Access Management in Multi-Administration Networks

S. P. Lord, N.H. Pope, and Susan Stepney

GEC-Marconi Research Centre, Chelmsford,UK.

SUMMARY

Consider the problems of linking together networks controlled by different administrations, and allowing these administrations to maintain autonomy but to use each others services. Consider also what happens if these administrations have different policies on how access should be controlled and security maintained.

We present a model for access control which has been developed to cope with these situations. The model allows administrations to control their own services and users from an "Authority". Services rely on their local authority to control access correctly, and Authorities can exchange information about users and services by the use of a controlled trust mechanism.

1. Introduction

As the need for communication between computers increases, existing systems are being incorporated into networks. These machines often provide good internal security relying on names and passwords for authentication. However, expanding this technique to be used over a network can lead to problems :

- Users will have to manage a multitude of passwords.
- The allocation of rights to users may require updating data on several machines
- The boundary between machines is very visible, for some types of application (e.g. Distributed file stores) this is not a desirable feature

These difficulties are compounded in large complex networks. Consider the problems of linking together networks controlled by different administrations, and allowing these administrations to maintain autonomy but to use each others services. Consider also what happens if the different administrations have different policies on how access should be controlled and security maintained.

These problems are being addressed under project Admiral (described in Pope *et al* (1)) a collaborative project supported by the Alvey programme. An "internet" of local networks at several sites are being interconnected via a high performance wide area network. The project is investigating management services to allow applications to be supported between computer systems across the sites. Under this project there is a need to provide controlled interworking between distributed systems whilst allowing the different partners to implement their own security policies.

In this paper we present a model for access control which is being used as the basis for the design of a system for access control for project Admiral.

2. Background

Our experience with the authentication experiment (Girling (2)) in Project Universe (Burren (3)) has led us to believe that access can be controlled in a distributed manner. The Universe system allowed the generation of secure representations for users and their access rights which could be passed around the network. However, the system did not use encryption, and it was possible to masquerade as an authentication server and steal passwords and representations.

S. P. Lord, N. H. Pope, and Susan Stepney.
Access Management in multi-administration networks.
IEE 2nd International Conference on Secure Communication Systems, London, 1986

The masquerade problem can be overcome by using the approach suggested by Needham and Schroeder (4). This provides a means of authenticating users and services to each other, but does not, in itself, solve the problems of access control.

3. The Model

The model provides a framework for administrators to build Access Control Systems to meet their differing requirements. It provides mechanisms which allow users and services from different administrations to communicate with each other while still allowing the administrators to retain control of their own parts of the network.

A system based on the model would allow users to login to a Distributed Computing System and to make requests for services in any part of the system without having to provide any more information about themselves. After this initial login all subsequent authentication and access control decisions are handled automatically, and remain invisible to the user unless access is refused.

The model does not concern itself with the organisational aspects of user management, that is, with the mechanisms used to give access rights to users, or to create new users and services. However, it does provide a framework whereby this information can be used to control users and services.

We describe the model by detailing the way it views the structure and operation of the network, the mechanisms it provides for access control and how these mechanisms may be used.

The network is divided into Entities which communicate with each other via Requests, these Requests are controlled by Authorities.

3.1 Entities

An Entity is something involved in the provision or use of a service. It could be a person or a software module. Entities are the objects controlled and protected by the Access Control System.

Entities can take on three different roles, these are Client, Server and Principal. When an Entity initiates a request for a service, either explicitly, or implicitly as a result of a previous request, it plays the role of Principal. When an Entity responds to a request it plays the role of Server. A Request is handled by an Entity playing the role of Client which passes it to the server. Principals are responsible for requests, Clients and Servers are the communicating parties involved in Requests. This expansion of the standard Client-Server model allows the Access Control System to base its decisions on Entities other than the Client.

The model treats all communications between Entities as Requests. A Request is initiated from a Client and goes to a Server. After this initial message the Request may include further messages in both directions. Access control decisions are made at the time the Request is initiated, after this, only the security of the Request is maintained.

Servers may respond to a range of Requests. For example, a file-server might respond to “read”, “write”, “delete”; a time-server to “now”, “reset”. Principals have access rights, permissions to make certain Requests of Servers. For example, a particular Principal could have the right to consult a time-server for the current time, but not to reset it.

Entities can enact different roles. For example, a Server might receive a request from a Principal which requires it to make a request of another Server. It could then either act as a Principal in its own right, or as a Client for the original Principal.

Software Entities might take on any of the roles. Human Entities will tend only to be Principals.

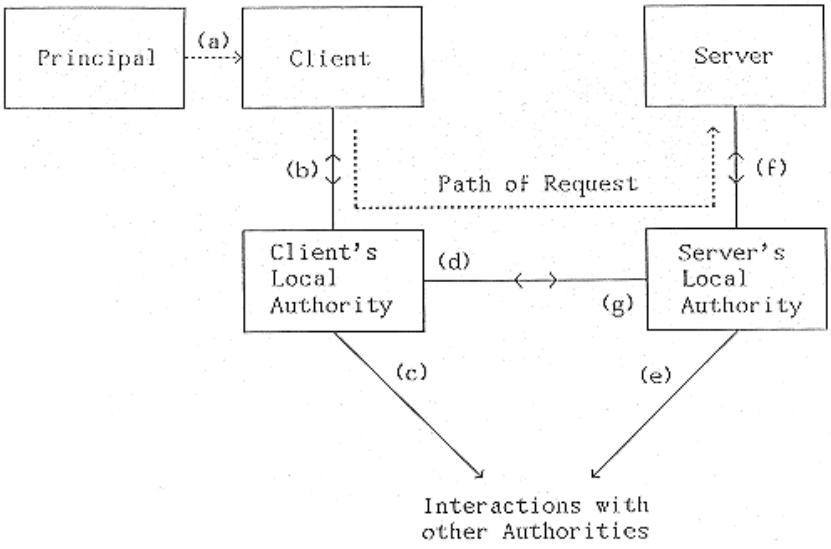

Figure 1. Interactions between Entities and Authorities

3.2 Authorities

The model divides the network up into Administrations, these Administrations manage the network by using Authorities.

Authorities provide the authentication and access control functions for a set of Entities. They store data about Entities' access rights, and can cache data obtained from other Authorities, for consultation during a Session of Requests.

Authorities provide the following functions:

- They allow Servers to check the access rights of Principals.
- They allow Clients to gain access to Servers for Principals.
- They obtain and generate statements about Principals' access rights for other Authorities and for Servers.
- They are used to authenticate Principals.

Each Authority is controlled by a single administration; an administration may control more than one Authority. The administration controls how its Authorities will behave in response to requests from Clients, Servers, other Authorities and external administrations. This behaviour forms part of the administration's security policy.

Each Client and Server has a Local Authority, which it trusts to make appropriate access control decisions. Local Authorities may use management information from other Authorities to help it carry out its function (see Fig. 1).

3.3 Trust

Different Authorities can interact with each other in two ways; by passing out management data on Entities with Requests, or by asking for this management data from other Authorities. These interactions are governed by the Trust mechanism. Each Authority is treated as an autonomous unit, and will only communicate with, or accept communication from, other Authorities which it trusts.

An Authority A Trusts Authority B if :-

a. Authority A accepts data or services provided by Authority B as being trustworthy and,
b. Authority A is able to authenticate Authority B as the source of the information it receives.

Similarly, B will only provide services and data to Authorities it Trusts.

For two Authorities to interwork, both their managers must set them up to Trust the other, thus maintaining the autonomy of the different administrations.

Trust could be used for finer grain control than this by specifying certain categories of data and services for which an Authority is trusted.

3.4 Control Mechanisms

The model provides a number of mechanisms whereby Authorities can control requests. The different mechanisms provide different types of control, and may be used in isolation, or in combination to provide the security required.

There are three mechanisms which can be categorised as giving control based upon WHO - Access Conditions and Rights, WHERE -Talk To and HOW - Quality of Service.

3.4.1 Talk To

Talk To allows access to a Server to be restricted to a set of named Clients. It also allows a Client to be restricted to using a set of named Servers. The Client's Local Authority will only generate messages for Servers on its list, and the Server's Local Authority will only pass on requests from acceptable Clients.

For example, a networked printer might be set up so that it would only accept requests to print files from particular hosts, similarly, a host might be restricted to using a subset of the available printers.

3.4.2 Access Conditions and Rights

A lot of access control is based on the user, or Principal who originated the Request rather than the Client making it. The model incorporates this by stipulating Access Conditions on servers, and giving Access Rights to Principals. In order to gain access to a Server, the Principal must prove to the Servers Local Authority that he meets the conditions. In fact, once the Principal has been authenticated to an Authority, the proving is carried out by Authorities without involving the Principal.

Access conditions can take two forms, access control lists and capabilities. The access control list details the names of Principals who can, or cannot, use the service. All requests to such a Server should include proof of the Principal's identity. Where the access condition is the possession of a capability, then the Principal's access rights must include this capability and this fact must be proved to the Server's Local Authority.

Access Conditions and Rights are controlled by the various administrators. Access lists will generally be close to the Server while capabilities can be passed around the network to allow delegation of management.

3.4.3 Quality of Service

The Quality of Service of a Request determines which security measures will be used to protect it (e.g. Data Encryption or Message Authentication Codes). Thus, the Quality of Service required to access a Server can be used to force prospective Clients to use specific security measures. The actual mapping used is not specified by the model, it will depend on the requirements of the network being protected.

Principals will authenticate themselves to an Authority at a particular Quality of Service. This will then represent the highest Quality of Service they can use for the duration of their connection to the Distributed Computing System. This Quality of Service could be determined from:

- Preset Tables,
- The method of authentication used for the Principal,
- The location of the Principal and
- The Quality of Service requested by the Principal.

Servers can be set up so that they will only accept requests within a particular band of qualities of service. Similarly Clients may be restricted to a band of qualities of service. For a request to succeed, the Principal must be allowed to generate requests within the range of qualities of service common to the Client and Server.

The Quality of Service mechanism can also be used to divide the network into groups of Entities. By restricting different sets of Entities to different bands of Qualities of Service the network can be partitioned into Domains. Communication between the Domains of such a partitioned network can be controlled by careful choice of the Quality of Service bands used. This might be used, for example, to create a multilevel secure environment.

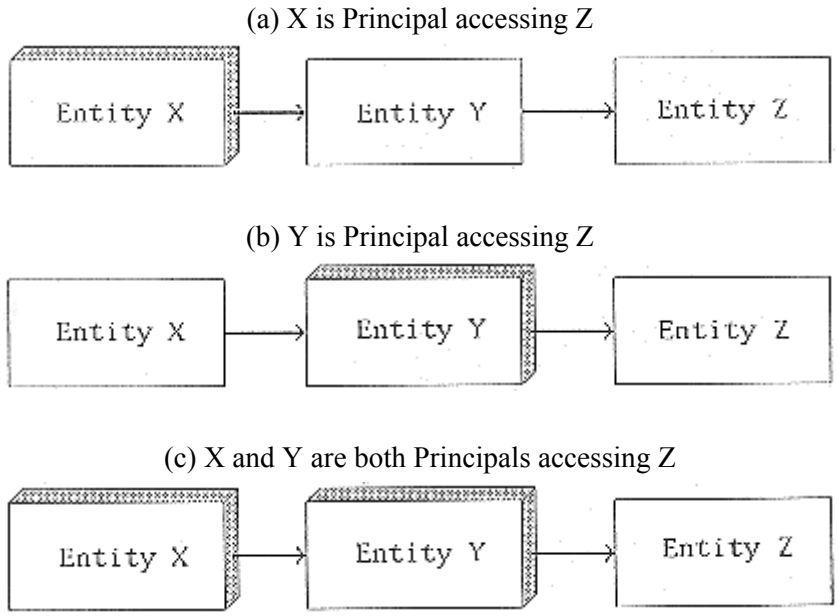

Figure 2. Multiple Principles

3.5 Multiple Principals

A request made to a Server can have more than one Principal; whether the request is honoured can depend on the access rights of more than one Entity. When several Principals are involved in a request, it must be possible to distinguish between them. For example, it might be necessary to know which Principal was initially responsible for the request, for accounting purposes.

Consider figure 2, where X has made a request to Y which requires Y to make a further request to Z. In the second request the Client is Y and the Server is Z, but the Principal could be X, Y, or X and Y.

If the sequence of requests gets longer, more combinations of Principals become possible. But in practice, access to a Server will tend to require Statements about one or two Principals rather than several.

3.6 Operation

This section describes how Entities communicate via Authorities, and how the access control mechanisms are used.

3.6.1 Associations and Sessions

Before two parties can communicate, there must be an Association between them. An Association implies that a securer communication route has been set up at a particular Quality of Service. The existence of an association also implies that the Talk To function has been satisfied. A single association may be used for many different requests between a Client and Server, even requests involving different Principals.

As well as Associations between Clients and Servers to carry Requests, there are Associations between Authorities for the exchange of management data.

A Entity's Session is the region of the Distributed Computing System from which requests can be made with that Entity acting as Principal. A Session is started when the Principal is first authenticated, and continues until the Principal can no longer use the System (for example, logout for a person, termination for a software module). The Entities contained in the Session can vary. It expands to include those Entities whose Local Authorities have been satisfied with the Principal's access rights. It contracts as cached data is discarded or removed.

All the Entities within a session will be satisfied with the identity of the Principal, so that Access Conditions may be checked and Access Rights used.

Setting up and maintenance of Sessions and Associations is carried out by Authorities.

3.6.2 Making a Request

If a Server is a member of a Principal's Session, the Principal can make a request to the Server. This request might still be refused, if the Principal does not have the right to make that particular request, or it is made at an inappropriate quality of service.

Figure 1 shows the communication paths between the Entities and the Authorities involved in the access control. Some of these paths are used when setting up Sessions and collecting Access Rights, others are used for the Requests.

The Principal is responsible for the request (a in Fig. 1), this path implies responsibility rather than data flow; the Client makes the request on the Principal's behalf (b). The request goes via the Client's Local Authority. The Client's Local Authority holds cached Statements about the relevant Principal and Server. These were obtained during the setting up of the Session, both from its own store and, optionally, from other trusted Authorities (c).

The request is passed on (d) to the Server's Local Authority. The Server's Local Authority checks the access right, using its cached Statements. These too were obtained during the setting up of the Session, both from the Client's Local Authority's cache, and, optionally, from its own store and from other Trusted Authorities (e). If the access conditions have been fulfilled, the request is passed on to the Server for processing (f). Further exchange of data may occur (g).

In the model, the setting up of an Association, the adding of a server to a Principal's Session, and the making of a request, are treated as logically separate issues. In practice, the first request to a Server could trigger the other two activities, and all could occur in parallel.

4. The Formal Model and Implementations

Originally the model existed as an informal English description. This was used to develop a formal version of the model in the Z specification language. Z is described in Sufrin (5), and a detailed description of the formal model in Stepney and Lord (6).

Writing a version of the model in Z was a useful exercise. It forced us to think about the problem carefully, and this resulted in several changes in the model, some of them significant. The concepts of Talk To and Qualities of Service were introduced during the development of the Z version. Trust, although introduced in the original informal model, only became fully described in the Z version. Various consequences of the choices made in the model can be deduced using the Z formalism, and certain desirable properties can be proved to hold. The model's structure has improved, making it more understandable.

Once the model had been formally specified the Z was used to produce a Prolog animation of the model. Because both Z and Prolog are based on predicate logic this was a straightforward task; in most cases there is one line of Prolog for each line of Z.

The implementation will be built around a package capable of supporting multiple Authorities. One of these packages will run on each computer, and will support the Local Authorities for Entities on that computer. Each package will have interfaces to communicate with Entities, other packages and a management program. This management interface will itself be a controlled Service allowing the construction of hierarchies of Authorities.

5. Conclusions

We have described what we believe to be a flexible model which can be applied to many configurations of network. The model allows administrators to control their own part of a network in a manner of their own choice while still allowing communication with other administrations.

The model avoids a single control centre by distributing control between cooperating bodies around the network. This should allow the construction of systems which do not rely on connectivity to such a central point to function. At the same time Authorities can be organised into hierarchies if central control is required.

Distribution of control is by means of the Trust mechanism, something we believe to be an important concept for distributed systems where access control is a requirement.

Auditing is an important part of maintaining the security of a network. Authorities are well placed to collect audit data. Sessions make it possible to link together requests and identify a source external to the network for them. Associations can be used to identify patterns of traffic flow.

Finally, this model is now being implemented, and will be evaluated over a real network.

6. References

1. Pope, N. H., Tolcher, D. J., Wilbur, S. R and Rosner, R. A., 1986, "Project ADMIRAL - Research into Networks and Distributed Systems". Conference Proceedings "Networks 86", Online Publications, Middlesex, England.
2. Girling, C. G., "Authentication in Project Universe", 1982, Proceedings of the 6th ICCC, "Pathways to the Information Society", 395-400, North Holland.
3. Burren, J. W., (Editor), 1985, "Project Universe - Overview of the Project", Universe Report No. 1, SERC Rutherford and Appleton Laboratory, Didcot, Oxon, England.
4. Needham, R. M., and Schroeder, M. D., 1978, "Using Encryption for Authentication in Large Networks of Computers", Communications of the ACM, Vol. 21 No. 12, 993-999.
5. Sufrin, B., Morgan, C. Sorensen, I., and Hayes, I., 1984, "The Z Handbook", Programming Research Group, University of Oxford, Oxford, England.
6. Stepney, S., and Lord, S.P., 1987, "A Formal Model of Access Control", Software Practice and Experience, Vol 17, No. 9, 575-593.