomplete evaluations and interference between concurrent evaluations can be provided by use of the ANSA Atomic Operation model [WARNE91].

The proposal above to control the right to invoke operations combines with enforced data abstraction to give 'data integrity'. Only those authorised are able to invoke operations, and invoking an operation is the only way to modify data. This means that only those who are authorised may modify data and they make only those modifications that have been authorised.

3.3.2.4 Activities

Activities in the computational model correspond to instances of the activities described in the enterprise model. Enterprise model activities are performed by agents; computational model activities proceed by evaluating the bodies of operations within objects. Since each object is affiliated to an agent, that agent must be performing the enterprise activities corresponding to the computational activities within that object.[1]

The agent performing an activity often acts as a proxy for some other agent. This happens whenever an activity being performed by one agent invokes an operation provided by an object affiliated to a different agent.

Where there are only two parties involved, the invoked object determines whether or not the agent, on whose behalf the invocation has been made, is authorised to invoke the operation.

Where the evaluation requires the invocation of an operation in an object affiliated to a third agent, the question of the second agent acting on behalf of the first can arise.

An invocation involves a transfer of obligations. How much of which obligations are transferred is a matter for further study.

3.3.2.5 Dynamics

Since the specific rights will change over time, there is an issue of how finely in time these changes occur and take effect.

In a distributed system, the effect of changes will take time to propagate. If the change is at one location it will take time before it can be observed from elsewhere. If the change needs to be made at a location different from that at which the decision to change was made, then there will some interval between the decision and the change being made.

Two particular issues need to be considered when specifying the policy and choosing which mechanisms to adopt.

The first is the ability to stop an activity that is in progress if a right which was required to initiate it is revoked. If the policy requires such a facility then mechanisms to identify such activities and stop them, without violating other aspects of the policy, will be required. Preserving integrity may require the kind of mechanisms that can be found in the atomic operation infrastructure.

The second issue is causal consistency. The policy may require that an activity that depends upon some right must be permitted to proceed if initiated after the right has been granted. Since there is no global clock in a distributed

1. This is a model in which "enterprise reality" is enacted in the computer. An alternative model is that in which the computing equipment is used as a means of communication between agents (in both space and time). The uses of these different models are being considered as part of the continuing work on the enterprise projection.

system, this must mean logically after the granting of the right in the sense of Lamport's logical clocks [LAMPORT78].

3.4 Naming issues

Many aspects of security depend upon the ability to identify things and to distinguish between them on the basis of that identification. In a computer system, names must be used to identify those things that are of interest.

Many of the things that need to be identified are intangible and could not be sensed directly, even if they were located at the place at which the identification is required. Even those things, such as human users, that could be directly sensed, will need to be identified; the point at which the identification is required is often remote from point at which the thing to be identified exists.

Since the security mechanisms must identify and distinguish entities by their names, security depends upon the naming model. The necessary properties the naming system must be identified, and it must be shown that those properties are preserved.

3.4.1 Federation and naming contexts

The ANSA Naming Model [AR.003] recognises that all names are relative to some context. Since identification and the making of distinctions depends upon matching names, it is also dependent on the contexts with respect to which the names are to be used.

The terms 'homonym' and 'alias' are used extensively in the discussion that follows. They are defined in The ANSA Naming Model as:

Alias: Two or more distinct names are aliases if and only if when they are resolved in the same context, they denote the same entity.

Homonym: A name which denotes more than one entity, in a particular naming context.

In a federation, each organisation is free to choose its own naming system. There is no global context available to all organisations in which all entities have names, and which is free from homonyms and aliases.

Each organisation may choose to establish a context in which it uses a standardised naming convention and standardised names for certain entities. This will simplify interaction with other organisations that have done the same. It is important to remember that each organisation has its own context and its own name resolution process. That a particular name denotes the same entity from contexts maintained by different organisations is a carefully constructed artefact, not a logical necessity.

The term 'identifier' is often used for a name that is to be used in a context in which it is neither a homonym nor an alias. It is also commonly used when the name is absolute, at least with respect to those contexts from which it will be resolved. It is important to establish that names actually have these properties if security depends upon them.

3.4.2 Implications for authorisation

The security policy defines what is authorised, and this must be expressed in some way that allows the policy to be enforced.

Any statement of the policy must be in terms of names. In an identity based policy, it is usual to start by assigning names to human users and then go on to define named groups whose members are identified by naming them. In a rule-based policy, names are assigned to sensitivity levels, and to the degrees to which users may be trusted; the rules are then expressed in terms of these names.

When defining the policy, the names will have been generated with respect to some context. When making an access control decision, it is important to know that the names in the statement of the policy denote the same entities as the names presented for a decision. This means that the relationships between the various contexts must be taken into account, and names must be translated as necessary.

To support integrity and confidentiality, it is important that there are no homonyms in the context associated with the statement of the policy. If a name is a homonym and denotes both an authorised entity and an unauthorised entity, and the policy is expressed as “X may …” statements, then access may be incorrectly granted to an unauthorised, but correctly identified, entity whose name is also the name of an authorised entity.

Similar problems arise if distinct resources have the same name, which is therefore a homonym. When that name is used, an entity authorised to use one resource may be granted access to a different resource which it is not authorised to use. This can occur either when a right is granted or when it is checked. The former corresponds to inserting a right into an incorrect entry in an access matrix, the latter corresponds to looking up the rights in the incorrect entry.

Unexpected aliases can lead to denial of service. If an entity has names X and Y and there is some right expressed as “X may …”, then access would be denied if the entity were identified as Y rather than X, both of which are correct.

The examples have been given in terms of an identity-based policy but the same naming issues apply for rule-based policies. The problems are less likely to arise in practice, since rule-based policies typically have fewer distinct named entities, and less need to introduce new names dynamically.

3.4.3 Implications for authentication

Standard ECMA-138 defines authentication as:

The process by which the identity of an entity is established.

Given the ANSA Naming Model, this means finding a name in a particular context which denotes some entity. This implies that there must be some other way to distinguish the particular entity in question, and a way in which a name in a given context can be ascertained to denote that entity.

Authentication is usually studied in the context of verifying a claim of identity. It is also usual to assume that all parties use the same name for the entity in question, and also that the name is neither a homonym nor an alias. Verifying a claim will still be an important case as it is likely to be easier than finding which of a set of names is the right one. The assumption that the same name is used by all parties must be questioned; it is unlikely to be true in a federation, especially if homonyms and aliases are to be avoided.

The immediate requirement for authentication is in support of authorisation. If some access is attempted, the authorisation mechanism must find out the

name by which it knows the invoker in the context in which the policy is specified. Given that there will be various mechanisms that have forwarded the request, the notion of 'the invoker' needs to be clarified. The only identity worth establishing is that of the entity that accepts the obligations of this invocation. Liability is likely to be the most important in this case. Given the possibility that there are proxies involved and obligations have been partially delegated, it may be necessary to establish a number of identities and the relationships between them.

The theory of authentication proposed by Lampson et al. [LAMPSON91] is described in terms of a naming system with a global root but appears to cover all of the other issues of identifying the holders of various obligations. (The role and delegation operators, although not described in those terms, are dealing with divided obligations.) If this theory can be adapted to accommodate the ANSA Naming Model then a basis for reasoning about authentication in federated distributed systems will have been established.

3.4.4 Implications for accountability

Accountability requires that history be recorded. This must be done in terms of names, the names of the entities that did things and the names of what they did.

The names must be recorded relative to some context, and, when the records are examined, this may be done in a different context. If the records are to deliver accountability then the means by which authentication will be achieved for both these stages must be defined.

A particular issue will be where there has been a repudiation. In this case one of the named parties will be attempting to deny that it is the entity named in the record of the incident in question.

3.4.5 Naming of groups

The security implications of groups have not yet been explored. How a group is named and how this relates to names of members needs to be determined. The question of groups with members from different organisations also needs to be considered.

3.5 Transparency

ANSA has placed particular emphasis on being able to use services without having to know the details of how or where they have been implemented. The concept can be applied to security in the same way as it can to many other issues.

When designing and implementing an object, it should not be necessary to know what security policy will be applied to its use, or to the services that it uses. There are various degrees of transparency that can be applied, the object may have security provided completely transparently, or it may call upon security services without needing to known which kind of policy those services are enforcing.

As with other transparencies, security transparency is modelled, if not implemented, as an infrastructure supporting the objects.

3.6 Relationship to other aspects of ANSA

This section describes some ways in which other aspects of ANSA could be combined with security features. The intention is to identify possible benefits of using other features of ANSA to support security and vice versa. These potential benefits could then be considered as goals for future work.

3.6.1 Atomicity

There appears to be scope for using atomicity mechanisms provided by the ANSA atomic operation model [AR.004] to solve some security problems.

3.6.1.1 Optimistic authorisation

It is usually assumed that authorisation must be obtained before proceeding with some operation. Having an atomicity infrastructure available opens up the possibility of performing the authorisation in parallel with the requested operation. Information may be related before commit, but cannot be changed so as to grant permission until commit. Since an atomic operation neither makes any visible changes nor reveals any information, both integrity and confidentiality can be preserved.

An issue to be explored would be how the ACID properties of an atomic activity would be guaranteed in the presence of malicious as well as accidental faults. Security would be more obviously dependent on these properties if this approach were adopted.

3.6.1.2 Support for dynamic (immediate) revocation

A question that sometimes arises in discussions of security is what to do if a right is revoked while some activity is still in progress that was authorised on the basis of that right. The problem is in stopping that activity and undoing what it has already done, should the policy require this.

These are the issues addressed by reversion in the atomic activity model. An extension to the proposal above for optimistic authorisation would allow revocation right up to the point at which the decision to commit is made.

The possibility of leaving a revocation mechanism visible when performing authorisation would need careful study. Being able to invoke an operation that affects an atomic activity from outside that activity may breach the independence property, although a server that has the power to refuse to commit may consult an authoriser before deciding to commit. The question of whether or not it actually violates serializability would need to be studied.

3.6.2 Replication

Some work has been done in ANSA on groups [AR.002] and replication. Fault tolerance is the most widely quoted reason for using replication.

Security is at risk if faults in the security services go undetected. Stopping all further activity upon detecting a fault provides an opportunity for a devastating denial of service attack. This suggests that fault tolerant security services would be beneficial. Groups offer an approach to this problem but would require the converse problem to be solved.

Since the security services must themselves be secure, the problem of making groups secure must be solved if groups are to be used to make security services fault tolerant.

The questions of how group members protect the communication between them, how server groups authenticate and authorise their clients and how client groups are authenticated and authorised all need to be addressed.

3.6.3 Existing ANSA services

Trading is an important concept in ANSA and one which has some security implications. Trading is the way in which previously unknown services are made visible.

The security policy may require that the visibility of offers in the trading service is limited. The contexts provided by the ANSA trading service can be used for this purpose. Each context has a separate interface, and multiple context services also have separate interfaces [DESCHREVEL92]. Access control at interfaces, as described above, can therefore be used to control visibility of contexts.

A client will wish to find a service provided by an object that it is prepared to trust to perform that service, and that will be prepared to perform the service for that client. These may be additional constraints used when importing.

Since trading is on the basis of claimed types and properties, the question of where and when these claims are verified arises. It would be unhelpful to fill a trader with spurious claims so authentication of export parameters may be required. Even if this has been done, a client should be able to perform an independent authentication of the service it is importing if it wishes to do so.

There are many unresolved issues both in the provision of a secure trading service and in the trading of secure services.

3.7 Lifecycle

Systems evolve in a way determined by those who control them. Decisions made at each stage can limit the freedom to make decisions in the future. This section presents a brief overview of some of the stages in the life of a system, and the issues involved in retaining control over the system.

3.7.1 The role of the administrator

When a system is installed an administrator is in sole control. He must have the authority to install other users, and assign to them rights to use services. He may, although he need not, retain authority to remove users. He may retain authority to revoke all rights assigned to a user rather than to remove all knowledge of a user. That knowledge may be required for accountability.

He could delegate authority and retain control, or delegate such as to relinquish control.

He may also delegate rights, including the right to delegate rights, and the rights to add or remove users, so initiating a dynamic environment where administrative authority is propagated.

All of these management services must be controlled by the security infrastructure, in the same way as for access to other services.

3.7.2 The role of the system designer

The facilities that the system provides for the administrator, or any other user, to propagate authority is determined by the system itself. A user can use only

the facilities provided to him. So the system designer, and subsequently the programmer, plays a key role in providing a suitable security infrastructure. A way to represent the role of the system designer is to view the system from the perspective of each service the system designer provides.

A service will be available at system start-up for the systems administrator to add, and possible also delete, users. This service will probably impose a relatively restrictive security policy. It is likely to be “hard wired” into the system to be isolated and self contained, so that transfer of management authority can be carefully controlled. At the opposite extreme, access to a trading service will probably be open to all other services, so that service offers may be deposited and sought. Between these extremes, services may have different motives for propagating authority and applying access restrictions.

3.7.3 System dynamics

A system is generally not “hard wired” in total, but may be dynamic in various ways. Services may be created and deleted dynamically by whoever holds authority in any instance. When a service comes into existence, it will contain its birth rights, which will be delegated by its creator to a degree determined by its system design. It is not necessarily true that, once created, its creator may have access to it, nor that the creator may have authority to delete it. Authority to access may be transferred to some other service on whose behalf the creator is operating.

Dynamic modification of security policy is another form of system dynamics. This notion is recursive, since for this proposition to be true, there must be a security policy that permits dynamic adjustment of security policy. Most services would offer the normal interfaces for general use, but would have interfaces which could allow some modification of the access policy that applies to other service interfaces. Although security policy ultimately derives from initial installation, to a degree permitted by the system design, a finer grain application of security policy may vary on a per service basis, and between classes of interfaces offered by those services.

A further complication comes from the evolution of the system itself. Most systems are not designed once, and remain static forever, but undergo change. This may arise both from fixing inherent errors, or from functional enhancement. Old services may be modified, and new services introduced into systems while they are in some state of dynamic evolution. This implies some perturbation in the provision of security services.

Although all of these forms of dynamics will apply to most commercial systems, there will be need for systems that are static. Such systems may offer high security, or high integrity features, for safety critical reasons. Even so, there will still be a need for these systems to integrate openly with others, albeit with much tighter and “hard wired” access restrictions.

3.7.4 The power to exercise authority

A great deal of attention in security is paid to ensuring that people do not have the power to do things that they are not authorised to do. It is also important to ensure that those who are authorised have the power to exercise that authority.

This is particularly important in the case of an administrator. If an administrator has the authority to revoke a right and a need arises for that right to be revoked then the administrator must have the power to do so. As

was the case for users, an administrator can use only the facilities provided to him.

If the administrator has chosen to install or to operate the security facilities in such a way that he loses the power to exercise his authority, then he cannot regain that power. The means by which a system is controlled is embodied in the design of the system and the facilities to control that system are limited by that design. Once that design has been chosen, so have those limits.

It is the mechanisms within the system that are in control once they have been installed. The administrator may request that things be done but it is the security mechanism that chooses how it will respond. Particular care is required for systems that will be installed in inaccessible locations and controlled remotely; in these cases, physical access cannot be used to regain control once it has been lost.

3.8 Summary

This chapter has used ANSA concepts and terminology to describe security issues. It has shown how security fits into ANSA.

The ANSA principle of avoiding assumptions has been applied to security:

- There is no single common infrastructure.
- Federation implies many policies and conflicting objectives.
- Failures can be observed and must be accommodated.

Concepts from the enterprise and information projections can be used to reason about security. It may answer questions such as:

- who decides what is authorised?
- what is to be protected?
- to whom or to what may rights be granted?
- how may information (data plus its interpretation) be protected?

The enterprise projection also identifies issues to be considered when choosing answers appropriate to a particular organisation, and offers concepts which may help in the choice of:

- Policy - objective, missions and constraints
- Obligations - responsibility, accountability and liability
- Authority structures - hierarchy and federation
- Trust

The ANSA object model supports the separation necessary for security.

- object - units of encapsulation, clear enforcement requirement.
- interfaces - unit of service provision, access control granularity.

The naming model is an essential prerequisite for security. The ANSA naming model provides the concepts and terminology necessary to identify potential problems and design mechanisms that enforce the required properties.

4 Monitoring and accounting

This chapter identifies requirements for monitoring and accounting for security. There are other monitoring requirements for ANSA [HOFFNER92], the relationship of those requirements to security is for further study.

The intention of this chapter is to identify when and in what way the history of the system might be used. This will determine what needs to be observed and, of that, what needs to be recorded. The question of how the observations are made and records are kept will be addressed only to the extent necessary to have confidence that the requirements can be met.

4.1 Attack detection and response

The system may be monitored to detect attacks. Some condition is defined as a trigger which leads to an immediate response.

4.1.1 Detection

An attack can be detected in a number of ways. The general approaches are related to how soon the attack is detected.

4.1.1.1 Incident monitoring

A single incident may be taken as the sign of an attack. This is a typical approach for physical security systems: opening a door connected to the alarm system is sufficient to trigger the alarm.

In the context of the model presented above, a specific invocation would be the trigger. An invocation with an authorisation token that has been reported as compromised would be a typical example.

This kind of detection does not require any history.

4.1.1.2 Threshold monitoring

A combination of events might be used as the trigger. Automated teller machines typically allow three attempts to enter a personal identification number before retaining a card; this is a simple form of threshold monitoring.

In general, any arbitrarily complex combination of events could be specified as the trigger. In the ANSA model, the events would be invocations, or possibly the requests and responses that correspond to the beginnings and endings of invocations. The trigger might be that certain invocations are being attempted concurrently.

This detection is less immediate than the previous one. It suggests that an attack is in progress and that various things have already happened. This kind of detection requires some history; all of the combinations of events that are prefixes of a trigger must be available. The storage and processing needed

to keep the history up to date will limit the complexity of trigger conditions that are of practical value.

4.1.1.3 Damage detection

An even less immediate detection is by observing the damage that has been done by an attack. In this case, the system is observed to have some incorrect value. This implies that there is some independent notion of what is a correct value. This would correspond to discovering that a semantic integrity constraint was wrong in a database. Balancing of accounts is a semantic integrity constraint and this has been the first observed sign of an attack [STOLL88].

Detecting damage requires the concepts and techniques which have been studied extensively in the field of databases if it is to be automated. Where the deviation is between a computer model and reality, detection usually depends upon human observation.

This kind of detection implies that an attack has taken place and has, at least to some extent, been successful. If the attacker intended to be unobserved, then the attack has been only partially successful if it leaves damage that is detected.

4.1.1.4 Log analysis

Another form of detection is through analysis of records kept explicitly for security purposes. This is similar to incident detection or threshold monitoring in terms of the conditions being tested but differs in being performed later.

4.1.2 Immediate response

When an attack has been detected, some action will be taken in response. There are many possible kinds of action so only the general possibilities will be described. Some kinds of response are always appropriate, others are appropriate only when detection takes place while the attack is in progress.

It is always possible to report the attack to a human security officer. It may be appropriate to report some but not all attacks, or to grade the attacks according to some degree of severity. Reports may be of individual attacks or summaries of attacks of a particular kind. The reports may also be made with different degrees of urgency.

The attack may activate an automated defence or recovery mechanism. This is most likely where there is detection while the attack is in progress. The TCSEC requires a defence mechanism for Class B3 and above, and specifies that the 'least disruptive action' be taken.

If no damage has yet been done then the requirement is to prevent the attack from succeeding. For example, if it appears that an attempt is being made to guess passwords then the response may be to change all the passwords so that the information collected by the attacker is no longer valid. In this case, although a threshold monitoring trigger is used, no damage has yet been done.

If damage has been done then it may be possible to repair the damage.

If the damage is to integrity then the issues are similar to those of atomicity. Reversion or compensation can be used to restore correct state, and under the right conditions, these can be automated.

If the damage is to confidentiality then it is likely to be harder to repair. If the revealed information is about something that can be changed then it may be

possible to render it worthless; for example, changing passwords that have been revealed.

4.1.3 Violation of trust

In a federation, it is difficult to determine whether or not other parties that are neither superior nor subservient are fulfilling their obligations. This is a particular problem where the contract between the parties specifies conditions that cannot be monitored by observing the interaction between them.

An example would be where organisations enter into a non-disclosure agreement covering information exchanged between their information processing systems. Neither organisation can tell from the interaction between them, that the other has not revealed information to a third party in breach of the contract.

Integrity, as well as confidentiality, may be at risk from violation of trust in a federation. An organisation that is supposed to store data for another may deliver incorrect data when retrieval is requested. The organisation whose data is at risk must take measures to ensure that it can detect corruption or loss of data.

In a federation, it is likely that problems will be discovered only by the detection of damage. Repair of damage and measures to limit its effect are of particular importance for security in federations.

4.2 Damage limitation

The failure model introduced in §3.1.3 must be recognised in the approach to monitoring and accounting. The monitoring and recording of significant events must continue in the presence of failures. The monitoring and accounting mechanisms also play a part in minimising the consequences for security of the various kinds of failures that may occur.

To isolate a faulty or compromised component from the rest of the system, it is necessary to identify which component is faulty: the fault must first be detected. In general, there may be several components that need to be isolated, either because several are faulty or because the effects of a fault have spread before it was detected. If faults are not contained, the effects may propagate. The monitoring mechanisms and the records that they keep may be used to determine the source of the fault and how far it has spread.

Where several policies are in force, there may be several monitoring mechanisms recording different kinds of events. In this case, it may be necessary for there to be cooperation between the various administrations in order to isolate the fault and perform whatever recovery action is required. It would be unhelpful for a recovery mechanism acting at one level to be mistaken for an attempted attack by another because of the unusual nature of what it is doing. An integrity recovery mechanism might revert to an old state and then repeat some subsequent invocations; this could easily be mistaken for a replay attack.

4.3 Investigation after attack

In addition to any immediate action to correct the state of the system, records kept by the monitoring mechanisms may be used to investigate attacks in order to identify those responsible and to prevent future attacks of that kind.

When an attack has taken place, it may be appropriate to hold an inquiry in order to discover what happened and what, if anything, ought to be done to prevent future attacks of that kind. Records kept by monitoring mechanisms will be important evidence for such an inquiry. It must be possible to determine both the correctness and completeness of the records.

The evidence may show that the rules were followed but this did not satisfy the objectives of the organisation. The evidence may show that there was a failure to enforce the rules correctly. In either case, the problem can be corrected once it has been identified so that no further attacks of that kind will succeed.

If it is intended that an offender be brought to trial, there will be a question of the admissibility of the records as evidence in the appropriate court. If there is any possibility that this use will be required, then the administrators who decide what is to be kept and how it is to be kept must ensure that the requirements of the relevant legislation are satisfied.

4.4 Measurement of success of countermeasures

To be confident that security objectives have been achieved, it is necessary to check that the security mechanisms have been effective.

It is not sufficient to wait for an alarm to be raised, the alarm mechanisms must be tested and shown to be working in order to have confidence that an attack would be detected. It is also important to test the system as it is and not a model of how the system ought to be.

In addition to testing the alarm mechanisms, the normal activity of the system must be checked to ensure that the security mechanisms have been invoked at the proper times and have given the correct decisions. The OSI Security Architecture calls this process a security audit and the data collected for this purpose a security audit trail.

The choice of this terminology shows the similarity of this process to other kinds of audit. There are well established techniques for keeping accounts in such a way that they can be shown to be correct and complete when audited. These techniques can be adapted to the keeping of a security audit trail.

4.5 Summary

Four major requirements for security monitoring and accounting have been described.

- Attack detection for response triggering
- Fault identification for damage limitation
- Evidence for investigation
- Security audit

For all these cases, the nature of the events to be observed, and how those observations are used, must be specified by the security policy. The guards in the infrastructure model can then be designed to gather the required information and pass it to where it is needed.

Some of the implications of federation have been described both in this chapter and in §3.4.4 which considered the impact of the naming model on the keeping of records.

Monitoring and accounting mechanisms are required as part of the enforcement mechanisms, but are also the means by which those responsible for security, and liable for its failure, can convince themselves that the mechanisms that have been put in place to enforce security satisfy the objectives of the organisation.

References

[ALPERN87]

Alpern, B., Schneider, F. B., "Recognizing safety and liveness". *Distributed Computing* 2 117-126 Springer-Verlag, (1987)

[ANDERSON72]

Anderson, J. P., "Computer Security Technology Planning Study". *ESD-TR-73-51* Vols I and II, USAF Electronic Systems Division, Bedford, Mass, (October 1972)

[ANSA91a]

"A System Designer's Introduction to the Architecture". Architecture Projects Management Ltd, (April 1991)

[ANSA91b]

"An Application Programmer's Introduction to the Architecture". Architecture Projects Management Ltd, (November 1991)

[AR.001]

R.T.O. Rees, "The ANSA Computational Model", Architecture Projects Management Ltd., Cambridge (UK), February 1993

[AR.002]

E. Oskiewicz, N.J. Edwards, J.P. Warne, "A Model for Interface Groups", Architecture Projects Management Ltd., Cambridge (UK), February 1993

[AR.003]

R.J. van der Linden, "The ANSA Naming Model", Architecture Projects Management Ltd., Cambridge (UK), February 1993

[AR.004]

J.P. Warne, R.T.O Rees, "The ANSA Atomic Activity Model and Infrastructre", Architecture Projects Management Ltd., Cambridge (UK), February 1993

[BAUER83]

Bauer, R. K., Berson, T. A., Feiertag, R. J., "A key distribution protocol using event markers". *ACM Transactions on Computer Systems*1(3) 249-255 (August 1983)

[BELL76]

Bell, D. E., LaPadula, L. J., "Secure computer system: unified exposition and Multics interpretation". *MITRE MTR-2997*, (March 1976). Available as NTIS AD-A023 588

[BELLOVIN90]

Bellovin, S. M., Merritt, M., "Limitations of the Kerberos Authentication System". *Computer Communication Review* 20(5), 119-132, ACM (October 1990)

[BHARGAVA87]

Bhargava, B. K., Lilien, L., "A Review of Concurrency and Reliability Issues in Distributed Database Systems". in "Concurrency Control and Reliability in Distributed Systems", B. K. Bhargava (ed.), ISBN 0-442-21148-1 Van Nostrand Reinhold (1987)

[BIBA77]

Biba, K. J., "Integrity Considerations for Secure Computer Systems".*Mitre TR-3153*, Mitre Corporation, Bedford MA (April 1977)

[BIRRELL85]

Birrell, A. D., "Secure communication using remote procedure calls". *ACM Transactions on Computer Systems* 3(1) 1-14 (February 1985)

[BIRRELL86]

Birrell, A. D., Lampson, B. W., Needham, R. M., Schroeder, M. D., "A Global Authentication Service without Global Trust". *Proc 1986 IEEE Symposium on Security and Privacy* 223-230 (April 1986)

[BURROWS88]

Burrows, M., Abadi, M., Needham, R.; "Authentication: a practical study in belief and action" *University of Cambridge Computer Laboratory Technical Report 138* (June 1988)

[CLARK87]

Clark, D. D., Wilson, D. R., "A Comparison of Commercial and Military Computer Security Policies". *Proc 1987 IEEE Symposium on Security and Privacy* 184-194 (April 1987)

[DENNING81]

Denning, D. E., Sacco, G. M.; "Timestamps in key distribution protocols". *Communications of the ACM* 24(8) 533-536 (August 1981)

[DENNING83]

Denning, D. E. R.; "Cryptography and Data Security". Addison-Wesley (1983)

[DESCHREVEL92]

Deschrevel, J. P., Watson, A. J., "A brief overview of the ANSA Trading Service". *APM/RC.324.00*, Architecture Projects Management Ltd, (February 1992)

[DIJKSTRA74]

Dijkstra, E. W.; "Self-stabilizing Systems in Spite of Distributed Control". *Communications of the ACM* 17(11) 643-644 (November 1974)

[DOD85]

"Department of Defense Trusted Computer System Evaluation Criteria". *DOD 5200.28-STD* (December 1985)

[ECMA88]

"Security in Open Systems — A Security Framework". *ECMA TR/46*, European Computer Manufacturers Association, 114 Rue du Rhône, 1204 Geneva (July 1988)

[ECMA89]

"Security in Open Systems — Data Elements and Service Definitions". *STANDARD ECMA-138*, European Computer Manufacturers Association, 114 Rue du Rhône, 1204 Geneva (December 1989)

[GASSER89]

Gasser, M., Goldstein, A., Kaufman, C., Lampson, B., "The Digital Distributed System Security Architecture". *Proc. 12th National Computer Security Conference*, 305-319 NIST/NCSC, Baltimore (1989)

[GASSER90]

Gasser, M., McDermott, E., "An Architecture for a practical delegation in a distributed system", *Proc 1990 IEEE Synmposium on Security and Privacy* 20-30 (1990)

[GRAHAM72]

Graham G. S., Denning, P. J., "Protection — Principles and practice". *Proc 1972 Spring Joint Computer Conference*

[HOFFNER92]

Hoffner, Y., "Monitoring in Distributed Systems". *APM/RC.315.00*, Architecture Projects Management Ltd, (February 1992)

[ISO89]

"Information processing systems — Open Systems Interconnection — Basic Reference Model — Part 2: Security Architecture" *ISO 7498/2* (February 1989)

[ISO92a]

"Basic Reference Model of Open Distributed Processing — Part 2: Descriptive Model"; ISO/IEC CD 10742-2 Nov 92

[ISO92b]

"Basic Reference Model of Open Distributed Processing — Part 3: Prescriptive Model"; ISO/IEC CD 10742-3 Nov 92

[ISO91c]

"Working Draft Security Frameworks Overview"; ISO/IEC JTC1/SC21 N5532

[ISO91d]

"Information Technology — Security Frameworks for Open Systems — Part 2: Authentication Framework (DIS 10181-2)"; ISO/IEC JTC1/SC21 N5727

[KARGER86]

Karger, P. A., "Authentication and Discretionary Access Control in Computer Networks"; *Computers & Security* 5:314-324 (1986) North-Holland

[LAMPORT77]

Lamport, L., "Proving the correctness of multiprocess programs". *IEEE Transactions on Software Engineering* SE-3(2) 125-143

[LAMPORT78]

Lamport, L., "Time, clocks, and the ordering of events in a distributed system". *Communications of the ACM* 21(7) 558-565 (July 1978)

[LAMPSON71]

Lampson, B. W., "Protection". *Proc. Fifth Princeton Symposium on Information Sciences and Systems,* Princeton University 437-443 (March 1979) reprinted in *Operating Systems Review* 8(1) 18-24 (January 1974)

[LAMPSON91]

Lampson, B., Abadi, M., Burrows, M., Wobber, E., "Authentication in distributed Systems: Theory and Practice". *Proc. Thirteenth Symposium on Operating Systems Principles,* 165-182 (October 1991)

[LINDEN91b]

van der Linden, R., "ANSA Enterprise Concepts: A Preview". *APM/RC.308.01,* Architecture Projects Management Ltd, (December 1991)

[LINDEN91c]

van der Linden, R., "ANSA Information Model: A Preview". *APM/RC.306.01,* Architecture Projects Management Ltd, (December 1991)

[LIPNER82]

Lipner, S. B., "Non-discretionary Controls for Commercial Applications". *Proc 1982 IEEE Symposium on Security and Privacy* (April 1982)

[MCLEAN87]

McLean, J., "Reasoning about security models". *Proc 1987 IEEE Symposium on Security and Privacy* 123-131 (April 1987)

[MULLENDER86]

Mullender S. J., Tannenbaum, A. S., "The design of a capability based distributed operating system". *The Computer Journal* 29(4),289-299 (August 1983)

[MUFTIC89]

Muftic, S., "Security Mechanisms for Computer Networks". Ellis Horwood (1989)

[NCSC87]

"Trusted Network Interpretation of the Trusted Computer System Evaluation Criteria", *NCSC-TG-005* National Computer Security Center (July 1987)

[NEEDHAM78]

Needham, R. M., Schroeder, M. D., "Using encryption for authentication in large networks of computers". *Communications of the ACM* 21(12),993-999 (December 1978)

[NEEDHAM87]

Needham, R. M., Schroeder, M. D.; Authentication revisited. *ACM Operating Systems Review* 21(1),7 (January 1987)

[OTWAY87]

Otway, D. J., Rees, R. T. O.; "Efficient and timely mutual authentication". *ACM Operating Systems Review* 21(1),8-10 (Jan 1987)

[RUSHBY83]

Rushby, J., Randell, B., "A Distributed Secure System", *IEEE Computer,* 16(7),55-67 (July 1983)

[SOLLINS88]

Sollins, K. R., "Cascaded authentication" *Proceedings of the 1988 symposium on security and privacy* IEEE (1988)

[STEINER88]

Steiner, J. G., Neuman, C., Schiller, J. I., "Kerberos: An Authentication Service for Open Network Systems". *Proc. Winter USENIX Conference* 191-202, Dallas (1988)

[STOLL88]

Stoll, C., "Stalking the wily hacker". *Communications of the ACM* 31(5),484-497 (May 1988)

[TR.038]

D.Iggulden, R.T.O. Rees, R.J. van der Linden, "Architecture and Design Frameworks", Architecture Projects Management Ltd., Cambridge (UK) February 1993

[VARADHARAJAN91]

Varadharajan, V., Allen, P., Black, S., "An Analysis of the Proxy Problem in Distributed Systems", *Proc 1991 IEEE Symposium on Security and Privacy* (1991)

[VOYDOCK83]

Voydock, V. L., Kent, S. T., "Security mechanisms in high-level network protocols". *ACM Computing Surveys*, 15(2),135-17 (June 1983)

[WARNE91]

Warne, J. P., "An ANSA conformant atomic operation infrastructure". *APM/RC.213.06*, Architecture Projects Management Ltd, (1991)

[WEGNER87]

Wegner, P., "Dimensions of Object-Based Language Design". *OOPSLA '87 SIGPlan Notices* 22(12), 168-182 ACM (December 1987)