

---

University of Nottingham

Department of Computer Science

SPACE: SPatial Access Control for  
collaborative virtual Environments

*by Adrian Bullock, B.Sc. (Jt. Hons)*

Thesis submitted to the University of Nottingham for the degree of  
Doctor of Philosophy, October 1998.

---

## Acknowledgements

I would like to thank my supervisor, Professor Steve Benford, for his guidance and encouragement throughout the work undertaken in this thesis.

I would also like to thank the members of the Communications Research Group, past and present, for their help and friendship. Especial thanks go to Marcus Roberts for explaining the finer points of C to me.

I would like to thank all those who responded to the informal questionnaire that was circulated at the beginning of this research. The model had its first airing in the Esprit COMIC project, and I thank my colleagues of the time for their feedback and encouragement.

Finally, I would like to thank Janet for providing motivation when it was most needed.

This thesis is dedicated to my father and mother, for their support, encouragement and love.

# Contents

<b>Acknowledgements .....</b>	<b>ii</b>
<b>Contents .....</b>	<b>iii</b>
<b>List of Figures.....</b>	<b>vii</b>
<b>List of Tables .....</b>	<b>viii</b>
<b>Abstract.....</b>	<b>ix</b>
<b>Chapter One: Introduction.....</b>	<b>1</b>
1.1. An introduction to CVEs .....	2
1.2. What is security? and Why is it important?.....	4
1.3. The change of emphasis.....	6
1.4. Requirements of access control in CVEs.....	7
1.4.1. Comments from access control and security related literature .....	8
1.4.2. Surveying the CSCW field .....	10
1.4.3. The requirements .....	13
1.5. Spatial access control.....	14
1.6. Thesis outline.....	15
<b>Chapter Two: A review of access models and an overview of VR spatial systems .....</b>	<b>18</b>
2.1. Review of security and access models.....	18
2.1.1. Criteria for reviewing security and access models.....	19
2.1.2. The Access Matrix model (Access Control Lists and Capabilities) .....	22
2.1.3. The Bell–LaPadula clearance classification model .....	25
2.1.4. The Shen–Dewan collaborative access model.....	29
2.1.5. The Group–based authorization model (BSCW).....	32
2.1.6. Other Access Models .....	32
2.1.7. Summary.....	33
2.2. Overview of VR spatial systems and approaches .....	40

---

2.2.1. Dive.....	41
2.2.2. NPSNET .....	44
2.2.3. RING.....	45
2.2.4. Spline .....	47
2.2.5. MASSIVE–2.....	48
2.2.6. TeamRooms.....	49
2.2.7. Summary.....	52
2.3. Matching the results to the requirements.....	54
<b>Chapter Three: SPACE – the model.....</b>	<b>56</b>
3.1. The model.....	57
3.1.1. Graph theory definitions .....	58
3.1.2. Boundaries .....	60
3.1.3. The access graph.....	63
3.1.4. Relative and absolute classifications .....	64
3.1.5. Teleportation and action at a distance.....	68
3.2. Understanding and managing access control.....	68
3.2.1. Defining the functionality of a management tool .....	69
3.3. Adjacency matrix techniques.....	70
3.3.1. The constraints applied to the universe.....	73
3.3.2. Constructing the paths .....	74
3.3.3. Storing the classification values .....	77
3.4. An example space.....	77
3.5. Answering the questions.....	80
3.6. Shortcomings of the model and workarounds .....	84
<b>Chapter Four: Implementing SPACE .....</b>	<b>86</b>
4.1. Introduction.....	87
4.2. The access model – the Bell–LaPadula clearance classification model .....	89
4.3. The collaborative environment – Dive 2.2 .....	91
4.4. The security broker .....	95
4.4.1. Extracting access information.....	96
4.4.2. Constructing the adjacency matrix.....	99

4.4.3. Initialising the security broker .....	99
4.4.4. Querying the access graph .....	102
4.4.5. Updating and writing back access information.....	105
4.5. An example application of SPACE .....	106
4.5.1. The environment.....	106
4.5.2. Analysing the space .....	108
4.5.3 The management interfaces .....	109
4.5.4. Answering the questions.....	112
4.6. A filter for MASSIVE-2 .....	116
4.6.1. Introduction.....	116
4.6.2. The filter .....	117
4.6.3. Examining the Panoptican Plaza environment from MASSIVE-2.....	118
<b>Chapter Five: A framework to apply the model to different scenarios/applications.....</b>	<b>119</b>
5.1. The applicability of SPACE to other areas .....	120
5.1.1 Why should the model be applied to other areas? .....	121
5.2. A cookbook for applying SPACE to other spatial systems .....	122
5.2.1. Examining the applications.....	122
5.2.2. Adding access information .....	124
5.2.3. Extracting access information.....	125
5.2.4. Using the management tool .....	126
5.2.5. Summary.....	126
5.3. Applying SPACE to the Spline system – a worked example .....	127
5.3.1. Examining Spline.....	128
5.3.2. Adding the access information .....	129
5.3.3. Extracting the access information.....	130
5.3.4. Using the management tool .....	130
5.4. Applying SPACE to Teamrooms.....	131
5.4.1. Examining Teamrooms.....	131
5.4.2. Adding the access information .....	132
5.4.3. Extracting the access information.....	133
5.4.4. Using the management tool .....	133

---

5.5. Applying SPACE to other systems .....	134
5.5.1. Examining Rodden’s awareness model .....	134
5.5.2. Controlling access.....	137
5.6. Summary.....	140
<b>Chapter Six: Assessment of SPACE .....</b>	<b>141</b>
6.1. Introduction.....	142
6.2. Security evaluation and assessment criteria.....	143
6.2.1. What is security evaluation? .....	143
6.2.2. Assessment criteria .....	145
6.3. Examining SPACE .....	147
6.4. Comparing SPACE with other access models.....	149
6.5. Limitations of the SPACE access model .....	152
6.5.1. The requirement of a spatial metaphor .....	152
6.5.2. The applicability of SPACE .....	153
6.5.3. Support for specific applications .....	154
6.5.4. Sensitive application areas: a lack of formalism .....	154
6.5.5. Vulnerabilities.....	155
<b>Chapter Seven: Further work and conclusions .....</b>	<b>157</b>
7.1. Summary of research goals and contribution made.....	158
7.2. Further work .....	160
7.2.1. Extensions to the access management tool .....	161
7.2.2. Access rights .....	162
7.2.3. Boundaries .....	165
7.2.4. Scalability and distribution .....	166
7.2.5. Real–world spaces .....	166
7.3. Concluding remarks.....	167
<b>References .....</b>	<b>168</b>
<b>Appendix A: Surveying security in CSCW – a questionnaire .....</b>	<b>180</b>

## List of Figures

Figure 1.1:	The definition of security.....	4
Figure 2.1:	The simple security property of the BLP model .....	26
Figure 2.2:	The *-property of the BLP model .....	27
Figure 2.3:	The Dive interface .....	42
Figure 2.4:	The Panoptican Plaza.....	50
Figure 3.1:	A simple graph.....	58
Figure 3.2:	A simple digraph.....	59
Figure 3.3:	A multi-component boundary .....	62
Figure 3.4:	A simple office scenario .....	65
Figure 3.5:	An example of relative classification.....	66
Figure 3.6:	An example environment.....	78
Figure 3.7:	Access graph to adjacency matrix.....	79
Figure 3.8:	The nine powers of the adjacency matrix .....	80
Figure 4.1:	The structure of the implementation.....	88
Figure 4.2:	Extended Dive schema to include gateway classifications.....	92
Figure 4.3:	Simple definition of a gateway within a Dive data file .....	93
Figure 4.4:	Macro gateway definition .....	93
Figure 4.5:	Extended definition of a person to include clearance .....	94
Figure 4.6:	User embodiment file with clearance .....	94
Figure 4.7:	Example CyCo file format for a simple five world space .....	98
Figure 4.8:	The text interface to the security broker .....	102
Figure 4.9:	The daVinci graph of the information in figure 4.7.....	103
Figure 4.10:	A subset of the access graph for a user with a clearance of 1.....	104
Figure 4.11:	The whole access graph for a user with clearance 1 .....	104
Figure 4.12:	An example environment.....	106
Figure 4.13a:	The Corridor .....	107
Figure 4.13b:	Conference Room 2 .....	107
Figure 4.13c:	The Reception.....	107
Figure 4.13d:	Conference Room 1 .....	107
Figure 4.13e:	The Manager's Office.....	107
Figure 4.13f:	Denied Access.....	107
Figure 4.14:	Access graph to adjacency matrix.....	108
Figure 4.15:	The nine powers of the adjacency matrix .....	109
Figure 4.16:	The access graph for the office environment.....	110
Figure 4.17:	Subset of access graph – clearance is 2 .....	110
Figure 4.18:	Access graph showing links above classification 2 as dotted lines.....	111
Figure 4.19:	The simple access graph for the Panoptican Plaza .....	118
Figure 6.1:	An overview of the Trusted Computer System Evaluation Criteria (TCSEC).....	144

## List of Tables

Table 2.1:	Overview of the access models being reviewed .....	22
Table 2.2:	Summary of review of access models.....	39
Table 2.3:	The spatial VR systems under consideration.....	41
Table 3.1:	The constraints for reducing walks and trails to paths in the powers of the adjacency matrix .....	74
Table 4.1:	Region name to alphabetic label mapping.....	108
Table 6.1:	Review of the access models from chapter two.....	150
Table 6.2:	Comparison of the access models.....	150



## Abstract

A vital component of any application or environment is security, and yet this is often one of the lower priorities, losing out to performance and functionality issues, if it is considered at all. This thesis develops a spatial approach to enabling, understanding and managing access control in collaborative virtual environments (CVEs). Access control is governed according to the space within which subjects and objects reside and whether a subject can traverse this space in order to get close to an object.

We begin by introducing the areas of CVEs and security. Chapter two then reviews access models and overviews notable CVEs, highlighting some of the shortcomings of traditional security approaches for CVEs. In chapter three we introduce the SPACE access model. The two main features of this model are a spatial approach to access control using boundaries to partition space, and the use of an access graph (a mathematical representation of a structured virtual space) that allows spatial access controls to be understood and managed.

A simple realisation of the SPACE model using an adaptation of the Bell–LaPadula access model and the Dive 2.2 CVE is presented in chapter four. The implementation of the security broker is described and a worked example given. Chapter five introduces a framework to apply SPACE to other applications and areas. Mappings of SPACE to Spline, Teamrooms and to Rodden’s awareness model are presented. Following this chapter six assesses SPACE, considering the success of the model in meeting the requirements of access control in CVEs, and also considering some of the limitations of the approach of SPACE. Finally chapter seven presents concluding remarks and considers how this work may be taken further.

## Chapter One

### Introduction

This thesis develops a spatial approach to enabling, understanding and managing access control in collaborative virtual environments (CVEs). Access control is governed according to the space within which subjects and objects reside and whether a subject can traverse this space in order to get close to an object. The two main themes running through this thesis are:

- i) a spatial approach to access control using boundaries to partition space. This approach fits in with the primary feature of virtual reality technology, movement through a 3D virtual space (Benedikt 1991, Rheingold 1991), is applicable to spatial 2D collaborative applications and complements other approaches to access control.
- ii) the use of an access graph, a mathematical representation of a structured virtual space, that allows spatial access controls to be understood and managed.

Chapter one is structured as follows:

- Section 1.1. introduces the subject area of collaborative virtual environments, outlining the key concepts and features associated with CVEs.

- Section 1.2. overviews the area of computer security and briefly explains why it is important. This section defines the small subset of the general security field that this thesis examines, namely that of authorisation through access control and the understanding of any access mechanisms employed.
- Section 1.3. moves on to examine the changes that have occurred in the way computer systems are used, from their invention to modern times, and how they have moved away from single users on individual workstations to multi-user scenarios where the context of the operation becomes important. We have groups, roles, individuals and the relationships between all of these to consider (Greif 1986). This section highlights that a new approach is needed to address this modern class of applications.
- Following on from section 1.3., section 1.4. considers the requirements of an access control model for a collaborative virtual environment. We examine two sources of information. First, quotes from a number of papers that consider security related issues spanning many fields are presented and discussed. Secondly excerpts of answers from a questionnaire that the author circulated to researchers in the field of computer supported cooperative work (CSCW) at the beginning of this research asking their opinions on secure collaborative environments are presented and discussed (see appendix A for the full questionnaire text). From these discussions a set of requirements for access control in CVEs are presented.
- Section 1.5. briefly considers why a spatial approach to access control is appropriate.
- Finally Section 1.6. provides an outline of the structure of this thesis.

## **1.1. An introduction to CVEs**

Collaborative virtual environments provide a common “space” for people to meet and interact. They have their roots in the earlier virtual reality research of the 1970s

(Rheingold 1991) and the communities of MUDs and MOOs from the 1980s. Basically a CVE is a shared virtual space, inhabited by multiple participants whose ability to communicate with each other is enabled through the common space they share. When one thinks about CVEs, often the images that tend to come to mind are of graphically complex, realistic 3D virtual reality environments, but it does not have to be like this. Alternate CVEs include 2D MUD/MOO environments such as Active Worlds (1998) and LambdaMOO (1998), or bulletin board and conferencing systems (e.g. USENET (Tanenbaum 1996), SuperCom (Palme 1990)); in fact any application that provides a shared space where multiple participants can interact and communicate can be thought of as a CVE.

The basic component of any CVE is the “E”, i.e. the environment or shared space within which any interaction takes place. This space is called a virtual environment, because it does not exist in the real world, only in the computer. Often this space is structured or organised into a number of regions which segment the whole space. This is done for a number of reasons. Regions allow

- the decomposition of large environments into smaller, more manageable chunks;
- computational load to be spread (you only concern yourself with what is taking place in the current region and its neighbours);
- different parts of the environment to be provided by different processes or services (it is the logical whole that is important and not how it is made up).

People or agents (representations of users with pre-programmed behaviours to perform certain tasks) inhabiting the shared space are another vital component of CVEs. Without participants there can be no collaboration, after all.

Finally, the interaction that takes place between participants inside the shared space is also a vital component of CVEs. This is the activity which the environment has been

created to support, be it asynchronous text messaging at one end or an immersive graphical 3D environment to simulate, say, aircraft maintenance at the other.

The acceptance of CVEs into mainstream research is evident through the number of research papers concerning CVEs presented at a wide range of conferences, and perhaps most tellingly the setting up of an international conference solely on CVEs (Snowdon 1996, Snowdon 1998) which attracts interest from around the world.

Having introduced the concept of a CVE, we now consider what security is and why it is an important component of any environment.

## 1.2. What is security? and Why is it important?

The word “security” is of Greek derivation, coming from the two words “se” which translates to “without” and “cura” which translates to “care”. Security is there to remove everyday worries and concerns, enabling those with responsibility over something to carry on without care, knowing that whatever it is they are responsible for is secure.

Se    Cura  
**Security**  
Without    Care

Figure 1.1: The definition of security

For a complex computing environment, where there are many tensions and potential conflicts, any mechanism that offers to remove worries is welcomed. In the past, secure systems have not been as idealistic as the origin of the word. Security mechanisms have tended to be bloated and overly complex in attempts to produce provably secure systems for a number of different circumstances (Landwehr 1981). This complexity has often led to security mechanisms being misunderstood and misused, with cumbersome security models detracting from the applications they are protecting, rather than complementing them and providing a useful service. Rather than reducing the care needed to be paid to any system, the adoption of these

techniques can introduce increased care. Mechanisms need to be carefully scrutinised, and even then mistakes can be made.

The field of computer security is a very large and diverse one, and there are many different topics which can be addressed. This thesis concentrates on access control (ISO 10181–3 1996), an authorisation technique that guarantees permission has been granted to perform the action or access the object being protected.

“**access control:** The prevention of unauthorized use of a resource, including the prevention of use of a resource in an unauthorized manner.” (ISO 7498–2 1989)

We do not concern ourselves with verifying the identity of the person who is doing the action; this is the job of an authentication service (Lampson 1992, Sandhu 1996). We assume that basic operating system security is in place and if actions need to be audited (NCSC 1985) then the appropriate mechanisms to achieve this are available. Data integrity (Jacob 1991) which safeguards against unauthorised changes to information, and denial of service (Amoroso 1990) where resources are consumed making them unavailable to others are other threats to be safeguarded against. We assume a combination of well known and emerging techniques can be used to perform these tasks.

Although work on collaborative systems, be they CSCW based systems or VR based systems, is reasonably mature, it is still rare to find work being carried out in the area of security in these fields. This is particularly true of CVEs where the author is unaware of any existing work in the field. We need only consider events from the early 1990s to see why security is an important area which needs careful and proper consideration, the following events having prompted a greater awareness of security across computer science in general:

- the activities of ‘phone phreaks’ in the US causing havoc to phone companies (Hafner 1991, Sterling 1993);

- the activities of young German hacker/crackers trying to steal defence information from the Americans (Stoll 1991);
- perhaps the most publicised case being the Internet Worm, created by Robert Morris Jnr in 1988 (Spafford 1989).

These incidents made people realise that security was something to be dealt with seriously, and that security was a real problem<sup>1</sup>. This realisation has had a swifter effect in some areas more than others, but if CVEs are to be a success in the real world then security is an important issue which needs to be addressed.

### **1.3. The change of emphasis**

Traditional security has the approach of preventing access to data, information and resources. One of the aims of a collaborative application is to foster information sharing, and traditional security approaches might seem to be inappropriate for this. We wish to place restrictions on information, resources and applications to limit their availability but these restrictions should not hinder the ability to share this information with the appropriate people, rather they should promote it. One reason for this may be that traditional approaches often have their roots in technologies that were developed the 1970s, when the default scenario for computer use was single–user use of computers and applications. The growth of multi–user and collaborative systems has meant that new approaches and techniques are required. It was not until first Greif (1986) and then Ellis (1991) specified requirements for access control in collaborative environments that the problem began to be better understood.

Collaborative applications offer perhaps the greatest challenge in developing security mechanisms and services as it can be argued that security and sharing are antithetical.

---

<sup>1</sup> Indeed, the weaknesses which the Internet Worm exploited were well known amongst the systems administration community but no–one felt there was any need to address the problems! Security

Perhaps the biggest problem to overcome when trying to provide a measure of security in a collaborative environment is the inherent obtrusiveness of security. This ranges from having to remember possibly many passwords for many different scenarios to having to use only certain resources to perform certain tasks (for example, you might have to move from a networked workstation to a stand-alone workstation to examine sensitive data). Any mechanisms provided must be unobtrusive in their nature otherwise people will not use the application being protected. Aside from this pragmatic problem the next biggest obstacle is that of access control. If we can limit access to information in shared applications then we are close to our goal. In fact we could probably replace the notion of “providing security services to a collaborative environment” with that of “being able to control the accessibility of information within the environment”. With this in mind we now consider what the requirements of access control are in CVEs.

#### **1.4. Requirements of access control in CVEs**

In this section I would like to state what I consider to be requirements for a successful and effective access control mechanism in CVEs. These personal requirements will largely be based on other people’s opinions. These opinions are derived in part from two sources of information, the first a series of quotes from access control and security related literature, and the second an informal survey of researchers in the CSCW field, carried out at the beginning of this research. The resulting requirements will be used to define a set of criteria, used for both reviewing access models (in chapter two) and assessing the success of the SPACE model (in chapter six).

By examining existing approaches to access control from the literature, it is possible to identify the strengths and weaknesses of current systems and applications. This review also identifies security issues that need to be addressed in the future. We supplement this source of information with comments from practitioners in the area of

---

Through Obscurity (STO) – hoping no-one can determine how to exploit the potential flaws – just doesn’t work.



CSCW. These comments were collected through a survey conducted by the author near the beginning of his research, the author being unaware of any similar surveys in this area. From these two separate but complementary sources of information, the requirements presented at the end of this section are derived.

#### **1.4.1. Comments from access control and security related literature**

To begin with we present the views of specialists in the security field. Extracts are given from CSCW and security papers that highlight what are considered basic requirements for security services.

“Our application programming experience has shown that constraints on group interactions are not easily expressed in terms of traditional file system access rights such as ‘read’ and ‘write’ for ‘owner’, ‘group’ and ‘public’. Instead, more sophisticated controls are needed, which take into account factors such as ... the operation being performed ... the role of the user ... the relationship between the user’s identity and object properties whose values are user identifiers ... the contents of the database:” (Greif 1986)

Greif here is stating the first set of requirements for access control in collaborative environments. This is important as it represents the first time someone stated that collaborative applications need a different approach, as traditional approaches are not easy to use in such circumstances.

“Groupware’s requirements can lead to complex access models, a complexity that must be managed, ..., there must be lightweight access control mechanisms that allow end–users to easily specify changes. User interfaces should smoothly mesh the access model with the user’s conceptual model of the system.” (Ellis 1991).

In this paper Ellis lists a host of issues and requirements for CSCW systems, including access control towards the end of the paper. The understanding and management of any access policy is stressed, avoiding overly complex unmanageable solutions.

“A fundamental tenet of security is that a chain is only as strong as its weakest link ... You need to be aware of the weak points of your defence so that you can take

steps to eliminate them, and so that you can carefully monitor those you can't eliminate ... Simplicity is a security strategy ... keeping things simple makes them easier to understand; if you don't understand something, you can't really know whether or not it's secure." (Chapman 1995)

While the subject of this extract is network security and firewalls, the principles expressed are the same for general security models. Being aware of weaknesses and keeping things simple result in an understanding of security issues and (hopefully) a more secure system.

"Users should be able to selectively enable or disable access control without having to learn a new language, running the risk of inadvertently exposing their data to unwanted access, or understanding the intricacies of the resource object model and access control rights." (Edwards 1996).

Access rights should be presented in the language of the user, rather than forcing the user to learn the specifics of the approach to access control. This way there is a greater chance that the user will make use of the controls properly, rather than ignoring the controls or worse still applying them inappropriately. Any security mechanisms should be seen as a natural extension of the application environment, and not something new and divorced from the application it protects.

"Existing systems often take a feature-based approach to access control in which multiple interacting access-control facilities are configured by security administrators to meet their policy objectives. Unfortunately, these access-control features are often poorly documented and their interactions poorly understood" (Sandhu 1996).

You have to be aware of what it is you are trying to protect, often across a wide area with many differing competing objectives to consider. To do this it is necessary to understand both the access model and the effects of any interactions with other policies. The full consequences of any interactions must be known.

“Existing, widely deployed access control tools are based on models devised 15–20 years ago. These are inflexible, do not provide all the functionality needed, and do not scale well to the collection sizes and user populations wanted” (Gladney 1997)

The way computers are used has evolved over the years, and so any access models and access control tools employed must also evolve to keep pace with the changes. Often this is not the case, hence the need for new approaches to access which complement modern applications and provide all the functionality required of them. The Bell–LaPadula model may be a long standing approach to access control, but the approach presented in this thesis demonstrates a flexible approach to access based on a refinement of this model.

#### **1.4.2. Surveying the CSCW field**

A questionnaire<sup>2</sup> was circulated to researchers in CSCW at the beginning of this research to elicit their opinions on security and co–operative work. This was undertaken as a background activity, the aim being that the comments received would inspire the focus and approach for this work. While the comments received were useful and interesting, it should be stressed that these results were only partially responsible for guiding the work undertaken. It should also be noted that the quotes represent a cross section of opinions from a relatively small number of people, few of whom are experts in security–related issues. However, the author is unaware of any similar survey requesting the opinions of end users on security issues. As such the survey represents an important, if not unique information source.

The following sections examine some of the answers received on a variety of subjects, and comment is made on the answers given. Chapter two will present a detailed justification of why some of the issues raised by the questionnaire are important, through a formal review of a number of access models.

---

<sup>2</sup> The text of this questionnaire can found in Appendix A.

**When asked if security was important in CSCW**

“CSCW systems should support users, not hinder them, thus security components must be as invisible as possible.”

This comment follows a discussion on CSCW systems being used by naive users, and the possibilities for group data to be accidentally destroyed. Clearly this is not an appealing or acceptable situation, but the capabilities of the users need to be taken in to consideration. Hence, any security mechanisms should form a natural part of the application, and not be bolted on as an obvious afterthought. This way the group data is protected and the applications are usable by naive users.

**When asked to choose between a secure or an easy to use application (an editor)**

“[easy to use] probably, but this would affect what the system is used for. I also think that the editor needs to be the same for mail, OA [office application], ... I am a non-techie and am fed up with having to learn what is essentially four or five different editors in the office when one should suffice.”

As well as being unobtrusive, any security mechanism should be understandable and easy to use. An ideal solution would provide the same basic techniques applicable across a number of different scenarios, but presented to the end user in a simple consistent manner. Users are not bothered how the underlying mechanism works, they just want a simple, consistent approach applicable across a range of applications.

“A good analogy is the systems analysis one. You cannot computerize an inefficient paper system to get an efficient computerized system. You must first improve the paper system, if only in theory, before you design the computer system.”

One cannot simply bolt security functionality onto a system and expect it to work if there has been no initial design consideration for security. There is a need to think about the design first and then apply it. Although many spatial systems have been developed without security in mind, they still share the spatial structuring with our model, so we have a firm foundation on which to build, common to the application and the security model.

“I would want the less secure, easier to use application”

“I want both, secure and easy to use”

These two comments speak for themselves – ease of use of a security mechanism is of high importance.

### **Should you put secure information in a shared setting?**

“Take shared work in a normal context. It has to be shared somehow and has to be protected. If computers can make this easier then all for the good ”

### **Does security impede information sharing as opposed to CSCW which promotes it?**

“Security is not a means of impeding information sharing, but is rather a way of not allowing unwanted sharing of information, which seems to be perfectly OK w.r.t. CSCW as well. However, it must be an underlying mechanism that is as unobtrusive as possible to the users”

“I think they are only restrictions because current systems are clumsy. Security should be ever-present but rarely (if ever) noticed.”

“It is certainly important to carefully design the system so that you get acceptable security without impeding normal usage.”

“Very often the implementation of these services into a computer system is done in a way that complicates the whole procedure”

“If the protect mechanism is extremely user-friendly it will enhance the collaboration”

Once again, this set of comments suggests that unobtrusive, easy to use security mechanisms are desirable.

### **Miscellaneous comments**

“My personal biases are towards sharing and I would like to do away with security altogether and use societal means to achieve the benefits attributed to security”

The opinions expressed here come from the starting point that security and sharing are antithetical, that is the more security you provide the less you are able to share information and vice versa. Hence, to achieve true sharing of information the need to ditch security and rely on trust and other social mechanisms is advocated. While trust is a recognised tool in security, it is not wise to build a complete solution based solely on this. As intrusive and complex as some mechanisms are, at least they afford some level of protection, whatever the case. The same cannot be said of trust, with one respondent quoting scripture: “All we like sheep have gone astray” (Isaiah:53(6))! We need to provide security services that foster information sharing, are pervasive and most importantly do not detract from the system or application they apply to.

“Of course everyone wants to have security AND flexible sharing at the same time”

This has to be the end goal of any approach to incorporate security services into collaborative applications.

### **1.4.3. The requirements**

From the discussions above, the following four basic requirements for a collaborative access model are now clear:

- The mechanism must be simple.
- The mechanism must be unobtrusive to users – there should be no extra overhead imposed on users to ensure secure over insecure operation. This means that the mechanism must be naturally integrated into the system philosophy and should utilise existing systems features where possible.
- It should be easy to inspect and change access rights.
- Above all, the effects of access controls should be understood, and the consequences of any changes made clear.

## 1.5. Spatial access control

The previous section has determined a set of requirements for access control. Although it may seem to be just common sense to say mechanisms should be simple and understood this has not been the case for many applications to date. There is clearly scope for an access control mechanism that is easily understood and does not impede normal usage of the application. This is the aim of the work described in this thesis.

General access control models typically utilise objects and attribute lists to govern access. These general models are applicable across a wide range of application areas. This thesis does not propose an approach applicable across the whole computing field, rather it concentrates on one particular area, based on the concept of awareness and spatial partitioning. This specialisation permits a more focussed approach to be taken when considering access control, and while not encompassing the whole of computer science it still leaves a large application area for study.

This thesis proposes a spatial approach to access control for CVEs. Such an approach is appropriate because it allows the simple, yet extremely powerful technique of spatial partitioning to be employed to provide access control. Controls are implemented through the positions of objects in the environment and potential awareness between objects and users. A natural part of the environment is exploited, making it possible to hide explicit security mechanisms from end users through the natural spatial makeup of the environment. This integration of access control into the fabric of the environment results in environments that are easy to use and which afford protection where required. Through the use of an appropriate management interface a full understanding of the environment is possible.

It should be noted that when we talk of CVEs we mean the general definition presented in section 1.1. A 3D virtual environment such as Dive is just one example of a CVE. Other examples include 2D spatially motivated CSCW systems such as Piazza, Teamrooms or Orbit. Teamrooms, for example, uses the spatial metaphor of

shared virtual rooms for its environment. Although the presentation is in two dimensions, the underlying structure is no different from three-dimensional graphical virtual environments. This is one of the strengths of the spatial approach to access control. Any systems sharing the same underlying structure can be controlled with the same mechanism, and moreover comparisons can be made between the security requirements and policies of each.

Thus, a spatially motivated access model seems an appropriate mechanism to provide access control within collaborative virtual environments.

## **1.6. Thesis outline**

The remainder of this thesis has the following structure:

Chapter two presents a review of access models and an overview of spatially motivated CVEs. It begins by examining four access models, from the classic Access Matrix model (Lampson 1971) to an authorisation model for the BSCW system (Sikkel 1997). The access models are reviewed according to a set of criteria, and comment is made on their effectiveness, comprehensibility and ease of use. Next, seven spatially motivated CVEs are examined, noting how they make effective use of a spatial approach and considering how the spatial structuring is achieved. Five of these CVEs are 3D virtual environments and two are 2D spatial collaborative applications. Finally, the chapter considers the results of the access review together with the examination of spatial CVEs to comment on the requirements for access control in CVEs from chapter one. This identifies which aspects of the review need to be addressed by the model in chapter three.

Chapter three presents the SPACE access model (**S**patial **A**ccess **C**ontrol for collaborative virtual **E**nvironments). After a brief overview of graph theory terminology, the key concepts that underpin the model, boundaries and access graphs, are introduced. It is the ability to partition environments into regions and then to represent these regions by an access graph which makes this approach particularly interesting. Standard mathematical techniques can be used to interrogate the access



graph and provide the information to prime a security broker. This broker is an access-control management tool and can be used to answer a number of key access-related questions. These are questions that have been specified to provide an understanding of the access requirements of the environment. An example application of the model to a simple environment is given, followed by outlines of the mathematical techniques which allow these questions to be answered. Finally some shortcomings of the model are considered and workarounds suggested.

Chapter four describes the implementation work carried out to realise the SPACE model in the Dive distributed virtual reality system, and the separate management application that allows users to extract access information from the CVE and inspect and manage it. The access model, based on the Bell-LaPadula model (1973), is presented followed by the CVE adopted, Dive2.2. Next the implementation of the security broker is described in detail, this being at the heart of the ability to understand and manage the access rights associated with a space. A 2D graphical interface based on the daVinci graph package (Fröhlich 1995) is introduced. The example from the previous chapter is examined in greater detail and example daVinci graphs given. Finally, a filter allowing MASSIVE-2 (Greenhalgh 1998) worlds to be examined using the security broker is presented. As a brief example the Panoptican Plaza environment is examined using SPACE.

Chapter five considers a framework for applying the SPACE model to other systems and applications. After reasoning why it is appropriate to apply the model to other areas, an example cookbook is presented that lists the step-by-step requirements to apply the model to other applications. An example of the model, applied to a system motivated by a 3D spatial metaphor, Spline (Barrus 1996), is given. A second example applies SPACE to the 2D CSCW system Teamrooms (Roseman 1997). The chapter concludes by considering an application of the model to more general systems motivated by a non-spatial metaphor, through a mapping of SPACE to Rodden's (1996) awareness model for co-operative applications.

Chapter six considers the area of security evaluation and presents an assessment of the SPACE access model. It first considers what we aim to show in our assessment of the model, and then examines the area of formal security evaluation. From this inspection, the requirements from chapter one and the review criteria from chapter two, a number of assessment criteria are identified. These criteria are then used to compare the SPACE model against the models reviewed in chapter two. This gives us an assessment of the model. Some limitations of the SPACE approach are next presented, and we conclude the chapter by examining some vulnerabilities of the model and the infrastructure in which it is used.

Chapter seven summarises the work presented in the thesis, looking at the research goals that have been achieved and the contribution made to the areas of CVEs and access models. Suggestions for areas of further work are made, concerning both the model and its implementation in Dive. Finally some concluding comments are made.

Appendix A lists the source of the questionnaire that was circulated to elicit the comments presented in chapter one.

## Chapter Two

# A review of access models and an overview of spatial systems

In reviewing related work two areas are considered. This first part of this chapter presents a review of access models from security-related literature, followed in the second part by an overview of a number of systems that use spatial partitioning in their approach. We begin by examining a selection of approaches to access control and security in a number of different systems. A list of criteria, which will be used to assess each of the models, is then given. This list reflects the requirements presented in chapter one. Each named system or approach is examined in turn with conclusions given about each, highlighting the basic requirements that any security service needs to provide (e.g. flexibility and understandability) and the shortcomings of traditional approaches for today's CVE applications (e.g. complex models that are hard to comprehend). Following this, overviews of some spatially-based systems are given, comparing and contrasting the approaches adopted for world structuring by various applications and systems. This chapter aims to demonstrate that there is indeed a need for something to address some of the shortcomings of traditional security approaches for CVE applications and exploit the spatial techniques used by some CVE systems, techniques which people find easy to reason about and understand.

Chapter two is structured as follows:

- Section 2.1 reviews a number of security and access models.
- Section 2.2 overviews some spatially motivated CVEs, notably 3D virtual reality (VR) systems and 2D collaborative applications.
- Section 2.3 considers the results from the access model review, together with the insights gained from the overview of spatial systems and compares the results of the two sections to the requirements identified in chapter one in section 1.3.

## **2.1. Review of security and access models**

At the most basic level, access control is concerned with managing the actions that different users can take on objects. One of the easiest and probably best known access control models is the access matrix model (Lampson 1971) which has acted as the basis for many of the access models which have been developed since the early 1970s. Later models such as the clearance classification model (Bell 1973) have been developed to consider the semantics of the information being controlled in more detail. This section examines four access models, presents a summary of overall strengths and weaknesses found in the examinations, and then gives its conclusions. Before we can do this, though, we need to identify a number of criteria which will be used to assess each of the models.

### **2.1.1. Criteria for reviewing security and access models**

In order to undertake a comparative assessment of the access models we need a common set of criteria to judge each model against. The criteria listed below will be used for this purpose. These high level criteria have been chosen to reflect the aims of the thesis, that is to produce an appropriate, simple, easily understood and easily managed approach to access control for CVEs.

- *Simplicity* of access information for users.
- *Complexity* of the access model that must be managed.

- *Understandability* of the model; both users and managers/developers must be aware of the consequences of any actions.
- *Applicability* of the model to the task at hand.
- *Discretionary or Mandatory access control*.
- *Support for collaboration*.

By **simplicity** we mean how easy it is to specify, examine or manipulate access rights. This concerns the day-to-day use of the information represented by the model and presented to the end user. Typically the simpler things are then the more useful and used they become.

**Complexity** concerns the nature of the underlying access model. It does not necessarily follow that a complex model will be difficult to use, but there is certainly more scope for misfortune with overly complex models. Complexity should be minimised, though not at the loss of functionality.

The **understandability** of a model dictates how obvious any consequences of different access rights combinations or changes are. If you can easily get an overview, or “*The Big Picture*” then a model is likely to be easily understood. A complex model with many twists and turns may be less easily understood. For a model to be effectively applied a full understanding of it is required.

We also need to consider the **applicability** of the model to the task at hand. We need to be sure that the model and access rights make sense, and an appropriate level of security with reasonable costs is being employed. It is no use providing a theoretically totally secure environment if it is not possible to put in place the infrastructure to support this (e.g. from cryptography, the one-time pad is a totally secure cryptosystem, but is not used widely due to the extremely high cost associated with distributing keys amongst participants).

We should consider whether the access model uses **Discretionary** or **Mandatory access control**. DAC is an ownership based approach where the access rights are transferable at a user's discretion. MAC stops the transfer of rights by enforcing a clearance–classification approach where the access rights are centrally managed and not user modifiable. DAC is open to Trojan horse attacks and user problems whereas MAC relies on the system to provide integrity.

Finally we need to consider what **support for collaboration** each of the models offers. Many of the models will have no collaboration support, being developed in the formative years of computing, but some of the later models incorporate collaborative techniques, having been designed for a collaborative setting.

There are differing levels of formality in the models we shall look at. Much work has been undertaken on formal security models, the reason for using formal models is that, as Landwehr (1981) says “the system must not only be secure, but must be demonstrably so”. However, this formality does come at a cost, as we shall see. We will also examine less formal models. While they are less formal, these models often offer a more basic approach which in itself can be seen as a potential bonus.

Before we look at each model in turn, it is useful to consider what each of the access models is trying to achieve or attain. This provides us with a context for the comparison against the review criteria and gives a good general overview of the different aspects of access control.

We can see in table 2.1 that although we have a range of access models, concerned with subtly different issues (for example collaborative versus non–collaborative and simple versus complex), they share the underlying *raison d'être*, of controlling access to resources.

Having set the scene, we now look at each of the models in turn in more detail.

Model	Description
Access Matrix model (Lampson 1971)	Access to objects controlled via a 2D matrix, which defines permissible actions.
Bell–LaPadula model (Bell 1973)	Clearance–Classification model developed with a military scenario in mind.
Collaborative access model (Shen 1992)	Collaborative access model based on a generalised editing model of collaboration.
BSCW Authorization model (Sikkel 1997)	General authorization model that emphasises conceptual simplicity.

Table 2.1: Overview of the access models being reviewed

## 2.1.2. The Access Matrix model (Access Control Lists and Capabilities)

### 2.1.2.1. Overview of the Access Matrix model

Lampson (1971) describes the “object system” consisting of an object set  $X$ , a set of domains  $D$ , and an access matrix  $A$ . It is the objects contained in  $X$  that are protected against the many different potential access domains. Lampson explicitly points out that these domains are objects in their own right, and should not be thought of as merely containers for members of  $X$ . As such they themselves can be acted upon by other domains. Each domain acts on a set of objects from  $X$ , and there can be intersections between domains, so domains can share objects so long as all the rules pertaining to the objects are obeyed. The access matrix,  $A$ , is used to represent the access rights of domains to objects. The heart of the access matrix approach can be summed up by the following one–line quote from the Protection paper:

“Element  $A[i,j]$  specifies the access which domain  $i$  has to object  $j$ ”

The elements, typically made up of sets of strings, then specify what types of access are permissible. They are the access rights associated with the model. A set of rules is

introduced to modify the access matrix,  $A$ , which permits domains with varying access rights to perform various actions such as removing, copying or adding rights to and from the elements of  $A$ . In reality, the access matrix  $A$  will often be very sparse if implemented as a straightforward 2D array, so alternative methods are used. By considering each row of the matrix, and thus a subject and set of associated objects and access rights, one is left with a list of *Capabilities*. If the columns are considered – that is which subjects have which type of access to a particular object – one has an *Access Control List*. These are the two most common implementations of the access matrix model, and are often used together.

The examples used in the Protection paper are based on the operation of a circa 1970 multi-user computer system, where resources were scarce and it was important to make sure each user had appropriate access to disk space, memory, CPU time, and so on. When domains are mentioned in the paper they are typically processes, although we can also look on them as users or subjects to use alternative terminology. The Access Matrix model is applicable across a wide range of applications, some examples include many UNIX™ operating systems and Microsoft Windows NT.

#### **2.1.2.2. Assessment of the Access Matrix model**

We consider the access matrix model in the light of the six review criteria listed above, and then provide an overall summary for the model.

*Simplicity*: The access rights are the elements of  $A$ . The example given in Lampson's paper contains easily understood rights such as *read*, *write*, *wakeup* and *call*, as well as specifying ownership constraints. If other uses of the model use similarly explicit access rights then the ability to specify, examine and manipulate these rights is simple. The indices of each element explicitly identify the domain and object involved.

*Complexity*: The access matrix model is not a complex one. A plain 2D matrix identifies all possible actions for each element, it being obvious which rights apply to which objects through the structuring of the access matrix.



*Understandability:* To gain a full understanding of the access model one just needs to consider all the elements associated with each domain, and to view the end results. The overall complexity, and thus the understandability, of the model depends on the access rights which have been specified as the elements of  $A$ . The underlying model is not inherently complex, rather the understandability depends on the specification of access rights in  $A$ .

*Applicability:* The access matrix model is suitable for scenarios where there are easily identifiable collections of objects and easily identifiable collections of domains acting upon those objects. The model itself is not concerned with the complexity of the access rights held within the matrix, hence the model is very flexible and widely applicable. For easier use, the access rights should be explicit statements of what is possible between a given object/domain pair, and to protect the whole environment it must be possible to exhaustively list all possible access rights in the access matrix. This model appears more suited to smaller scale environments where it is easier to enumerate all objects and domains.

*Discretionary or Mandatory access control:* The access matrix model is a discretionary access model where users can alter rights held within the access matrix.

*Support for collaboration:* There is no explicit support for collaboration in the access matrix model.

The access matrix model can be considered the Father of access models, representing the classic mechanism for controlling access in multi-user computer systems, that of Access Control Lists (ACLs) and Capabilities through the use of an Access Matrix. It is possible to exhaustively determine all the access states for any system by considering each element of the Access matrix in turn. The access matrix does not scale very well for large numbers of objects, as the matrix just gets bigger and bigger and generally sparser. However, by simplifying the access matrix into ACLs and Capabilities only the pertinent access information is displayed in a concise form. The definition of access rights is determined according to the scenario the model is being used in. For example, to protect a manuscript while a book is being written one can

imagine rights such as read, create new version, delete text, append text, overwrite text and so on. The access matrix itself specifies who has permissions to amend the rights it holds.

The next model to have a significant influence in the area of access control models was the Bell–LaPadula clearance classification model.

### **2.1.3. The Bell–LaPadula clearance classification model**

#### **2.1.3.1. Overview of the Bell–LaPadula clearance classification model**

The Bell–LaPadula access model (BLP) (Bell 1973) was developed at Mitre Corporation in the early 1970s and has since been adapted to suit many different circumstances. We shall examine the classic BLP access model here, a model based on the Access Matrix described above. If the reader is interested in its many variations then Landwehr (1981) provides a useful starting point. The original specification of the model is very complex, using finite state machines and much systems theory notation to define access states and transitions between them. However, it is possible to express the model informally without this notation, and that is what we will do.

Being based on the Access Matrix model, the BLP model consists of a set of subjects  $S$  and a set of objects  $O$ . In addition there is a collection of security levels that consist of *classifications* which are associated with objects in  $O$  and *clearances* associated with subjects in  $S$ . Finally a collection of access modes defines the different kinds of access permissible. The model uses a finite state machine to determine whether a given state is secure, and also to examine what transitions are possible to ensure that this state remains secure. If all the rules are obeyed then it is impossible to transit from a secure state to an insecure state. Four access modes are defined for the model:

- |           |   |
|-----------|---|
| read-only | subject can read but cannot modify the object.            |
| append    | subject can write but cannot read the object.             |
| execute   | subject can execute, but cannot read or write the object. |

read–write      subject can read and write the object.

Two security properties are defined in the model, the simple security property and the \*-security property (star property). For a state to satisfy the simple security property, every subject with read access to an object in this state must have a clearance greater than or equal to the classification of the object being accessed. The star property is more complex and considers the modification of objects as well as their being read. To satisfy the star property a subject with clearance  $s$  can write to an object with classification  $o$  if and only if the value of  $o$  is greater than or equal to that of  $s$ . That is, a subject is not allowed to write to objects with lower classifications than itself. This is a gross simplification of the star property, which often involves append and execute accesses, but suffices to make the two properties conceptually symmetric and contains the essence of the star property. These two properties can be summarised as the two informal axioms below, and a system is said to be secure if and only if all reachable states are simple secure and \*-secure.

- i) No user may read information classified above his clearance level (“no read up”)
- ii) No user may lower the classification of information (“no write down”)

Amoroso (1994) expresses the above rules very clearly using simple diagrams. We reproduce and annotate Figures 9.5 and 9.6 from page 105 showing the above two axioms in Figures 2.1 and 2.2.

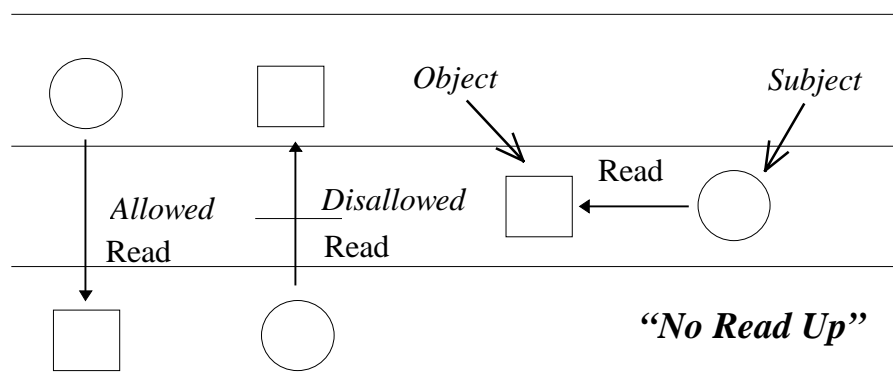


Figure 2.1: The simple security property of the BLP model

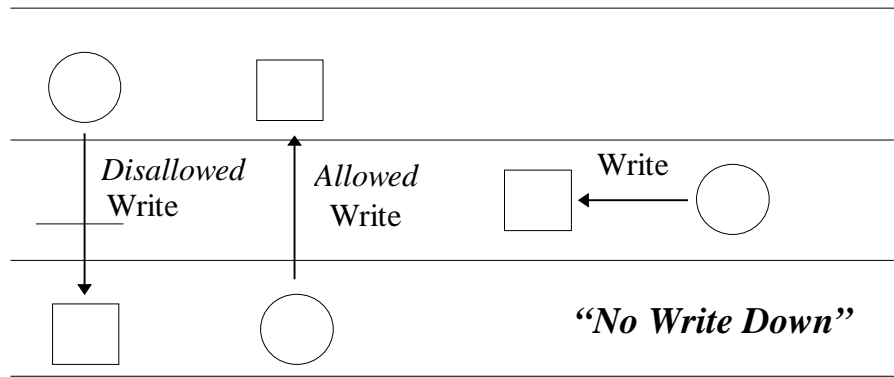


Figure 2.2: The \*-property of the BLP model

### 2.1.3.2. Assessment of the Bell–LaPadula clearance classification model

We consider the Bell–LaPadula clearance classification model in the light of the six review criteria listed above, and then provide an overall summary for the model.

*Simplicity:* The access rights associated with the BLP model are defined by the clearance and classification values which make up the security levels. These are values which are directly comparable, the result being obvious which value is greater than the other, otherwise known as the dominant value. Typically a collection of hierarchical security rights can be mapped onto a set of integers, and it is a very simple task to examine, compare and manipulate such integer values. In these terms the BLP model is a simple one.

*Complexity:* While the model is conceptually simple, the formalisation of the model and fine detail involves much systems theory. If the model is considered purely from a mathematical standpoint then it is very complex indeed. However, we are able to counter this by restating the model in layman’s terms, presenting a simplistic approach to the end user while hiding the systems formality in the implementation. Indeed, in our description of the BLP model above we have avoided a mathematical explanation in favour of simpler pictures. However, to gain the full benefits of the BLP model one needs to understand the theory behind it.

*Understandability:* If we adopt the simple axioms of “No Read Up” and “No Write Down” then it is very easy to understand the model and its effects. We are a user with

a certain clearance and it is obvious which objects we can read and which objects we can write. If the complexities of a full \*-property are introduced then it becomes more difficult to understand the model, as the ability to write to an object becomes dependent upon a number of factors. For simple read accesses, though, the model is clear and easily understood.

*Applicability:* The BLP model is applicable where access rights can be specified in strict hierarchies as classifications associated with objects being accessed and clearances with the subjects or people performing the accesses. Military and Governmental institutions with their strict infrastructures and departmental levels are the ideal setting for this model, but the model is applicable anywhere access rights can be ranked. In practice the model forms the backbone of the American National Computer Security Centre's evaluation system for trusted systems and it is widely deployed in national security. However, it is not applicable to systems where users need to change security levels.

*Discretionary or Mandatory access control:* The BLP model uses Mandatory access control. The access rights are associated with the objects and the subjects must ensure they have sufficient access rights before they can gain access to the objects. Objects are externally managed and users cannot alter the classifications of objects.

*Support for collaboration:* There is no support for collaboration in the BLP model. Indeed, a noted drawback is the model's inability to express requirements that cannot be performed by a single individual (McLean 1990) e.g. policies regarding missile launches in government security. To launch certain missiles (e.g. nuclear warheads) requires the actions of two people to be performed simultaneously, in this case turning a key. If the keys are not turned at the same time then the launch cannot take place. The BLP model was developed with individuals in mind.

The BLP access model is arguably the most influential access model in the field of computer security and has generated much debate. There have been a number of adaptations of this model, most notably the Feiertag et. al. (1977) model that introduced the tranquillity principle to differentiate between active objects that can be

read but not changed and inactive objects that can be modified but not read. The other model extensions have provided easier automated proofs (Feiertag 1977), integrity (Biba 1977) and secure database management systems (Hink 1975). A number of problems with the BLP model have been identified, notably its static presentation of information can be restrictive and the model often disallows many operations which in fact would be secure. If the state transfers do not satisfy the underlying model then the operation will be disallowed, even if it appears to be a secure operation. At the end, two choices are faced when using the BLP model, you either use it as stated and impose severe constraints on system functionality, or else you must rely on trusted processes to allow the desired functionality to be securely achieved.

## **2.1.4. The Shen–Dewan collaborative access model**

### **2.1.4.1. Overview of the Shen–Dewan collaborative access model**

Shen (1992) and Dewan present a collaborative access control model based on a generalised editing model of collaboration that assumes users interact with a collaborative application by concurrently editing its data structures. Once more the Access Matrix model is taken as the starting point, and the model aims to overcome a number of drawbacks identified with the access matrix model in a collaborative setting. Notably, the access matrix model does not support multiple users or roles, it does not specify how access rights are defined and offers no management support. Shen and Dewan suggest four extensions to the model that overcome these limitations. A realisation of their model has been made based on the Suite multi–user framework. Their extensions are:

*Collaboration Rights* – a new set of access rights.

*Negative Rights* – allows the explicit denial of access.

*Inheritance–based Specification* – the access matrix is extended to support the possibility of groups by inferring an element’s value from other elements.

*Automation* – the access model is automatically implemented in Suite.

The Suite framework allows two or more users a common view onto a shared environment. By default there is no access control and the effects of one user can be viewed by another, and in the most extreme case one user is able to delete another's work! Negative access rights can simplify access control greatly. For a group of 100 students, if you wish to deny access to one, instead of positively granting access to the other 99 students you can explicitly deny access to the one student using the negative access right. This permits fine-grained definitions of access rights to be made.

An inference function,  $F$ , is introduced that allows inheritance to be used to simplify access to groups of variables. This function allows rights that have not been explicitly specified to be inferred. The main bulk of the model is introduced as a number of formal rules that finely control access, inheritance of objects, rights and subjects, and conflict resolution of objects, rights and subjects. These rules are expressed using set theory notation and are complex in their makeup, although clear examples of each are given. A novel feature is, an *include* relationship that permits access rights to be stored in logical categories, the grouping being independent of a particular operation on an object is also introduced. For example, the rights associated with a area manager might include specific *hire\_local\_manager* and *fire\_local\_manager* rights, as well as including all the rights associated with a local manager.

To make use of the rules requires application developers to implement them for their particular application, often a tedious task. However, through the use of Suite dialogue managers it is possible to automatically generate implementations of the rules.

#### **2.1.4.2. Assessment of the Shen–Dewan collaborative access model**

We consider the Shen–Dewan collaborative access model in the light of the six review criteria listed above, and then provide an overall summary for the model.

*Simplicity:* The access rights used in this model are relatively easily understood. The basis of the rights are those used in the Access Matrix model. These have been augmented by the notion of negative access rights, a useful and powerful technique, and collaboration rights that can be inherited and shared between users and groups of

users. The only drawback with the specification of access rights is that it is necessary to understand the hierarchies of rights, and the inheritances between them, to fully know what is going on in the model, so a full understanding of the formal set theory notation is required.

*Complexity:* In its full form this model is very complex. However, the trade-off against this complexity is the flexibility afforded by the model. Making the model less complex would undoubtedly make it less flexible. However, there is the option of incrementally learning and using the model, starting off in situations where security is less critical and gaining a firm footing with the techniques employed.

*Understandability:* As stated in the previous two sections, there is a degree of formalism with this model that makes it complex and not the easiest model to understand. At the most basic level the use of normal access rights and negative rights are easy to comprehend. As the flexibility of the model is increased, through added functionality, the complexity is also implicitly increased. To make successful use of this extra functionality it is necessary to understand how it operates, something that requires an individual to learn the complexities of the model. This includes features such as *Multi-dimensional Inheritance* and *Extended Access Lists* from the model. In essence, the understandability of the model depends on what level you wish to use it at, and if you wish to use the full flexibility of the model you have to understand the complex mathematics that goes with the functionality.

*Applicability:* For the environment it has been developed in, (Suite), and for the tasks Suite supports, the access model is ideally suited. Its mechanisms are automatically implemented and flexible working practices between users are supported. For applications areas outside the Suite environment considerable work will be required to re-implement the rules.

*Discretionary or Mandatory access control:* Discretionary access control is used with users able to set the access rights for the editing sessions they are involved in.



*Support for collaboration:* The model offers considerable support for collaboration. It is a collaborative–access model first and foremost.

The Shen–Dewan collaborative access model is both complex and flexible. It proposes a number of useful concepts for controlling access in collaborative environments, including negative access rights, the ability to infer and include access rights between groups and fine grained specification of access rights.

## **2.1.5. The Group–based authorization model (BSCW)**

### **2.1.5.1. Overview of the Group–based authorization model (BSCW)**

The BSCW system (“Basic Support for Cooperative Work”) (Bentley 1995) offers simple cross–platform data sharing, in distributed groups, within the regular working environment through shared workspaces on the World Wide Web. The authorization model discussed here has been motivated by BSCW, but it is intended to be generally applicable to other groupware systems requiring an authorization model. It is based on ACLs, with groups being built up from the structures describing which subjects have which rights on a particular object. Simple set theory notation is used to achieve this. Two explicit groups are identified, *user groups* within an organisational setting also known as roles, and *access groups* which are used to define rights that collections of users might have in a shared environment (e.g. “writing” and “reading” rights when editing a document).

User groups consist of one or more users, and groups cannot be contained within themselves. Thus it is possible to build up hierarchies of user groups and to represent them graphically. A number of operations which act on these group hierarchies are defined: *NewGroup*, *AddSubGroups*, *DeleteSubGroups*, *RemoveGroup*, *DissolveGroup* and *RenameGroup*. Objects acted upon by these groups are split into two categories. *Ordinary objects* act just like all other objects we have encountered so far, such as documents, and require access rights to be defined for them. *Attributed objects* on the other hand cannot be independently accessed, an example being the description of a document held within the BSCW system. The description is an

attribute of the document, totally dependent on it, and access to it is governed by the access rights defined for the document.

Groups of users are split into two categories, regular user groups (*Rgroups*) that consist of regular objects, and attributed user groups (*Agroups*) that consist of attributes of regular objects. For an *Agroup* to exist there must be an object for which it is an attribute. If this object ceases to exist then the *Agroup* also ceases to exist. Access rights of arbitrary semantic value are defined for each regular object in the system. For each right an *Agroup* is defined. Through this definition of access rights it is possible to redraw the traditional access matrix as a graph. Object rights are the graph's sources and the users the graph's sinks. Groups of users sharing common access rights are linked to nodes in the graph, and auxiliary nodes are introduced to simplify the mapping of these groups of users to access rights (an auxiliary node is used to link a group of access rights together). This graphical representation of the access matrix enables a *view* onto an object to be defined as the subset of the rights defined for that object. Views can then be used to simplify and manage access rights through partitioning access rights in the user interface into possibly overlapping subgroups. Views are a consequence of the model, and they permit users to see and interact with access rights in a simple manner, hiding extraneous information from users and only presenting what they are interested in.

This basic model has been extended to support (i) negative access rights through exclusion from group membership, (ii) conditional access rights through the addition of a Boolean statement which is evaluated at the time of access, (iii) explicit roles modelled as user groups requiring additional authorization to be adopted (e.g. the systems administrator role) and (iv) the delegation of access rights from the owner of an object through single-step delegation, recursive delegation and delegation within a trusted group.

The BSCW authorisation model has been developed as a minimal access model, introducing concepts only as and where needed and adopting a modular approach so that only those parts of the model necessary need be used.

#### **2.1.5.2. Assessment of the Group-based authorization model (BSCW)**

We consider the Group-based authorization model (BSCW) in the light of the six review criteria listed above, and then provide an overall summary for the model.

*Simplicity:* This model lists conceptual simplicity as one of its main motivational factors, aiming to keep the model minimal. This has been achieved by considering the process of access control in a step-wise manner. By considering users, groups and the relationships between them as hierarchical graph structures it has been possible to easily identify groups who can subsequently have access rights applied to them. Manipulating who has which rights is simply a case of re-organising the graph using the group functions the model provides.

*Complexity:* The model is moderately complex. It uses standard set theory notation and mathematical graphs to represent user groups and the relationships between them. However, this complexity is managed through the introduction of views, which allow users to examine subsets of the access matrix that are only pertinent to themselves, without regard for the remainder of the matrix. More complex elements of the model are possible, such as negative rights, conditional access and delegation, but these are optional elements. If one wants the full flexibility of the model then the cost is the extra complexity.

*Understandability:* Through the graphical representation of access rights, it is easy to see what rights are associated with which objects, given a reasonably small object space. As the number of objects involved increases it becomes more difficult to obtain an overview of the entire system, but views permit individuals to view particular access rights associated with particular objects very clearly and concisely.

*Applicability:* The model has clearly been developed with a particular use in mind, the BSCW system. However, it deals with users and the relationships between them in a general manner, so the model should be applicable to applications where groups of users negotiate shared access to resources, i.e. most collaborative applications.

*Discretionary or Mandatory access control:* Discretionary access control is used, each object having a Responsible, Control right and Control group. Responsibles have permission to make changes to access rights, add them or delete them.

*Support for collaboration:* The model has been developed to provide access control in a collaborative environment, that of BSCW. Through its use of groups and the possibility of defining arbitrary access rights it offers full support for collaboration.

The BSCW authorization model has investigated a conceptually simple approach to access control for collaborative applications. A modular approach, allowing as much or as little of the model to be used as appropriate, has been presented with legitimate practical requirements for access acting as the driving force.

### **2.1.6. Other access models**

We have examined four approaches to access control in some detail. These are not the only approaches in the literature, rather they represent a good cross-section of the origins and ongoing work in the area of access control models. Here we briefly mention some of the other ongoing work on access control.

Gladney (1997) proposes a Document Access Control Method (DACM) whose aim is to scale to both large user populations and large document collections. Although this model has been developed with library services in mind, it should be applicable across other kinds of data.

Coulouris and Dollimore present case studies on the preparation of an examination paper and a course directory (1994a), and then propose a security model for cooperative work (1994b). Collaborative tasks where information security is critical is the intended application area for this model. A distinction is drawn between user level and programming level access control.

Sandhu overviews a number of Lattice-based access models (1993) and later proposes a Role-based access model (1996b). Roles are used to define specific individuals and also the extent to which resources are accessed. A framework is presented that

separates the administration of role-based access control from its access control functions.

Dewan and Shen (1998) continue their work on access control and have recently developed a general framework for supporting access control in multiuser interfaces. Based on the access matrix model, it has been designed to support flexible control of shared operations and high level access policies. Like their previous model it has been instantiated in the Suite system.

### **2.1.7. Summary**

The above reviews have highlighted the strengths and weaknesses of the various access models. We now summarise the discussion in a per criterion manner, and then present a comparative table summarising the review of the access models against the review criteria.

#### **2.1.7.1. Simplicity**

Many of the access models are not concerned with the semantic definition of access rights. Instead entities known as “access rights” exist and are used to grant permissions to perform the associated actions. The Access Matrix model, which has acted as the basis for most of the models discussed here, is extremely simple to understand, as long as the semantics of the access rights are appropriately specified. You just examine the elements of the matrix  $A$ , or alternatively generate ACLs or Capabilities to provide useful summaries. Modifying access rights is a case of modifying elements of  $A$ . The other models add sophistication and extra functionality, and although they plan to be as simple as possible the extra functionality can reduce their simplicity. On the whole the specification and examination of access rights is adequately covered by all the access models, and difficulties only begin when access rights are manipulated and the consequences of the manipulations are not clear. This issue is covered in greater detail in section 2.1.6.3 on understandability.

### **2.1.7.2. Complexity**

The task of controlling access to resources in a collaborative multi-user setting is inherently complex. The more sophisticated the system then the more scope there is for subtle interactions, all of which must be captured by an effective access model. To provide a fully flexible approach to access control all of the subtleties and nuances of the system need to be modelled. It would appear that there is a trade-off between aspects of complexity and flexibility. The aim of any access model must be to provide maximum support and flexibility with the minimum of complexity. In the models we have looked at, the level of complexity seems to be generally in proportion to the flexibility of the approach. The Shen-Dewan collaborative access model offers tremendous flexibility but at the cost of being a highly complex model. The Bell-LaPadula access model is conceptually simple (the detail of the model involves much complexity, though the underlying model is simple) but inflexible in its applicability across a wide application area (it is only really ideal in its basic form for environments based on the National Security model of classification and clearance levels). Our aim should be to provide an access model which supports the natural interactions that take place in cooperative applications, while providing an easily understood interface to end users so they can easily manage access rights, and a straightforward model that can be easily applied across a number of systems by application developers.

### **2.1.7.3. Understandability**

With understandability we are concerned with the overall clarity of the access model, and the consequences of any actions that are possible. Generally speaking the simpler, less sophisticated, models tend to be more understandable than complex, more sophisticated models. There is typically much functionality in complex access models and extra checks are required to ensure that the result of any manipulation of access rights is secure, and the system has not been compromised. The added functionality can include relationships between objects and users of the system, and it is not a trivial task to determine the global consequences of a small change to the access rights. This makes the model more difficult to understand, unless an intuitive user

interface can be provided to display this information. For an access model to gain wide acceptance this criterion must be met – people want to know how the access model operates, and what the global consequences of any actions will be.

#### **2.1.7.4. Applicability**

The applicability of an access model depends very much on the application area. The Access Matrix model has found wide applicability, being used as the basis for many later models and indeed most of the models discussed here. None of the models present insurmountable problems when using them in the areas of collaborative systems, though the models designed for complicated editing tasks do have a price to pay in their general comprehensibility and hence there are questions raised about their suitability. An incorrectly applied access model can be potentially worse than having no access model and relying solely on social conventions. If there is not too high a price to pay in understanding and managing the access model then it is likely to be applicable to the area it was designed for.

#### **2.1.7.5. Discretionary or Mandatory access control**

Mandatory access control (MAC) gains over Discretionary access control (DAC) by not being vulnerable to Trojan Horse attacks or user mistakes. However MAC policies are more inflexible than DAC policies, as it is not possible for users to change or pass on their own access rights, as rights are set and managed at a system rather than user level. It is difficult to argue which of the two approaches is better – each is suited to its own application area. For example, in medical circumstances MAC may be preferable (e.g. there are confidentiality rules that doctors have to follow. They are not allowed to disclose any information obtained from a consultation. This is organisational rule, and not something the doctor can alter personally). In environments where access rights need to change dynamically a DAC approach may be better, permitting users to modify the rights accordingly (e.g. in a shared environment someone would want to protect a document they are currently working on, but when the document is finished they want to make it available to others. To

revise the document after comments one would again want to make it private during the revision).

#### 2.1.7.6. Support for collaboration

The early models that were discussed were developed for single user/process environments, with atomic actions taking place so decisions were made on an atomic basis. Recent work has seen the appearance of access models tailored specifically for collaborative environments, namely the work of Shen and Dewan and that of Sikkel, discussed above. These models have been developed with particular systems in mind, Suite and BSCW respectively. It is argued that they can be applied to general collaborative applications, but the complexity of the models may well make their implementation a far from trivial task, although this will certainly vary from application to application. There lies the crux of the matter. There is a great variety of functionality across the broad range of collaborative applications, and it is unlikely that one model will be able to provide a complete solution.

In Table 2.2 we summarise the review of the access models against the review criteria. This concludes our examination of access models. In the next section we examine spatially motivated CVEs, examining five such systems.

Criterion \ System	Access Matrix	Bell – LaPadula	Shen – Dewan model	BSCW model
Simplicity	High	Medium	Low	Medium
Complexity	Low	Medium	High	Medium
Understandability	High	Medium	Low	Medium
Applicability	Wide	National Security	Collaborative Editing	BSCW
DAC or MAC	DAC	MAC	DAC	DAC
Collaboration	No	No	Yes	Yes

Table 2.2: Summary of review of access models



## 2.2. Overview of spatial systems and approaches

This thesis proposes a spatial approach to access control for CVEs. It is only correct that we examine the approach of a number of spatially motivated CVEs to see how they partition space so we can learn from their experiences. After we have done this we can use the conclusions drawn from this section together with those from the review of access models to identify shortcomings in approaches to access control that may be overcome by the adoption of a spatial metaphor. Before we can do any of this we need to examine some systems that have adopted a spatial approach, looking in general at how the spatial structuring works and in particular at any security mechanisms or techniques that are employed. This is less of a formal review than the previous section, and rather than list a set of criteria used to compare and contrast the models, an overview of each approach will be given in separate sections. A comparative summary will then be presented, followed by some conclusions drawn from the comparisons.

We will examine six systems that adopt a spatial approach. Five of these systems are 3D-based CVEs. These range temporally from Dive, one of the earliest widely available distributed virtual environments, to the recent MASSIVE-2 system, which is being developed at Nottingham to examine issues such as persistence and concurrency within CVEs. The other system is a 2D-based CVE, which supports collaborative working. All these systems structure their environments according to a spatial metaphor. Table 2.3 introduces and summarises each model.

For each of these approaches we consider amongst other things:

- What is the aim of the system?
- What approach is used for spatial structuring?
- What security mechanisms are built into the system?

System	Summary
Dive (Carlsson 1993)	A general collaborative virtual environment consisting of worlds and portals.
NPSNET (Macedonia 1995)	Hexagonal cells associated with multicast groups make up the environment for large distributed simulations.
Ring (Funkhouser 1996)	Servers scope interaction to the local area based on visual occlusion. A large user base with real time interaction.
Spline (Barrus 1996)	Regions are combined using arbitrary 3D transformations to create a large, detailed virtual environment.
Massive-2 (CVE) (Greenhalgh 1998)	A general collaborative virtual environment consisting of regions and third party objects.
Teamrooms (Roseman 1996)	A groupware environment based on the metaphor of shared virtual rooms.

Table 2.3: The spatial CVE systems under consideration

## 2.2.1. Dive

### 2.2.1.1. Description of Dive

Dive (Carlsson 1993) has been developed at the Swedish Institute of Computer Science and is a general purpose CVE. It has been developed “to create a fully distributed software platform enabling several participants to meet and interact in a shared virtual environment”. Dive is one of the more mature examples of a CVE, being freely available since the early 1990s for non-commercial work, and it is still evolving. Rather than being an application created with an explicit goal, Dive has been developed to allow prototyping and testing of applications within the Dive environment. The idea has been to make the system as flexible as possible, supporting any desired configuration or use. Dive can be used in desktop mode or immersively, with a wide array of available input devices, and runs over a wide range of heterogeneous platforms from PCs to high end graphics supercomputers.

The current version of Dive is built on top of a multicast architecture, each user having a replica of a distributed database held locally, with updates between different databases being propagated via reliable multicast messages. Previously Dive relied upon the Mbone multicast backbone to provide wide area connectivity, but recent work on a multicast bridging protocol called the DiveBone permits distributed participants to connect to each other or the Mbone using this DiveBone. A nameserver running on a specified multicast address provides references that enable users to connect to each other.

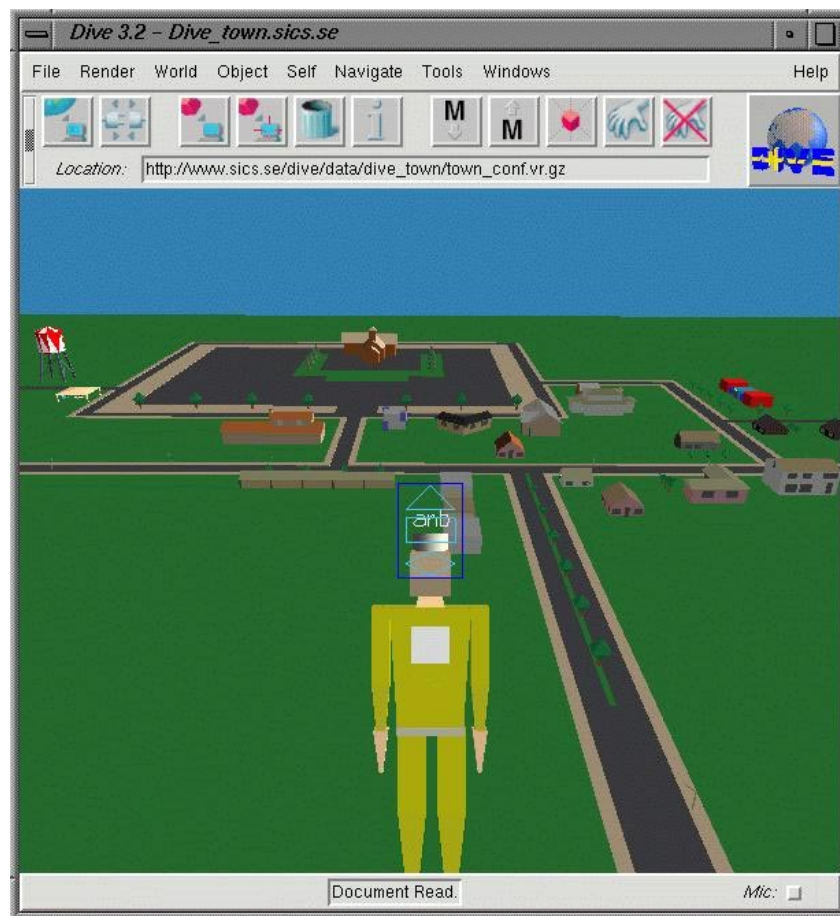


Figure 2.3: The Dive interface

Figure 2.3 shows the visualiser, a rendering application that permits the user to see and interact with the VE. In this view we are looking from behind our own shoulder, so the embodiment we see in the foreground is ourselves. The embodiment shown is not very realistic; it is possible to use highly complex bodies (Pandzic 1997) but these

can have considerable computational and rendering requirements. The user can navigate through the environment in a number of ways. Most simply the cursor keys can be used, a mouse vehicle offers a full 6 degrees of freedom and more rapid movements (the blue rectangle in the centre of the figure), and it is also possible to use trackers to navigate.

#### **2.2.1.2. Spatial structuring in Dive**

Dive is structured as a collection of individual worlds connected via portals. Worlds are text data files which describe geometry and behaviour for the world and objects within the world. They either exist on the local filestore, or are available from Web sites using the http transfer protocol. Portals are connections between worlds. By walking into a portal a user is transferred from the current world into the destination world. Portals are not necessarily two way, while it may be possible to move from world A to world B it does not necessarily follow you can move from world B to world A. There are many advantages to such an approach. Geometry is defined on a world by world basis, so you only have to render the current world and are not concerned with the geometric complexity of the other worlds in the environment. It is also possible to support many groups of users in different worlds, as each world is only concerned with its own state and inhabitants. Finally, the world/portal approach easily lends itself to representing the entire environment as a directed graph, the worlds being the nodes of the graph and the portals between worlds the edges. Later versions of Dive have introduced the ability to teleport between worlds – you just type the destination world into a dialogue box and you are transported there, regardless of whether a route via portals exists between the worlds.

#### **2.2.1.3. Support for security in Dive**

Dive has no explicit support for security in the platform. Any participant in a Dive session is able to go to any other world. Hence, if you are holding a meeting there is nothing to stop another user from connecting to your diveserver (given that they know the multicast address it is running on) and joining your meeting by entering the world. Clearly there are many circumstances where this behaviour is not acceptable.

## **2.2.2. NPSNET**

### **2.2.2.1. Description of NPSNET**

NPSNET (Macedonia 1995) has been developed at the US Naval Postgraduate School to “provide[s] a network software architecture for solving the problem of scaling very large distributed simulations”. A motivating factor behind the work has been to support more than 1000 simultaneous entities in a single virtual environment, there being many industries interested in modelling such large numbers of users (e.g. the military for large scale battlefield simulations and training, and the telecommunications industry to provide support for interactive multi-user games). Previously the Distributed Interactive Simulation (DIS) standard (IEEE 1993) had provided simulations for up to 300 players, but faced fundamental problems when scaling up to thousands of users through bandwidth, computation, replication and efficiency factors. NPSNET offered a solution by partitioning up the environment into workable, smaller environments using an Area of Interest Manager (AOIM).

### **2.2.2.2. Spatial structuring in NPSNET**

Groupings of objects called functional classes are identified, for example an “air control” group, and they have an IP multicast group associated with them. This way only messages relevant for a particular group are forwarded. Temporal classes are defined to control the timing of the updates, with different groups able to handle different update rates. The AOIM is then used to manage this partitioning of the environment into the different groups. It tiles the environment into hexagonal cells and maps collections of cells known as “Areas of Interest” (AOI) onto multicast groups. This reduces the computation load on hosts, minimises network traffic and makes reliability a local issue.

Objects in NPSNET can be members of more than one multicast group, allowing the multicast groups to act as labels (e.g. one group may be a live video stream while another is a live audio feed). As hexagons tessellate efficiently, it is possible to specify the AOI of an object to be the immediate cell (the associated cell) and the surrounding cells, to whatever radius is necessary. As an object transits between cells, then so can

the AOI associated with the object. The object listens to messages across its entire AOI, but only sends traffic to its associated cell. The size and scope of the hexagonal cells used to represent the AOI have been determined using battlefield statistics for movement of units during combat situations to ensure that no more than about 800 objects are communicated with at any one time.

### **2.2.2.3. Support for security in NPSNET**

The author has been unable to locate mention of security or access mechanisms in the literature describing NPSNET.

## **2.2.3. RING**

### **2.2.3.1. Description of RING**

Ring (Funkhouser 1996), developed at AT&T Bell Labs, is a client–server approach to large scale collaborative virtual environments. It “supports real–time visual interaction between a large number of users in a shared 3D virtual environment”. Larger numbers of users are supported through a reduction in the number of network messages generated for each event. In early CVEs point–to–point communication models were used, which scale very badly generating  $O(N^2)$  messages per update, and even later broadcast systems such as NPSNET generate  $O(N)$  messages per update. When a thousand users are being considered, this large amount of network traffic quickly becomes a limiting factor.

In RING the VE consists of a number of entities, each having a geometric description and behaviour. A client workstation manages an entity, executing necessary programs to generate the behaviour of the entity, and also maintaining surrogate entities which are representations of entities managed by other clients. Clients do not communicate directly with each other, rather they use servers that forward messages as necessary. The servers are able to process any messages they receive, culling, augmenting or altering them. They can also tailor messages to particular client performance capabilities. Much of the processing is taken away from the clients and placed at the

servers. This permits users with lower-specification client workstations to take part in sophisticated simulations, so long as powerful enough servers are used.

### **2.2.3.2. Spatial structuring in RING**

RING has been designed for densely occluded environments (e.g. the interior of buildings) where users have visibility cones, and are only concerned with actions occurring inside their cone. If an event takes place outside a particular user's visibility (i.e. they do not see it) then there is no need to make that user aware of the event. This greatly reduces the number of messages that have to be sent between entities simultaneously inhabiting a virtual environment.

The current implementation of RING is best summarised by a quote from Funkhouser (1996):

“RING servers forward update messages in real-time only to other servers and clients managing entities that can possibly “see” the effects of the update. Server-based message culling is implemented using pre-computed line-of-sight visibility information. Prior to the multi-user simulation, the shared virtual environment is partitioned into a spatial subdivision of cells whose boundaries are comprised of the static, axis-aligned polygons of the virtual environment ... servers keep track of which cells contain which entities by exchanging “periodic” update messages when entities cross cell boundaries”.

To use the example of a building, cells within RING can be thought of as the individual rooms and corridors which make up the floor of a building. Line-of-sight algorithms determine which collections of cells are involved in state updates.

### **2.2.3.3. Support for security in RING**

As with NPSNET, the author has been unable to find mention of security in the literature describing RING.

## **2.2.4. Spline**

### **2.2.4.1. Description of Spline**

Spline (Barrus 1996) arose through the developers' (from Mitsubishi Electric Research Labs) desire to make CVEs "detailed in the sense that there are interesting things to see if you look close and large in three senses: in spatial extent, in numbers of objects, and in numbers of users interacting with the environment". This mirrors the real world, an enormous, highly complex environment as a whole but built up from many small localised activities that are dealt with locally by each inhabitant on a daily basis.

### **2.2.4.2. Spatial structuring in Spline**

Adopting the approach of decomposing complex environments into small localised activities, Spline partitions a virtual environment into independent "compact chunks" with their own descriptions and communication possibilities. These chunks are known as "locales", each locale object consisting of the following four fields:

1. A unique ID.
2. A distinct multicast address.
3. A binary space partitioning (BSP) tree (Naylor 1990) describing the boundary of the locale.
4. Neighbours – a list of structures describing the relationship between the current locale and its neighbours.

The spatial partitioning in Spline is realised as binary space partitioning (BSP) trees. The boundary of a locale need not be a regular geometric shape, indeed one of the strengths of locales is that they can have arbitrary geometric definitions. The boundary scopes the extent of a particular locale. What is on the other side of the boundary is governed by the list of neighbours held by the locale, and the BSP tree that is used to determine to which other locale an object should be transferred once it has left the current locale. Each locale also has a separate co-ordinate system that allows detailed interaction in the local vicinity.



Each object in Spline occupies exactly one locale. Having decomposed an environment into manageable chunks, the last task to perform is the ability to locate objects within locales – we cannot interact with an object if we do not know where it is! To overcome this problem Spline introduces the concept of “beacons”. A beacon has two attributes, an identifying tag and the multicast address of the locale containing the beacon. It is possible to identify beacons without knowing which locale they are in through the identifying tag. By issuing a search query, a number of beacons are returned whose tags match the search. Each of these beacons identifies a locale, and all the objects contained therein. Hence it is possible to decide which locale to visit by inspecting the search results. Beacons are very flexible and, through their tags, they can be used to publicise locales (e.g. placing the tag on a the World Wide Web), or for reference purposes so that people who encounter this locale can easily return to it.

Spline has been used to realise a virtual environment known as Diamond Park (Waters 1997), a one square mile terrain containing a dozen buildings, consisting of a number of locales (e.g. one locale for the entire outside world, separate locales for the interiors of each building).

#### **2.2.4.3. Support for security in Spline**

As with the other systems, no explicit security mechanisms are mentioned when Spline is described.

### **2.2.5. MASSIVE-2**

#### **2.2.5.1. Description of MASSIVE-2**

MASSIVE-2 (Greenhalgh 1998) has been developed at the University of Nottingham, and is “concerned with the problem of constructing CVEs that scale to large numbers of simultaneous participants and yet which still afford rich and varied possibilities for communication.” The name of the system is an acronym, and stands for “Model Architecture and System for Spatial Interaction in Virtual Environments”. It is based on the Spatial Model of Interaction (Benford 1993), a model that introduces concepts of aura, focus and nimbus to manage awareness in CVEs. Through experiences with

MASSIVE-1 (Greenhalgh 1995, 1997), a preceding VR tele-conferencing system also based on the spatial model of interaction, a number of drawbacks were identified, and MASSIVE-2 represents an attempt to address these.

Third party objects were introduced (Benford 1997) to provide contextual awareness and aid the process of scalability. In the original spatial model, awareness calculations were based on information pertaining to users within an environment. There was limited scope to include contextual effects, i.e. those from the environment itself, through the use of adapter objects. However it was not possible to consider situations such as nested bounded spaces or shared objects in the virtual world. With regard to scalability, the MASSIVE-1 approach suffered from the large number of pair-wise awareness calculations required for highly populated spaces (an awareness calculation is needed for each pair of users in the same space), resulting in a bottleneck for systems wishing to support large numbers of users. Through the use of third party objects MASSIVE-2 supports large numbers of mutually aware users, providing both audio and graphical feedback.

Third party objects, a much extended notion of adapters from the original model, are at the core of MASSIVE-2. They may be objects in their own right, have spatial extent, they may be mobile or static and may be applied recursively. They affect the awareness calculations between a pair of objects (hence the name third party object). Two effects on awareness are supported by third party objects in MASSIVE-2:

- *adaptation* where existing awareness relationships between objects are manipulated (e.g. increasing or decreasing audio), and
- *secondary sourcing* which introduces new indirect awareness relationships between objects (e.g. representing a crowd of individuals by a single crowd aggregate when the crowd is viewed from a distance).

#### **2.2.5.2. Spatial structuring in MASSIVE-2**

One of the many ways in which third party objects are used by MASSIVE-2 is for world structuring and regions. They make it possible to create nested structures such

as rooms within buildings within zones. A region is typically a membership-activated third party, that is it becomes aware of users as they cross the boundary that is its extent. The effects of a region may be different across different media. The ability to manipulate awareness values across boundaries makes it possible to build large scale environments with many inhabitants. The ability of third party objects to introduce adaptive effects at boundaries makes this approach particularly interesting in comparison to the other approaches discussed above where boundaries are used solely for spatial partitioning. MASSIVE-2 realises third party objects using dynamic, spatially-related multicast groups.

To demonstrate the MASSIVE-2 system we will consider a demo called the Panoptican Plaza (Greenhalgh 1998). This shows how third party objects are used in MASSIVE-2 to enrich the interaction possibilities for the space. Figure 2.4 shows an overview of the plaza, an environment constructed as a thought experiment to investigate different boundary combinations across media. The plaza consists of the main plaza region and three third party objects, a prison cell in the centre, a jury box to the left and a witness stand to the right.

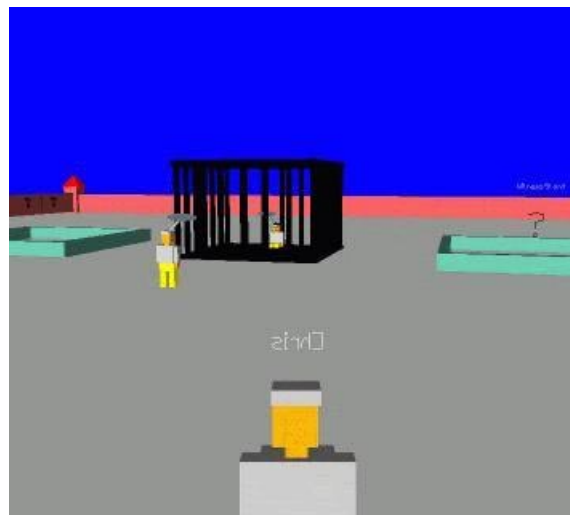


Figure 2.4: The Panoptican Plaza

The prison cell permits those outside the prison full awareness of ongoing activity within the cell (both visual and audio) while anyone unfortunate enough to find

themselves inside the cell has no audio or visual awareness of anything taking place outside the bounds of the cell.

The witness stand works in the opposite way to the prison cell. The person occupying the stand has full audio and graphical awareness of events occurring outside the stand. However, those in the plaza have no visual awareness of the occupant of the stand (they are represented graphically as a question mark) and the audio of the witness is attenuated to disguise her identity. There is no direct awareness from the plaza to the witness box.

The jury box is similar to the witness stand, in that it preserves the anonymity of the jury members whilst providing a clear and full view of the proceedings in the plaza. The audio from inside the jury box is not audible at all from outside the box, and the jurors are represented by an aggregate crowd that conveys the current state of the deliberations by providing a text commentary on the state of the voting.

### **2.2.5.3. Support for security in MASSIVE-2**

MASSIVE-2 has recently been extended to support simple access control using clearances and classifications. Clearances are specified in a user embodiment profile file as simple integers. Classifications are specified for each artefact in a given world description file as text attributes. The “region” the access controls affect is the solid hull of the associated artefact. This may or may not be a CVE region. By default MASSIVE-2 runs without access control enabled, as an additional overhead is imposed on the system when a user is very close to non-secure solid objects.

## **2.2.6. TeamRooms**

### **2.2.6.1. Description of TeamRooms**

TeamRooms (Roseman 1996), a groupware environment developed at the University of Calgary based on the metaphor of shared virtual rooms, “fills the role of a team room for groups whose members can work both co-located and at a distance.” It provides a shared environment where a team of possibly geographically disparate people can work closely together on a common project or task. A central server

manages a collection of user-defined rooms on a per team basis. Members of the team can make use of the different rooms to support their work. TeamRooms provides a persistent meeting space, containing a number of standard tools to aid collaboration. These include a shared whiteboard, a text chat tool and a postit applet to leave text messages. It is possible to customise rooms, providing specialised groupware tools to support tasks as necessary. The system also contains mechanisms to support awareness of other users and their activity. One window provides a list of the rooms making up the environment, another a list of the current occupants of the environment and the room they are in. TeamRooms can be used either synchronously or asynchronously.

#### **2.2.6.2. Spatial structuring in TeamRooms**

A TeamRooms environment consists of a collection of shared rooms managed by a central server process. Once you have connected to the server it is possible to move freely between the rooms, each room being accessible from all others. The server manages many teams through the use of different ports for each team environment. To move between teams you must connect to the new team via the central server. The topology of the teams can be thought of as a star structure. We begin at the centre of the structure and can move to any of the teams. A user is only able to be in any one room at a time.

#### **2.2.6.3. Support for security in TeamRooms**

To access one of the teams (i.e. enter the collection of rooms associated with that team) you must first provide the central server with appropriate authentication information. This takes the form of a username/password pair for users on a per team basis. Once a user has been authenticated for a particular team there are no further access constraints. She is able to examine, modify and delete any information contained in any of the rooms for that team, and can even delete rooms from the environment if she wishes.

### 2.2.7. Summary

We have examined six CVEs, each having adopted spatial structuring of some sort to achieve their goals. One common goal across all the systems studied is the desire to support a large user community. By utilising small, independent regions it is possible to reduce computational complexity, network messaging and complex geometries. The realisations of spatial structuring differ amongst the systems. Dive uses a collection of individual worlds connected via portals. NPSNET tiles its environment into hexagonal cells and then uses an Area of Interest Manager to map groups of these cells onto network multicast groups. RING scopes interaction based on visual occlusion. Spline uses locales to build up complex environments, each locale having a unique identifier, an associated multicast group, a BSP tree describing its extent and a list of neighbours. MASSIVE-2 creates regions through membership-activated third party objects that affect the awareness calculations between objects and are realised as dynamic spatially-related multicast groups. TeamRooms supports provides groups of users with a collection of rooms where they can work and persistently store information.

The spatial approach to world structure appears to be a good concept to use, enforced by the widespread use noted above. Not only do we gain from the possibility of a simpler access model, but also the model should naturally scale to large communities of users so long as a suitable infrastructure is put in place (e.g. similar to the servers utilised by RING).

Notable by its absence is work on security. The majority of the systems above are concentrating on providing detailed large scale environments, a laudable goal, but if these environments are ever to take off commercially there need to be constraints on behaviour and safeguards. There is therefore a need to address the issue of security and access control for large scale CVEs, as such functionality will be required in the future. It may well be possible to utilise standard techniques, but these can be cumbersome and detract from the application or system. If possible an approach to security which complements the aims of the system would be highly desirable.

### **2.3. Matching the results to the requirements**

We have reviewed approaches to access control in section 2.1, and overviewed CVE systems that utilise spatial partitioning as an integral part of their approach in section 2.2. In this final section we look again at the requirements that were postulated in chapter one (section 1.4) for an access model for CVEs. Four requirements were identified:

1. The mechanism must be simple.
2. The mechanism must be unobtrusive to users – there should be no extra overhead imposed on users to ensure secure over insecure operation.
3. It should be easy to inspect and change access rights.
4. Above all, the effect of access controls should be understood, and the consequences of any changes be clear.

Our review of access models has shown that those models avoiding complex formal notations for specifying access rights and the relationships between them are simpler to use and understand. However, for an access model to provide a provable level of security it must be possible to determine and verify the different possible access states. To achieve this requires a degree of formalism in the specification of access rights, so we need to achieve a fine balance between formal access rights which are easily specified and understood.

CVEs are complex environments, the systems examined above having adopted different approaches to overcome this complexity. However all these approaches share the common theme of spatial partitioning. We can make use of this partitioning to simplify the approach to access control as well. Typical access models act on every object in an environment simultaneously, so a knowledge of all the objects in the environment is required. If we make use of spatial partitioning it is possible to introduce a two-level access model. The high level model would concern movement between regions in an environment. For this high level model we do not concern

ourselves with controlling access between objects within the region, rather we control the ability to enter the region to access the object in the first place. It should then be possible to employ separate access models within each region, models tailored to the activity taking place in any particular region. These lower level access models complement the high level model controlling access between regions. It also means that regions do not need to be aware of the content of other regions, only the fact that certain credentials are required to enter them.

The possible use of a different access model within a region offers great flexibility. Of course it would still be possible to use spatial structuring within a region to control access, but this can lead to problems of granularity and the problems of identifying more and more boundaries.

The main aim of a CVE is to support interaction in the VE. The systems examined above have used many different techniques to ensure that the users of their systems are offered the maximum support and flexibility at as low a cost as possible. Any access model should adopt this reasoning, being a natural part of the CVE, as unobtrusive to the user as possible, giving as much support as possible to the normal interactional process.

We have examined a number of access models and spatially motivated CVEs. In the next chapter we present the *SPACE* access model. This is a model based on the fact that we can divide an environment into regions using spatial partitioning, and then control the access between these regions. This is the high-level access model talked about above, a model which, we propose, offers an easy understandable and manageable approach to access control for spatially motivated CVEs.



## Chapter Three

### SPACE – the model

This chapter presents SPACE (Spatial Access Control for collaborative Environments), a model that adopts a spatial approach to access control for collaborative environments. The model concentrates on movement and navigation, and is suited to overview purposes, rather than controlling fine-grained interactions. By default collaborative virtual environments (CVEs) will be used in any examples and discussions. We will see in Chapter five that the model is applicable to other areas beyond virtual environments. The model adopts a spatial approach to access control where boundaries segment space into regions. First and foremost this exploits people's natural spatial reasoning abilities, which allies with the whole philosophy of virtual reality, as was discussed in the previous chapter. The spatial approach can also be used as a complement to other approaches to access control. Perhaps the biggest advantage, as far as this model is concerned, is the ability to represent the spaces being examined as mathematical "access graphs". An access graph is used to represent (possibly very complex) spatial structures, and allows an understanding of access possibilities and the ability to manage these complex structures and spaces.

Chapter three is structured as follows:

- Section 3.1 introduces the model and its associated concepts. To start with some graph theory terminology and definitions are presented. This is followed

by an introduction to the two main components of the access model, *boundaries* and the *access graph*, followed by definitions of relative and absolute classifications. Finally issues concerning teleportation and action at a distance are considered.

- Section 3.2 explains how access control can be understood and managed. A list of questions that define the functionality of a management tool is presented.
- Section 3.3 introduces the adjacency matrix that is determined from the Access Graph of the environment. It outlines techniques for using this matrix to determine the access information needed to manage the environment.
- Section 3.4 presents a simple example environment that demonstrates the features of the model.
- Section 3.5 answers the questions posed in section 3.2 using the techniques outlined in section 3.3 and the example from section 3.4 for illustration.
- Finally Section 3.6 points out some potential shortcomings in the model, and suggests ways of overcoming these problems.

### **3.1. The model**

The access model consists of two key components, *boundaries* that perform the segmenting and structuring of the virtual environment, and the *access graph* which is a concise mathematical summary of the constraints on, and possibilities for, movement around the virtual environment. The model also makes use of definitions concerning classifications associated with boundaries and different modes of traversal within the virtual environment (Bullock 1994, 1994b).

First of all it is necessary to introduce some mathematical graph theory concepts and definitions. We briefly overview graphs, digraphs and their associated terminology (Wilson 1985, Cattermole 1979) and introduce the notions of paths and an adjacency matrix, which will be seen to be an integral part of the approach.

### 3.1.1. Graph theory definitions

#### 3.1.1.1. Graphs

A graph is a discrete structure consisting of sets of vertices (nodes) which may be joined by edges (arcs). A simple graph involving 5 vertices and 5 edges could be of the form shown in figure 3.1.

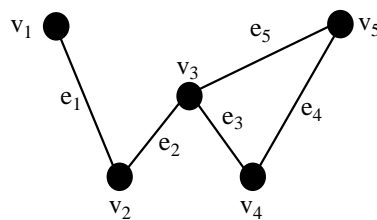


Figure 3.1: A simple graph

A graph  $G$  has a number of significant constituent parts

$V(G)$  is a set of objects called vertices.

$E(G)$  is a multiset<sup>3</sup> of edges. Each element of  $E$  can be mapped to a multiset of two vertices from  $V$ .

In figure 3.1 above  $V(G) = \{v_1, v_2, v_3, v_4, v_5\}$  and  $E(G) = \{e_1, e_2, e_3, e_4, e_5\}$  where  $e_1 = \{v_1v_2\}$ ,  $e_2 = \{v_2v_3\}$ ,  $e_3 = \{v_3v_4\}$ ,  $e_4 = \{v_4v_5\}$  and  $e_5 = \{v_3v_5\}$ .

Some simple definitions:–

Two vertices are said to be **adjacent** if they are joined by an edge. For example  $v_2$  and  $v_3$  are adjacent.

An edge is said to be **incident** to the vertices it joins. For example, edge  $e_1$  is incident to vertices  $v_1$  and  $v_2$ , and  $v_1$  is a **neighbour** of  $v_2$ .

---

<sup>3</sup> A multiset is a set which allows repetition of elements.

If edges are incident with the same vertex they are termed **adjacent edges**; if edges are incident with the same pair of vertices they are termed **multiple edges**

If an edge is incident with only one vertex it is said to be a **loop**

A **simple graph** contains no multiple edges or loops

A **multigraph** contains multiple edges but not loops

A **general graph** or **pseudo graph** contains both multiple edges and loops

### 3.1.1.2. Digraphs

The terminology for directed graphs is much the same as above, except that the graphs are considered as ordered graphs. The direction of edges that join vertices is important.

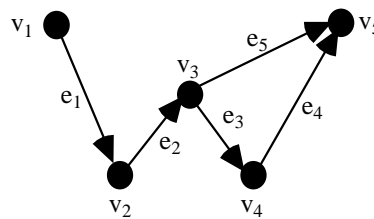


Figure 3.2: a simple digraph

A Digraph  $D$  has the following constituent parts:

$V(D)$  is a set of objects called vertices.

$A(D)$  is a family of ordered pairs of elements of  $V(G)$  called edges.

So, considering figure 3.2,  $V(D) = \{v_1, v_2, v_3, v_4, v_5\}$  and  $A(D) = \{e_1, e_2, e_3, e_4, e_5\} = \{v_1v_2, v_2v_3, v_3v_4, v_4v_5, v_3v_5\}$

Some simple definitions:–

Two vertices are said to be **adjacent** if there is an edge joining one of them to the other. For example  $v_1$  and  $v_2$  are adjacent.

An edge is said to be **incident to** the vertex it joins and **incident from** the vertex it leaves. In the above example, edge  $e_2$  is incident from  $v_2$  and incident to  $v_3$ .

If there are two or more edges joining vertices together then these are called **parallel edges**.

If an edge joins a vertex to itself it is said to be a **loop**

A **simple digraph** contains no parallel edges or loops

The **underlying graph** of digraph  $D$  is the graph obtained from  $D$  with the directed edges replaced by undirected edges (Figure 3.1 is the underlying graph of figure 3.2)

### 3.1.1.3. Paths

A sequence of edges  $v_1v_2, v_2v_3, \dots, v_{n-1}v_n$  is called a **walk** of length  $n$  from  $v_1$  to  $v_n$

$v_1$  is the **initial vertex**;  $v_n$  the **terminal vertex**. If each edge is distinct then the walk is called a **trail**. If each edge and each vertex are distinct then the walk is called a **path**.

A **directed path** is similarly defined.

The length of any shortest path between two distinct vertices is called the **distance** between them.

Vertex  $v_j$  is said to be **reachable** from vertex  $v_k$  if there is a directed path from  $v_j$  to  $v_k$ .

### 3.1.1.4. Adjacency Matrix

The **adjacency matrix** of a graph  $G$  that has vertices  $(v_1, v_2, \dots, v_n)$  is the  $n \times n$  matrix  $A(G) = (a_{ij})$  where  $a_{ij}$  is the number of edges joining  $v_i$  and  $v_j$ .

With the knowledge of these definitions we can now define the model.

## 3.1.2. Boundaries

The most fundamental component of the model is the *boundary*. Boundaries provide a way of segmenting virtual spaces into distinct regions. The boundaries that are talked

of here come from Bowers' definition (Bowers 1993) originating in the ESPRIT funded COMIC project (Comic 1995). A boundary can act in a twofold manner. It can affect and control traversal between regions, and it can affect the awareness between and within regions.

“... I want to suggest that the (arguably) most elementary operation – that of drawing a boundary in space – is rich in significance. As Cooper (1990) argues, the boundary is primary. The entities which come into existence through such divisions are the boundary's secondary effects.”

John Bowers (1993)

There has been much work in the social sciences on the significance of space and how it can be organised to encourage or discourage (social) interaction, acting as a resource for any such interactions (Giddens 1985, Goffman 1967). What Bowers is suggesting above is that it is not the resulting spatial regions from organising a space that are of most interest, rather the process of performing the partitioning in the first place, as without the boundary there can be no spatial structures to examine! Once the boundary has done its work and a spatially-structured environment has been constructed the collection of regions and boundaries can be examined. The boundary is a fundamental element of this thesis work and it cannot be stressed too strongly that without it we would not be able to adopt a spatial approach, an approach which fits in with the philosophy of virtual reality and also complements other approaches, which is one of the strengths of this work.

Chapter two gave examples of systems that use spatial world structuring. There are many ways in which boundaries are manifest. They might take the form of portals between worlds as in systems such as Dive (Carlsson 1993) and MASSIVE-1 (Greenhalgh 1995), or they might be used to partition single worlds as in Spline (Barrus 1996), NPSNET (Macedonia 1995), RING (Funkhouser 1996) and MASSIVE-2 (Greenhalgh 1998, Benford 1997). Boundaries are therefore the ideal place to enforce access control. The basic technique used to control access is through the ability of a person or object to traverse a boundary. To gain access to an object one

must be capable of traversing the boundary or boundaries between the current location and the location of the target object.

The access model introduces the fundamental idea that in order to traverse a boundary, one must first possess necessary and adequate *credentials*. These credentials are presented to the boundary and it determines whether traversal is possible. In the most general case, these credentials involve matching an arbitrarily complex combination of attributes of the user (or indeed group of users). There are many ways in which this is possible. A simple example involves the use of integer numeric values where one must present a suitable integer value to the boundary to traverse it. This particular example is based on an extended clearance–classification scheme (Bell 1973). Another technique involves matching attributes of users’ descriptions such as their name, status, occupation, position or other appropriate attributes. Possession of other objects such as keys and passwords may be sufficient to permit traversal. Whatever the case, these credentials are presented to the boundary which determines whether or not the user can cross. For more details on the specification of credentials, also known as security labels, see chapter six in (Amoroso 1994).

It is possible, and indeed highly probable that more than one boundary may exist between two given regions. When this occurs action must be taken to reduce the multi–component boundary to a single component. To use a real–world example, consider a meeting room with a window onto a corridor that runs along one side of the room. There is also a door into the room from the corridor along this wall (figure 3.3).

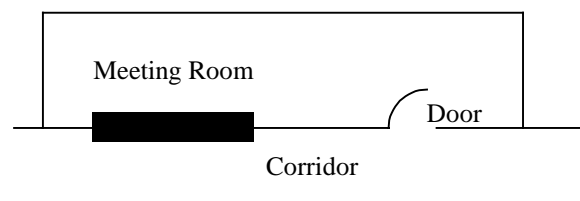


Figure 3.3: A multi–component boundary

The boundary between the corridor and the meeting room has three components, that is there are three different boundaries between these two rooms; the door, the window

and the wall. Obviously each of these boundaries has its own requirements for traversal. During a normal working day the door to the meeting room would be left unlocked, and as such would present the easiest way of entering the room from the corridor. It is possible to make a hole in the wall or break the window, but each of these actions requires greater credentials than simply walking through an unlocked door.

In general, when faced with a number of possible boundary components between a pair of regions, we reduce the list of components to the single boundary that requires minimal credentials to traverse. This is described in the next section on the access graph.

Having introduced the concept of boundaries we continue with the access graph.

### **3.1.3. The access graph**

Multiple boundaries might be used to create complex spatial structures such as many worlds inter-linked by different portals or nested and tiled regions within a single world. In such cases, it is possible to construct a useful framework called an “access graph” that provides a concise summary of the possibilities for, and constraints on, movement around the spatial setting. In particular, access graphs allow us to reason about the possibilities for accessing the objects contained within a given space.

In general, each discrete (i.e. bounded) region of space is represented by a node in the access graph and each boundary by an edge. The edges are labelled with their associated credentials and may be unidirectional if one-way boundaries are supported. Where there are multiple boundaries between two spaces (e.g. the example in figure 3.3 above), there will be parallel edges between the two nodes in the access graph. However, in those instantiations of the model where the possible credentials can be ordered in some way according to their degree of power to grant access (i.e. where there is a well defined hierarchy of credentials which will typically be represented as an enumerated type), it will be possible to replace parallel edges with a single edge



which represents the easiest boundary to cross (i.e. the one requiring the minimum credentials).

Using the meeting room/corridor example, the enumerated type `meeting_room` is defined for the possible boundary types and the credentials required to traverse them for this small space.

```
enum meeting_room (Wall=9, Locked Door=6, Window=4, Unlocked Door=1)
```

Other orderings of credentials for the boundaries found in any space are similarly defined.

In the above example (figure 3.3) it is the unlocked door that is the easiest to traverse and which therefore becomes the critical boundary. Furthermore, the edges of the access graph can then be labelled with an integer weighting which represents the position of its credentials within the overall ordering (i.e. a label on an edge indicates the relative difficulty of crossing the associated boundary). Then, the application of standard mathematical techniques to this access graph will allow us to easily reason about and manage access rights across the entire space (see sections 3.3 and 3.4).

In order to specify our model fully, we must first introduce some concepts related to boundaries and access graphs that are vital if we are to successfully model collaborative environments. These concepts concern how classifications are determined and how it might be possible to move around an environment.

#### **3.1.4. Relative and absolute classifications**

The *relative classification* of two regions describes how difficult it is to move between them. More specifically, the relative classification of region A to region B is defined as the minimum credentials required to move to A from B via *any* valid path. Where directional boundaries are supported (e.g. one way portals), this need not be the same as the relative classification of B to A. We derive this relative classification through two steps.

- 1) The relative classification of a particular path between two regions is the maximum of the classifications of the edges traversed along the path. In other words, whether one can traverse a specific path is determined by the most difficult boundary to cross that is encountered on the path.
- 2) The relative classification of region A relative to region B is then the minimum of the relative classifications across *all* possible paths from B to A. In other words, whether one can move from one region to another is determined by the easiest of all possible paths between them.

The following example illustrates relative classification. Figure 3.4 shows a space with six regions representing a simple office scenario.

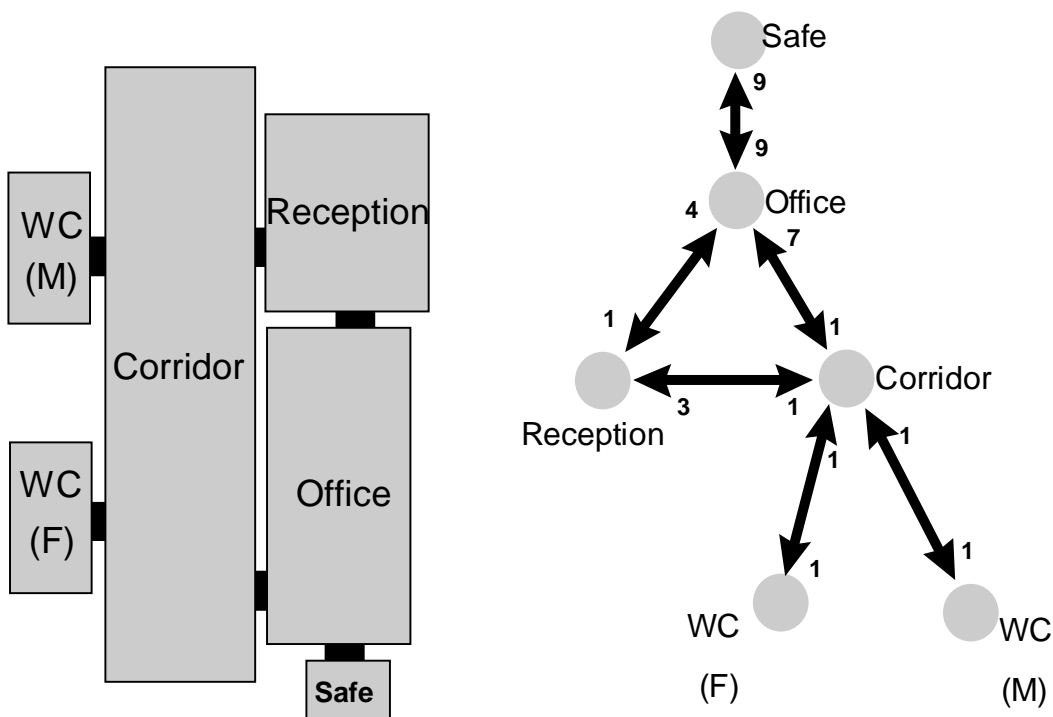


Figure 3.4: A simple office scenario

Three different classes of users populating this space are defined, `visitor`, `employee` and `manager`, with the following clearance definition:

```
Clearance (visitor={1,2,3}, employee={4,5,6}, manager={7,8,9})
```

The access graph for this space has been constructed and is seen on the right side of the figure. Numeric credentials relating to the above clearances have been applied to each boundary and can be seen as edge weights on the access graph.

Some initial observations on this space show that employees need to pass through the reception to gain access to the office, whereas the manager can go directly into the office from the corridor. Also, visitors are not allowed past the reception area unless they have been given the required clearance first. However, anyone is able to leave the office via either of its exits.

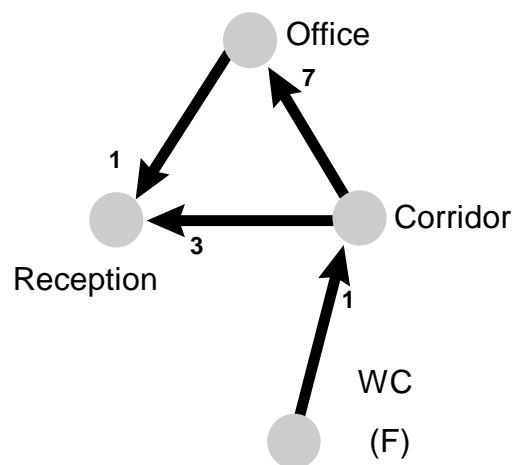


Figure 3.5: An example of relative classification

To demonstrate relative classification we consider the task of going from the Female Toilets to the Reception area. There are two possible paths between these two areas; figure 3.5 shows only the relevant details extracted from the Access graph. The two paths are {Female WC to Corridor to Office to Reception} and {Female WC to Corridor to Reception}. To determine the relative classification of each route we consider each step of the path in turn. For the first path we get {1, 7, 1} and for the second path we get {1, 3}. Therefore the classification of the first route is 7 and the

classification of the second route is 3 (by step 1). By following step 2 we see (trivially) that the second route offers the lower classification, therefore the relative classification of the Reception to the Female Toilet is 3.

We define the notion of *absolute classification* to be the minimum credentials required to enter a given region. In the most general case, this is at least the minimum of the classifications of all edges incident to the node representing this region. In order to enter this region a person must have credentials at least equal to this absolute classification. To increase the absolute classification of a region, the edge incident to this node with the minimum credentials must be modified.

Taking the Reception node in figure 3.4 as an example, its absolute classification is the minimum of {1,3}, which is 1.

An additional possibility is to constrain the entry points into a virtual environment so that, in effect, a user can only join this environment's access graph at one of a designated set of start nodes. In this case, the definition of the absolute classification of a node can be extended to be the minimum of all of the relative classifications across all possible paths between a valid start node and the target node. This will be greater than or equal to the minimum of all of the edges incident to the node, as defined above.

Considering our example of the Reception once more, if we designate the Female Toilet as the only valid start node then the absolute classification of the Reception becomes the relative classification of the Reception to this start node. We have previously calculated this to be 3, so the absolute classification of the Reception has risen to 3.

The second, more specific, definition of absolute classification can be used when there are real concerns about the access security of the environment being examined and protected. By defining a subset of all regions to act as starting points the absolute classification of each region is known taking *every* other region in the environment into account. The general definition only defines absolute classification taking into

account regions that are neighbours of the one being examined. There may well be circumstances where either definition is appropriate, and it is a matter of choice on a per application basis on which one is adopted for implementation.

### **3.1.5. Teleportation and action at a distance**

It would seem that the model requires that a user must actually move to an object's current location in order to access it, perhaps crossing many intervening boundaries on the way. Although this might often be the case, two other possibilities need to be considered:

- *Teleportation* – users directly specify the address of a destination region in order to move there without ever being present in any intervening region (e.g. directly supplying a URL in a VRML browser).
- *Action at a distance* – users act on a remotely located object without ever leaving their current location.

Both of these possibilities are allowed by the access control mechanism, so long as the user's credentials would have allowed them to traverse at least one valid path through the access graph. In other words, the access control mechanism is concerned with whether the journey could logically have been made; how the system actually enables navigation and interaction, given suitable access rights, is another matter.

The basic components of the model have now been introduced. Attention is now drawn to how the model can be used to understand and manage access control.

## **3.2. Understanding and managing access control**

In order to provide effective access control, users must be supported in understanding the effects of different boundary configurations and the consequences of any proposed changes to these. Many traditional access control mechanisms are overly complex (Landwehr 1981) and it is difficult to both specify and understand the controls in place. There is much formalism involved in many definitions (Abadi 1993, Shen

1992) and there is a great danger that the security provided may be overridden by misuse through ignorance or confusion. For example, changing the credentials associated with a boundary at a critical junction might have a major impact on the access characteristics of an entire environment. Users need to be made aware of such possible consequences. A key goal of the proposed approach is to actively encourage this kind of understanding by creating a system which can answer a range of access related questions. In this section, we identify a number of such questions. Indeed, this list of questions defines the functionality of an access control management tool called the security broker, which has been implemented to work with Dive and will be discussed in chapter four.

This list is not intended to be the definitive list of access control questions, rather it represents those considered most pertinent and obviously helpful. Hopefully it covers the main concerns for access security, as this is its aim.

### **3.2.1. Defining the functionality of a management tool**

To manage access control in a CVE we need an understanding of the access mechanisms in place, what security they provide, and what effect this has on the environment they control. Perhaps the easiest way to gain this understanding is to take the point of view of an everyday user of such an environment and ask ourselves what they might be asking themselves. In addition we also consider the interests of system managers in terms of the access related questions that they might ask.

We identify the following key questions that would be relevant to the users and managers of a virtual environment access control mechanism.

- Can I go from A to B?
- What credentials would I need to go from A to B?
- What are the valid paths from A to B for me?
- How secure is A relative to B?

- Which is the most secure region?
- Which is the least secure region?
- Where can the object currently located in A be moved to so that its level of security is not lessened?
- What paths can be taken when moving this object, so that its level of security is not compromised during the move?
- If I have security credentials x where can I go and where am I unable to go?
- What properties do I require to be able to traverse the whole environment? (i.e., to become a Super User)
- What is the effect of changing a specific boundary's credentials on the rest of the space?
- If I add a region and associated boundaries what properties do the new boundaries need to have to maintain confidence in the rest of the graph?
- What is the effect of removing a region (node) from the graph? Do certain conditions have to be met for this to occur?

This list of questions acted as the driving force for the implementation work that is detailed in the next chapter. We now consider the mathematical techniques that can be applied to the access graph for a given space in order to answer these questions.

### **3.3. Adjacency matrix techniques**

When the access graph was introduced it was stated that standard mathematical techniques could be used to interrogate this graph. These techniques centre on the Adjacency matrix of the graph. This matrix was defined at the beginning of this chapter together with the general graph theory definitions, and we re-iterate that definition here.

**Definition:** The adjacency matrix of a graph  $G$  which has vertices  $(v_1, v_2, \dots, v_n)$  is the  $n \times n$  matrix  $A(G) = (a_{ij})$  where  $a_{ij}$  is the number of edges joining  $v_i$  and  $v_j$ .

This matrix is used to form the basis of a powerful mechanism that enables the understanding and management of access control within CVEs.

The adjacency matrix for the environment,  $A$ , is a matrix that identifies all adjacent nodes within the graph of the environment. The indices of each element in the matrix identify a unique pair of vertices in the graph. If these vertices are adjacent, i.e. an edge or edges exist between them, then that element of the adjacency matrix is set to the number of those edges. If the vertices are not adjacent, i.e. there are no edges between them, then the matrix element is set to zero. When the access graph was introduced in section 3.1.3 it was stated that parallel edges would be simplified to a single edge between any two nodes of the graph representing the easiest to cross. Thus whenever there are adjacent vertices this will be shown by an entry of 1 in the Adjacency matrix, and when the vertices are not adjacent by an entry of 0.

The adjacency matrix by itself is of little use, but with the help of simple matrix multiplication we can use this basic matrix to generate every possible walk, trail or path within the graph.

At this point a review of earlier section 3.1.1.3 concerning path definitions in graph theory is recommended to clarify the difference between walks, trails and paths, as these are three different things and not three words used to describe the same thing. To summarise:

a **walk** is a sequence of edges

a **trail** is a sequence of edges where each edge is distinct

a **path** is a sequence of edges where each edge *and* each vertex is distinct

Whenever each of these words is used in the following description it carries the above specific meaning, and is not meant to be read as an alternate name for the same thing.



An alternative way of looking at the adjacency matrix is to consider it as holding all of the one-step walks, trails or paths in the graph of the space. If we multiply the matrix by itself, producing  $A^2$ , the resulting matrix defines all of the two-step walks, trails or paths in the graph, i.e. if  $(a_{ij})^2 \neq 0$  (the matrix element in the  $i^{th}$  row and  $j^{th}$  column), there are that many two-step walks, trails or paths between nodes  $i$  and  $j$ . While the elements of  $A$  are confined to being either 0 or 1, the elements of the powers of  $A$  do not have such constraints (there may be many walks, trails or paths between a given pair of vertices with there still only being a single edge between adjacent vertices). The same result holds for higher powers of the adjacency matrix, i.e. they identify all the walks, trails and paths of length one less than the power of the matrix between any given pair of nodes.

For an environment with  $n$  regions (hence  $n$  nodes in the access graph) we only need calculate the  $(n-1)$  powers of the adjacency matrix, from  $A^2$  up to  $A^{n-1}$ , in order to generate all possible walks (some of which will be trails and some paths) between regions that are of interest. There will exist walks whose lengths are greater than  $(n-1)$  but the longest path in a graph with  $n$  nodes has length  $(n-1)$ , as a path can only visit each node in the graph once by definition.

As  $n$  becomes large, the values the elements that  $A^{n-1}$  can hold become potentially high. Considering a simply-connected 10-node space as an example it is highly likely that some elements of the ninth power of  $A$  will be of the order  $2^8$  or higher<sup>4</sup>, that is there will be around 256 or more possible nine-step walks (some of which will be trails and some paths) between some nodes. The reason for this large growth in the number of walks is that many of these are walks or trails which either visit the same

---

<sup>4</sup> For a reasonably populated adjacency matrix, when  $A^2$  is produced some of the elements will hold values of 2 or higher as a result of the multiplication process. As the multiplication process continues it is not unreasonable to expect similar multiplications of the path lengths to occur. Indeed, the growth of the number of walks could well be much higher than simply powers of two keeping pace with the one less than the power of the matrix. However, the exact number of walks generated is not of real concern, only demonstrating the fact that this number can grow very large very quickly.

node many times or traverse the same edge many times. It would appear that the task of generating all the walks within the graph is overly complex and unfeasible for even moderately sized environments. All is not lost though, as through the application of a few simple constraints, which we shall now describe, the complexity of this task can be reduced considerably.

### **3.3.1. The constraints applied to the universe**

Three constraints are applied when considering all the walks, trails and paths identified by the matrix multiplication process. The effect of these constraints can be summed up as simply reducing all the walks and trails to paths. By definition each node is visited only once and each boundary traversed only once in any path.

The first constraint we apply is that loops are not permitted. We are concerned with environments where boundaries partition the space into regions, the boundary appearing between two different regions. A loop is boundary between the same region. A consequence of this constraint is that we can set the main diagonal of the Adjacency matrix to zero.

The second constraint concerns cyclic walks or trails and those containing cycles. These walks and trails can always be simplified to shorter paths. To do this the cyclic part of the walk or trail is identified and all iterations after the first one are removed from the walk or trail. If this is done for all cycles that appear in the walks and trails then we are assured that we will be left with a path. If we can remove all such cycles from the earliest powers of the adjacency matrices then we drastically cut down the complexity in the higher powers of the adjacency matrices.

Our final constraint concerns re-entrant walks and trails, which pass through a particular node more than once on their way to their destination. These are similar to cyclic walks and trails, but there does not need to be any pattern to where the walk or trail becomes re-entrant. To identify such walks or trails we look for the second appearance of any node in the walk or trail and then remove all the nodes between the first and second appearance of this particular node, and continue searching for other

duplicate nodes in the walk or trail. Through this process the re-entrant walks and trails are simplified to shorter paths, which do not contain the re-entrant section of the walk or trail. Again, this constraint helps reduce complexity in the calculation of the powers of the adjacency matrix.

To summarise:


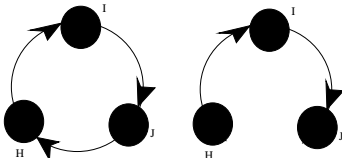
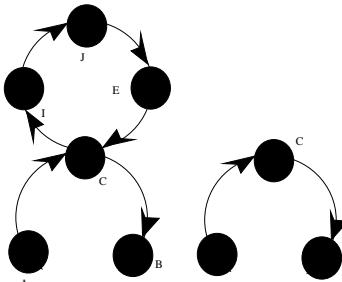
1)	loops are not permitted	AA is not defined	
2)	cyclic walks and trails are simplified	HIJHIJ becomes HIJ	
3)	re-entrant walks and trails are simplified	ACIJECB becomes ACB	

Table 3.1: The constraints for reducing walks and trails to paths in the powers of the adjacency matrix

The key to the analysis of the access graph is the identification of all the paths generated by the powers of the adjacency matrices. The next section outlines a generic technique for generating all paths for a general  $n$ -dimensional space.

### 3.3.2. Constructing the paths

We now consider the process of identifying possible paths and their classifications in more detail.

The starting point is the Adjacency matrix,  $A$ , so our first task is to construct this matrix from an analysis of the structure of the VE (e.g. by inspection of its world data

files). After having done this we next consider the constraints introduced above. The first constraint concerning loops means the main diagonal of the Adjacency matrix can be set to zero (if it is not already). The next two constraints do not apply to the adjacency matrix. By removing loops all possibilities for cycles at this early stage have been removed, and as paths only consist of one-step (i.e. involving only a start node and an end node) at this stage there can be no re-entrant walks or trails either.

The modified adjacency matrix is noted (it holds all the one-step paths) together with a matrix which holds the classification values. This classification matrix is constructed using the adjacency matrix as a base, and wherever there are adjacent nodes instead of entering a value of 1 in the matrix (as done in the adjacency matrix), the edge weight (equal to the classification of the boundary the edge represents) is entered.

Multiplying the modified  $A$  by itself identifies all the two-step walks, trails and paths in the environment. We are able to set the main diagonal of  $A^2$  to zero once again, but this time we use constraint three to achieve this. Any walk, trail or path identified by an element on the main diagonal of the matrix has the same start and end node and is therefore re-entrant. Removing the re-entrant section of the trail leaves just the start node, hence a node rather than a walk, path or trail has been identified. Thus, the main diagonal of  $A^2$  can be set to zero. To identify the walks, trails and paths we make use of the information we already have about one-step paths. For any two-step walk, trail or path, the end node of the first step must be equal to the start node of the second step. We know what the start and end nodes of the walk, trail or path are (the indices of the matrix element tell us this) and identify possible candidates from the one-step paths identified earlier. We then compare the end node of the start step with the start node of the end step. For a valid two-step path these two nodes must be the same (e.g. a path from  $A$  to  $B$  consists of start step  $AX$  and end step  $YB$ , so for a valid path we must find one-step paths  $AX$  and  $YB$  where  $X=Y$ ). By considering each entry in  $A^2$  in turn we identify all two-step paths. There can be no cyclic or re-entrant walks or trails when constructed this way.

Multiplying the modified  $A^2$  by  $A$  gives us  $A^3$ , from which we can identify all three-step walks, trails and paths. The main diagonal of the matrix is set to zero for the reasons concerning re-entrant paths explained above. The start and end nodes of each walk, trail or path are identified from the matrix. As before we match valid one-step paths identified by the adjacency matrix to the start and end node constraints of the walk being considered. This leaves us needing to identify a valid one step path which joins together the end node of the start step and the start node of the end step (e.g. a three step path from  $H$  to  $B$  consists of start step  $HX$ , end step  $YB$ , and a one-step path  $XY$ ). All valid one-step paths are identified by the adjacency matrix so it is a simple case of inspection to find a path which fits. We must be careful, however, to remove all re-entrant and cyclic paths (e.g.  $ACAF$ ,  $ACBC$  and  $AFAF$  are valid three-step paths, but the first two are re-entrant and can be simplified to  $AF$  and  $AC$ , whilst the third contains a cycle and can be simplified to  $AF$ . These are paths which have already been identified ).

By the time we get to four-step paths, a pattern is emerging. After multiplying the modified  $A^3$  by  $A$  to get  $A^4$  we just consider the start and end nodes of each walk, trail or path, identify a valid path between these nodes and store the result. For four-step paths we look at all three-step paths between start and end steps, but we have already identified all of these, and as we were careful to remove all cyclic and re-entrant paths before we stored the results we just use the stored results to build up the path database. This way we are guaranteed to reduce the complexity of the powers of  $A$ . We only need consider powers of  $A$  up to  $n-1$  for a universe with  $n$  worlds, because the longest possible path which satisfies our constraints can visit each world only once, so for  $n$  worlds this path has length  $n-1$ .

To generalise, for a matrix power of  $m$  we identify the start and end nodes of each potential path and then find a suitable path of length  $(m-1)$  which satisfies the start and end node constraints.

### 3.3.3. Storing the classification values

To store the classification values associated with each path we determine the relative classification of the start node to the end node along that particular path. Using the earlier definition of relative classification this is just equal to the maximum arc weight of all the arcs which make up the path from start node to end node. In the simplest case there is simply one arc between the nodes, and we construct the classification matrix. The indices of each element of this matrix identify the two nodes between which this weight applies, and we use this matrix as a base for determining classification values for the longer paths in the environment.

Each path identified through the matrix multiplication and simplification technique is broken down into its constituent one-step elements. The classifications of each of these elements are noted through examination of the classification matrix, and the element with maximum weight is noted. By exhaustively determining the classifications for each path in turn, the result is a complete list of all valid paths and their associated classification values for the environment represented by the access graph.

By examining the access graph, its nodes, arcs and arc weights, we have been able to determine the access rights for the entire space by considering only the arc weights between individual nodes.

The information identified above, that is the set of paths and their associated classifications, is all that is needed in order to check for the existence of any paths between two nodes, to identify the paths should they exist and to inform of the classifications associated with these paths.

## 3.4. An example space

The adjacency matrix techniques that have just been outlined above provide the key information that permits us to answer the questions posed earlier in section 3.2. It is the stored information on paths and their classifications which is of importance. If this

information is combined with the classification definitions presented in section 3.1.4. then we are able to determine answers to the questions. However, before answering the questions we give a simple worked example which illustrates the ideas introduced above.

A 10 node environment with 26 boundaries is shown in figure 3.6 (in this example all of the boundaries are bi-directional, but that may not always be the case). This environment consists of four offices leading off a common corridor and a number of information resources available from various offices and the corridor. Circles are used to represent the information resources whereas the offices and corridor are represented by rectangles in the diagram.

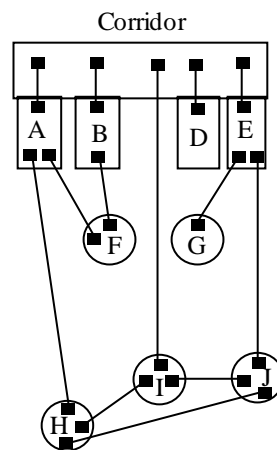


Figure 3.6: An example environment

The access graph for this environment has been constructed (figure 3.7) and boundary classification values have been added as weights on each of the arcs. For the sake of clarity a single edge, with a directional arrow on each end, connects adjacent nodes. In reality this represents a pair of edges which join adjacent nodes in either direction, with a different weighting applying to each edge.

The adjacency matrix and classification matrix for this environment can be seen to the right of figure 3.7. These matrices are constructed from information held in the access graph. The adjacency matrix is constructed by examining pairs of adjacent nodes in the access graph and setting the appropriate elements of the matrix accordingly. The

classification matrix is constructed by examining the arc weights for each pair of adjacent nodes in the access graph. For example, nodes *A* and *F* are adjacent, and so an entry of 1 appears in the adjacency matrix for row *A* and column *F*. The weight of the arc connected *A* to *F* is 2, so the corresponding element of the classification matrix is set to 2. This is done for all adjacent nodes, and thus the adjacency and classification matrices are constructed. Thus the access graph has provided all the raw information needed to reason about access control in the environment.

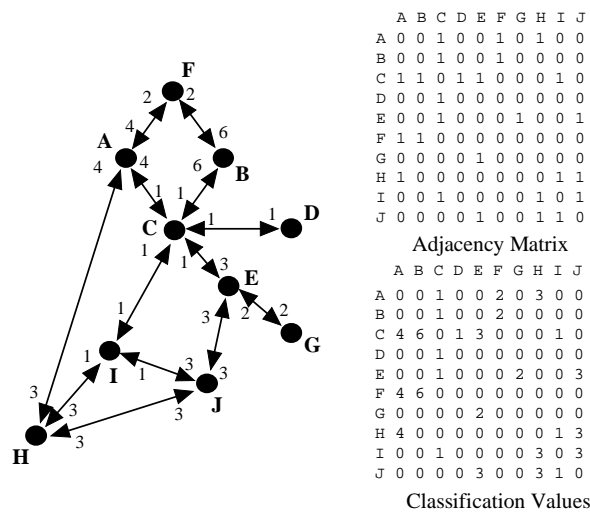


Figure 3.7: Access graph to adjacency matrix

We now construct the powers of the adjacency matrix following the steps outlined in section 3.3. As the environment consists of 10 nodes we only generate the first nine powers of the adjacency matrix. Figure 3.8 shows the resulting powers of *A*. Notice that the highest powers are the most sparsely populated, as the constraints have removed all cyclic and re-entrant paths. We can now use this collection of matrices to answer questions about the environment. For example, is it possible to move between nodes *I* and *E*? We need to examine all elements in the matrices whose indices are 9 and 5. These elements have been highlighted bold and underlined in figure 3.8. By inspection there are 9 possible paths between these two nodes.



A	A <sup>2</sup>	A <sup>3</sup>	A <sup>4</sup>	A <sup>5</sup>
0010010100	0201100021	0020111113	0122401221	0212214223
0010010000	2001100010	0010011302	1001200243	2010312234
1101100010	000021202	2100100222	2000211221	1100122011
0010000000	1100100010	0000021202	2100100222	2000211221
0010001001	1101000120	1010020311	4221010112	2312040221
1100000000	0020000100	1102200031	0010102315	1121401433
0000100000	0010000001	1101000120	1010020311	4221010112
1000000011	0020110011	1322301010	2222133013	2202241020
0010000101	2101200101	1020132101	2422111101	2312231201
0000100110	1020001110	3222110010	1312251310	3411132010
A <sup>6</sup>	A <sup>7</sup>	A <sup>8</sup>	A <sup>9</sup>	
0101122211	0000111110	0000001000	0000000000	
1021423211	0012214121	0001002000	0000000000	
0200111011	0100101000	0000001000	0000000000	
1100122011	0200111011	0100101000	0000001000	
1411040011	1211010121	0001000000	0000000000	
2212404242	1101104000	0000001000	0000000000	
2312040221	1411040011	1211010121	0001000000	
2200022002	1100100010	0000001000	0000000000	
1111142001	1201201101	0000002000	0000000000	
1111121210	0101101010	0000001000	0000000000	

Figure 3.8: The nine powers of the adjacency matrix

Determining the paths for the environment is done following the techniques in section 3.3. Once these paths have been determined we can calculate the classification of each path and be in a position to answer the rest of the questions.

The next section details the general techniques used to answer each question, making brief reference to this example. A fuller working of this example appears in chapter four, where the realisation of the model using Dive2.2 is used to answer each question in specific terms relating to the example.

### 3.5. Answering the questions

We consider each question in turn.

#### Can I go from A to B?

To find out if we can move between two nodes we examine the powers of the adjacency matrix. If there exist any non-zero entries for the element whose indices are equal to the nodes under consideration, in any of the matrices, then it is possible to move between the nodes. By looking into the specific paths identified by the matrices and their classification it is possible to deduce much more information about these two nodes. However, to simply find out if we can possibly go from A to B we just look for a non-zero entry in at least one of the powers of A for the element whose

indices are equal to the two nodes being considered. For the example above the answer is yes.

### **What credentials do I need to go from A to B?**

Once it has been established that at least one path exists between two nodes we need to consider how many paths there are and what their classification values are. The non-zero matrix elements identified when proving the path's existence tell us how many paths of what length there are between the two nodes. As a list of every path for the environment has been generated, while the matrix powers of A were being generated, it is just a case of identifying those paths that fit the starting and finishing node requirements. The classification of each of these paths is then determined, and the minimum value from the path classifications identifies the minimum credentials required to go from the start node to the end node. A consequence of this technique is that the path associated with this credential value is also identified.

### **What are the valid paths from A to B for me?**

We use the same technique as above to identify the credentials needed to move between a pair of nodes, but this time we optionally state a clearance level (defaults to the minimum) for a user and just return those paths between the two nodes which the user is cleared to traverse.

### **How secure is A relative to B?**

After confirming that at least one path exists between the nodes we examine all the paths between A and B to find the path with the lowest relative classification. This will identify the weakest link in the security of A relative to B and may be useful in identifying and eliminating weaknesses from an environment.

### **Which is the most secure region?**

The node with the highest absolute classification identifies the most secure node. Depending upon which version of absolute classification is adopted this can either identify the node whose lowest 'incident to' edge has the highest weight, or else the node whose classification is highest, relative to a set a valid start nodes.

**Which is the least secure region?**

The node with the lowest absolute classification identifies the least secure node. This node is similarly defined to the most secure region, except we look for the lowest weightings.

**Where can the object currently located in A be moved to so that its level of security is not lessened?**

The absolute classification of the object's current location is identified. Each of the other nodes in the environment is then examined in turn to determine its absolute classification. Those nodes whose classification is greater than or equal to the current classification are locations where the object can be moved to.

**What paths can be taken when moving this object, so that its level of security is not compromised during the move?**

The previous question has identified a list of nodes where the object can be moved. The paths database is inspected to identify all paths between the current node and each of the target nodes in turn. This collection of paths contains all the possibilities for movement around the environment. The classification of each constituent part of each path is known, so any path that has a component whose classification (i.e. the value held by the edge weight, identified in the Classification matrix) is less than the absolute classification of the current node is discarded. The absolute classification of the node linked to also must be greater than or equal to the object classification. This may or may not reduce the number of available target nodes available in which to locate the object. This resulting path set contains all paths which may be taken in moving the object in the environment so that its level of security is not lessened. It may be necessary to increase the credentials of the object to traverse some of the paths identified, but we are guaranteeing a minimum level of security.

**If I have security credentials x where can I go and where am I unable to go?**

By identifying all nodes whose absolute classification is less than or equal to our clearance we identify where we can go. All other nodes (those with classifications greater than our clearance) are no-go areas. The paths available to be traversed can be

identified by considering the subset of all paths whose classifications are less than or equal to the user credentials.

**What properties do I require to be able to traverse the whole environment?  
(i.e., to become a Super User)**

To traverse the whole space clearance equal to the maximum absolute classification for the space is required.

The final three questions posed have a common theme. They concern changes to the environment and the effects of these changes. These questions are answered through simulation rather than analytical techniques. The requested changes are made to the information base representing the environment. The access information is then recalculated and a new set of path and classification data is generated. This new information can then be compared to the original to determine the effects of the changes. If the changes are acceptable then they can be applied permanently to the environment. If further investigation and modification is required then it is possible to change the values once more and regenerate the access information.

**What is the effect of changing a boundary's credentials on the rest of the space?**

The classification value for the boundary held in the classification matrix is updated and the classifications recalculated for all the paths identified in the environment. The new values should be stored in a new location and compared with the classification values held before the changes. Any differences are reported to users and they can immediately see the effect of their changes.

**If I add a region and associated boundaries what properties do the new boundaries need to have to maintain confidence in the rest of the graph?**

The adjacency matrix for the space is extended to include the new node(s) and the classification matrix is updated accordingly. The new classification values for all paths are calculated and compared against the values before the new node(s) was added. Any differences are reported to the users and they can then change the

classification of the new node(s) so that the classification values are at least equal to the original values.

**What is the effect of removing a region (node) from the graph? Do certain conditions have to be met for this to occur?**

The adjacency matrix for the space is reduced so as to represent fewer regions in the space (any links to this node disappear) and the classification matrix changed accordingly. The paths database is recalculated and compared to the original set. Careful examination of the differences between the old and new path sets should show if any regions are cut off from others and if new links need to be added between regions. In some cases it may not be possible to delete the region without redesigning the whole structure of the environment.

### **3.6. Shortcomings of the model and workarounds**

A model that adopts a spatial approach to access control for collaborative environments, through the use of boundaries to segment space, has been presented. It is argued that the exploitation of people's natural spatial reasoning abilities makes this model particularly suitable for collaborative virtual environments. A powerful approach for providing such mechanisms has been given.

However, there are some potential disadvantages to the spatial approach.

It is more difficult to specify fine-grained restrictions on the ability to perform specific actions on individual or small groups of objects. This would require wrapping objects in ever smaller spatial boundaries and finding appropriate metaphors for associating specific actions with these boundaries. It is extremely difficult to group such objects together according to non-spatial relationships (e.g. where objects that are far apart need to be treated in a common way). It makes more sense to employ access control list and capability based techniques at this level; using a broad spatial approach for general access control and a specific approach for the detailed fine-grained access.

Furthermore, the possibilities for accessing an object can change as it migrates around a virtual environment making it more difficult to specify globally consistent and stable access rights (this would at least require the ability to move bounded spaces through other bounded spaces as realised in the MASSIVE-2 system).

Another problem concerns action at a distance. The spatial approach assumes that one moves to the space in which an object is located in order to access it. Support for action at a distance requires some relaxation of this principle. Indeed, this is a general problem with spatial metaphor and section 3.1.5 offered a possible solution where the physical action of movement is discounted and it is only the logical ability to make the journey that matters. This may have consequences for social interaction in an environment (Bowers 1995) and it is left to the application developer to decide what approach to adopt. Neither possibility is excluded by the model.

In conclusion it is argued that the spatial approach potentially offers a natural and understandable access mechanism for virtual environments. There is a price to pay in lost granularity because of having to look at the environments under consideration through spatial eyes, but there is nothing to stop other approaches to access control being employed in a complementary manner. Indeed, it may be highly desirable to provide top level access management through a spatial approach but to retain individual access rights for key objects within an environment through established access models.

This chapter has outlined a model for access control in spatially motivated virtual environments. In the next chapter, chapter four, a prototype implementation of this model using the Dive virtual reality system and an access model based on the Bell-LaPadula clearance classification access model is presented.

## Chapter Four

### Implementing SPACE

Chapter three presented an access control model for collaborative virtual environments. This chapter presents an implementation of this model as an extension to the Dive system (Carlsson 1993) from the Swedish Institute of Computer Science. Users inhabit applications within Dive and a simple integer-based clearance classification scheme has been implemented at gateways between worlds. The management features described in chapter three have been provided through the construction of a security broker application. This broker holds all the information concerning paths in the VE and provides an interface that enables users to view, manipulate and present the access information in a number of different ways. This broker gives the user an overview of the environment they inhabit.

Chapter four is structured as follows:

- Section 4.1 introduces the structure of the implementation and shows that although the implementation has been heavily tailored to a specific version of Dive, it is possible to extend the work to cover other systems through the use of filters.
- Section 4.2 presents the access control model adopted, based on the Bell-LaPadula clearance-classification model.

- Section 4.3 presents the virtual environment adopted, Dive 2.2, and outlines the modifications to enable access control at gateways.
- Section 4.4 describes the security broker. The techniques used to construct this access information repository are detailed, explaining how the raw access information is extracted from the Dive data files, and how this data is used to generate all the paths in the environment using the Adjacency matrix and its powers. This information provides the key to analysing the virtual environment. Two different interfaces are then presented, one which presents the results in a textual form, the other which generates directed 2-D graphs using the daVinci graph package (Fröhlich 1995). The broker is also able to update the access information held in the Dive data files.
- Section 4.5 presents an example session using the implementation. This example was first discussed in chapter three, and with the help of a realisation in Dive we can examine this scenario in more detail.
- Finally Section 4.6 presents a filter that allows simple access control information from MASSIVE-2 world description files to be examined using the security broker.

## **4.1. Introduction**

This implementation work demonstrates the model in use in a collaborative virtual environment. At the outset it should be noted that this realisation has been tailored to a specific version of Dive, version 2.2, and constraints have to be applied in the construction of the Dive data files, which describe the virtual environment and define the access information. The aim was not to create a generic access control and management environment, rather to demonstrate a realisation of the model, which provides effective access control and a broker tool enabling easy management of access information.



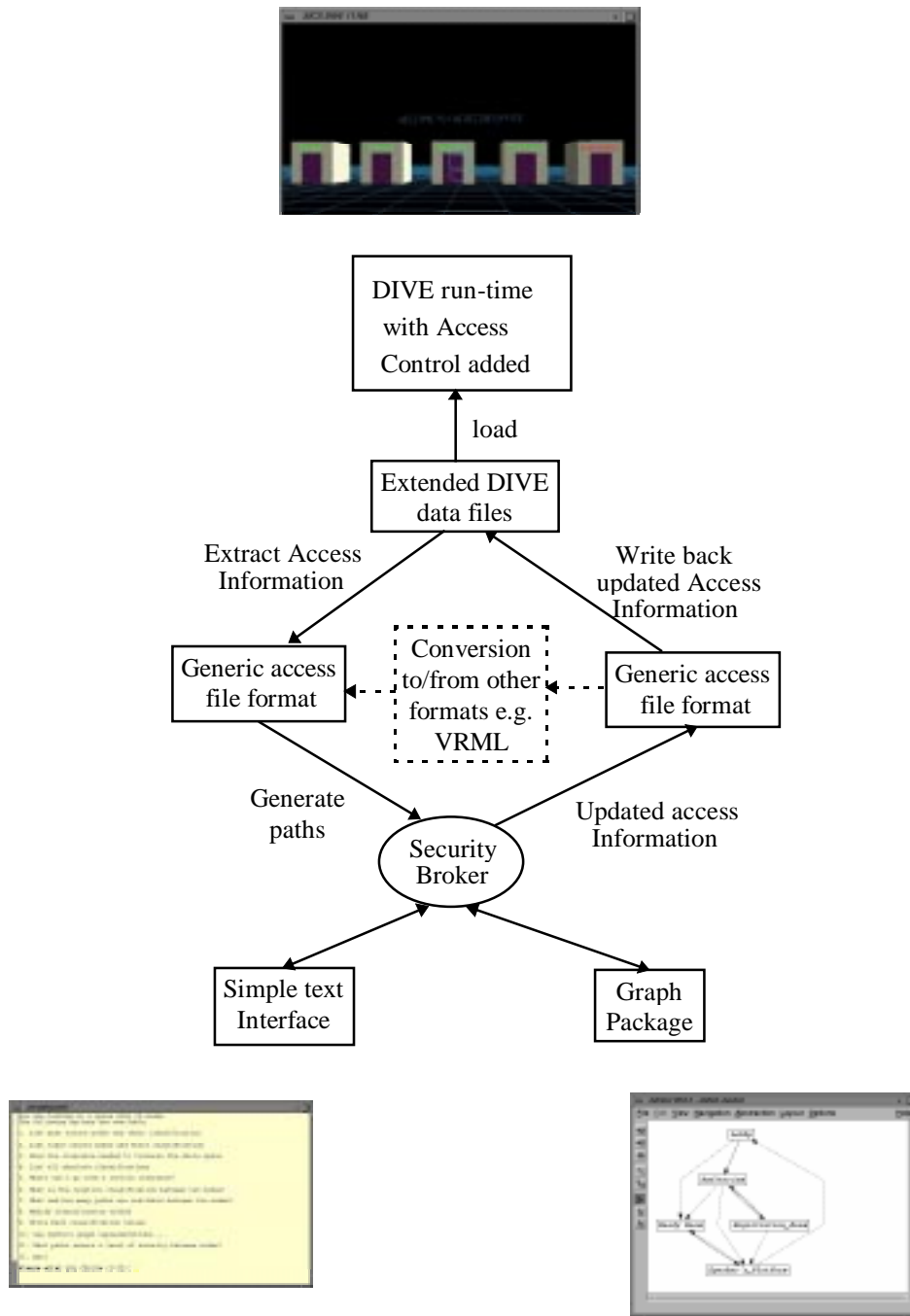


Figure 4.1: The structure of the implementation

A modular approach has been adopted in realising this implementation. There are three main components to consider

- The virtual environment – Dive

- The security broker – the management tool at the heart of the implementation
- The interface to the security broker

A distinction should be noted between the environment that is inhabited and controlled and the process of controlling the environment. Dive is the application or environment that a user inhabits. The broker and its associated interfaces are a means of controlling this environment or application. That is to say we have two different processes here, but that they complement each other and need to be considered in conjunction to understand and manage access control in the environment fully.

The majority of the work has been undertaken in creating the security broker tool which extracts relevant access information from the virtual environment, analyses this information using the techniques discussed in chapter 3, and provides the results to be viewed, either textually or as a 2D directed graph. Figure 4.1 shows the overall structure of the implementation.

Other systems could make use of the management and interface features through the use of a conversion tool between system-specific file formats that define boundaries, regions and access information and the generic access file format used by the broker.

The features of the implementation will now be presented, including details on the access model, virtual environment, the security broker and the management interfaces.

## **4.2. The access model – the Bell–LaPadula clearance classification model**

Chapter two identified a number of different access models; Access Control Lists, the Access Matrix and the Bell–LaPadula access model to name a few. The main requirement from a model for this work is that it supports access credentials that can be ordered hierarchically. This permits multiple boundaries to be reduced to the single, weakest boundary (see section 3.1.2) and also introduces the possibility of using integer values for classifications and credentials that greatly simplifies the

specification and checking of security credentials<sup>5</sup>. The most appropriate model that fits this bill appears to be the Bell–LaPadula clearance classification access model (Bell 1973). However, it is possible to use other models, and one could easily imagine using the Access Matrix model instead of the Bell–LaPadula model.

To briefly review, the Bell–LaPadula model was developed with military security in mind and is based on the access matrix model. Objects have classifications applied to them that define their security rating and accessibility. Users are given clearances that determine which objects they have access to and what sort of access this is. It is possible to informally express the key characteristics of the model in terms of two fundamental axioms:

- i) No user may read information classified above his clearance level (“no read up”)
- ii) No user may lower the classification of information (“no write down”)

There have been a number of adaptations of this model, most notably the Feiertag (Feiertag 1977) model. These extensions have provided easier automated proofs (Feiertag 1977), integrity (Biba 1977) and secure database management systems (Hink 1975). However, none of these extensions provides us with exactly what is required, so we adapt the Bell and LaPadula clearance–classification model for our own purposes.

For our initial scenario nine distinct credentials are specified by the integers 1 to 9 with the level of security increasing as the integer values increase<sup>6</sup>. These credentials can be used to specify both the classification and clearance values associated with the VE. Each of the values corresponds to both a classification and a clearance. Hence we define nine classification values which can be applied to boundaries and nine

---

<sup>5</sup> Using, for example, an enumerated type.

<sup>6</sup> For complex scenarios the number of credentials may have to be increased.

clearance values that can be applied to users or subjects in the VE. The first axiom from the Bell–LaPadula model is strictly observed, that is users may not traverse a boundary whose classification is greater than their clearance. The second axiom concerning the modification of access values in a secure system is a management issue. Everyday users of the system are not allowed to modify classification values, only system managers are able to do this. This is because for management purposes, it is desirable to make changes to gateway classifications and see the consequences of these changes, even if these changes may decrease the overall system security. The original BLP model applied in much more strict and controlled circumstances.

Having introduced the access model used in this implementation we continue with an introduction to the CVE adopted, Dive.

### **4.3. The collaborative environment – Dive 2.2**

Dive was one of the first distributed collaborative virtual environments to appear, dating back to the early 1990s, and represents relatively mature technology in a quickly developing field. It was chosen because it offered an environment consisting of a number of worlds interconnected by gateways, conforming to the spatial structuring approach. There was also full access to the source code. Version 2.2 of Dive was used for this work.

In overview, simple numerical credentials have been associated with gateways between Dive worlds. Access control has been enabled by comparing a user's clearance value against the classification value held by the boundary. If the user's clearance is greater than or equal to the gateway classification she can traverse the gateway. If her clearance is insufficient she is stopped in her tracks and denied access to the world the gateway leads to, remaining in the initial world. She is informed of this denied access in two ways. First a graphic appears on her display, a simple no entry sign, and remains there for a few seconds. Secondly on audio-capable systems a message is played informing the user that she has insufficient clearance to traverse the gateway.

Modifications to three parts of Dive were required to implement access control. The first of these was to extend Dive's world description files to include the specification of numerical credentials as part of gateway definitions. An object is identified as a gateway by the presence of a gateway attribute in its definition. This attribute has a target world and a set of co-ordinates as arguments that specify which world is linked to and where within that particular world. The schema that defines an object within Dive, and thus a gateway object as well, was extended to include a classification parameter, thus whenever a basic gateway definition appeared an integer classification value was also required. An extract showing the attributes of a *vr\_obj*<sup>7</sup> definition relevant to gateways can be seen in figure 4.2, with the classification attribute highlighted at the bottom.

```

struct vr_obj {
    DIVE_object_type type;          /* Object datatype == VR_OBJ */
    DIVE_vrobj_type sub_type;      /* type of vr_obj          */
    objid_t id;                    /* Unique identifier in time and space */
    ...
    char gateway_world[MAX_WORLD_LENGTH]; /* Optional world connected by
                                           gateway */
    point_t gateway_T[1]; /* Optional position in gatewayd world */
    mat_t material;
    int texture;           /* Texture number */

    char maj_descr[MAX_DESCRIPTION]; /* Use for classification etc */
    char min_descr[MIN_DESCR_SIZE]; /* Specific information */

    int classification;      /* security classification */
    ...
}

```

Figure 4.2: Extended Dive schema to include gateway classifications

Gateway definitions appear in two forms within Dive data files. The first form is as simple object components where a gateway attribute is part of the object definition. The appropriate arguments to make this conform to the schema definition must then be given. Below in figure 4.3 we see an example of an extended simple gateway definition from a data file. A gateway to the world called *glashall* is defined. The position of the gateway within the world in which it is located is specified next as an

---

<sup>7</sup> *vr\_obj* is one of the basic objects in the virtual environment.

$(x,y,z)$  co-ordinate, followed by the classification of the gateway. Finally the geometric appearance of the gateway is defined, in this case a four-sided polygon.

```

object {
  material "GATEWAY_M"
  gateway "glashall"
    v      10      1.8      1
  classification "5"
  view 0 {
    N_POLY 4
    v      16.05    2      26
    v      16.05    2      28
    v      16.05    0      28
    v      16.05    0      26
  }
}

```

Figure 4.3: Simple definition of a gateway within a Dive data file

The second form of gateway definition is as a macro. A number of different macro definitions for gateways already existed in the standard Dive distribution. These macros not only defined where and within which world you would appear by traversing the gateway, but also geometric information concerning the appearance and location of the gateway within the original world. Macros are later expanded into their basic schema representations (as in figure 4.3) when the data files are parsed by Dive. Based on an existing gateway macro, a new classification macro was defined which included an argument which specified the classification of the gateway. Below in figure 4.4 we see an example of this macro. The first argument specifies the world the gateway is a link to, `conf_world`. The second argument is a text label which will appear above the gateway in the VE, `conference`. The third argument is the classification of the gateway, 2 in this case, and this is followed by three arguments which define the materials used for the appearance of the gateway and its associated text. Finally the location of the gateway is given by the last three arguments which are  $(x,y,z)$  co-ordinates.

```

object {
  translation v 45 0 3
  fixedxyz v 0.0 3.141 0.0
  class_gateway("conf_world", "conference", "2", "GATEWAY_M",
"WHEAT_M", "GREEN_M", 0.0, 1.8, 9.0)
}

```

Figure 4.4: Macro gateway definition

Through these extensions it was possible to impose integer classification values on gateways in the Dive data files.

The second modification to Dive was to associate integer clearance values with users of the system. To do this the definition of a user had to be extended to include a clearance attribute in the schema defining a person. Figure 4.5 shows an extract from the definition of a person object (only the relevant sections have been shown).

```
typedef struct person_t{
    DIVE_object_type      type;           /* PERSON */
    objid_t id;           /* Unique identifier */
    ...
    int clearance; /* personal clearance level */
    ...
} person_t, *person_p;
```

Figure 4.5: Extended definition of a person to include clearance

Having given a person the possibility to have a clearance value we defined how this value was specified on a per user basis. This was done by extending the embodiment configuration file for a user (the `.person.vr` file). This is a file which defines body geometry and user behaviour. Below in figure 4.6 we see an extract from a simple embodiment file with clearance added. A person is made up of a number of components, i.e. a body, head and eyes grouped together, and clearance has been added as one of these integral body components.

```
...
#define TOP_ID 1
#define BODY_ID 2
#define HEAD_ID 3
#define EYE_ID 4
#define LEFT_EYE_ID 5
#define CLEARANCE 4

person "Adrian" {
    "top" id TOP_ID
    "body" id BODY_ID
    "head" id HEAD_ID
    "eye" id EYE_ID
    "left_eye" id LEFT_EYE_ID
    "clearance" id CLEARANCE
}
...
```

Figure 4.6: User embodiment file with clearance

The final extension to Dive was to enable the checking of credentials before allowing users to move through gateways. Dive 2.2 executes a world transition by checking that a user has intersected with a gateway object as part of a movement. If this is the case a function is called which transports the user to the world defined by the gateway. To implement credential checking the key function that controls world traversal was identified and extended. A new function was added to compare the user credentials against the gateway classification before any of the world transition actions were performed when a user intersected with a gateway. If the user clearance is greater than or equal to the gateway classification then normal world traversal takes place. If it is not, then no world traversal takes place. Users are informed of this failure by a graphic *no entry* sign appearing for a short time on their client interface and a short audio message informing them of the insufficient clearance.

This section has explained how a basic access control mechanism has been added to Dive. The next section explains how access control is understood and managed within Dive, through the extraction of access information from the data files and its subsequent manipulation by the security broker.

#### **4.4. The security broker**

The security broker is an access control management application. It allows users to view the access rights associated with a space, and to manipulate those access rights allowing experimentation with different access scenarios before any changes are applied to a space. Through an appropriate interface the security broker is able to help answer the questions raised in section 3.2.

The security broker operates by first inspecting a collection of specified extended Dive world description files that define the VE, and converting them into a general intermediate format which retains only topology and access information. One of the advantages of working with such an intermediate format is that converters to and from other world description formats (such as VRML (ISO 14772-1 1997)) can be easily implemented in the future. The broker inspects the resulting files in order to construct



an access graph and its associated adjacency matrix. The techniques outlined in chapter three are then used to construct all valid paths in the VE and their associated classifications. This information is used to answer the common access control questions from earlier. Text and daVinci (a directed 2D graph drawing package) based interfaces have been written to interrogate the access information.

The broker adopts an approach that relies on compilation to produce its results, rather than an interpreted approach. There are a number of pragmatic reasons why this compiled approach has been adopted. Dive, like most other VR systems to date, places semantic information about the environment that it is modelling in the geometric world description files that describe the appearance of the environment. This is a known problem, and future systems should strive to separate the definitions of semantic and topological information. If this were the case an interpreted approach would be possible, but until this separation occurs we are forced to adopt a compiled approach. The main drawback of this is the performance of the broker. If we change only a single value we still have to recompile the entire contents of the security broker, re-evaluating the entries for each path held as we have no way of knowing the consequences of the change for the global picture.

#### **4.4.1. Extracting access information**

The first step to understanding a space is to extract the access information associated with that space. The data extraction tool has been implemented using the standard UNIX shell *sh* (Kernighan 1984) and pattern matching and manipulation tools *sed* and *awk* (Dougherty 1992). A collection of data files are given as arguments, and these files are then parsed and the gateway and access information extracted through pattern matching techniques.

First, the raw gateway definitions are extracted, a check being made to make sure the name attributed to a world within its data file is the basename of the data file (sometimes this is not the case, in which case the file basename is adopted as the name of the world). Next, the extraneous information and blank spaces contained in the definition of a gateway are removed, making sure to retain the distinction between

the different information fields. The resulting file is then checked for consistency, making sure that for each gateway the target world exists. If this world does not exist an extra world is created and added to the collection specified on the command line, with all dangling world links pointing to this world. (For example, if a subset of Dive worlds, from a given environment, is being examined it is possible that there will be links from this subset to worlds not contained in the subset. For consistency in generating the access graph and adjacency matrix we create an extra world and point all dangling links to it).

We are now ready to construct the intermediary file that will hold the information from which the access graph and adjacency matrix are constructed. This format is one that has been developed at Nottingham in other projects, and is known as the CyCo (Benford 1993) file format. It is a simple structure, which allows easy graphical interpretation and the ability to store attributes and other information about the structures that it represents.

A CyCo file assigns each world a unique numeric ID and contains the total number of worlds being considered as the first line of the file. It lists for each world in turn various attributes, including a brief one-line description, the worlds linked to and the classification values associated with the links. The annotations on figure 4.7 detail the exact file format. Using the information that has been extracted from the Dive data files and simplified, the first step in constructing the CyCo file is to replace each world name with a unique numeric identifier. The extracted information is then parsed twice and written out in CyCo file format. The first pass establishes the structure of the CyCo file while the second pass fills in the details of the worlds linked to and the classification values, and makes sure that a consistent CyCo file is output. The resulting file contains a list of worlds linked to for each world, and the associated classification value for each link.



#### 4.4.2. Constructing the adjacency matrix

The CyCo file, a concise representation of the pertinent access information about the space, is used as the primer for the security broker, and from it we are able to understand and manage the space. This is because from this information we can construct the adjacency matrix for the space. First of all a null matrix is initialised, having dimensions equal to the number of worlds in the CyCo file (the first line in the file marked ① in figure 4.7). This matrix is then filled in a row at a time, taking each world in the CyCo file in turn. Worlds linked to are identified by their numeric ID within the CyCo file (the line marked ② in figure 4.7), so it is simply a case of setting these elements within the row to ‘1’ to show that a link exists between the worlds. After this has been done for each world in the CyCo file we have constructed the adjacency matrix for the space.

A classification matrix is also constructed, which holds the classification values associated with each link between a pair of worlds. As with the adjacency matrix, a null matrix with the same dimensions is initialised. The classification values associated with each world are read into an array in turn (the line marked ③ in figure 4.7). The links to which these values refer are then read in (line ② again), and the matrix elements to which these links refer to are filled in with the previously stored classification value. Taking the CyCo file from figure 4.7 as an example, the following adjacency and classification matrices are constructed.

0 1 0 0 1	0 5 0 0 1
0 0 1 0 0	0 0 1 0 0
1 1 0 0 0	2 2 0 0 0
0 0 1 0 1	0 0 6 0 1
0 1 1 1 0	0 3 7 1 0

Adjacency matrix      Classification matrix

#### 4.4.3. Initialising the security broker

By constructing the adjacency matrix of the space and extracting the credentials into the classification matrix we have obtained the necessary information to construct the

access graph of the VE. This graph is used to reason about and manage access rights in the VE. We use the adjacency matrix to generate all paths associated with this environment applying the techniques outlined in chapter three. Once the broker has been initialised it can be interrogated to provide insights into the VE.

By following the method outlined in section 3.3.2 an array holding all valid paths for the space is constructed. First, the one-step paths are noted directly from the adjacency matrix. Next, this matrix is multiplied by itself and the main diagonal entries set to zero. Possible paths are identified by non-zero matrix entries, the paths being identified by considering the possible start and end steps which satisfy the matrix element indices and each other (i.e. the start node of the first step and the end node of the last step equal the  $i,j$  matrix element index pair, and the end node of the start step equals the start node of the end step). This identifies which paths have been generated by  $A^2$  which only shows the number of paths by itself. Hence all the two-step paths are stored, and there are no cyclic and re-entrant paths within the two-step paths.

The modified  $A^2$  is multiplied by  $A$  and the main diagonal set to zero. Possible three step paths are identified similarly with all valid start-steps and end-steps being initially identified. Next the one step paths formed between these start and end steps are compared against the valid one-step paths and any that do not match are discarded. Thus all the valid three step paths are identified. Since we were careful when we selected the start and end steps there can once more be no cycles or re-entrant elements.

At this point a pattern is emerging. By considering an  $m$ -step path as a start-step, an end-step and an  $(m-2)$ -step path in between, it is easy to generate all paths for the space. This is because the one-, two- and three-step paths have already been determined, and these paths contain no cycles or re-entries. If we are careful in the choice of start and end steps for each power of the adjacency matrix, we are guaranteed to generate paths free from cycles and re-entries. By generating paths in

ascending length, the results established two iterations previously are used to generate the next set of paths, and so on.

In this particular implementation the total number of nodes in a space is limited to 50, hence the maximum path length will be 49. This should allow reasonably complex environments to be considered without too great a performance hit on the machine being used. When this work was being undertaken a Silicon Graphics Indy workstation was the default platform. Obviously with O2 workstations available today computational power has increased tremendously and it should be possible to increase the number of nodes under consideration. However, for truly large-scale environments a network of communicating security brokers, each having their own management domains will be required. Such an extension is discussed in the further work section in chapter seven. For the purposes of this implementation a 50-node space should be more than sufficient to demonstrate the principles of the model.

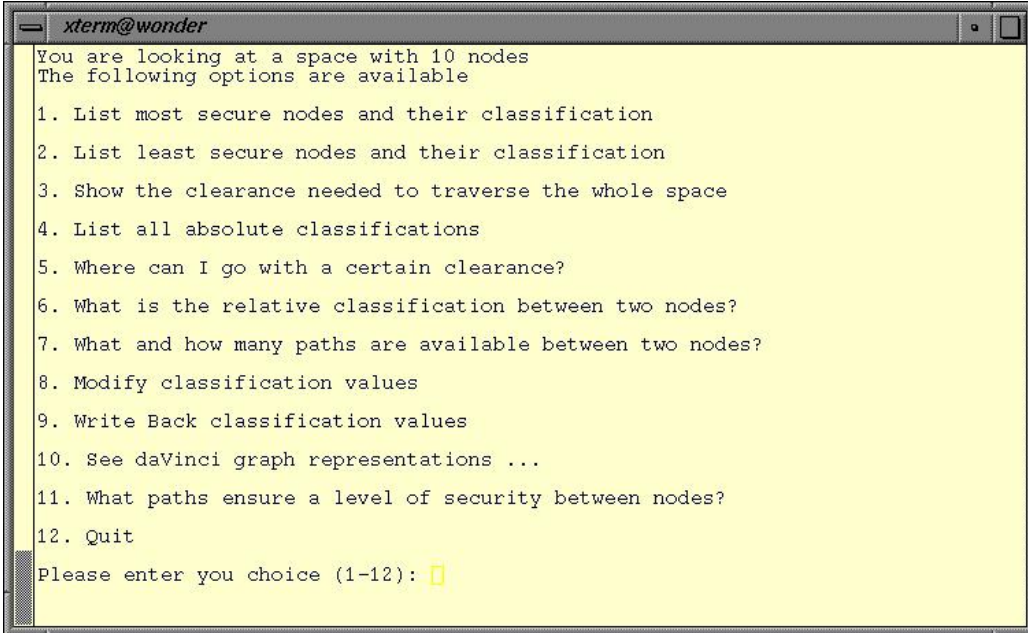
Once all the paths have been generated the next step is to associate a classification value with each path. This is done by considering each path in terms of its basic elements, that is a collection of successive one-step paths. Beginning at the start node of the path, the classification for each step is noted (these are simply elements from the classification matrix), noting the maximum value before finishing at the end step of the path. This maximum value is the classification of the path.

As a result of these calculations, the broker now holds the collection of the powers of the adjacency matrix (after the simplifications have been applied), all the paths which exist in the virtual universe, and the classifications associated with these paths. This information is all that is needed in order to check for the existence of any paths between two nodes, to identify these paths and to inform us of the classifications associated with the paths.

To end the presentation of the security broker details on how the access graph is queried and the results portrayed to the end user are now given.

#### 4.4.4. Querying the access graph

Two interface styles for viewing the information stored in the security broker have been developed, a simple text-based interface and an interface which uses the daVinci (Fröhlich 1995) 2D graph drawing package to display results graphically.



```
xterm@wonder
You are looking at a space with 10 nodes
The following options are available

1. List most secure nodes and their classification
2. List least secure nodes and their classification
3. Show the clearance needed to traverse the whole space
4. List all absolute classifications
5. Where can I go with a certain clearance?
6. What is the relative classification between two nodes?
7. What and how many paths are available between two nodes?
8. Modify classification values
9. Write Back classification values
10. See daVinci graph representations ...
11. What paths ensure a level of security between nodes?
12. Quit

Please enter your choice (1-12):
```

Figure 4.8: The text interface to the security broker

The text interface lists the questions from chapter three (section 3.2) and can be seen in figure 4.8. The answers to the questions appear at the bottom of the screen. The techniques described in section 3.4 are used to answer the questions. For example, to calculate the clearance required to move between two nodes (e.g. the auditorium and the speaker's platform) all possible paths between these nodes are examined in the broker and the classifications of each path compared with the others. The path or paths with the lowest classification provide the answer about the clearance needed to move between the two nodes. In this instance just the clearance required is returned, but it is also possible to request the paths which offer a route at this clearance. Once the answer has been noted the screen is refreshed and another selection can be made.

One of the selections lets the user view the access graph for this space through daVinci. There are three variations on the information viewable via the daVinci interface. First, the access graph for the entire space under examination is available. This can be seen in figure 4.9 where the graph of the space defined by the CyCo file in figure 4.7 is shown.

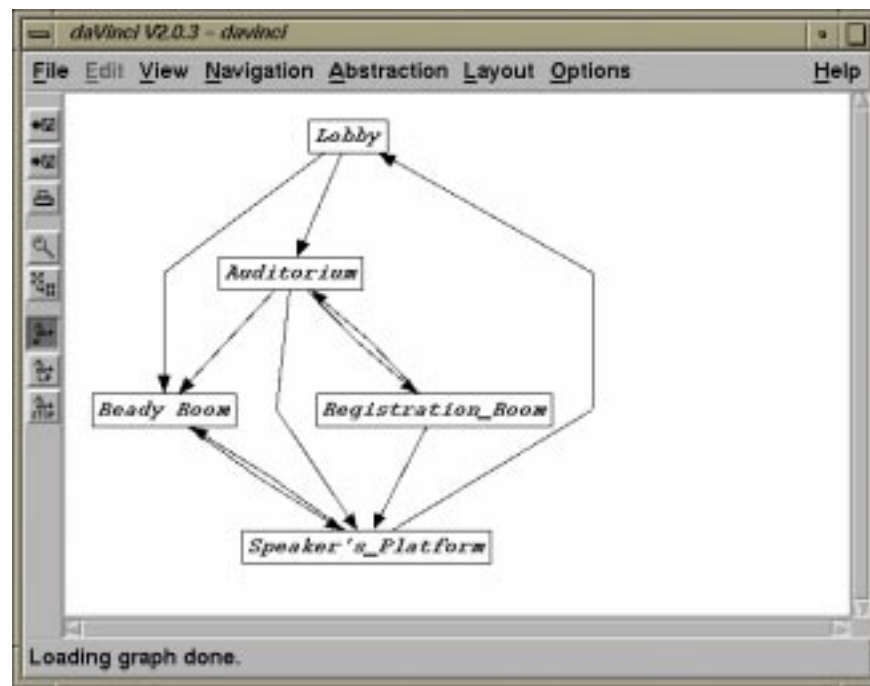


Figure 4.9: The daVinci graph of the information in figure 4.7

Second, a clearance value is presented and an access graph generated based on this given value. This graph only shows the nodes and edges available for the given clearance (figure 4.10). The third option shows the graph for the entire space once more. However, this time a clearance value is specified and this option shows links that can be crossed with solid edges and those that cannot with dashed edges (figure 4.11).



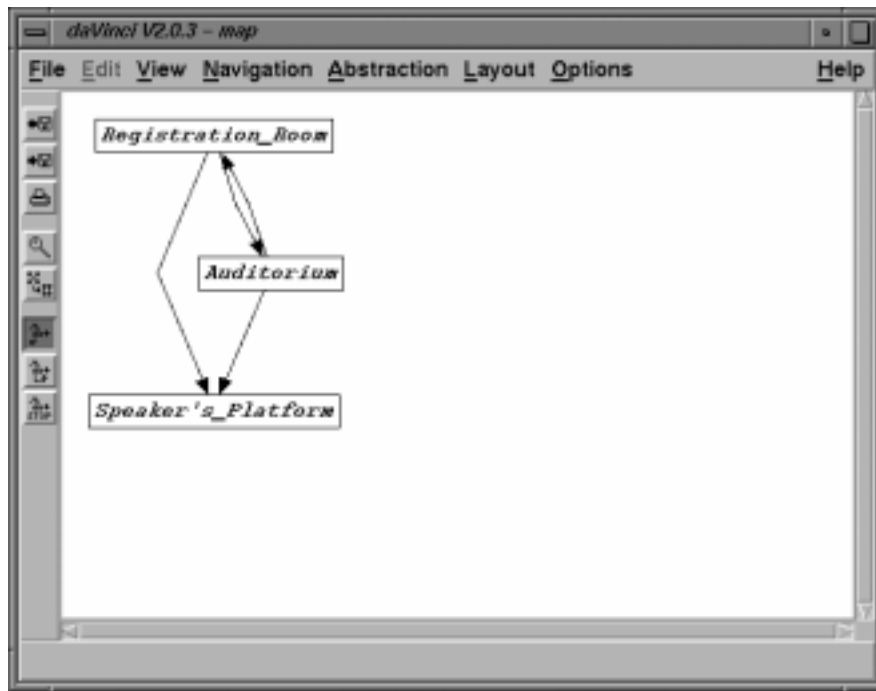


Figure 4.10: A subset of the access graph for a user with a clearance of 1

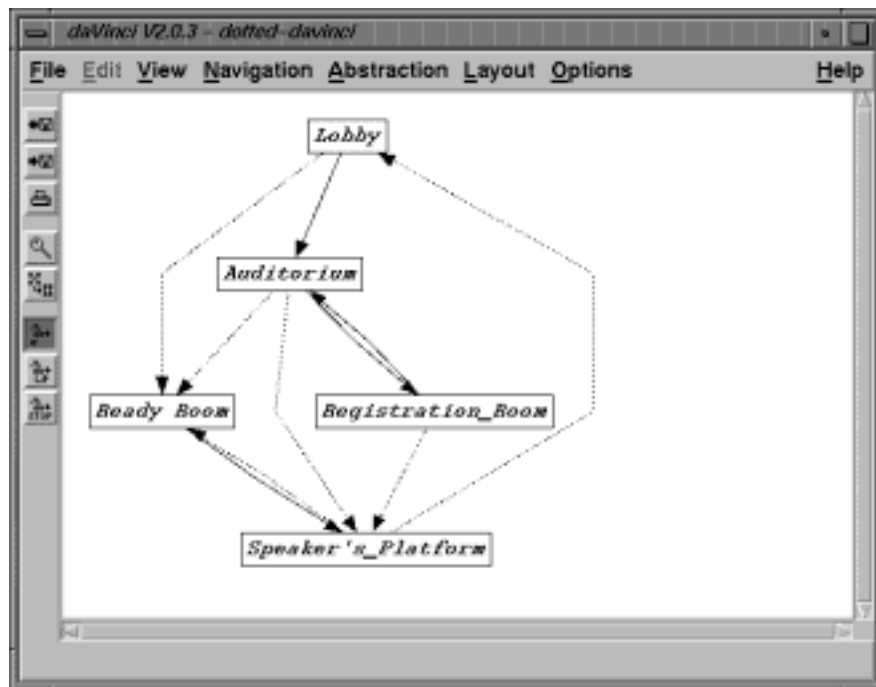


Figure 4.11: The whole access graph for a user with clearance 1

All input is done using integers, hence whenever a query is issued where start and/or end nodes need to be given the user is prompted with a list that maps world names to their unique IDs as stored in the broker. The results are passed through a similar process so that the end user is shown world names and not mystifying numbers.

Thus it is possible to produce personalised and tailorable access graphs for a space that not only list the access constraints in place but also act as navigation aids to help understand the composition of the space.

Finally in this section on the security broker we consider the possibility of modifying the values held by the security broker and writing these back as changes to the original Dive data files.

#### **4.4.5. Updating and writing back access information**

The security broker holds the access information within an array. It is possible to modify the values held within this array using option 8 from the text interface. Currently there are no restrictions on who can make modification, but it should be a relatively straightforward task to add on simple password protection to various interface options if desired. After every change the absolute classification of each node and the values held in the paths database are recalculated using the new values. The space can now be analysed using the available tools, using these new values.

When any changes to the access information held by the security broker have been agreed upon they can be applied to the virtual environment under consideration by being written back to the Dive data files. This is achieved by comparing the values held within the classification matrix inside the security broker against the values within the CyCo file that was produced at the beginning of the session (this CyCo file contains the original access information from the Dive environment). If there are any differences between the classification held in the CyCo file and those in the broker, then the CyCo file is updated. The values held in the CyCo file are then used to update the values held in the original Dive data files. Once these files have been updated their

values become effective in the virtual world once a state change has occurred for any updated worlds.

## 4.5. An example application of SPACE

Chapter three gave a brief example to illustrate the model in use (section 3.5). We will now consider this example using the implementation outlined above.

### 4.5.1. The environment

Figure 4.12 shows the environment under consideration, a simple office scenario. There are two conference rooms, a reception and a manager's office connected to a central corridor. There is also a door to the records office reception from the corridor, where various archives are kept. There are a number of different ways to gain access to the company records, some directly and others going via the records reception.

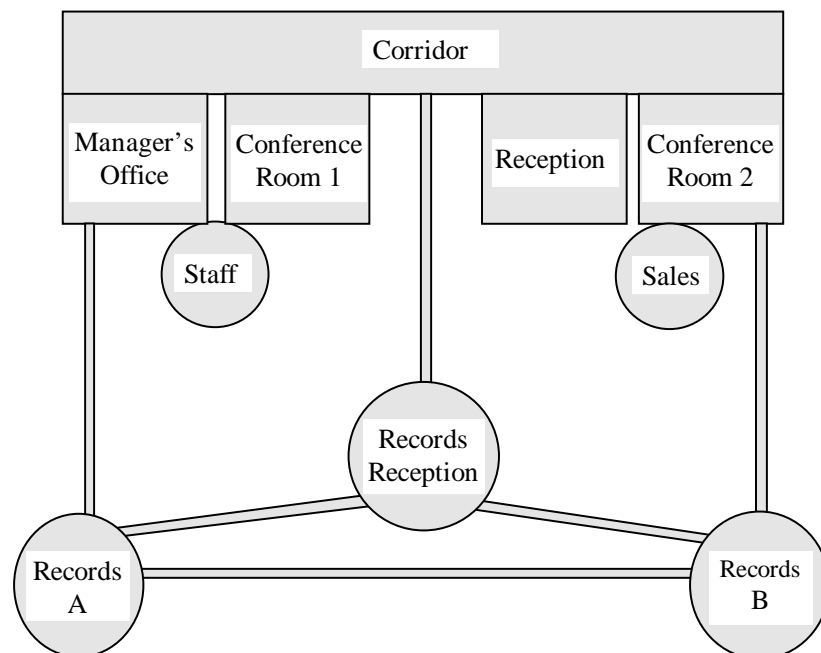


Figure 4.12: An example environment

A set of Dive worlds was constructed to represent this environment and some views of the environment can be seen in figures 4.13a to 4.13f.

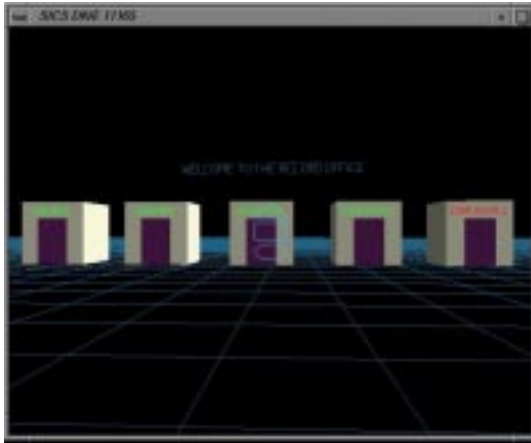


Figure 4.13a: The Corridor



Figure 4.13b: Conference Room 2



Figure 4.13c: The Reception

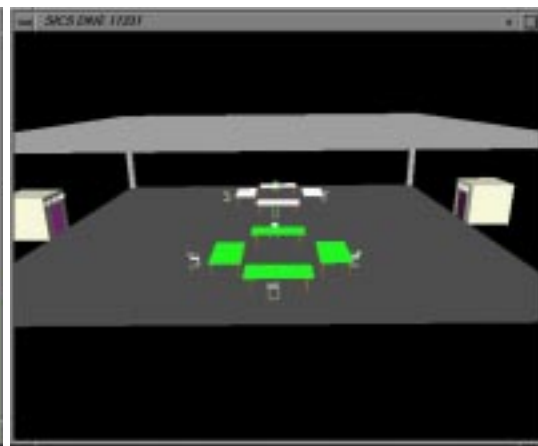


Figure 4.13d: Conference Room 1



Figure 4.13e: The Manager's Office



Figure 4.13f: Denied Access

Classifications were applied to the boundaries between the different regions in the environment by marking up the Dive data files accordingly. Thus we have a space that is ready to be examined.

### 4.5.2. Analysing the space

When the SPACE application is run the first thing prompted for is the location of the Dive data files (currently they have to be located in a single data directory) followed by a list of individual data files. This information is passed as arguments to a call to the main security broker program, which first extracts access information from data files into a CyCo file, and then proceeds to analyse this information. For simplicity in labelling any access graphs of this environment, a mapping from region names to letters is presented in table 4.1.

Manager's Office	A	Staff Records	F
Conference Room 1	B	Sales Figures	G
Corridor	C	General Records A	H
Reception	D	Records Office Reception	I
Conference Room 2	E	General Records B	J

Table 4.1: Region name to alphabetic label mapping

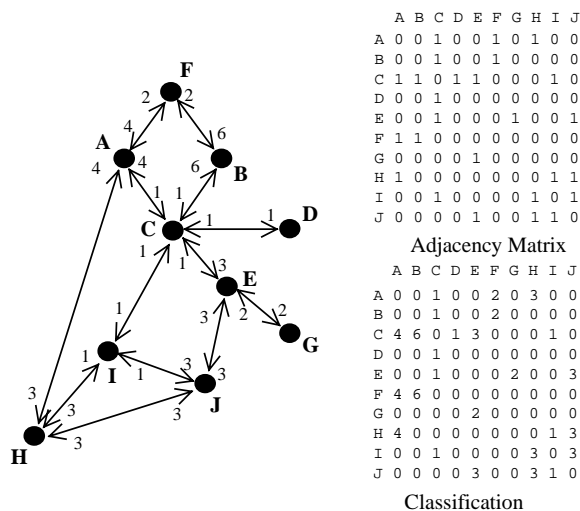


Figure 4.14: Access graph to adjacency matrix

Figure 4.14 is a hand drawn access graph and shows the environment with classifications for each boundary added as edge weights. The adjacency and classification matrices for this space can be seen to the right of figure 4.14. These have been defined through simple inspection of the access graph. The construction of these matrices is the first step carried out by the broker application. After the adjacency matrix has been noted, powers of the adjacency matrix up to one less than the number of nodes in the environment are calculated.

A	A <sup>2</sup>	A <sup>3</sup>	A <sup>4</sup>	A <sup>5</sup>
0010010100	0201100021	0020111113	0122401221	0212214223
0010010000	2001100010	0010011302	1001200243	2010312234
1101100010	0000021202	2100100222	2000211221	1100122011
0010000000	1100100010	0000021202	2100100222	2000211221
0010001001	1101000120	1010020311	4221010112	2312040221
1100000000	0020000100	1102200031	0010102315	1121401433
0000100000	0010000001	1101000120	1010020311	4221010112
1000000011	0020110011	1322301010	2222133013	2202241020
0010000101	2101200101	1020132101	2422111101	2312231201
0000100110	1020001110	3222110010	1312251310	3411132010

A <sup>6</sup>	A <sup>7</sup>	A <sup>8</sup>	A <sup>9</sup>
0101122211	0000111110	0000001000	0000000000
1021423211	0012214121	0001002000	0000000000
0200111011	0100101000	0000001000	0000000000
1100122011	0200111011	0100101000	0000001000
1411040011	1211010121	0001000000	0000000000
2212404242	1101104000	0000001000	0000000000
2312040221	1411040011	1211010121	0001000000
2200022002	1100100010	0000001000	0000000000
1111142001	1201201101	0000002000	0000000000
1111121210	0101101010	0000001000	0000000000

Figure 4.15: The nine powers of the adjacency matrix

In this case the all powers up to the ninth power of the adjacency matrix are found. These matrices can be seen in figure 4.15, after the simplification techniques have been applied. This information is then used to determine all possible paths in the environment together with their associated classifications. Once this information has been determined it is possible to answer the questions from chapter three.

Before each of the questions are gone through in turn, examples of the management interfaces (textual and graphical) are presented.

### 4.5.3 The management interfaces

Two forms of interface exist, a simple text based interface and a daVinci 2D graph interface. The text interface is not particularly exciting, looking much the same whatever space is being examined, and figure 4.8 adequately presents a view of this.

The daVinci interface however shows the structure of each individual space. Figure 4.16 shows the 2D graph of the current example.

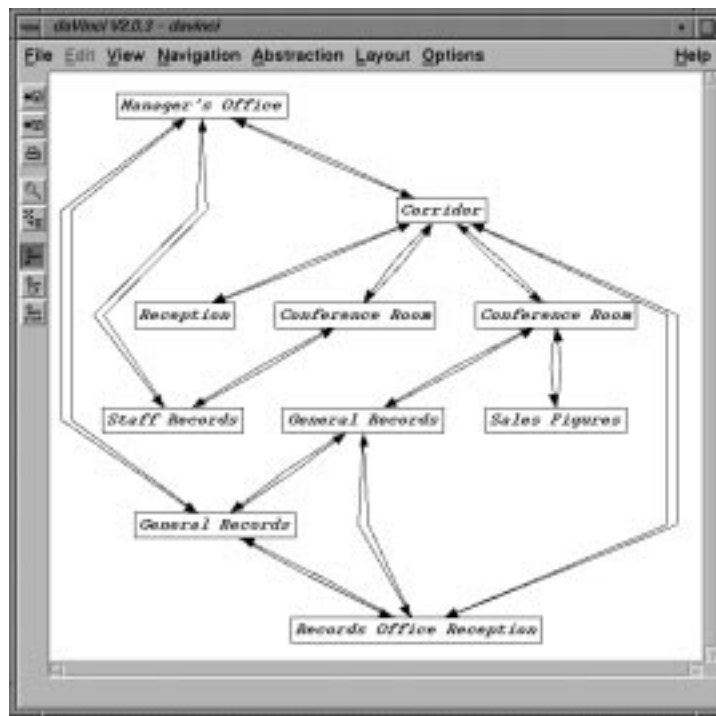


Figure 4.16: The access graph for the office environment

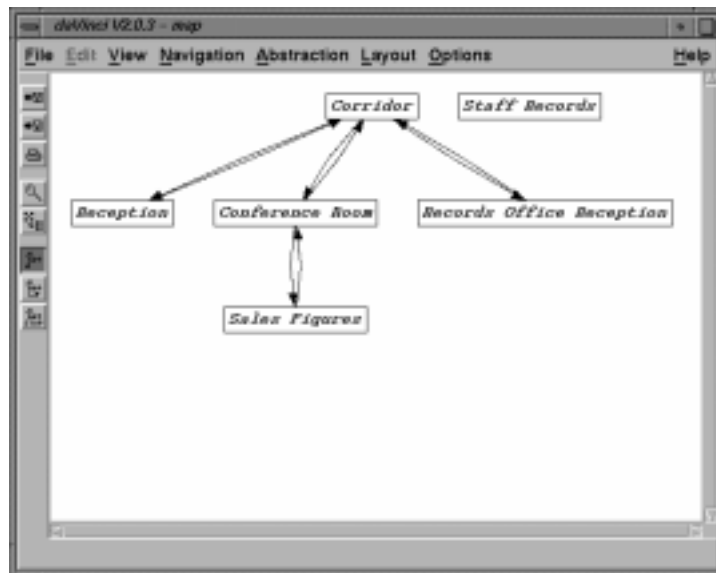


Figure 4.17: Subset of access graph – clearance is 2

There are two other views that can be adopted, both dependent on a clearance level being specified by the user. Figure 4.17 shows the regions in the office available to a user with clearance 2. The regions where access is denied have been removed from the graph representation.

Sometimes this view can be misleading though, as demonstrated by the Staff records node above. Clearly we can gain access to this node, but we do not know from where. This is where the third type of access graph available comes in. Figure 4.18 shows the full access graph for the space, but for this graph a clearance level was presented before it was generated. In this case the value was 2, and all links (boundaries) with classifications higher than this value are displayed with dotted lines.

We see from this full representation of the environment that it is possible to access Staff Records from the Manager's Office, but unfortunately with a clearance of only 2 we are unable to gain access to the Manager's Office in the first place. This explains why Staff Records appears as an isolated node in figure 4.17 above.

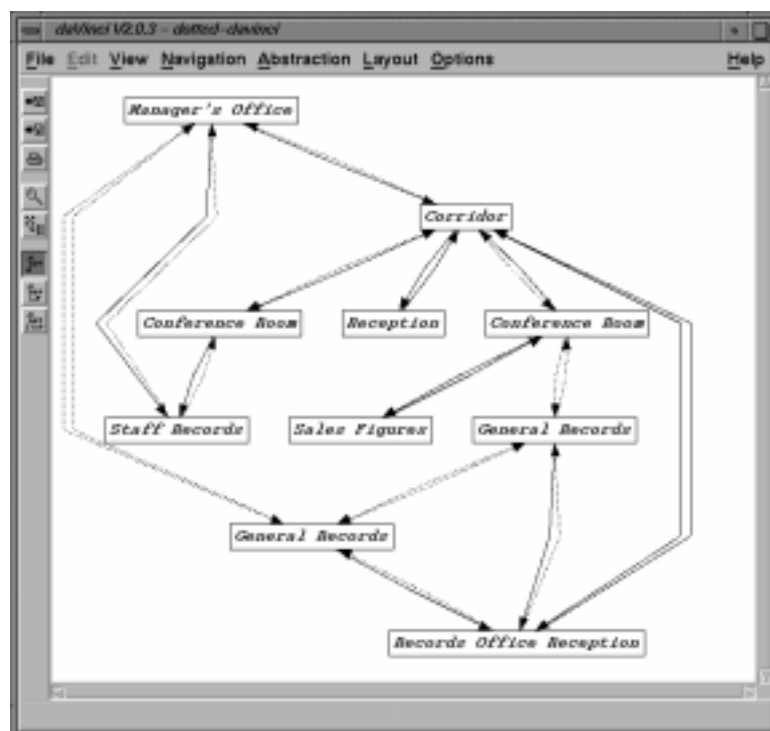


Figure 4.18: Access graph showing links above classification 2 as dotted lines



#### 4.5.4. Answering the questions

Chapter three identified a list of access related questions and gave general techniques for answering each question. Here we answer each question with the specific example above in mind.

##### **Can I go from the Reception to the Sales Figures Archive?**

Yes, use option 7 to find this out, entering the start and end nodes (The valid paths question below shows a full worked example of this).

##### **What credentials would I need to go from the Reception to the Sales Figures Archive?**

Use option 6, entering the two nodes when prompted.

To go from the reception to the Sales Figures Archive you need a clearance of 3.

##### **What are the valid paths from the Reception to the Sales Figures Archive for me?**

Use option 7 entering the start and end nodes.

There are seven valid paths from the Reception to the Sales Figures Archive.

- 1 path of length 3 exists between Reception and Sales Figures  
Reception to Corridor to Conference Room 2 to Sales Figures with classification 3.
- 1 path of length 5 exists between Reception and Sales Figures  
Reception to Corridor to Records Office Reception to General Records B to Conference Room 2 to Sales Figures with classification 3.
- 2 paths of length 6 exist between Reception and Sales Figures  
Reception to Corridor to Manager's Office to General Records A to General Records B to Conference Room 2 to Sales Figures with classification 4.  
Reception to Corridor to Records Office Reception to General Records A to General Records B to Conference Room 2 to Sales Figures with classification 3.
- 1 path of length 7 exists between Reception and Sales Figures  
Reception to Corridor to Manager's Office to General Records A to Records Office Reception to General Records B to Conference Room 2 to Sales Figures with classification 4.
- 1 path of length 8 exists between Reception and Sales Figures  
Reception to Corridor to Conference Room 1 to Staff Records to Manager's Office to General Records A to General Records B to Conference Room 2 to Sales Figures with classification 6.
- 1 path of length 9 exists between Reception and Sales Figures  
Reception to Corridor to Conference Room 1 to Staff Records to Manager's Office to General Records A to Records Office Reception to General Records B to Conference Room 2 to Sales Figures with classification 6.

**How secure is the Sales Figures Archive relative to the Reception?**

Use option 6 entering the start and end nodes.

The relative classification is 3.

**Which is the most secure region?**

Use option 1

The most secure region is Conference Room 1 with absolute classification 6.

**Which is the least secure region?**

Use option 2

The least secure regions are the Corridor, Reception and Records Office Reception with absolute classification 1.

**Where can the object currently located in Record Store A be moved to so that its level of security is not lessened?**

First we examine the absolute security of Record Store A and the other nodes in the space using option 4:

```
Absolute Classification of Manager's Office is 4
Absolute Classification of Conference Room 1 is 6
Absolute Classification of Corridor is 1
Absolute Classification of Reception is 1
Absolute Classification of Conference Room 2 is 2
Absolute Classification of Staff Records is 2
Absolute Classification of Sales Figures is 2
Absolute Classification of General Records A is 3
Absolute Classification of Records Office Reception is 1
Absolute Classification of General Records B is 3
```

We see that we can move the object to any of the following nodes: Manager's Office, Conference Room 1 or General Records B.

**What paths can be taken when moving this object, so that its level of security is not compromised during the move?**

First we identify all the paths between Record Store A and the three nodes identified above. Then each element of each of these paths has its classification compared to the classification of the object. The classification of every element in the path has to be greater than or equal to the object classification. We assume a classification of 2 for this object. By inspecting the results below we see that there are only 3 paths out of a

possible 26 over which we can move an object from Record Store A and be sure that its classification is never below 2. For all of these paths we must raise the object's credentials so it can traverse the paths. The paths are identified by selecting option 11.

There are 9 possible paths between General Records A and Manager's Office

- 1 path of length 1 exists between General Records A and Manager's Office  
General Records A to Manager's Office with classification 4
- 1 path of length 3 exists between General Records A and Manager's Office  
This path has elements of insufficient classification
- 2 paths of length 4 exist between General Records A and Manager's Office  
This path has elements of insufficient classification  
This path has elements of insufficient classification
- 2 paths of length 5 exist between General Records A and Manager's Office  
This path has elements of insufficient classification  
This path has elements of insufficient classification
- 2 paths of length 6 exist between General Records A and Manager's Office  
This path has elements of insufficient classification  
This path has elements of insufficient classification
- 1 path of length 7 exists between General Records A and Manager's Office  
This path has elements of insufficient classification

There are 10 possible paths between General Records A and Conference Room 1

- 3 paths of length 3 exist between General Records A and Conference Room 1  
This path has elements of insufficient classification  
General Records A to Manager's Office to Staff Records to Conference Room 1  
with classification 6.  
This path has elements of insufficient classification
- 2 paths of length 4 exist between General Records A and Conference Room 1  
This path has elements of insufficient classification  
This path has elements of insufficient classification
- 2 paths of length 5 exist between General Records A and Conference Room 1  
This path has elements of insufficient classification  
This path has elements of insufficient classification
- 2 paths of length 6 exist between General Records A and Conference Room 1  
This path has elements of insufficient classification  
This path has elements of insufficient classification
- 1 path of length 7 exists between General Records A and Conference Room 1  
This path has elements of insufficient classification

There are 7 possible paths between General Records A and General Records B

- 1 path of length 1 exists between General Records A and General Records B  
General Records A to General Records B with classification 3
- 1 path of length 2 exists between General Records A and General Records B  
This path has elements of insufficient classification
- 3 paths of length 4 exist between General Records A and General Records B  
This path has elements of insufficient classification  
This path has elements of insufficient classification  
This path has elements of insufficient classification
- 2 paths of length 6 exist between General Records A and General Records B  
This path has elements of insufficient classification  
This path has elements of insufficient classification

**If I have security credentials 2 where can I go and where am I unable to go?**

Use option 5.

With clearance 2 you can go to nodes:

- Corridor
- Reception
- Conference Room 2
- Staff Records
- Sales Figures
- Records Office Reception

With clearance 2 you cannot go to nodes:

- Manager's Office
- Conference Room 1
- General Records A
- General Records B

**What properties do I require to be able to traverse the whole environment?  
(i.e., to become a Super User)**

Use option 3

The clearance required is 6.

**What is the effect of changing a specific boundary's credentials on the rest of the space?**

We will lower the classification of the boundary between Record Store A and the Manager's Office to 1. This reduces the absolute classification of the Manager's Office to 1. The effect of this is that the Manager's Office becomes one of the least secure nodes in the space and it is possible to go there with a clearance of 2. The reduction in absolute classification means that this is no longer a valid location for our object from Record Store A. In fact, if we examine the possible secure paths to move our object along we are left with only one possible location for the object: Record Store B. The path to Conference Room 1 involved passing through the Manager's Office and this is no longer a secure region. To retain the old confidence in the space the Manager's Office classification must be restored, or else an alternative route must be put in place to enable objects to move between Record Store A and Conference Room 1.

**If I add a region and associated boundaries what properties do the new boundaries need to have to maintain confidence in the rest of the graph?**

A new node New Office is added having links to the Manager's Office, Corridor and Records Store A with classifications 1,1 and 3. The links the other way (from the three regions to the New Office) have classifications of 1,3 and 3 applied. After inspecting the new figures in the broker it becomes apparent that to maintain the previous security levels the boundary classifications between the New Office and the Manager's Office need upgrading to 4. This ensures the new possibilities for traversal introduced by the new node adhere to the existing security policy.

**What is the effect of removing a region (node) from the graph? Do certain conditions have to be met for this to occur?**

If we decide to remove Conference Room 2 from our example this directly affects access to the Sales Records and to Records Office B. In fact it no longer becomes possible to gain access to the Sales Records, an area that becomes an isolated node. A link to the Sales Records needs to be added to the environment from an appropriate location. We must also make sure that any paths which made use of the link between Records Office B and Conference Room 2 can be satisfied by alternatives. These additions should conform to the overall security policy for the environment and maintain confidence in the space. The Conference Room 2 node should only be removed once these additions are in place and working.

**4.6. A filter for MASSIVE-2****4.6.1. Introduction**

Simple clearance-classification access control was implemented in MASSIVE-2 towards the end of this research. This presented an ideal opportunity to demonstrate how a filter could be written to extract the access information from MASSIVE-2, convert it to the CyCo file format discussed above, and thus be able to examine MASSIVE-2 environments from an access standpoint using the security broker.

Revisiting Figure 4.1, this is a filter that sits between the management interface and the CVE.

#### **4.6.2. The filter**

Unlike the filter to extract access information from Dive that was written using shell tools, this time the filter was written in *C* (Kernighan 1988). The filter extracts access information from the MASSIVE-2 data files and writes a corresponding CyCo data file. The ability to update MASSIVE-2 data files on the fly from changes made by the security broker has not been implemented. The filter simply extracts access information and converts it into a form that the security broker can understand.

Two functions make up the *C* program. First the MASSIVE-2 data file is read. The position of the region is noted as a co-ordinate triple, followed by the extent of the region as another co-ordinate triple. Finally the classification required to enter this region is noted, together with the name of the region. It transpires that you effectively need minimal access credentials to leave any region (if you have insufficient credentials then you are automatically expelled from a region, and if you have sufficient credentials to enter you can leave at any time). As there is support for transition between worlds in MASSIVE-2 each application of the broker tool will consider a single environment file and the regions that are contained within that environment.

Each region is represented as a single node in the CyCo file, with one exit to the parent world. The parent world has a number of links equal to the number of regions in the environment. There is one drawback with the approach that has been adopted in writing this filter, and that concerns nested regions. To determine whether a region is nested in another requires careful examination of the position and extent of each region within the parent world. However, the only circumstance where we can potentially experience difficulties in defining the access controls for an environment is when a low security region is nested within a high security region. Whether this makes any sense in a practical sense is open to debate (unless you have the credentials to enter the outer region there is no way you are going to gain access to the region nested

within). When a high security region is nested within a low security region we still get an accurate overall picture of the access requirements for the environment, but we lack the information to know that the lower classified region must be traversed to enter the higher classification region. It should be possible to extend the filter written to date to compare the extents of each region identified to determine which regions are nested within which other regions.

#### 4.6.3. Examining the Panoptican Plaza environment from MASSIVE-2

We conclude this chapter by briefly examine the Panoptican Plaza environment from MASSIVE-2. This a simple environment consisting of three regions, and the parent plaza (see Figure 2.4 from chapter 2). Access constraints have been added to each region and the following access graph has been produced.

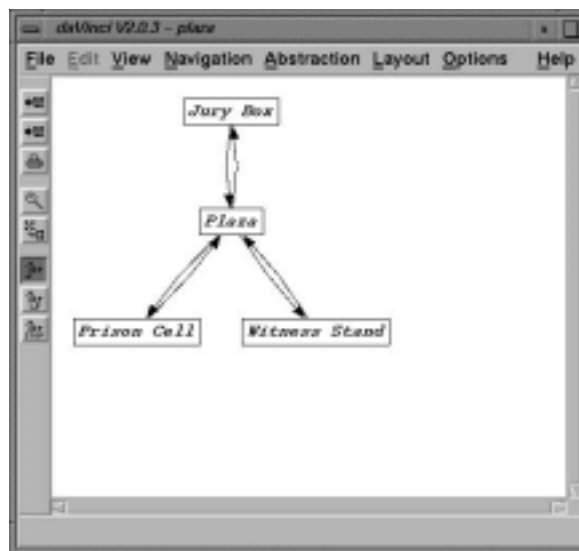


Figure 4.19: The simple access graph for the Panoptican Plaza

A simple but effective realisation of the access model and access management tool have been presented in this chapter. In the next chapter we consider a framework for applying the SPACE access model to other systems and areas. We begin by considering spatial systems, and then show how the model can also be applied to non-spatially motivated systems.

## Chapter Five

### **A framework to apply the model to different scenarios/applications**

Chapter three described SPACE, a model to apply, understand and manage access control in spatial systems, and an example of the model applied to a simple environment was given at the end of the chapter. Chapter four then described a prototype implementation of the model using the Dive virtual environment, and gave further details on the example from chapter three. This chapter discusses how SPACE can be more widely exploited to aid in the understanding and management of access control in general spatial systems and other application areas. We show that, given a suitable framework, SPACE can be applied to a number of different systems and areas (for example wideband virtual worlds such as VRML, battle simulation systems such as NPSNET or network routing scenarios). A “Cookbook” is presented which details how SPACE can be applied to these different spatial scenarios. It will also be demonstrated that SPACE is not only relevant or useful to 3D spatial systems, as was originally intended when developing the model, but that there are a wide range of systems that are not motivated by a 3D spatial metaphor to which the model may be applied. We examine this possibility by examining Rodden’s (1996) awareness model. To end the chapter three worked examples are given, a mapping of the model to the



Spline system (Barrus 1996), a mapping to the 2D CSCW system Teamrooms (Roseman 1996), concluding with a mapping to the awareness model of Rodden.

Chapter five is structured as follows:

- Section 5.1 discusses the applicability of the SPACE model to other areas.
- Section 5.2 specifies a “Cookbook”, a step by step guide for applying the SPACE model to other spatially-based systems.
- Section 5.3 presents an example use of the cookbook as the model is applied to the Spline system.
- Section 5.4 presents another example use of the cookbook as the model is applied to Teamrooms.
- Section 5.5 considers the applicability of SPACE to systems without a strong 3D spatial metaphor. In particular it considers how the model can be applied to Rodden’s work on awareness and graph structures.
- Finally section 5.6 summarises the chapter.

## **5.1. The applicability of SPACE to other areas**

The SPACE model was devised with spatially motivated CVEs as its intended application area. However, it has become clear that while the model is suited to such environments, it may also be possible to make use of the model in other application areas. These areas could be either spatially motivated, for example other CVEs, or even non-spatial systems. This chapter considers how the model can be applied to these different application areas. However, before the model is applied to other areas it seems sensible to ask, and answer, the question “Why should the model be applied to other areas?”

### 5.1.1 Why should the model be applied to other areas?

The SPACE access model is built upon two key foundation ideas. The first of these is that a space or environment can be partitioned into regions through the use of boundaries. The second is that we can use an access graph to represent this structuring, thus enabling the environment to be understood and managed with the help of standard mathematical techniques. It is the ability to easily understand and manage possibly complex structures which makes it interesting to consider using this approach in other application areas.

There are a number of candidate application areas where the model may prove useful. Simulation of real world environments is one area that could prove fruitful, with examples as diverse as building security and network routing to name but two. Both of these could be complex environments where the extent of any changes may be difficult to fully evaluate. The SPACE model would make these evaluations easier and highlight potential problem areas. Indeed, one possible use of SPACE could be to incorporate it into architectural design tools, giving practical feedback about proposed designs in a simple manner.

The model also offers the possibility of a common management interface onto a number of different systems. The specification of access rights is application dependent and done on a per application basis, whereas the management facilities of the model can be shared across a number of applications, so long as the access information from a particular application can be presented to the security broker in a standard form. For the prototype implementation described in the previous chapter this format is the CyCo file format. As long as access information from an application can be converted into CyCo format the same management interface can be used. This simplifies the specification and understanding of access rights and also provides the added bonus of being able to compare and contrast the access rights across different systems.

Another advantage of applying the model to other systems is that it can be used in conjunction with any existing security policies that may be in place. Hence this model

could be used to provide an overview of the access constraints at a high level while specific access models could be used to provide more detailed restrictions and fine grained controls.

It appears the ability to apply the model to other systems is beneficial. In section 5.2 we consider how the model can be applied to other systems, and a step-by-step cookbook is defined.

## **5.2. A cookbook for applying SPACE to other spatial systems**

As mentioned above, there are two components to the SPACE model, (1) applying access control to an environment using boundaries and (2) understanding and managing the access requirements of the environment. Figure 4.1 in the previous chapter showed the modular approach adopted for the implementation of the model. To apply the model to other systems we need only concern ourselves with the top half of this diagram, that is point (1) above, extracting the access information from the environment and converting it into the necessary generic file format. If the CyCo file format is chosen then we are able to re-use the security broker and management interfaces introduced in chapter four. Alternatively this management functionality can be re-implemented using the guidelines in chapter three.

For simplicity's sake we will assume that the security broker and management interfaces introduced in chapter four will be re-used and that the CyCo file format will act as the generic access information file format. We now go through the steps which will be required to add access controls to a spatial system and ensure that a CyCo file can be produced from the access credentials.

### **5.2.1. Examining the applications**

#### **5.2.1.1. Identify boundaries**

The first step to take when applying the model to an application is to determine how the spatial structuring is performed in the application, i.e. identify the boundaries.

There are many different approaches to structuring the virtual environment. The Dive system is partitioned into individual worlds, with interconnecting gateways identifying which worlds are connected to which others. The worlds and gateways are described in data files, one for each world. Hence the boundary is the gateway. NPSNET tiles its world into hexagonal cells and maps collections of cells known as “Areas of Interest” onto multicast groups. An Area of Interest Manager (AOIM) is used to partition the world into these smaller, more manageable components. The boundary is the extent of the multicast group represented by each hexagon. In RING, a client server system, the shared VE is partitioned into a spatial subdivision of cells based on visual occlusion, these cells are then used to inform servers of the position of entities from boundary traversals. The boundary is the extent of the cell. Once the structuring method has been identified, i.e. the boundaries have been identified, we can consider how access rights are going to be associated with these boundaries. However, we must first examine any existing security mechanisms in place in the system.

#### **5.2.1.2. Existing security mechanisms**

The SPACE model should complement any existing policy and not compromise or conflict with it. In an ideal world the existing security credentials can be mapped onto a hierarchy of classifications, and these classifications are then used at the boundaries to enforce access. For example, this may entail mapping existing security credentials onto a hierarchical enumerated type, as previously described in section 3.1.3. Alternatively a new hierarchy of access rights may be needed, one which maps existing object based access rights into spatially governed access rights. Fine grained access rights, for example those pertaining to collaborative editing, should coexist with the high level rights, each set of rights being independently defined. The most important thing to avoid is compromising the existing security policy when the SPACE model is introduced.

At this stage we have identified how the world structuring is performed and whether there is a security policy in force for the application. The next section describes how new access information can be added to the application.

## **5.2.2. Adding access information**

The definition of the boundary in the specific application will need extending to introduce a classification attribute. It will also be necessary to extend the definition of a person or entity that interacts with the system to incorporate a clearance value in their definition. Classifications and clearances should be defined in such a way as to be comparable, yielding a pass or fail answer.

### **5.2.2.1. Extending the boundary**

Having identified how boundaries are defined in the previous section, we now examine this definition to determine how it can be extended to incorporate a classification attribute. There is no generic answer to how access classifications can be added to boundaries, they will be defined on a per application basis depending on the boundary definition. Typically one might expect to extend the schema definition of a boundary and incorporate the new attribute (“Classification”) in all uses of the boundary in the system. For examples see section 4.3 which included an example of extending boundaries in Dive and section 5.3.2 which will consider boundaries in Spline. An alternative approach is possible if we consider the RING system. Here it is likely that the classification values associated with a cell’s boundary would be controlled by the servers which control messaging, the functionality in the server being triggered when an entity tries to traverse the cell boundary.

Typically boundaries will not be symmetrical, the access credentials needed to traverse the boundary in one direction being different to those needed to traverse it in the other.

### **5.2.2.2. Defining user clearance**

Clearance values are also subject to differing definitions depending on the specific application. The two most obvious places where these values are defined are in embodiment configuration or personal profile files in a user–specified location (typically their home directory), or in an embodiment configuration file defined by the

system and stored centrally, a more likely general solution as this permits centralised control of access rights by those authorised to do so. It is assumed that well known integrity and authentication techniques will be used to ensure that this clearance information is only modifiable by those authorised to do so. It should be possible to adopt such techniques without too large an overhead, but it should be noted that there is the possibility that significant changes may be required to an application to ensure sufficient integrity and authentication mechanisms are in place.

Once more, examples of clearance defined in Dive can be seen in section 4.3 and in Spline in section 5.3.2.

#### **5.2.2.3. Comparing classification and clearance values**

The final addition we need to address is functionality to compare clearance values against boundary classifications. This is the mechanism that enforces access control, and should be added to the functionality that controls boundary traversal, be it a gateway, a state change or something else. It should be possible to identify a function or set of functions that control this change and place a control statement around it based on the outcome of the clearance–classification comparison.

These three extensions allow access control to be enabled in an environment using the SPACE model. We must also specify how this access information can be extracted from the system and converted into a format which can be used by the management tools. The next section considers how the newly added access information can be extracted and converted in the appropriate file format.

#### **5.2.3. Extracting access information**

The format of the access information in the application is known from the earlier extensions to the boundary. A filter can therefore be constructed that extracts the access information from the world definition format for the application for a pair of regions, a source or originator region and a target or destination region. By considering each region in turn, each of its links to another region can be identified

and the classification associated with the boundary noted. This is done for every region in the environment. This provides the base information from which a CyCo file can be constructed. The tool to extract this information can be implemented in a number of ways. The example in Chapter Four uses the standard UNIX shell, *sh*, together with the *sed* and *awk* tools. Any high-level programming language should suffice.

To complement the data extraction tool a second tool is also required to write back the access information from the CyCo files into the application world definition format. This enables any changes made in the management interface to be written back to the application data files.

#### **5.2.4. Using the management tool**

The management functionality introduced in chapter four can be reused if the CyCo file format has been adopted as the intermediate file format. Alternatively, this functionality can be re-implemented using the guidelines in chapter three.

#### **5.2.5. Summary**

The above discussion can be summarised into the following eight-point plan:

1. Identify how boundaries are defined in the application.
2. Check for existing security mechanisms.
3. Determine a security policy (ideally hierarchical) and map existing access rights/policy onto this policy.
4. Extend the boundary definition to include a classification value.
5. Extend the user definition to include a clearance value.
6. Extend the boundary traversal mechanism to include a comparison of classification and credential values – i.e. enable access control.

7. Write a conversion program to extract the classification information from the application and convert it into the generic file format (CyCo in this case).
8. Write a program to write back changes from the generic (CyCo) file format to the application boundary classification values, however they are held in the application.

A generic outline of how to apply the SPACE model to spatially motivated applications has been presented above. To aid in the comprehension of this outline two worked examples are given below. These show how the model might be applied to a system motivated by a 3D spatial metaphor, the Spline system, and a system that allows the spatial metaphor to be applied to non-3D spaces, Rodden's awareness model for cooperative applications.

### **5.3. Applying SPACE to the Spline system – a worked example**

Spline is a virtual environment whose aim is to support large, detailed multi-user virtual environments:

“There is a natural desire to make multi-user virtual environments both detailed and large – detailed in the sense that there are interesting things to see if you look close and large in three senses: in spatial extent, in numbers of objects, and in numbers of users interacting with the environment. However, doing this brings up many problems: ... Locales are an efficient method for solving all of these problems by breaking up a virtual world into compact chunks that can be described and communicated independently”

(Barrus 1996)

The fact that every object must be in exactly one locale, and Spline's use of locales makes it an ideal candidate for an application of the SPACE access model. We begin by taking a close look at how Spline works.



### **5.3.1. Examining Spline**

Spline uses locales to divide a virtual world into compact regions that can be processed separately. A locale consists of (Barrus 1996):

1. A unique ID.
2. A distinct multicast address.
3. A binary space partitioning (BSP) tree (Naylor 1990) describing the boundary of the locale.
4. Neighbours – a list of structures describing the relationship between the current locale and its neighbours.

Locales have been implemented in the Spline software platform, which in turn has been used to implement a demonstrator application called Diamond Park. In our examination of Spline we need to identify two things; how boundaries are defined and operate and if there are any existing security mechanisms in place.

#### **5.3.1.1. Identifying boundaries within Spline**

Boundaries are simply the limits of each locale in Spline, and are specified as a binary space partitioning (BSP) tree. The boundaries need not be regular geometric shapes, indeed one of the strengths of locales is that they can have arbitrary geometric definitions. The boundary scopes the extent of a particular locale. What is on the other side of the boundary is governed by the list of neighbours held by the locale, and the BSP tree that is used to determine to which other locale an object should be transferred once it has left the current locale.

To summarise, the boundary is defined by the “Boundary” attribute of a locale, and the worlds linked to are defined by the “Neighbours” structures.

#### **5.3.1.2. Existing security mechanisms**

There appear to be no explicit security mechanisms in place in the Spline system. Anybody is able to go to any of the locales from any of the other locales. Like many

other systems, Spline has been built with the intention of facilitating interaction between many users in large scale virtual environments, but as yet no attention seems to have been paid to security aspects, as these do not help overcome the immediate problems. Fortunately we are able to apply the SPACE model to provide protection between locales.

### **5.3.2. Adding the access information**

Before we add access information to the boundary we must first identify a security policy. Keeping things as simple as possible, the integers 1 through 9 will be used to represent nine corresponding, hierarchical, classification and clearance values (the same as in chapter four). We now extend the boundary to include access information.

#### **5.3.2.1. Extending the boundary**

We know that boundaries are defined in Spline by BSP trees (Naylor 1990), a technique which removes the topological information from the spatial partitioning and results in simpler algorithms. So, given an arbitrary locale, we know the extent of the boundary of the current locale, defined by the BSP tree, and we also know which other locales are connected to our current one through the list of neighbours. We need to extend the definition of the current locale, to include in each structure in the Neighbours list a classification value that represents the clearance required to move from the current locale into that neighbouring locale. This attribute should be called “Neighbour Classification”.

#### **5.3.2.2. Defining user clearance**

We also need to add an attribute to the definition of the user in Spline to enable clearance values to be specified. This should entail adding a new variable, CLEARANCE, to the definition of the user avatar, and setting it appropriately. This variable will hold simple integer values between 1 and 9 inclusive.

### 5.3.2.3. Comparing classification and clearance values

The final addition is to enforce the checking of credentials at boundaries. Fortunately Spline provides a function that will check whether a “thing” has left the boundary of its locale and will transfer it to the appropriate neighbour  $N$  if necessary. All we need do is extend this function to perform a comparison between the clearance attribute of the user and the classification attribute of the boundary being traversed. If the comparison is successful normal traversal should occur. If not the user should remain in the initial locale, no transfer taking place.

### 5.3.3. Extracting the access information

The access information can be extracted to produce a CyCo file by considering each locale in turn. The first line in the CyCo file is the total number of locales. Then each locale is examined in turn. We assign the first locale examined the ID of 1, and a file is created holding the mapping from the locale ID  $L$  to this CyCo ID  $N$ . The neighbours of  $L$  are identified and added to the mapping file in the order they occur, so  $N1$  maps to CyCo ID 2,  $N2$  to 3 and so on. The integer values are used for internal references within the CyCo file, and the Spline Ids are specified in the name field of each CyCo record. The number of neighbours for the locale is determined and output. This is followed by the CyCo Ids for each neighbour on the next line. Finally the classification values associated with the link to each neighbour, read from the extended Neighbours definition, are output. This is done for every locale in the environment, adding to the mapping file when new locales are encountered, using the existing mappings for locales we have already seen.

### 5.3.4. Using the management tool

The CyCo file can be examined using the daVinci and text interfaces introduced in Chapter four. This will show us the form and structure of the locales and permit us to reason about the newly defined access rights and simulate changes. The fact that the CyCo intermediary file format has been used means that we can simulate access

requirements from one system in another, for example it is now possible to adopt a set of access requirements from Dive into Spline.

We have outlined the application of SPACE to the Spline system, demonstrating the ease of which the model can be applied to other systems and the powerful consequences of providing a common management interface through the CyCo file format. We end this chapter by considering an application of the model so that the advantages of the spatial metaphor can be applied to other applications which are not based on a 3D spatial metaphor.

## **5.4. Applying SPACE to TeamRooms**

TeamRooms is a 2D CSCW application that supports groups of people who need to work closely together, but who may be geographically disparate.

“TeamRooms is a groupware environment based on the metaphor of shared virtual rooms. The system contains user-defined rooms, each with a shared whiteboard, chat tool and customisable groupware applets.”

(Roseman 1997)

### **5.4.1. Examining TeamRooms**

TeamRooms provide shared network spaces for people to meet and work together. Each environment in TeamRooms is represented by a process on a central server, and consists of an arbitrary number of rooms, which can be occupied by pre-defined team members. The rooms provide a persistent workspace, and contain a number of tools to aid the group working process. Multiple teams are supported using different ports on the central server.

#### **5.4.1.1. Identifying boundaries within TeamRooms**

Within a team environment, each room represents a distinct bounded space. Each team member must be located in one room within the environment. The act of

entering a room represents crossing the boundary surrounding that room. In a single TeamRooms environment, all of the rooms are accessible from each of the other rooms. The rooms are fully connected.

If we consider a server supporting multiple teams, there are two potential overview topologies. The first is a star topology, where any movement between teams involves passing through the central server and providing the appropriate authentication information. An alternative approach could be to allow direct connections between teams. In such a diagram all nodes would be independently accessible and the links between nodes would show the potential allowable movements between teams. In either case appropriate security information must be presented to allow a user to undertake a team member role for that particular TeamRooms environment.

#### **5.4.1.2. Existing security mechanisms**

In order to join a TeamRooms environment a user must first authenticate himself with the central server. Once this authentication process has taken place, through a username/password pair, the user is able to visit any of the rooms within that TeamRooms environment. The applets with each particular room may provide their own security services, but there is no underlying security model in place for the room structure within each TeamRooms environment. Users can examine, modify and delete any information contained in any of the rooms for that team, and can even delete rooms from the environment if they wish.

#### **5.4.2. Adding the access information**

In this section we present a mapping of SPACE to control access within a single TeamRooms environment. We do not concern ourselves with inter-environment accesses, assuming that the authentication service is sufficient to control individual access to teams. We adopt the same security policy as was used in the mapping to Spline, i.e. integers in the range 1 to 9 define a set of security credentials.

#### **5.4.2.1. Extending the boundary**

The boundary around each room is crossed when the room is entered. To extend the boundary we define a classification attribute that holds the minimal value required for accessing the room. This attribute is added to the existing definition of the room that details the applications that exist in the room.

#### **5.4.2.2. Defining user clearance**

In a personal configuration file for each user we need to define a simple numerical attribute representing user clearance.

#### **5.4.2.3. Comparing classification and clearance values**

We locate the function that places users in a room within the TeamRooms environment. This function is then extended to add an access function that can be evaluated to true or false when presented with a user's credentials. For the adopted policy, users are allowed to traverse the boundary if their credentials are greater than or equal to the access rights held at the boundary.

#### **5.4.3. Extracting the access information**

We know that every room within an environment is connected to every other room. We just need to read the classification attributes associated with each room and fill these value into a CyCo template file that has been constructed for the environment (such a file can be simply constructed from the number of rooms in the environment).

#### **5.4.4. Using the management tool**

We apply the security broker to the CyCo file and use the existing daVinci and text interfaces to examine the resulting information.

## **5.5. Applying SPACE to other systems**

We have applied SPACE to both 3D and 2D systems sharing a spatial motivation, i.e. systems using spatial properties to structure and to lay out their environments. This is a limited set of applications, and in this section we aim to demonstrate a mapping that allows a greater variety and scope of applications to make use of SPACE. We do this by considering Rodden's awareness model for cooperative applications (Rodden 1996).

The awareness model was developed to show that the concepts of presence, sharing and awareness (central to the spatial model of interaction (Benford 1993) and spatial systems) could be applied to applications lacking an explicit spatial metaphor. Rodden successfully demonstrates this, and identifies a wide range of CSCW applications that can be reasoned about using these spatial concepts. If we can demonstrate a mapping from the SPACE model to this awareness model we argue that this enables us to apply, understand and manage access control across the range of CSCW applications already identified.

### **5.5.1. Examining Rodden's Awareness model**

The awareness model of Rodden has been developed to exploit the way in which cooperative applications are shared and is a general model for reasoning about shared applications. As well as presenting a general model, where the "space" for activity is defined as a pool of objects shared by a community of users, Rodden also presents two specialisations of the model. The first specialisation considers how the model can exploit the geometric properties of a shared spatial frame. The second considers the application of the general model to shared graph structures. This second specialisation is particularly interesting for us. Not only does this mapping open up a wide range of CSCW applications that can have graph structures applied to them, it also ties in with the use of the access graph in SPACE. Any application that can be represented as a shared graph can be examined and compared to similar applications. Thus the ability

to compare access rights between 3D virtual spaces and general CSCW applications (e.g. a general multi-user hypertext system) becomes possible.

In the following subsections we first consider the most general form of Rodden's awareness model, and then later on the mapping to non-spatial domains through shared graphs.

#### **5.5.1.1. Identifying boundaries**

The general awareness model represents space as a collection of objects shared by a number of users. Unlike the previous examples we have considered, there does not have to be an explicit spatial frame – the objects may constitute the space themselves. Hence the task of identifying boundaries is not as straightforward as before. Rather than being objects in their own right, boundaries may be thought of as the relationships between objects in the “space”.

Consequently we define access rights in this model using awareness levels between objects. In order for one object to access the other some awareness threshold must be passed. Meeting this awareness requirement can be considered the same as traversing a boundary. Hence boundaries are mapped onto awareness thresholds. In order to understand this further we need to examine how awareness values are calculated and controlled in Rodden's model.

#### **5.5.1.2. Presence Positions, Nimbus and Focus**

The awareness model uses set theory notation to define and represent the spatial concepts it examines. Here we will summarise the key concepts of the model. The collection of users and objects associated with a space constitute the shared space. Each user or object can have a number of presence positions associated with them, this being their position and a collection of adjacent objects. The presence position maps users to objects within the space. The set of all presence positions is known as the presence space.



*Nimbus* in a spatial sense is a subspace in which an object makes some aspect of itself available to others. The awareness model uses nimbus to map users to the shared space through a mapping to potential presence positions in the space. A user may have many different nimbi. *Focus* in a spatial sense is a subspace in which an object receives information from other objects, i.e. the objects it pays attention to. The awareness model uses focus to map users to a position point in the space. Again a user can have many different foci. Thus focus and nimbus have been represented in terms of objects of interest and the definition of the space no longer relies on spatial arrangement.

### **5.5.1.3. Awareness**

We can now consider the role of awareness in all this. Awareness is a composite value, made up of a combination of focus and nimbus values. For example, to be fully aware of an object you must be within its nimbus and be directing your focus towards it. Rodden considers both continuous and discrete forms of awareness. He notes that the form of awareness being considered concerns the likelihood of actions by one user being noticed by another. This fits a potential mapping to SPACE well, using awareness concepts to control whether a user is able, rather than likely, to notice the actions of another (or even the ability to perform an action on another object).

We will limit ourselves to discrete awareness values, as this permits us to build a credential hierarchy that can be used in comparisons. Rodden identifies sixteen different awareness modes (p.90 Rodden 1996), the awareness between objects being determined by the relationships between focus and nimbus for the objects. Ten of these modes concern the placement of users in others' focus and nimbus; the other six are where focus and/or nimbus overlap.

We define a hierarchical set of access rights by mapping each of the awareness states identified above onto an access classification (e.g. an integer). Here we are using the awareness and potential awareness between objects and users to control access. In effect this identifies a security policy, a statement of how access can be managed in an application without giving details on how the policy may be implemented.

#### **5.5.1.4. Mapping to non-spatial domains**

So far we have considered the general awareness model. We now move on to examine the mapping of the awareness model to mathematical graph structures, a mapping that opens up non-spatial domains and their associated applications to us. Rodden achieves this through the use of graphs and directed graphs, considering how the spatial model concepts can be represented using these techniques. This allows a number of different application areas to be considered, rather than adopting a per-application approach and providing a different mapping for each.

In chapter three we introduced the necessary graph theory terminology for this discussion. The objects that constitute the application can be expressed as a graph structure. The nodes of the graph represent the objects while the arcs of the graph represent the relationships between the objects. These relationships concern the focus and nimbus values between objects, and Rodden makes use of the graph structuring to define focus and nimbus in a graph. He further defines two functions to aid in the construction and interrogation of adjacency sets of objects. A linked function identifies if objects are linked to a location. A sphere function returns all objects that lie within a specified distance from the location object. These functions make use of distance and adjacency properties in graphs to determine relevant sets of objects, and allow focus and nimbus to be more conveniently defined for graph structures.

#### **5.5.1.5. Existing security mechanisms**

The awareness model has been developed without any explicit security mechanisms in place.

#### **5.5.2. Controlling access**

The previous applications of the SPACE model have involved the addition of explicit access credentials to boundaries. Here we define access control according to the passing of an awareness threshold between two users or objects, rather than a

comparison of attributes between objects. To add access information we need some way of controlling potential awareness between users and other objects in the space.

Specifically we must be able to calculate discrete awareness values for any given pair of objects, and we must be able to determine whether any sensitive information is being made available when it should not be. To achieve this we need to consider not only the awareness value between a pair of objects, but also the individual focus and nimbus values used in the awareness calculations.

#### **5.5.2.1. Calculating awareness values**

In this section a subject is the person or object performing the accessing and the target is the subject or object being accessed. The subject has full control over his focus and can express many levels of interest across many different targets in an application. The subject is always able to try to gain access to a target. However, the target can determine how much it wants its presence to be felt in the application through its nimbus. Hence the definition of access rights is a composite value, and cannot be calculated by considering objects in isolation.

Awareness depends on the nimbus of the target and the focus of the subject. We can define two values for each potential access, the relative awareness and the absolute awareness. Previously, in section 3.1.4, the concepts of relative and absolute classification were introduced. Here we use an analogous definition to define corresponding awareness concepts. The relative awareness between a pair of objects is simply the current, instantaneous awareness value based on the target's nimbus and the subject's focus. Further, we define the absolute awareness value to be the special case where the subject has maximal focus while the target has minimal nimbus. This is the case where the target is making as little of itself available to the environment while the subject is trying its hardest to see what there is in the environment.

To determine what kind of access is possible between a pair of given objects we just map the discrete awareness level identified between the objects to the policy we identified in section 5.4.1.3. To do this we must identify a mapping between the

awareness modes and access rights. The range of access rights identified by this mapping runs from having no access at all to, and hence no awareness of an object, through to having access to all objects, hence full awareness of everything.

We conclude our examination of Rodden's awareness model by considering how the flow of information may be controlled within a shared graph structure. We use the concepts of focus, nimbus and awareness to determine who can do what and what information is available to whom.

#### **5.5.2.2. The controls associated with objects and subjects**

We are able to determine awareness values between subjects and objects, mapping the resulting awareness value onto an access right. This is achieved through a power sharing act, and there is potential for information to be made available to unauthorised users if they are able to utilise a suitably large focus. We need to employ a complementary mechanism alongside the awareness mapping to ensure that levels of access control can be maintained. If this mechanism can ensure a guaranteed level of security then all the better.

Currently the awareness value, and hence access credentials between a pair of objects, is determined through focus and nimbus calculations. To ensure that too much power does not end up with the subject (the focus part of the calculation), we may introduce an attribute associated with the focus and nimbus of an object. We can tag a nimbus with a particular value that ensures only objects whose focus has a tag of corresponding or greater value can be involved in awareness calculations with the object with the tagged nimbus.

This provides a mechanism for controlling the flow of information through awareness calculations, and it becomes possible to guarantee the minimum access credentials associated with objects. These tags may be applied to the edges of the shared graph representing the application. The graph shows the objects and the potential relationships between them, and the edges provide the extra detail to prevent unwanted information drainage.

It should be noted that the management of access rights requires controls over how subjects and objects are able to change their focus and nimbus values.

## **5.6. Summary**

This chapter has presented a “Cookbook” for applying the SPACE access model to other systems. After discussing the applicability of SPACE to other areas, the steps required to apply the model to a generic system were outlined, giving short examples from a number of CVEs of the different steps. To demonstrate how this can be done we first considered the Spline CVE, following the “cookbook” and discussing how Spline could be extended to support access control using SPACE. Following this we applied SPACE to the 2D CSCW system Teamrooms. Finally we considered the application of SPACE to an awareness model for cooperative applications. This mapping opened up the possibility of applying the SPACE model to applications not motivated by a 3D spatial metaphor.

The SPACE access model, an example implementation of the model and a framework for applying this model to other areas have been presented in chapters three, four and five. In the next chapter, chapter six, an evaluation of the model will be presented.

## Chapter Six

### Assessment of SPACE

Previous chapters have presented an access model for collaborative virtual environments, its implementation and a framework to apply the model to other application areas. In this chapter an assessment of the model is carried out, comparing this approach to other access models. This chapter aims to provide justification as to why the model has been proposed as a solution applicable to CVEs in particular and across a wider range of spatially-based applications in general. The chapter ends by considering the limitations of SPACE, examining the weaknesses of its approach and suggesting alternative functionality not offered by SPACE, but which is available in other approaches.

Chapter six is structured as follows:

- Section 6.1 examines the ethos behind assessment, i.e. what we are trying to show in assessing the model.
- Section 6.2 considers the process of security evaluation, the formal process through which security related mechanisms are tried and tested, and then sets out the criteria to be used for our assessment.

- Section 6.3 examines the SPACE model with respect to the assessment criteria from chapter two and the requirements for access control from chapter one.
- Section 6.4 compares the space model against other access models, commenting on its overall effectiveness and suitability.
- Finally section 6.5 considers some limitations of the SPACE model.

## 6.1. Introduction

In chapter two we examined a number of different access models. There were certain commonalities between the models, but also many differences that set them apart. In what follows we show how the SPACE model compares to these other access models, looking at the conclusions from chapter two and commenting on the overall appropriateness of the SPACE model for controlling access in CVEs.

To demonstrate that the SPACE model fulfils at least some of the criteria and requirements presented at the beginning of this thesis, three propositions are used to guide the comparisons with other access models (Mariani 1997). In this assessment we aim to show that:

- *at worst* the Space model is *at least as good as* existing models
- *at best* it is *better than* existing models
- it captures aspects of different models in an integrated way

We continue by first examining the area of security evaluation, and then we identify the criteria on which to base our assessment of our access model against the other contenders by revisiting the review criteria from chapter two. Next we present the results of the comparisons, bearing in mind the three propositions above. This process allows us to finally make comments on the appropriateness and successfulness of SPACE. We finish the assessment by comparing the results gained for SPACE against

those of the other access models. To draw the chapter to a conclusion we finally examine some of the limitations of the SPACE approach to access control.

## 6.2. Security evaluation and assessment criteria

### 6.2.1. What is security evaluation?

The following definitions come from the UK IT Security Evaluation and Certification Scheme (UKSP01 1991):

“**Computer security evaluation:** the detailed examination of a system or product to search for vulnerabilities and to determine the extent to which its security target is met by its implementation.”

“**Target of evaluation (TOE):** an IT system or product which is subjected to security evaluation”

“**Security target:** a specification of the security required of a TOE, used as the baseline for evaluation. It will specify the security functions of the TOE and may also specify the security objectives, the threats to those objectives and the particular security mechanisms that will be employed.”

The American Orange book (NCSC 1985), a policy document produced by the National Computer Security Center (NCSC), defines four criteria for system evaluation; Security Policy, Accountability, Assurance and Documentation. The relationships between these four criteria are shown in Figure 6.1.

*Security Policy* concerns the protection of information through access control mechanisms, both discretionary and mandatory, and security labels.

*Accountability* considers the issues of identification and authentication – who someone is and proof that they are indeed that person, audit to record security related events and trusted path to ensure only correct programs are used.



*Assurance* ensures that the system being evaluated was built within correct practices and functions properly, basically that all the system “building blocks” are safe and valid.

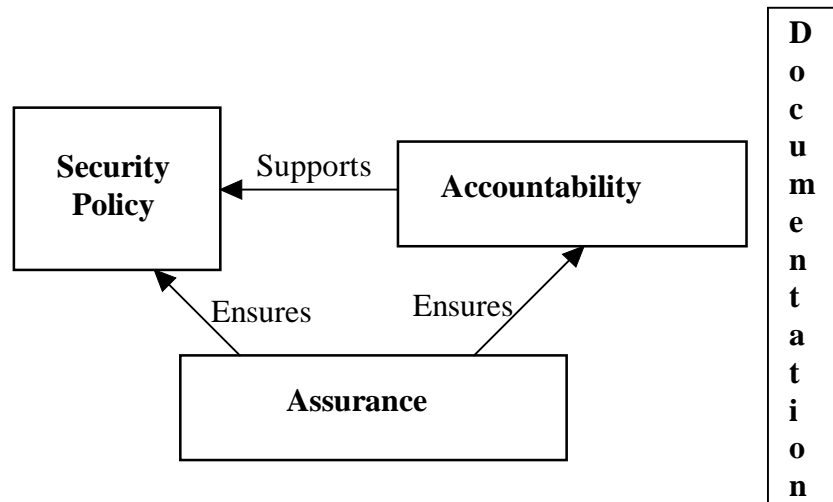


Figure 6.1: An overview of the Trusted Computer System Evaluation Criteria (TCSEC)

*Documentation* is required for developers, maintainers, users, operators and administrators.

Chokhani (1992) overviews the TCSEC in detail if further information is required.

We are not concerned with evaluating a particular product, rather a technique that can be usefully employed by a number of systems or products. Security policy will be the main factor when evaluating the access model. Documentation is covered to some extent by the security broker tool, which provides an interface to the access information which should be easy to use and understand. The other aspects are assumed to be in place, i.e. those relating to accountability and assurance, provided by appropriate external tools. We are only concerned with evaluating the access model.

So, to be able to evaluate our access model, we require a security policy that can be used across a range of access models, the results being compared against each other. This policy can be expressed as a set of assessment criteria.

### **6.2.2. Assessment criteria**

We can use the review criteria presented in chapter two to provide an initial assessment of the SPACE access model. This gives us a good starting point, and also provides the possibility for a comparison of SPACE against the models from chapter two. This does not give us the full picture though, as our mechanism must also be effective for CVEs. We also need to consider the requirements for access models for CVEs that appeared in chapter one. Four requirements for an access model were postulated there:

- The mechanism must be simple.
- The mechanism must be unobtrusive to users – there should be no extra overhead imposed on users to ensure secure over insecure operation.
- It should be easy to inspect and change access rights.
- Above all, the effect of access controls should be understood, and the consequences of any changes should be clear.

By examining SPACE using the review criteria, and then comparing the results of this review against the requirements listed above, we should be able to comment on the effectiveness and appropriateness of the model, thus assessing its performance. We begin by revisiting the review criteria from chapter two.

#### **6.2.2.1. Simplicity**

By simplicity we mean how easy it is to specify, examine or manipulate access rights. This concerns the day-to-day use of the information represented by the model and presented to the end user. Typically the simpler things are, then the more useful and used they become.

**6.2.2.2. Complexity**

Complexity concerns the nature of the underlying access model. It does not necessarily follow that a complex model will be difficult to use, but there is certainly more scope for misfortune with overly complex models. Complexity should be minimised, though not at the loss of functionality.

**6.2.2.3. Understandability**

The understandability of a model dictates how obvious any consequences of different access rights combinations or changes are. If you can easily get an overview, or “*The Big Picture*” then a model is likely to be easily understood. A complex model with many twists and turns will be less easily understood. For a model to be effectively applied a full understanding of it is required.

**6.2.2.4. Applicability**

We also need to consider the applicability of the model to the task at hand. We need to be sure that the model and access rights make sense, and an appropriate level of security with reasonable costs is being employed.

**6.2.2.5. DAC or MAC**

We should consider whether the access model uses Discretionary or Mandatory access control. DAC is an ownership based approach where the access rights are transferable at a user’s discretion. MAC stops the transfer of rights by enforcing a clearance–classification approach where the rules must be obeyed. DAC is open to Trojan horse attacks and user problems whereas MAC relies on the system to provide integrity.

**6.2.2.6. Collaboration**

We need to consider what support for collaboration each of the models offers. Many of the models will have no collaboration support, being developed in the formative years of computing, but some of the later models incorporate collaborative techniques, having been designed for a collaborative setting.

### 6.3. Examining SPACE

The SPACE access model adopts a spatial approach to access control for collaborative virtual environments, using boundaries to segment space into regions, and representing the space being examined as a mathematical access graph. This graph allows an understanding of access possibilities in the space and introduces the ability to manage possibly complex structures and spaces. Chapter three presented a full exposition of the model. Before we examine the model we will take a little time to reflect on the nature of this assessment. By examining the SPACE model against our own review criteria, a set of criteria that motivated the research in the first place, one would expect the SPACE model to come out favourably. We will show, at the end of this chapter, that there exist weaknesses and limitations to our approach, in a general sense. However, for the focused research area this work covers, that of spatially motivated CVEs, the assessment aims to demonstrate that the SPACE model is highly relevant, appropriate and powerful mechanism for controlling access.

We now assess the SPACE model in the light of the five criteria listed above:

***Simplicity:*** The specification of access rights for SPACE is very simple with users being associated with clearance rights that determine what they can and cannot do by comparison with an object's classification rights. These comparisons are Boolean and yield a pass/fail result. Any access rights that can be ranked hierarchically and compared against each other are suitable candidates. The easiest specification of access rights involves simple integers. Manipulating and examining access rights involves just replacing one hierarchical access right with another.

***Complexity:*** The SPACE model is not a complex one. The environment is presented to end users as a map, the construction of which is done automatically by the system. The security broker provides an easy to use interface, removing much of the complexity from the end user.

***Understandability:*** The access graph representation of the environment under study is one of the major advantages of this approach. A clear, intuitive presentation of access

information is possible. While the realised security broker from chapter four is limited in functionality, there is much scope to provide extensions to further enhance a user's comprehension of access rights for an environment. Chapter seven, in section 7.2.1. examines some of these issues. Also the ability to view the consequences of changes to the environment resulting from different access rights *before* the changes have been applied is another of the advantages of this approach. The user is able to experiment and gain a full understanding of the consequences of any actions before they happen.

***Applicability:*** The SPACE model is applicable any system that uses spatial partitioning to structure itself. This encompasses many of the current CVE systems, but other areas might include network routing (where there are pieces of hardware and connections between them), and chapter five has shown how the model can be applied to systems without a 3D spatial motivation.

***Discretionary or Mandatory access control:*** The SPACE model can be used with either a DAC or MAC approach to access control. The realisation of the model in chapter four uses MAC based access rights, being based on the BLP access model. Generally, SPACE will utilise other access models to enforce the checking of access rights at boundaries whilst keeping its spatial framework to provide an overview of the environment and the ability to understand and manage access rights. Hence it is tied to neither approach, making it possible to use the most appropriate control mechanism for the task at hand.

***Support for collaboration:*** The SPACE model can implicitly support group access rights, something which is touched upon in chapter seven in section 7.2.2. When a group of users tries to traverse a boundary then some aggregate representation of the group might be presented to the boundary. The boundary can then compare the presented aggregation against a number of group access functions it holds. If a suitable value has been presented then the group are able to traverse the boundary, else they are prevented from doing so. The model does not provide support for fine-grained collaborative actions, rather it relies on employing suitably powerful access mechanisms, in a complementary manner, to provide this functionality.

We have examined SPACE with respect to the review criteria from chapter two. We now consider how it stands up to the requirements from chapter one. Hopefully we have shown that SPACE is a simple, effective access control model. By placing the constraints on access at boundaries we have been able to take advantage of people's natural spatial reasoning skills, with parallels to the real world. People are familiar with concepts such as locked doors, so being prompted for access rights when entering a region is natural. Also, with suitable organisation, most of the access checks can be hidden from end users, only making them aware of the security mechanisms when insufficient privileges are presented. This way access control does not interfere with normal working practices, and is unobtrusive. Given the appropriate permissions, it is easy to inspect and modify access rights (the model is not concerned with how access rights are specified or held, rather that they exist and can be used to allow or deny access). Finally, through the security broker tool, it is possible to gain a full understanding of the access constraints within the environment being protected by SPACE, and simulations of changes can be performed to assess their impact before they are applied.

To summarise, SPACE is a simple access model, backed up by an easy to use security broker tool that permits a full understanding of the access rights associated with a given environment. We now compare SPACE to the access models from chapter two.

#### **6.4. Comparing SPACE with other access models**

Having examined SPACE in isolation, we now consider it with respect to the four access models examined in chapter two. Here we hope to demonstrate that SPACE is a useful addition to the area of CVEs by considering the three propositions from section 6.1 above. Before we do this though, a reminder of the access models from chapter two is given in Table 6.1. Table 6.2 then summarises the discussions on each of the models for the review criteria. Again, these comparisons are from the standpoint of the focused area of CVE research, and are not generally applicable access approaches across a wide-ranging set of applications.

	Model	Aims
1	Access Matrix (Lampson 1974)	Controls access to objects via a 2D matrix which defines permissible actions
2	Bell/LaPadula (Bell 1973)	Clearance–Classification model developed with a military scenario in mind
3	Shen/Dewan (Shen 1992)	Collaborative access model based on a generalised editing model of collaboration
4	BSCW (Sikkel 1997)	General authorization model that emphasises conceptual simplicity
5	SPACE (Bullock 97)	Spatial Access Control for Collaborative Virtual Environments.

Table 6.1: Review of the access models from chapter two

Criterion \ Model	1	2	3	4	5
Simplicity	High	Medium	Low	Medium	High
Complexity	Low	Medium	High	Medium	Low
Understandability	High	Medium	Low	Medium	High
Applicability	Wide	National Security	Collaborative Editing	BSCW	CVEs
DAC or MAC	DAC	MAC	DAC	DAC	Either
Collaboration	No	No	Yes	Yes	Yes

Table 6.2: Comparison of the access models

We comment on the success of the SPACE model in the light of Table 6.2 and the three propositions from section 6.1.

**At worst the SPACE model is at least as good as existing models**

The effectiveness of the SPACE model depends on the specification of the access rights and the ability to partition up an environment into regions for analysis. If the same complex access rights are used across the range of models, then it will become harder to understand the interactions between the access rights in SPACE. However, this will also be true of the other models too. If you are going to use a complex scheme to specify access rights then you cannot avoid introducing complexity into the access control mechanism. If there is much fine grained access this means the environment has to be partitioned into smaller and smaller regions. This could result in overly complex access graphs. However, it should be possible to consider subgraphs and present the results to users in as clear a manner as possible. This avoids the complexity involved in resolving fine-grained access rights present in other approaches (e.g. Shen–Dewan model).

**At best it is better than existing models**

The main advantages of SPACE are (1) the adoption of a spatial partitioning model which ties in with people’s natural spatial reasoning skills, (2) the provision of a security broker tool that permits the easy understanding and management (including simulations) of possibly complex environments and (3) the representation of CVEs as access graphs which present easy to understand, concrete overviews of possible access within the environment.

The other models include easy to use but inflexible approaches (classic BLP model), fully flexible but highly complex access models (Shen–Dewan collaborative access model), or else the model has been developed for a different setting than CVEs, and the techniques it employs are not appropriate for collaboration (the Access Matrix model). SPACE offers a simple, understandable approach that is applicable to a wide



number of applications including CVEs, and provides a highly functional management interface.

### **It captures aspects of different models in an integrated way**

One of the powerful properties of the SPACE model is that it does not have to be used in isolation. You can use the SPACE model to control high level, inter-domain access in an environment, whilst adopting a different access model within each domain. These models would be highly dependent upon the activity taking place in each region, and the ability to specify models as and when needed offers great flexibility. As such, access models whose strength is controlling fine-grained accesses could be used for those tasks, while the SPACE model holds everything together and explains the big picture to the user.

## **6.5. Limitations of the SPACE access model**

We have examined the SPACE access model from the point of view of CVEs, and have shown it to be particularly suited to that area. However, the model is not necessarily suited to all circumstances, and there will be situations where applying the SPACE model either does not make sense or does not provide the controls required for that particular environment or application. In this closing section we will consider some limitations of the SPACE access model.

### **6.5.1. The requirement of a spatial metaphor**

The SPACE model has been developed for systems that explicitly use a spatial metaphor to partition the environment under consideration. Therefore to make use of the model in non-spatial systems it is necessary to try and produce a mapping of the elements of the non-spatial system onto a spatial layout. It may not be easy, or indeed possible to do this. In such circumstances the access matrix model offers a widely applicable access model to control interactions that include objects and users or entities acting upon those objects. This lacks the sophistication and management interface of the SPACE approach, but does permit access rights to be defined as

required for objects and subjects using either ACLs or capabilities for example. To address non-spatial systems with the SPACE model a filter is needed that converts the traditional access information into a form recognisable by the security broker application. There is unlikely to be a general solution to this problem across a wide area of systems and applications. So for now we suggest that only systems that can be represented with a spatial mapping of some sort are examined using SPACE, and the other systems are controlled using the more traditional approaches based on the access matrix model.

### **6.5.2. The applicability of SPACE**

In the previous section we have stated that only systems using a spatial approach or systems that can be mapped onto a spatial layout are suitable for the SPACE model. This is a limited set of systems and applications, and the security and access requirements of the remaining areas still need addressing by mechanisms of some description. However, even for spatially motivated systems there are circumstances where the SPACE model experiences difficulty. Applications that involve very large numbers of objects with fine-grained access to these objects will be difficult to model using SPACE. Our approach relies on the ability to partition environments up into distinct regions. The following example demonstrates the problem. If we consider a library, there are many books, organised by subject classification. These classifications are then further subdivided by a string to identify the author. One can imagine laying out the library contents spatially. If we are to place access constraints on each individual author, we need not only to divide the library up into subject areas (and there are many of those) but also to divide the subject areas into authors. If we wish to control access to individual books then we must go one level further still. A great effort is needed to identify boundaries and regions in order to isolate each author/book. It is not clear that spatially dividing a library layout is a sensible approach. Rather a scheme that permits individual access rights to be applied on a per book and/or author basis, and then the books/authors can be examined as and when necessary. More traditional object-based approaches are suited to this, the standard Bell-LaPadula model being an ideal candidate. So, not only do we need to find

applications that adopt a spatial metaphor of some description to be able to apply SPACE, the applications must also be appropriate.

### **6.5.3. Support for specific applications**

The SPACE model has been proposed as a high level, general access model for CVEs. This is fine for controlling general accesses to resources, but there will be circumstances where specific actions and tasks need to be supported by the access model. For example, the BLP model was developed with the task of securing sensitive military information and the Shen–Dewan model is concerned with collaborative editing. These models have been designed with particular applications or tasks in mind, and so support these applications well with much flexibility. The SPACE model has been developed with a general framework, rather than a particular application in mind, and so the gains it makes from being widely applicable play off against the losses when application–specific functionality is required. As mentioned previously, it is possible to use other approaches alongside SPACE, but SPACE should not be considered a complete, standalone solution.

### **6.5.4. Sensitive application areas: a lack of formalism**

While simplicity is a strength of the SPACE approach, it may also be considered a potential weakness, in the sense that it becomes very difficult to demonstrate that an environment is provably secure. This may involve the long, drawn–out task of manually or semi–automatically inspecting a possibly complex environment to ensure there is confidence in each of the regions in the environment. Other approaches, such as the BLP model, ensure that each state is secure through the definition of the model. There is nothing to stop users misapplying the SPACE model and creating insecure regions and the potential security hazards that go with them. For situations where the level of security provided is paramount then it is better to adopt more complex but provably secure approaches than those provided by the SPACE model.

### 6.5.5. Vulnerabilities

We finally consider some potential weaknesses of the model that do not necessarily arise from our specification of the model; rather they concern the overall environment within which the model is set. A major weakness that affects the general applicability of the model to a wide range of applications is the reliance on systems developers to provide an appropriate infrastructure to which SPACE can be applied. We have seen earlier that a compile driven approach to parsing information has been adopted because current applications do not separate topological and semantic information. An interpretive approach would be a much better solution. However, as current applications do not provide an infrastructure to support this we are limited by other peoples' assumptions. More acutely, if it is not possible to represent an application in terms of regions and boundaries, or map its structure to such concepts, then we are unable to apply the SPACE model to it at all. We need an appropriate infrastructure to exploit.

Apart from the need for an infrastructure that permits an application to be examined, controlled and managed, there is also the requirement to safeguard against more traditional threats. We must ensure that users are who they say they are, i.e. authentication, and physical threats such as eavesdropping on the network need to be protected against. Here we need to employ complementary, external services that have been developed to explicitly overcome these problems. For example the Kerberos service (Neuman 1994, Bryant 1988) could be used to authenticate users before they can access the management interface. The networks that are used should be secure, and if we cannot guarantee the integrity of our connections, then network traffic should be encrypted so that if network snooping is taking place, then the results of the snooping are useless.

However, there are trade-offs that we must consider when trying to make our environment as secure as possible, as each action has an associated cost. One of the strengths of the SPACE model is its ease of use. If we introduce more and more external functionality there is a risk that our approach becomes too cumbersome to

use. We need to employ services that are as transparent as the SPACE model. To some extent we are limited by the network infrastructure in place. The network environment needs to provide authentication, secure wires and encryption services that are easily employed and do not make it more difficult to use the services they protect.

We have shown that the SPACE model represents a simple, uncomplicated, easily understood approach for controlling access in CVEs. We have also highlighted a number of limitations of this approach. We draw this thesis to a close by summarising the research goals and contribution of this work, considering areas for further work and present some concluding remarks in chapter seven.

## Chapter Seven

### Further work and conclusions

This thesis has developed an access model for collaborative virtual environments based on a spatial approach by attaching credentials to boundaries that separate or partition virtual worlds. In order to access an object, users must first be able to make a journey (at least logically) from their current location to that of the object, crossing any intervening boundaries. A key aspect of the model is the way in which the access properties of a given virtual environment can be represented as an access graph. The application of standard techniques to this graph allows a range of access related questions to be answered and supports the understanding and management of access rights. The approach has been implemented as an extension to the gateway mechanism within the Dive system and as a separate security broker application to support access management within Dive. This chapter examines the research goals attained within this thesis, considers the contribution made to the fields of collaborative virtual environments and access control, presents a number of extensions and enhancements which could be applied to the SPACE model and its realisation, and finally ends with some concluding remarks.

Chapter seven is structured as follows:

- Section 7.1 gives a summary of the goals of the research undertaken in this thesis, considering the contribution made to the fields of collaborative virtual environments and access control
- Section 7.2 presents a number of extensions that could be made to both the SPACE model and its realisation in Dive described in chapter four.
- Finally section 7.3 gives some concluding remarks.

## 7.1. Summary of research goals and contribution made

At the beginning of this thesis the overall goal of the work was given:

“This thesis develops a spatial approach for enabling, understanding and managing access control in collaborative virtual environments. Access control is governed according to the space within which subjects and objects reside and whether a subject can traverse this space in order to get close to an object. The two main themes that have been running through this thesis are:

- i) a spatial approach to access control using boundaries to partition space.
- ii) the use of an access graph, a mathematical representation of a structured virtual space, that allows spatial access controls to be understood and managed.”

We now consider how successful this work has been in addressing these issues by looking at the research achievements and contributions made to the areas of Access Models and CVEs.

An access model, based on the notion of partitioning virtual space into regions and then applying constraints at the boundaries between these regions, has been outlined. We reason that this model takes advantage of people’s natural understanding of 3D spatial layouts, and it also affords the possibility of representing potentially complex environments by mathematical graphs. These graphs can be interrogated using

standard, formal techniques, allowing users to reason about the security provisions in any virtual 3D space. The access graphs can potentially be tailored to any requirement the user has, and not only act as a security information base, but also as maps or navigation aids.

Through a simple realisation of the SPACE access model in Dive 2.2 it has been possible to demonstrate the extraction, manipulation and modification of access rights for a given environment. A security broker, a management interface, permits the user to examine and interrogate the environment under consideration to obtain a full understanding of it. It was finally demonstrated how a filter could be written to use the existing management functionality to examine another CVE, MASSIVE-2.

Chapter five presented a cookbook to enable the SPACE access model to be applied to other systems. A step-by-step guide was given, exemplified through the Spline system and Rodden's awareness model for collaborative systems. This chapter demonstrated the general applicability of SPACE to other systems. Finally we presented an evaluation of the SPACE model and demonstrated that its simple approach was in line with the requirements of CVEs.

To summarise, this work has made the following research achievements and contributions:

- SPACE, a simple access model where the location of an object or entity determines how secure it is, has been presented. This model relies on the fact that an object or person has to traverse boundaries, fundamental objects which partition space into regions, before they can gain access to a region.
- A management tool has been constructed that allows access rights to be examined and modified, presenting the information in both textual and 2D directed graph forms. With the provision of appropriate filters this management tool can be used to compare security credentials across a range of different applications.



- A mapping of the SPACE model to an awareness model, which in turn permits the model to be mapped to non-spatially motivated applications, has been given.
- This work has raised the profile of security related issues in the field of CVEs. Research into security issues and aspects in CVEs is noticeable by its absence, and if CVE technology is to become (commercially) successful then these issues need to be addressed.
- A model has been presented which can be used as the sole provider of access control, or used in conjunction with existing access mechanisms.
- A prototype realisation of the model within Dive 2.2 has been made, with a text interface and a 2-D directed graph interface (using the daVinci graph package) to the management tool being provided.
- A tool to extract access information from MASSIVE-2 data files and convert it into CyCo file format has been constructed. This permits the analysis of MASSIVE-2 environments using the broker described in chapter 4.
- A review of access models and security literature in general has shown that simple, easily understood access models are preferred. This may appear obvious, but many current approaches are still overly complex.
- An assessment of the SPACE model against existing popular access models has been given, demonstrating the suitability of SPACE for CVEs.

## **7.2. Further work**

We concentrate on two main areas for further work; extensions to the realisation of the model in Dive and the accompanying broker application, and extensions to the underlying SPACE model. There is much scope for extending the implementation work through both added sophistication to existing functionality and by extending this functionality. We consider ways in which the access management tool could be improved. We then look at access rights, discussing extensions to both the model and

its realisation. This is followed by some possible extensions to boundaries. Finally issues of scale and distribution are examined.

### **7.2.1. Extensions to the access management tool**

Extensions to the access management tool mainly concern more sophisticated presentation and manipulation of the access information held by the security broker. These are all implementation specific extensions.

#### **7.2.1.1. Viewing and manipulating the access graph within the CVE**

In improving the broker's interface we might exploit graph-drawing techniques to show and query an access graph. First, it would be desirable to view and manipulate a representation of the access graph within the virtual environment, enabling users to use the access graph as a map. This could be achieved using the FDP system (Snowdon 1995). This system is an implementation of a Force Directed Placement algorithm (Fruchterman 1991) and allows 3D graphs to be created and viewed in Dive. Such a representation should permit a user to manipulate the access values and world structure from within the virtual environment. Given a suitable API the changes made to the 3D graph could be written back to the broker information. Alternatively the current use of the daVinci graph package for 2-D graph drawing (Fröhlich 1995) could be extended to incorporate an API to the security broker. A TCL interface to the security broker could be written using the TCL/TK toolkit and this interface combined with the daVinci graph package. This would allow you to both view and manipulate the access graph generated for a set of worlds in the daVinci system and have the changes manifest in the underlying information held by the security broker. This approach makes it possible to manipulate the 2D graph representation of the space and have the changes directly manifest in the 3D virtual world.

#### **7.2.1.2. Tailoring the appearance of the access graph**

The appearance of the access graph in the above two cases should be configurable. For instance the use of colour in the access graph opens up many possibilities. Different coloured arcs could be used to represent the different classifications associated with

each boundary, with the colours of the nodes showing how secure each region is. This may make it easier to spot potential problems after changes have been applied. Also, the ability to highlight various nodes and arcs in the access graph means it can be used as a navigational aid, showing routes between various regions as highlighted nodes and links. In this case we would have an approach to access control which promotes traversal and awareness within a VE!

#### **7.2.1.3. World structure**

The ability to change the world structure (i.e. add and delete worlds) would be a useful extension. Currently it is only possible to modify access values for an existing world structure. If you want to add or delete worlds this must be done at the application level, in the application data files. The new set of data files must then be fed to the broker for analysis. This limits the usefulness of the management tool for dynamic environments where regions are created and deleted with regularity.

#### **7.2.1.4. Embodying users in the access graph**

It may be interesting to provide representation of users in the access graph as they move around the environment. This makes it possible to use the access graph as a monitoring tool, showing which regions are the most popular and how people traverse the environment as a whole. The access graph also offers the potential to associate security violations with users in a quick easy manner.

### **7.2.2. Access rights**

Extensions to both the model and its realisation are given. We consider how general access rights may be defined for individuals, and then groups, and then examine how access violations may be presented at a system level.

#### **7.2.2.1. Specifying clearances and classifications**

The current implementation uses the integers 1 through 9 to represent the classification values at boundaries. This is quite limiting as it means the policy for access rights has to be defined in terms of these nine numbers. A better approach

would be to use a scheme where more sophisticated attributes could be checked at the boundary. This way classifications could be defined in terms of different attribute combinations rather than as an explicit integer value (e.g. senior managers working on the COVEN project at Thomson, Division and TNO have role, project and organisation attributes). The only consideration would be that there would have to be a mapping of the different attribute combinations onto a hierarchy to enable the management techniques to work. For example, all university lecturing staff might have access to one region but only lecturing staff at the University of Nottingham have access to a second region.

#### **7.2.2.2. Group access**

We might extend our approach to explicitly support notions of group access. What should happen when a group of users tries to cross a boundary together? Should their access be determined by the credentials of the weakest member (i.e. as strong as the weakest link); by the strongest (i.e. we can all cross if one of us has a key); or by the average or sum of their credentials (i.e. we can “storm the boundary” if enough of us get together)? A general solution to this problem that introduces different aggregation techniques for determining group clearance from those of individual members would open up many interesting possibilities. For example, by taking the sum of individual credentials, we could create spaces which no one could enter or leave on their own (i.e. whose contents were so sensitive that no one could be left alone with them). One approach to this aggregation problem might be to extend the crowd modelling framework described in (Benford 1997b) which defines techniques for generating aggregate representations of dynamic crowds of participants in collaborative virtual environments.

#### **7.2.2.3. Storing clearances**

In order to perform an access check a user’s clearance value must be compared against a boundary’s classification value. In the realisation presented in chapter four clearances are defined in the Dive embodiment configuration file. While this is sufficient for a prototype system, any real security mechanism would need a dedicated

location where access credentials are defined, modifiable only by trusted users (e.g. system managers). If the more sophisticated classification schemes suggested above are used then authentication and integrity tools (Lampson 1992) are required to ensure the attributes being presented are valid for a given user.

#### **7.2.2.4. How to deal with access violations – subjectivity and user feedback**

Finally concerning access rights, a greater consideration is needed in terms of how access rights and violations are presented within a virtual environment. Ideally, we need to be able to inform users of the classifications associated with boundaries within the VE as well as through the management tool. Subjectivity (Smith 1996, Snowdon 1997) might provide one solution. If a user is not authorised to traverse a boundary then they may see the boundary represented as a locked door (the size of the lock could depend on how their clearance compared to the door's classification), whereas an authorised user would just see a door. It might even be desirable to show highly authorised users a view into the region this boundary protects. Once a user is aware of where and where not they can go, we next need to consider what happens when attempted access right violations occur, either accidentally or deliberately.

There is much scope for different feedback models when a user's clearance is insufficient. Does an alarm sound? How are users told they have insufficient clearance? Should they be informed of alternative routes to their eventual destination? Depending on the sensitivity of the environment being protected the attempted violation of access rights could be reported to those with administrative privileges for the environment. It would even be possible to grant access to someone with insufficient privileges, though whether the person knew they were violating the access rights associated with the boundary would be a policy matter. Indeed, much of this discussion comes down to local policy definition (Fites 1989, Fraser 1997, Edwards 1996).

The positioning of the user in the VE after they have attempted to traverse a boundary is an important issue if access has been denied. Should they be returned to their starting position, if it is possible to determine where they started before trying to

traverse the boundary?. If the VE supports collision detection then colliding with the boundary should be sufficient. However, if there is no collision detection a user will pass through the boundary but remain in the initial region and may well become lost. Empirical studies into how people react to different scenarios seems to be a good way to address this issue.

### **7.2.3. Boundaries**

#### **7.2.3.1. Boundary feedback techniques**

The functionality of the boundary might be extended to provide more feedback when unsuccessful attempts to cross it occur. If there is another route into the region which the presented clearance can traverse, then a dialogue could be presented showing the alternative route to the desired location. Here we are proposing intelligent boundaries which are aware of their surroundings. Alternatively, if the person's clearance is far below what is required then some form of strong warning might be issued. It may also be desirable to have boundaries which remember who has crossed or attempted to cross them, and more importantly how many times. If someone continually tries to cross the boundary with insufficient rights then, after a pre-determined number of attempts, they could at one extreme be issued with a warning or at the other extreme automatically teleported into a holding area to await punishment for their transgression!

#### **7.2.3.2. Dynamic boundaries**

Another boundary extension could be the introduction of dynamic boundaries. Here the classification associated with a boundary is not a static value, but rather a dynamic one, dependent upon external influences. We might consider temporal effects on a boundary where the boundary has different effects at different times of the day. For example the door to an office reception may have a low classification value during the day when the office is open and clients are visiting, but changes to a high classification at night when the office is closed and only employees can gain access. The issue here is how to specify classifications to any dynamically changing boundaries.

#### **7.2.4. Scalability and distribution**

The application of our approach to large-scale distributed virtual environments raises the issues of scale and distributed access to world data. For example, the construction of a single local access graph representing a globally distributed environment of millions of inter-linked worlds (such as a future VRML environment) will be impossible due to the scale of the computation involved and problems with gaining access to the necessary world data. Future work should therefore address the problem of developing a distributed access model and security broker so that responsibility for managing and presenting access information is spread across many local but communicating security brokers.

Related to distribution issues is the task of dynamically creating and modifying access graphs. The present approach represents a snapshot of the access rights at any one time, and if multiple users are actively interrogating the same environment and making changes then some form of locking mechanism is required to make sure the environment under examination remains consistent. Other user's updates also need to be propagated through to all management interfaces.

#### **7.2.5. Real-world spaces**

It should be possible to use the SPACE model as an analytical tool to examine real world spaces and environments. Architectural walk-throughs or plans could be scrutinised, checking the integrity of the building layout and making sure there are no hidden routes to locations or rooms that are meant to be only access through recognised routes. Much of this will depend on the building being constructed, but you can imagine if, for example, you are constructing a prison you want to know about every possible route around the prison.

Another possible use for the SPACE model could be to examine network routing tables and layouts. Typically such routing tables can be complex with many entries. Using the security broker it should be possible to find out about particular routes

between sites and more importantly manage a collection of routing entries, making potential changes and observing the consequences before the changes are made.

### **7.3. Concluding remarks**

The use of CVE technology is increasing. The world is becoming a smaller place, with more and more people being connected to each other via computer networks. If CVEs are to provide the next generation interface to collaborative tasks, steps must be taken to ensure this interface offers a secure and consistent working environment. Due to advances in technology and software it is no longer the case that performance and functionality issues take precedence over security issues in applications. More and more importance is being placed on networked solutions, and these solutions need to be secure.

Security has a reputation for being a necessary evil, and all too often it is left to the end of a project or piece of work to provide some security (if at all), rather than making it an integral part of the work in the first place. For CVE technology to gain widespread use, as it is anticipated it will, provision needs to be made for security mechanisms that protect both developers and users of a system alike. The trend of favouring functionality over security has to be bucked.

In this thesis we have proposed a simple approach to access control for CVEs based on spatial partitioning. If security mechanisms are to be successfully employed in the future, it is important to ensure they are simple to apply, understand and manage. Only then will a widespread take-up of security take place, and people's natural prejudices against security be overcome.



## References

- (Abadi 1993) M. Abadi, M. Burrows, B. Lampson, and G. Plotkin, “A calculus for access control in distributed systems”, ACM Transactions on Programming, Languages and Systems, Vol. 15, No. 4, pp.706–734, 1993.
- (Active Worlds 1998) Home page for Active Worlds at <http://www.activeworlds.com> [last accessed 2<sup>nd</sup> July 1998].
- (Amoroso 1990) Edward Amoroso, “A Policy Model for Denial of Service”, Proceedings of the Computer Security Foundations Workshop III, 1990.
- (Amoroso 1994) Edward Amoroso, “Fundamentals of Computer Security Technology”, Prentice Hall International, ISBN 0–13–305541–8, 1994.
- (Barrus 1996) John W. Barrus, Richard C. Waters and David B. Anderson, “Locales and Beacons: Efficient and Precise Support for Large Multi–User Virtual Environments” in Proc. 1996 IEEE Virtual Reality Annual International Symposium (VRAIS’96), San Jose, March 1996.

- 
- (Bell 1973) D. E. Bell and L. J. LaPadula, "Secure computer systems: Mathematical foundations", ESD-TR-73-278, Vol.1, ESD/AFSC, Hanscom AFB, Bedford, MA, November 1973.
- (Benedikt 1991) Michael Benedikt, "Cyberspace: Some Proposals" in "Cyberspace: First Steps", Ed. Michael Benedikt, MIT Press, ISBN 0-262-02327-X, 1991.
- (Benford 1993) Steve Benford and Lennart Fahlén, "A Spatial Model of Interaction in Large Virtual Environments", 3rd European Conference on Computer Supported Cooperative Work (ECSCW'93), pp. 109-124, Milan, Italy, 13<sup>th</sup>-17<sup>th</sup> September 1993.
- (Benford 1997) S. Benford and C. Greenhalgh, "Introducing Third Party Objects into the Spatial Model of Interaction", 5th European Conference on Computer Supported Cooperative Work (ECSCW'97), pp. 189-204, Lancaster, UK, 7<sup>th</sup>-11<sup>th</sup> September 1997.
- (Benford 1997b) S. Benford, C. Greenhalgh and D. Lloyd, "Crowded Collaborative Virtual Environments", Proc CHI'97, pp. 59-66, Atlanta, Georgia, US, March 22<sup>nd</sup>-27<sup>th</sup>, 1997.
- (Bentley 1995) R. Bentley, T. Horstmann, K. Sikkell and J. Trevor, "Supporting collaborative information sharing with the World Wide Web: The BSCW Shared Workspace system", 4<sup>th</sup> International WWW Conference, Boston, December 1995, pp. 63-74.
- (Biba 1977) K. J. Biba, "Integrity considerations for secure computer systems", ESD-TR-76-372, ESD/AFSC, Hanscom AFB, Bedford, Mass., April 1977. (MITRE MTR-3153, NTIS AD A039324).

- 
- (Bowers 1993) J. M. Bowers, "Modelling Awareness and Interaction in Virtual Spaces", in Supplement to Proceedings of the 6th MultiG Workshop, pp. S9–S24, Stockholm–Kista, Sweden, May 1993.
- (Bowers 1995) John Bowers, "The Social Logic of Cyberspace" in COMIC Deliverable 4.3 *Assessment and Refinement of Models of Interaction*, Eds. A. Bullock and J. Mariani, pp. 91–147, University of Lancaster, August 1995.
- (Bryant 1988) Bill Bryant, "Designing an Authentication System: a Dialogue in Four Scenes", 1988, at <http://web.mit.edu/kerberos/www/dialogue.html> [last accessed 12<sup>th</sup> January 1999].
- (Bullock 1994) A. Bullock and S. Benford, "An Approach to Access Control for Collaborative Virtual Reality", 6th ERCIM Workshop Participants' proceedings, pp.233–246, Stockholm–Kista, June 1994.
- (Bullock 1994b) A. Bullock, "An Access Control for Distributed Virtual Environments", 2nd UK VR–SIG Workshop, Silicon Graphics, Theale, Reading, 1<sup>st</sup> December 1994.
- (Bullock 1997) A. Bullock and S. Benford, "Access Control in Virtual Environments", ACM Symposium on Virtual Reality Software and Technology (VRST'97), pp. 29–35, Lausanne, Switzerland, September 15<sup>th</sup> – 17<sup>th</sup> 1997.
- (Carlsson 1993) Christer Carlsson and Olof Hagsand, "DIVE – a Platform for multi–user virtual environments", *Computer and Graphics*, Vol. 17, No. 6, pp. 663–669, 1993.

- 
- (Cattermole 1979) K. Cattermole, "Graph Theory and Communications Networks" in "Applications of Graph Theory", Eds. Robin J Wilson and Lowell W Beineke, 1979. ISBN 0-12-757840-4.
- (Chapman 1995) Brent Chapman and Elizabeth Zwicky, "Internet Security Strategies", *Connexions: The Interoperability Report*, Vol. 9, No. 12, pp. 10-17, December 1995. ISSN 0894-5926.
- (Chokhani 1992) S. Chokhani, "Trusted Products Evaluation", *Communications of the ACM*, Vol. 35, No. 7, pp. 64-76, July 1992.
- (Comic 1995) CSEG at Lancaster University, The COMIC Project, at <http://www.comp.lancs.ac.uk/computing/research/cseg/comic> [Accessed 15<sup>th</sup> April 1998].
- (Cooper 1990) R. Cooper, "Organisation/Disorganisation", *Organisation Studies*, 1990.
- (Coulouris 1994a) G. Coulouris and J. Dollimore, "Requirements for security in cooperative work: two case studies", Technical Report 671, Dept. of Computer Science, Queen Mary and Westfield College, University of London.
- (Coulouris 1994b) G. Coulouris and J. Dollimore, "A security Model for cooperative work", Technical Report 674, Dept. of Computer Science, Queen Mary and Westfield College, University of London.
- (Dewan 1998) Prasun Dewan and Honghai Shen, "Controlling Access in Multiuser Interfaces", *ACM Transactions on Computer-Human Interaction*, Vol. 5, No. 1, pp. 34-62, March 1998.

- 
- (Dougherty 1992) Dale Dougherty, "Sed & Awk", O'Reilly & Associates, Inc., ISBN 0-937175-59-5, 1<sup>st</sup> Edition November 1992.
- (Edwards 1996) W. Keith Edwards, "Policies and Roles in Collaborative Applications", Computer Supported Cooperative Work '96, pp. 11-20, Boston, MA, USA, 1996.
- (Ellis 1991) Clarence A. Ellis, Simon J. Gibbs, and Gail L. Rein, "Groupware: Some issues and experiences", Communications of the ACM, Vol. 34, No. 1, pp. 38-58, January 1991.
- (Feiertag 1977) R. J. Feiertag, K. N. Levitt and L. Robinson, "Proving multilevel security of a system design", in Proc. 6<sup>th</sup> ACM Symp. Operating Systems Principles, ACM SIGOPS Operating Syst. Rev., Vol. 11, No. 5, pp. 57-65, November 1977.
- (Fites 1989) P. E. Fites, M. P. J. Kratz and A. F. Brebner, "Control and Security of Computer Information Systems", Computer Science Press, Rockville (MD), 1989.
- (Fraser 1997) B. Fraser (Ed.), "Site Security Handbook", RFC 2196, at <ftp://nic.merit.edu/documents/fyi/fyi8.txt>, September 1997 [Accessed 15<sup>th</sup> April 1998].
- (Fröhlich 1995) M. Fröhlich, M. Werner: "Demonstration of the interactive Graph Visualization System daVinci", in: R. Tamassia, I. Tollis (Eds.), Proceedings of DIMACS Workshop on Graph Drawing '94, Princeton (USA), 1994, Lecture Notes in Computer Science No. 894; Springer Verlag; January 1995.
- (Fruchterman 1991) T. M. J. Fruchterman and E. M. Reingold, "Graph Drawing by Force Directed Placement", Software Practice and Experience, Vol. 21 No. 11, pp. 1129-1164, 1991.

- 
- (Funkhouser 1995) Thomas A. Funkhouser, "RING: A Client-Server System for Multi-User Virtual Environments", ACM SIGGRAPH Special Issue on 1995 Symposium on Interactive 3D Graphics, Cambridge MA, pp. 11-20, 1992.
- (Giddens 1985) A. Giddens, "The Constitution of Society", Cambridge: Polity, 1985.
- (Gladney 1997) H. M. Gladney, "Access Control for Large Collections", ACM Transactions on Information Systems, Vol. 15, No. 2, pp. 154-194, April 1997.
- (Goffman 1967) E. Goffman, "Interaction Ritual", Harmondsworth: Penguin, 1967.
- (Greenhalgh 1995) Chris Greenhalgh and Steve Benford, "MASSIVE: a distributed virtual reality system incorporating spatial trading", 15<sup>th</sup> International Conference on Distributed Computing Systems, pp. 27-34, Vancouver, Canada, May 30<sup>th</sup> - June 2<sup>nd</sup> 1995.
- (Greenhalgh 1997) C. Greenhalgh, "Analysing movement and world transitions in virtual reality tele-conferencing", 5th European Conference on Computer Supported Cooperative Work (ECSCW'97), pp. 313-328, Lancaster, UK, 7<sup>th</sup>-11<sup>th</sup> September 1997.
- (Greenhalgh 1998) Greenhalgh, C. M. and Benford, S. D., "Supporting Rich And Dynamic Communication In Large Scale Collaborative Virtual Environments", Presence: Teleoperators and Virtual Environments, MIT Press (accepted for publication - to appear in 1998).

- 
- (Greif 1986) Irene Greif and Sunil Sarin, "Data Sharing in Group Work", ACM Computer Supported Cooperative Work '86 (CSCW'86), pp. 175–183, Austin, Texas, USA, 1986.
- (Hafner 1991) Katie Hafner and John Markov, "Cyberpunk: Outlaws and Hackers on the Computer Frontier", Corgi Books, 1991. ISBN 0-552-13963-7.
- (Hink 1975) T. H. Hink and M. Schaeffer, "Secure data management system", RADC-TR-75-266, Rome Air Dev. Center, AFSC, Griffiss AFB, N. Y., November 1975 (NTIS AD A019201).
- (IEEE 1993) Institute of Electrical and Electronics Engineers, International Standard, ANSI/IEEE Std 1278-1993, "Standard for Information Technology, Protocols for Distributed Interactive Simulation", March 1993.
- (Isaiah:53(6)) Isaiah chapter 53, verse 6, The Holy Bible, Revised Standard Version.
- (ISO 7498-2 1989) Information processing systems – Open Systems Interconnection – Basic Reference Model – Part 2: Security Architecture. ISO 7498-2 : 1989 (E). International Standards Organisation, Geneva, Switzerland.
- (ISO 10181-3 1996) Information Technology – Open Systems Interconnection – Security Frameworks for Open Systems: Access Control Framework. BS ISO/IEC 10181-3 : 1996. International Standards Organisation, Geneva, Switzerland.
- (ISO 14772-1 1997) Information Technology – Computer graphics and image processing – The Virtual Reality Modelling Language (VRML) – Part1: Functional specification and UTF-8 encoding. ISO/IEC 14772-1 : 1997.

- 
- (Jacob 1991) Jeremy Jacob, “The Basic Integrity Theorem”, Proceedings of the Computer Security Foundations Workshop IV, 1991.
- (Kernighan 1984) Brian Kernighan and Rob Pike, “The UNIX programming environment”, 1984. ISBN 0–13–937699–2.
- (Kernighan 1988) Brian Kernighan and Dennis Ritchie, “The C Programming Language”, 2<sup>nd</sup> Edition, Prentice Hall Software Series, ISBN 0–13–110362–8, 1988.
- (LambdaMOO 1998) Humbert Humbert’s LambdaMOO archive at <http://vesta.physics.ucla.edu/~smolin/lambda> [last accessed 2<sup>nd</sup> July 1998].
- (Lampson 1971) B. W. Lampson, “Protection”, in Proc. Fifth Princeton Symposium on Information Sciences and Systems, Princeton University, March 1971, pp. 437–443, reprinted in Operating Systems Review, 8, 1, January 1974, pp. 18–24.
- (Lampson 1992) B. Lampson, M. Abadi, M. Burrows, and W. E. Wobber, “Authentication in distributed systems: Theory and practice”, ACM Transactions on Computer Systems, Vol. 10, No. 4, pp. 265–308, November 1992.
- (Landwehr 1981) Carl E. Landwehr, “Formal Models for Computer Security”, ACM Computing Surveys, Vol. 13, No. 3, pp. 247–278, September 1981.
- (McLean 1990) John McLean, “The Specification and Modeling of Computer Security”, IEEE Computer, Vol. 23 No. 1, January 1990.
- (Macedonia 1995) M. Macedonia, M. Zyda, D. Pratt, D. Brutzman and P. Barham, “Exploiting Reality with Multicast Groups: A Network Architecture for Large–scale Virtual Environments”,



- 
- in Proc 1995 IEEE Virtual Reality Annual International Symposium (VRAIS'95), 11<sup>th</sup>–15<sup>th</sup> March, 1995, RTP, North Carolina.
- (Mariani 1997) John Mariani, Personal communication on Lac Leman, Switzerland, September 1997.
- (Naylor 1990) Bruce Naylor, John Armanatides and William Thibault, “Merging BSP Trees Yields Polyhedral Set Operations”, *Computer Graphics*, Vol. 24, No. 4, pp. 115–124, August 1990.
- (NCSC 1985) National Computer Security Centre, “Department of Defence Trusted Computer Security Evaluation Criteria”, aka “The Orange Book”, DoD 5200.28–STD, 1985
- (Neuman 1994) B. Clifford Neuman and Theodore Ts'o. “Kerberos: An Authentication Service for Computer Networks”, *IEEE Communications*, 32(9) pp. 33-38. September 1994.
- (Palme 1990) Jacob Palme, “SuperCOM – A Distributed Computer Conference System”, in *Message Handling Systems and Application Layer Communication Protocols*, Proceedings of the IFIP WG 6.5 International Symposium, October 1990. Edited by P. Schicker and E. Stefferud, North–Holland 1991.
- (Pandzic 1997) Pandzic I.S., Capin T.K., Lee E., Magnenat Thalmann N., Thalmann D., “A flexible architecture for Virtual Humans in Networked Collaborative Virtual Environments”, *Proc. Eurographics 97*, Budapest, Hungary, 1997.
- (Prinz 1992) Wolfgang Prinz, “Representing Authorization Information in the X.500 directory”, *ULPAA'92 IFIP6.5 conference*, pp. 295–311, Vancouver, Canada, 1992.

- 
- (Rheingold 1991) Howard Rheingold, "Virtual Reality", Mandarin, 1991. ISBN 0-7493-0889-3.
- (Rodden 1996) Tom Rodden, "Populating the Application: A Model of Awareness for Cooperative Applications", Computer Supported Cooperative Work (CSCW'96), pp. 87-96, Boston, MA, USA, 1996.
- (Roseman 1996) M. Roseman, and S. Greenberg, "TeamRooms: Network Places for Collaboration", ", Computer Supported Cooperative Work (CSCW'96), Boston, MA, USA, 1996.
- (Roseman 1997) M. Roseman, and S. Greenberg, "A Tour of TeamRooms", Video Proceedings of the ACM SIGCHI'97 Conference on Human Factors in Computing Systems, Atlanta, Georgia, March 22-27, ACM Press. Two page summary.
- (Sandhu 1993) R. S. Sandhu, "Lattice-based access control models", IEEE Computer, Vol. 26, No. 11, pp. 9-19, 1993.
- (Sandhu 1996) R. S. Sandhu and P. Samarati, "Authentication, Access Control and Audit", ACM Computing Surveys, Vol. 28, No. 1, pp. 241-243, March 1996.
- (Sandhu 1996b) R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based Access Control Models", IEEE Computer, Vol. 29, No. 2, 1996.
- (Shen 1992) HongHai Shen and Prasun Dewan, "Access Control for Collaborative Environments", ACM Computer Supported Cooperative Work (CSCW'92), pp. 51-58, Toronto, Canada, November 1992.

- 
- (Sikkel 1997) Klaas Sikkel, "A Group-based Authorization Model for Cooperative Systems", Proceedings of the Fifth European Conference on Computer Supported Cooperative Work (ECSCW'97), pp. 345–360, Lancaster, UK, September 1997.
- (Smith 1996) Gareth Smith, "Cooperative Virtual Environments: lessons from 2D multi user interfaces", Computer Supported Cooperative Work (CSCW'96), pp. 390–398, Boston, MA, USA, 1996.
- (Snowdon 1995) Dave Snowdon, Steve Benford, Chris Brown, Rob Ingram, Ian Knox and Luke Studley, "Information Visualization, Browsing and Sharing in Populated Information Terrains", BCS Seminar series on New Directions in Software Development, University of Wolverhampton, UK, 8<sup>th</sup> March 1995.
- (Snowdon 1996) Dave Snowdon, Elizabeth Churchill and Jolanda Tromp (Eds.), Proceedings of Collaborative Virtual Environment (CVE)'96, University of Nottingham, 19<sup>th</sup>–20<sup>th</sup> September 1996.
- (Snowdon 1997) Dave Snowdon and Kai-Mikael Jää-Aro, "A subjective Virtual Environment for collaborative information visualization", in Virtual Reality Universe '97, Westin Santa Clara Hotel, California, USA, April 2<sup>nd</sup>–5<sup>th</sup>, 1997.
- (Snowdon 1998) Dave Snowdon and Elizabeth Churchill (Eds.), Proceedings of Collaborative Virtual Environment (CVE)'98, University of Manchester, 17<sup>th</sup>–19<sup>th</sup> June 1998.
- (Spafford 1989) Eugene Spafford, "The Internet Worm: Crisis and Aftermath", Communications of the ACM, Vol. 32, No. 6, June, 1989.

- 
- (Sterling 1993) Bruce Sterling, "Hacker Crackdown: Law and Disorder on the Electronic Frontier", Viking Books, 1993. ISBN 0-670-84900-6.
- (Stoll 1991) Clifford Stoll, "Cuckoos Egg: Stalking the Wily Hacker", Pan Books, 1991. ISBN 0-330-31742-3.
- (Tanenbaum 1996) Computer Networks, Andrew Tanenbaum, Prentice Hall, 3<sup>rd</sup> Edition, pp. 669-680, 1996. ISBN 0-13-349945-6.
- (UKSP01 1991) Communications-Electronics Security Group and the Department of Trade and Industry, "UK IT Security Evaluation and Certification Scheme", 1st March 1991.
- (Waters 1997) Waters R.C., Anderson D.B., Barrus J.W., Brogan D.C., Casey M.A., McKeown, S.G., Nitta T., Sterns I.B., & Yerazunis W.S., "Diamond Park and Spline: Social Virtual Reality with 3D Animation, Spoken Interaction, and Runtime Extendability," Presence: Teleoperators and Virtual Environments, 6(4):461-480, August 1997.
- (Wilson 1985) Robin J. Wilson, "Introduction to Graph Theory", Longman, 3<sup>rd</sup> Edition, 1985. ISBN 0-582-44685-6.

## **Appendix A**

### **Surveying Security in CSCW – A Questionnaire**

#### **A.1. Introduction**

At the beginning of this research I circulated a questionnaire, both electronically and on paper, to researchers in the area of CSCW. In particular CoTech working groups, CSCW project mailing lists and USENET newsgroups were targeted. In all over thirty responses were received via e-mail, with another five to ten coming in via paper mail. Useful information to help focus and guide my future research was gained from the responses, and chapter one contains a small selection of the more pertinent quotes from the answers received.

This appendix contains the original text of the questionnaire that was circulated.

#### **A.2. The Questionnaire**

##### **Introduction**

I have just started studying for a PhD in computer science in the fields of CSCW and security. I am hoping to take an in depth look at security as it relates to CSCW and try to discover just exactly what is needed, generally, and also what is needed specifically at an application level.

As an initial step towards my thesis I should like to try and find out what has already been done in this area. To this end I have compiled a questionnaire which asks details about any CSCW projects you might have developed/be developing and in particular about how security affected these projects (if at all). It also asks your opinions about certain security issues and how they may or may not affect CSCW.

### **General Details**

Name:

Organisation:

E-mail address:

Snail Mail address:

### **Project Details**

Project Title:

Brief Project Description:

When planning and designing the project what thoughts and special considerations, if any, were given to possible security requirements?

Was any implementation of a security based nature undertaken? If so what?

How important were these requirements felt to be?

Did any security aspects occur naturally (for example the use of passwords)?

### **General Questions**

How would you say the requirements of CSCW applications, in terms of security, are different from other applications?

How importantly would you rate effective security as being desirable in a CSCW activity? (please underline one).

---

essential      important      unimportant      not necessary

Why do you believe this?

If you were given the choice between an easy to use collaborative editor with no restrictions on who can edit what, and an editor which guarantees the integrity of the information being edited, but at the expense of the ease of use, which would you choose and why? (Assuming here that you cannot switch between editors for different tasks – you only want to learn the one editor.)

Should the issue of security and protection of information be relevant to CSCW? Surely if the information is that important it shouldn't be in a shared tool. To what extent would you say this is true?

There are two classes of entities who need to be protected against:

- 1) legitimate users within a system or application.
- 2) outsiders trying to break into systems and/or applications to eavesdrop/steal information.

Should CSCW concentrate on providing protection against the second threat, as the first threat only serves to impede the collaborative process?

To what extent can trust be relied upon as a security mechanism?

all of the time    most of the time      some of the time      hit and miss    won't work

Here accidents, mistakes, jokes and grievances amongst others must be guarded against.

Security is sometimes seen as a way of impeding information sharing and so seems to try and achieve the opposite of CSCW whose aim is to promote information sharing. Is this really the case? Can the restrictions normally associated with security be overcome in any way? (e.g. use of tickets in Kerberos allows the user to make many authentication requests while her/his password needs only be entered once).

What would you say is the one most important security requirement for a CSCW application/environment? Why?