Towards an Analytical Role Modelling Framework for Security Requirements

Robert Crook[1] Darrel Ince Bashar Nuseibeh

Security Requirements Group
Department of Computing, The Open University
Walton Hall, Milton Keynes, MK7 6AA, UK

Email: Robert.Crook@t-online.de, D.C.Ince@open.ac.uk, B.A.Nuseibeh@open.ac.uk

Abstract

Pressures are increasing on organisations to take a more systematic approach to incorporating security into their software development process. The key to this is analysing security requirements early on, rather than treating security as an add-on, as is often the case. An important component of security requirements is access control, and *roles* have been found to provide an effective basis for defining access restrictions. Current requirements engineering methods are generally inadequate for eliciting and analysing these types of requirements, because they do not allow the complex organisational structures and procedures that form the basis of role-based security policy to be represented adequately. In this paper, we outline the concepts that underpin role-based access control, and relate these to organisational theory, to give a basis for defining roles. We then propose an analytical role modelling framework that enables us to model and analyse access restrictions based on these concepts. The framework is illustrated by a detailed example taken from the healthcare domain.

1. Introduction

There is increasing recognition of the role of requirements engineering in the formulation and analysis of security policies [AE01]. Information security is concerned with the confidentiality, integrity, and availability of data in storage and in transmission [Anderson01]. Protecting information during transmission involves the development and use of secure protocols and cryptography. The key to protecting stored data – the focus of this paper – is restricting access to authorised users. Access is controlled by allocating permissions to users, allowing them to access particular information only. In order to specify access control requirements, an understanding of organisational structures and procedures is needed [Moffett98]. While some requirements methods incorporate notions of actors or agents, they do not explicitly allow relating actors to organisational groups or modelling authority, both of which are key to identifying responsibilities[2] of users with respect to the information assets that are to be protected.

Research into access control has become focused increasingly around Role-Based Access Control (RBAC) [SCFY96]. The basic premise underlying RBAC is that in order to simplify administration, permissions are assigned to roles rather than users; a user gains permissions by being assigned a role. Roles are a way of defining positions in organisations, bundling responsibilities, or perhaps representing a qualification.

The basic questions that need to be addressed in the early stages of system development, and the ones that are addressed by this paper, are concerned with how roles are derived from organisational structures and how these can be related to access restrictions and to the core business functions.

[1] Robert Crook is also an independent consultant.
[2] We use the term *responsibility* to denote *obligation* and *accountability*.

The paper is organised as follows. In section 2 we examine concepts that underlie RBAC and weaknesses of models proposed in the literature. In section 3 we relate this research to organisational theory, and identify categories of roles and how they may be derived from organisational structure. In section 4 we introduce a framework for specifying and analysing role-based access restrictions, and illustrate its use on a healthcare example. We conclude the paper with a short summary and discussion of future work.

2. Background and Related Work

Research on computer security policy has been ongoing since the 1970's. The Bell-LaPadula (BLP) model [BL73] has been particularly influential, and forms the basis of a family of multi-level security models, usually referred to as *mandatory access control* [Dod85]. *Discretionary access control* has been accepted as a less rigorous way of controlling access. Clark and Wilson [CW87] identified important principles for protecting the integrity of data in commercial organisations and in particular the *separation of duties*. The idea being that actions can be divided into smaller tasks that must be carried out by different individuals, so preventing one individual alone from being able to defraud the system.

Over the last few years RBAC has been recognised as an effective mechanism on which to define access restrictions. Sandhu et al. [SCFY96] argue that RBAC is policy neutral, thereby allowing the enforcement of a variety of security constraints. A good example is the separation of duties, an important policy to ensure integrity. Separation of duties can be achieved by defining mutually exclusive roles that have to be invoked for a collaborative task, in order to ensure that a sequence of tasks cannot be carried out by a single individual. Furthermore, RBAC can coexist with, or be used to support, mandatory access control policy [NO95] such as BLP or a discretionary access control policy [SM98], where users are assigned permissions individually. There is quite a significant variation in the interpretation of RBAC, differing in the level of sophistication that different models support. The authors illustrate this by presenting a family of models (RBAC96) with varying levels of complexity. More recently, Sandhu et al. [SFK00] have proposed the NIST RBAC model, which is an attempt to define a unified standard.

The NIST RBAC model is actually a sequence of models with each subsequent model containing an increased set of capabilities. Three important characteristics are identified. The first is the basic ternary user role permission relationship. The idea is that permissions are assigned to roles rather than to users. The second key characteristic is a role hierarchy that is relatively simple, whereby senior roles inherit the permissions of junior roles. For permissions that should not be inherited, Sandhu et al. propose an activity hierarchy in which senior roles only inherit permissions from junior roles if they have been activated. The final key characteristic is the principle of role constraints, which is important for enforcing separation of duties by, for example, defining two roles as mutually exclusive.

The idea of a single hierarchy based on inheritance has been called into question [Moffett98]. A manager does not necessarily inherit the roles of his juniors. For example, in a project, a manager does not necessarily possess the necessary competence to carry out specialised activities. Moffett has derived a set of hierarchies based on organisational control principles. The three main principles are separation of duties, decentralisation and supervision, and review. In order to capture these characteristics, Moffett proposes three types of hierarchies.

The "is-a" hierarchy based on generalisation. It is often possible to identify common responsibilities amongst members of an organisation. These responsibilities can be bundled together to form a generalised role. An example of this in a hospital, say, would be to define a generalised role of health care provider that provides permissions shared by both doctors and

nurses. The role doctor is itself a generalised role for a physician or a surgeon. The inverse of generalisation is specialisation.

An activity hierarchy based on aggregation. In this hierarchy, permissions are bundled together to form a collection of permissions that are needed to carry out various tasks that logically belong together from an organisational standpoint. For example, a sequence of tasks may be required to provide a specific customer service, such as booking a flight.

A supervision hierarchy based on the organisational hierarchy. This hierarchy is what is normally considered to be the organisational hierarchy in that it represents the lines of supervision showing the seniority of the members of staff. This is the hierarchy that can also be used to differentiate permissions between junior and senior members of staff.

Bacon et al. [BLM01] also demonstrate how roles based on function and seniority can be combined. In order to be assigned certain roles, a user must have been assigned other prerequisite roles, for example, a doctor can only be assigned the role of senior haematologist if the roles senior doctor and haematologist have already been assigned.

More recent work in this area is concerned with access control models referred to as *active security models,* which are aware of the context of an ongoing activity. Bertino et al. [BBF00] describe how temporal constraints can be defined for roles, for example, when a role is activated for a shift, and then subsequently deactivated. In addition, an administrator can activate roles *ad hoc*. Covington et al. [CLSDAA01] have explored applications for the home and suggest how environmental roles could be useful. Access can be permitted based on environmental factors, such as location or time of day. Georgiadis et al. [GMPT01] combine contextual information with team based access control. Team based roles identified by Thomas [Thomas97] are useful for collaborative working environments, where users are assigned to teams and get access to the team's resources. This can be combined with other contextual information, such as location or time intervals. Yao et al. [YMB01] present an access control model (OASIS), whereby users can activate roles, provided they satisfy prerequisite conditions, such as having an appropriate qualification, assigned function, task competence, or environmental constraint.

Research into access control has thus demonstrated that roles provide an effective basis for defining restrictions. However, the questions remain: where do roles come from, what are their relationships with one another, and how do we address these questions during requirements engineering?

3. On Organisational Structures and Roles

There is no universally accepted definition of roles. As we have seen in the previous section, researchers have different views on what constitutes a role and how role hierarchies can be defined. One of the reasons for this is that a role is a concept rather than a reality. Organisations do not normally have lists of roles, but they do have organisational structures, often documented with hierarchical charts. In this section we discuss different categories of roles, and how they can be derived from organisational structures.

3.1. Organisational Structure

The organisational structure is designed by the top management of an organisation and defines the lines of authority and the division of work. These are the two principle characteristics that determine the responsibilities for each individual member of the organisation.

Organisations can take many forms ranging from very simple structures, such as are normally found in small start-ups, to the complex divisionalised structures found in large multinational corporations; and although no two organisations are exactly the same, neither are they truly

unique. There are common aspects that enable us to categorise the structure and generalise about it.

An organisation is a composite structure made up of organisations that can themselves also be divided into even smaller units. There are different ways in which individual positions in an organisation can be grouped together to form these units, depending on the needs of theatorganisation. In defining roles it is useful to understand how these groupings come about.

3.2. Organisational Groupings

Mintzberg [Mintzberg92] identified the key characteristics that are used to form organisational groupings. Essentially there are two basic types of characteristics that are used to form organisational groups. The first is function defined around the tasks that a group performs, and the second is market defined around responsibility for product, service, or customers.

Often, particularly in large organisations, several of these characteristics are used. The National Health Service in the UK is divided into regional health authorities that, in turn, are composed of hospitals to serve the different population centres, so that the authority and the hospitals are organised on a geographical basis. A hospital, however, is organised on a functional basis according to administration, the medical specialities, and supporting services. Similarly, retail banks have autonomous branches dispersed to serve local markets with a functional structure in each branch.

Another way in which an organisation can be structured according multiple factors is through a matrix structure. In this case, each member of the organisation will belong to two groups. One group is responsible for the product or market, and the other has a functional responsibility. An example of this is in an engineering company undertaking projects. Each project consists of a multidisciplinary team of engineers and each member of the team reports to the project manager, but there are also departments that carry responsibility for staff development and maintaining standards in the different engineering disciplines.

3.3. Organisational Roles

Each member of the organisation has a set responsibilities that is the ultimate determinant of access privileges for applications. It is worth stepping back briefly to consider what organisational roles are, so that we can relate them to the groupings we defined above.

Role theory is largely a sociological science focusing on how individuals interact with one another. Handy [Handy85] describes the concepts of role theory. The individual who is the centre of analysis is called the *focal person,* and the group of individuals with whom he interacts is called his *role set*. Thus, depending on the situation and the person with whom he is interacting, an individual will adopt a specific role, perhaps as a father, customer, friend, or advisor, and there are certain societal expectations of people in these respective roles and in the way they are supposed to behave. Some roles are occupationally defined, such as doctor or lawyer, and for which there are legal as well as cultural expectations.

In an organisation, the situation is similar in that members of this organisation in a particular position may have multiple roles such as manager, specialist, or subordinate depending on the task they are currently undertaking, and particularly with whom they are interacting. Thus, roles in an organisation, and the expectations of an individual adopting a role are defined by management, and relate to the responsibilities and tasks that have been assigned to that individual.

3.4. Role Categories

So, we need to create a link between roles and assets, or more appropriately between sets of roles and assets, because, as we have seen, an individual will have several roles. Seniority is one of the important dimensions and the other is scope of work, for which Mintzberg has defined two groups of characteristics based on functions and markets. These can be used as a basis for the identification of roles and are summarised as follows:

- Roles based on supervision
  - Seniority

- Roles based on function
  - Qualification
  - Function
  - Work-process

- Roles based on market
  - Market/Customer/Client
  - Product/Service
  - Location
  - Time

For each of these categories, there is often a potential for a hierarchy, perhaps reflecting hierarchical structures in the organisation. The characteristics of this hierarchy also need to be determined, for example, whether permissions are automatically inherited.

It is also necessary to establish the relationship between these roles and the access to information assets. Having established the key characteristics by which responsibilities are assigned, it is then a matter of identifying the criteria by which access is restricted. This needs to be done for each type of access or operation on each asset.

3.5. Roles based on Seniority

Roles that are based on seniority are reflected in the hierarchical lines of authority. Supervision is a key co-ordination mechanism and seniority is the cornerstone of this mechanism. But what are the characteristics of a role based on seniority, and what is the relationship to other roles?

Mintzberg has identified three types of authority: line, staff, and functional authority. Staff and functional authority are very similar in nature in that they transcend organisational groups. Line authority is the most common form, and is the kind of authority found within a single organisational unit. This is what we focus on in this paper.

One of the key characteristics of seniority is span of control, that is, which subordinates are under the control of a supervisory role. There will be subordinates who are directly controlled and those who are indirectly controlled through delegated authority. There are two properties that need to be considered: the first is the cardinality representing the span of control, and the second is the transitivity that represents the extent to which control transcends levels.

An important issue is how the authority relationship between two users can be modelled. There are two ways that this can be done. The first is to link the relationship through the users, i.e. a user with a subordinate role is directly assigned to a supervisor. The alternative is to link the relationship through other roles representing situational factors, such as a project, a ward in a hospital, or a product line in a sales department. In the latter case, the line of authority in an organisational grouping is linked to the functional or market factor on which the group is based.

Subordinates may have more than one supervisor. There are several situations in which this can occur. One situation is when the subordinate answers to superiors with varying degrees of delegated authority. In the absence of the immediate superior, one of the subordinates could act in a temporary capacity as a supervisor to whom the others report. A second situation occurs in matrix organisations, where subordinates answer to two superiors: one with responsibility for assigning the tasks, and the other with a more indirect responsibility for the quality of work, standards, and personal development. In fact, in some organisations that operate in this way, an individual may be assigned to more than one project. In such cases, the individual reports to a senior in the functional hierarchy and several project managers in a market-based hierarchy.

Delegation is also a key aspect a supervisory structure [Moffett98]. Delegated authority is an inherent part of any hierarchy and there are several forms of this. Barka and Sandhu [Bsa00] identify the following characteristics.

- ***Permanence***. This determines whether authority delegated on a temporary or permanent basis.
- ***Totality***. This determines the extent to which all or only some permissions from a user or role are delegated.
- ***Duration***. This determines the length of time for which a delegation is valid. ***Delegation Levels.*** This determines the extent to which delegated permissions can be further delegated

Each of these characteristics needs to be modelled in order to capture the principles of delegated authority as they are commonly practised in organisations.

3.6 Roles based on Function

It is difficult to envisage a role structure for an organisation that does not include roles with functionally-based characteristics. Seniority alone is not normally sufficient to identify a position in an organisation effectively. As listed in section 3.4, three ways in which roles are based on functions are qualification, function, and work-process.

Roles based on qualifications or functions are similar in that they basically model the capability of an individual. In section 2, the idea of a generalisation or "isa" hierarchy [Moffett98] was reviewed. In fact, a specialisation hierarchy would be a more appropriate term, as it reflects the principle of specialisation either through qualification or assignment to a function. More specialised roles higher up inherit responsibilities from roles further down in the hierarchy.

Tasks that logically belong together can be grouped together to form work-process-based roles. In section 2, the idea of aggregating activities was presented by Moffett [Moffett98].

It is normal that a dependency exists between work-process based roles, the assigned qualification or function roles, and the level of seniority. In order to assign a work-process role, the user must have the appropriate qualification or function and seniority assigned to him.

3.7. Roles Based on Market

The roles based on functions and seniority determine the tasks that a user can carry out, but it is roles based on market characteristics that determine on which targets a user may carry out the tasks. There needs to be a correspondence with the asset affected. The role needs to link the role with the asset through a context, as we will demonstrate in section 4. We therefore refer to them as contextually-based roles. As with roles based on functions, there is the potential for a hierarchy similar to the aggregation of activities. In the previous section the concept of roles based on contextual information was reviewed [GMPT01], which has certain similarities; location is an example of this.

3.8. Concluding Remarks

Using organisational structure in this way – to define roles to form a basis of a security framework – has some significant advantages:

- It provides a clear focus for analysts and users eliciting requirements, as organisational groupings and the lines of authority are relatively easy to identify;
- Users are able to relate more easily to the defined roles; and
- Organisational procedures can be more readily translated into a security framework, because roles more accurately reflect the organisation.

4. An Analytical Role Modelling Framework

In this section, we present an analytical framework based on the discussions in the previous section. Our hypothesis is that the framework can be used by analysts in order to model and analyse access control requirements. We present the framework formally (in Z [Spivy92]) to reduce ambiguity, increase precision, and explore the scope of automated reasoning and analysis of security policies. We demonstrate the application of the framework through the use of an example of restricting accesses to patient records.

4.1. Analytical Framework

The framework comprises three types of roles: functional, seniority, and contextual. The contextual roles represent the market-based roles that we described in the last section. These roles relate to information assets through a context. For example, in a regional branch of a retail bank, bank tellers have access to accounts of customers of their branch only. Therefore, the branch outlet represents a context that has to be assigned both to the account and to the bank teller in order for access to be granted.

An access policy is modelled as a ternary relationship between a set of role sets, sets of operations, and an asset category. A role set is required to model restrictions where a combination of market, functional, and seniority-based characteristics are a prerequisite for accessing an operation or set of operations. The set of role sets can be considered as alternative conjunctions of roles, i.e. a user must satisfy at least one of the specified role sets. This enables allowing users with different functions or responsibilities to access the same functions. The reason for including sets of operations in an access policy is that sometimes tasks may be bundled to form a logical unit of work. An access policy relates only to a single asset category, however, an asset hierarchy has been included so that, through inheritance, the policy can be applied to more than one category of asset.

The basic types in the framework are as follows:

[*ROLE*]
Role is a basic type representing either functional, seniority, or contextual roles.

[*ROLE_INSTANCE*]
This is an instance of an assigned role. Users are assigned role instances rather than roles. In effect, a role instance is an assigned role. There are number of purposes for doing this. By having a role instance, useful instance information about the role assigned can be maintained. Primarily the context; if for example a role is based around a client of a consultant or a patient in a hospital, then a role instance modelling responsibility for the patient or client would be related to a context. Using a role instance enables the tracking of other information, such as who assigned the role, when it was assigned, and when it should expire.

[*CONTEXT_TYPE*]
This represents the type of context that can be used to match user contextual role instances with asset instances. A context type can be location or patient.

[*CONTEXT_INSTANCE*]
This represents the actual context that can be used to resolve an authorisation request.

[*ASSET_CATEGORY*]
An asset represents a category of information entities that we are trying to protect. All secure entities must be assigned to a category.

[*ASSET_INSTANCE]*
This represents an instance of an ASSET.

[USER]
This basic type represents the user in the system.

[OPERATION]
This represents the basic type for operation.

Using these basic types we can define the compound schema for an operation policy that models the access restriction as described above.

$$OPERATION\_POLICY \cong [RoleSets: \mathbb{P}(\mathbb{P}ROLE);\ Operations:\ \mathbb{P}\ OPERATION;\ AssetCategory: ASSET\_CATEGORY]$$

The schema below includes finite sets of seniority, functional, and contextual roles. A role-role mapping is used to represent the role hierarchy. For functional roles, a hierarchy serves to model inheritance whereas for seniority roles it models the lines of authority. A similar hierarchy has also been included for asset categories, rather similar to the generalisation hierarchy of a functional role model. At the bottom of the hierarchy are the most general asset categories and as we move up the hierarchy the categories become more specific. An example of a very basic asset type is static data. Static data is a term used in business process systems to describe data that rarely changes such as names and addresses account numbers and so on; another common category is that of transactional data. A more specific category could be financial transactions and even more specific would be a debit.

The hierarchies enable us to define policies at the desired level of abstraction. For example, a general policy could be defined for a financial transaction and this policy would be applicable to debit and credit transactions. *UserRoles* is a function that gives the role instances assigned to a user and the role type can be accessed through the *RoleType*. *AssetCategory* gives the category of an asset instance. *AssetContext* is a function that gives the context instances assigned to asset instances. In this model a policy containing a context role, such as patient, location or product group, is resolved by matching the context of a contextual role assigned to a user with one assigned to an asset instance. If a doctor wishes to access a patient record then this needs to be associated with a patient for whom the doctor has responsibility. In this instance the context is the patient.

Security

Roles: $\mathbb{F}$ ROLE
SeniorityRoles: $\mathbb{F}$ ROLE
FunctionalRoles: $\mathbb{F}$ ROLE
ContextualRoles: $\mathbb{F}$ ROLE
Contexts: $\mathbb{F}$ CONTEXT_TYPE
RoleContext: ROLE $\pfun$ CONTEXT_TYPE
RoleInheritance: ROLE $\leftrightarrow$ ROLE
RoleSeniority : ROLE $\leftrightarrow$ ROLE
Policy: $\mathbb{F}$ OPERATION_POLICY
AssetHierarchy: ASSET_CATEGORY $\leftrightarrow$ ASSET_CATEGORY
Assets: $\mathbb{F}$ ASSET_CATEGORY
Operations: $\mathbb{F}$ OPERATION
Users: $\mathbb{P}$ USER
UserRoles: USER $\pfun$ $\mathbb{P}$ ROLE_INSTANCE
RoleType: ROLE_INSTANCE $\fun$ ROLE
AssetCategory: ASSET_INSTANCE $\fun$ ASSET_CATEGORY
AssetContext: ASSET_INSTANCE $\fun$ $\mathbb{P}$ CONTEXT_INSTANCE
ContextType: CONTEXT_INSTANCE $\fun$ CONTEXT_TYPE
RoleInstanceContext : ROLE_INSTANCE $\pfun$ CONTEXT_INSTANCE

CheckAuthorisation

ΞSecurity
o?: OPERATION
u?: USER
ai?: ASSET_INSTANCE
r?: ROLE_INSTANCE

---

let InheritedAssets == ran ({AssetType ai?} $\lhd$ (AssetHierarchy *));
InheritedRoles == ran ((RoleType (UserRoles u?))
$\lhd$ (RoleHierarchy *))

• $\exists$p: Policy | o? $\in$ p . Operations $\land$ p . Asset $\in$ InheritedAssets
• $\exists$rs : p .RoleSets
• $\forall$fr: rs | fr $\in$ FunctionalRoles • fr $\in$ InheritedRoles $\land$
$\forall$sr: rs | sr $\in$ SeniorityRoles • sr $\in$ UserRoles $\land$
$\forall$cr: rs | cr $\in$ ContextualRoles •
($\exists$uri: UserRoles u? • (RoleType uri = cr $\land$ ($\exists$ci: AssetContext ai?
• (RoleInstanceContext uri = ci
$\land$ ContextType (RoleInstanceContext uri) = RoleContext cr
$\land$ ContextType ci = RoleContext cr))))

Check Authorisation above is used to check whether a policy exists for a user wishing to carry out an operation on an asset instance; e.g., a Doctor accessing a patient record of John Smith.

Two temporary variables are defined. *InheritedAssets* represents all asset categories to which the asset instance belongs. *InheritedRoles* is the set of roles and indirectly inherited roles that are attributed to the user. The predicate is true when at least one policy exists that is applicable to an asset category to which the asset instance belongs, relevant for the operation that is being requested, and the user possesses all the prerequisite roles of one of the policy role sets. If the policy contains a contextual role then a contextual role instance must be assigned to the user that is related to a context instance that matches a context instance assigned to the asset instance.

4.2. Example: a patient record access policy

We now present a healthcare example to illustrate how the above framework can be used. In this example there are two types of record: the medical record that contains diagnoses, observations, and treatment plans that are updated by medical practitioners, and the nursing record that records the treatment and observations from the nursing staff. A patient is assigned to a consultant who is generally either a physician or a surgeon. The consultant needs read and update permissions for the patient's medical records and read-only access for the nursing records. All nurses on the ward where the patient is located need read access to the medical records, and must be able to read and update the nursing record. As far as nursing care goes, the role set that is needed for read access includes a nursing qualification and an assignment to a ward. Medical practitioners who have access must be the consultant to whom the patient has been assigned.

This following schema represents the domain definitions that we need to enforce the policy. It includes the necessary role, asset, and policy definitions. The functional roles *Surgeon* and *Physician* inherit permissions of the role *MedicalPractitioner*. The seniority hierarchy for medical staff is partially represented with *Consultant* and *Registrar*. A registrar is a junior doctor who reports to a consultant. Access to a medical record by a registrar is achieved by delegation, but this has not been modelled here. In this simplified example, nurses are represented by a single functional role of *Nurse.*

The two basic asset categories are *MedicalRecord* and *NursingRecord*. The asset categories *TreatmentPlan* and *Diagnosis* are inherited from *MedicalRecord* that is modelled in the mapping *AssetHierarchy* and therefore policies that apply to *MedicalRecord* apply also to these. The context role *ResponsibleForPatient* models the assignment of patient to a consultant and *WardAssignment* the assignment of a nurse to a ward.

There are four policies. *ReadMedicalRecordPolicy* restricts read access to medical records to nurses that are assigned to the ward where the patient has been stationed and the consultant who has responsibility for the patient. Similarly, the policy *ReadNursingRecord* restricts read access of the nursing records to the same groups. *UpdateMedicalRecordPolicy* restricts the update of medical record to the consultant who has responsibility for the patient, while *UpdateNursingRecordPolicy* restricts update access of nursing records to nurses assigned to the ward where the patient is located.

HospitalAdminSecurityDomain

Security
MedicalPractioner, Physician, Surgeon, Nurse, AssignedWard,
ResponsibleForPatient : ROLE
Consultant, Registrar, Sister, Staff: ROLE
Location, Patient: CONTEXT_TYPE
ReadMedicalRecord, UpdateMedicalRecord,
ReadNursingRecord, UpdateNursingRecord: OPERATION
UpdateMedicalRecordPolicy: OPERATION_POLICY
ReadMedicalRecordPolicy: OPERATION_POLICY
NursingRecordPolicy : OPERATION_POLICY
TreatmentPlan,Diagnosis,MedicalRecord, NursingRecord: ASSET_CATEGORY

FunctionalRoles = {Physician,Surgeon,MedicalPractitioner}
SeniorityRoles = {Consultant, Registrar}
ContextualRoles = {AssignedWard, ResponsibleForPatient}
RoleContext = {ResponsibleForPatient ↦ Patient, AssignedWard ↦ Location }
RoleInheritance = {Surgeon ↦ MedicalPractitioner , Physician ↦ MedicalPractitioner}
AssetHierarchy = {TreatmentPlan ↦ MedicalRecord, Diagnosis ↦ MedicalRecord}

ReadMedicalRecordPolicy . RoleSets = {{MedicalPractitioner, Consultant
ResponsibleForPatient} , {Nurse,Location}}
ReadMedicalRecordPolicy . Operations ={ReadMedicalRecord}
ReadMedicalRecordPolicy . Asset = MedicalRecord

UpdateMedicalRecordPolicy . RoleSets = {{MedicalPractitioner, Consultant
ResponsibleForPatient}}
UpdateMedicalRecordPolicy . Operations = {UpdateMedicalRecord}
UpdateMedicalRecordPolicy . Asset = MedicalRecord

ReadNursingRecordPolicy . RoleSets {{Nurse,Location},{MedicalPractitioner, Consultant
ResponsibleForPatient }
ReadNursingRecordPolicy . Operations = {ReadNursingRecord}
ReadNursingRecordPolicy . Asset = MedicalRecord

UpdateNursingRecordPolicy . RoleSets = {{Nurse,Location}}
UpdateNursingRecordPolicy . Operations = {ReadNursingRecord}
UpdateNursingRecordPolicy . Asset = NursingRecord

The next schema shows instance data of an example.

HospitalAdminInstance

HospitalAdminConfig
JohnSmith, JudyClegg : USER
PhysicianInst, ResponsibleForPatientInst, ConsultantInst, NurseInst,
AssugnedWardInst: ROLE_INSTANCE
RichardCargill , GeriatricWard : CONTEXT_INSTANCE
MedicalRecordCargill,NursingRecordCargill : ASSET_INSTANCE

UserRoles
= {JohnSmith ↦ {PhysicianInst, ResponsibleForPatientInst, ConsultantInst},
JudyClegg ↦ {NurseInst , AssignedWardInst}
RoleType
= {PhysicianInst ↦ Physician, NurseInst ↦ Nurse,
ResponsibleForPatientInst ↦ ResponsibleForPatient,
ConsultanInst ↦ Consultant}
RoleContext = { AssignedWardInst ↦ GeriatricWard,
ResponsibleForPatientInst ↦ RichardCargill }
ContextType = {RichardCargill ↦ Patient}
AssetCategory = {MedicalRecordCargill ↦ MedicalRecord,
NursinglRecordCargill ↦ NursingRecord }
AssetContext = {MedicalRecordCargill ↦ {RichardCargill,GeriatricWard},
NursingRecordCargill ↦ {RichardCargill,GeriatricWard} }

There are two users. *JohnSmith* is a consultant physician who has responsibility for patient *RichardCargill*. In the mapping *UserRoles,* he is therefore assigned the roles instances *PhysicianInst* and *ResponsibleForPatientInst* that are of role types *Physician* and *ResponsibleForPatient* respectively, defined in the mapping *RoleType*. The role instance *ResponsibleForPatient* is linked to the patient context instance *RichardCargill* through the mapping *RoleContext.* The asset instance *MedicalRecordCargill* is linked to the patient of *RichardCargill* through the mapping *AssetContext* . These assignments mean that *JohnSmith* has the prerequisite role assignments to satisfy policies *ReadMedicalRecord*, *UpdateMedicalRecord* and *ReadNursingRecord* so that he can read medical and nursing records as well as update medical records associated with the patient *RichardCargill*.

JudyClegg is a nurse assigned to the ward *GeriatricWard* where the patient *RichardCargill* has been stationed. In the mapping *UserRoles* she is therefore assigned the roles instances *NurseInst* and *AssignedWardInst* that are of role types *Nurse* and *AssignedWard* respectively. The role instance *AssignedWardInst* is linked to the ward context instance *GeriatricWard* which in turn is mapped to the patient record *MedicalRecordCargill* in *AssetContext. JudyClegg* therefore has the prerequisite role assignments to satisfy the policies *ReadMedicalRecord, ReadNursingRecord* and *UpdateNursingRecord.*

5. Conclusion

In this paper, we have explored how the concepts underlying role-based access control can be related to organisational theory and used as a basis for an analytical framework for modelling access restrictions. This is a first step towards bringing security policy concepts into the requirements engineering domain. Through a simple example, we demonstrated how this framework can be used to specify an access policy.

There is still much that needs to be done, both in the general area of security requirements engineering [CILN02] and to extend the work in this paper. The example given in this paper does not demonstrate how the role assignments and access requests can be integrated with the core business functions. Other issues that remain to be explored include delegation, authorisation procedures, and role constraints that can be used to enforce the separation of duties.

Although formal notation has served well to demonstrate the principles of this framework, amongst practitioners formal methods are not widely used and it is still not clear what kind of formal and automated analysis would be useful in this context. Further research may also be carried out to explore how the framework could be integrated into more widely accepted method or expressed in a less formal notation such as the Unified Modelling Language.

Checking the consistency of security requirements is also crucial. Role and asset hierarchies offer a way of defining policies at different levels of abstraction but may also introduce inconsistency, particularly when access rights arise indirectly through the inheritance structure that are not foreseen.

Finally, the link between the requirements model and the implementation model needs to be investigated, when implementation options range from custom solutions to the use of standard access control models, such as OASIS.

Acknowledgements

We would like to acknowledge the advice and feedback to our colleagues in the Security Requirements Group at The Open University, particularly Jonathan Moffett for his comments and views on earlier drafts of this paper. Thanks also to Pat and John Crook, former Clinical Director of Wrightington Hospital, Wigan, and Leigh NHS Trust, for insights into the procedures for accessing patient records.

References

[AE01] A.I. Anton & J.B.Earp,"Strategies for Developing Policies and Requirements for Secure Electronic Commerce Systems", *Recent Advances in Secure and Private E-Commerce*, Kluwer Academic Publishers, 2001.

[Anderson01] R. Anderson, *Security Engineering,* Wiley, 2001.

[BBF00] E.B. Piero, A. Bonatti, and E. Ferrari, "TRBAC", *Proceedings of the 5th ACM Workshop on Role-based Access Control*, July 2000, pages 21-30.

[BL73] D. Bell and L.J. La Padula, "Secure Computer Systems: A Mathematical Model", *MITRE Technical Report 2547*, Volume II 1973.

[BLM01] J. Bacon, M. Lloyd, and K. Moody , "Translating Role-based Access Control within Context", *Proceedings of International Workshop Policies for Distributed Systems and Networks (Policy 2001)*, Bristol, Jan 2001, LNCS Springer-Verlag, , pages 107-119.

[BSa00] E. Barka and R. Sandhu, "A Framework for Role Based Delegation Model", *Proceedings of 23rd National Information Systems Security Conference*, Baltimore Oct 16-19 2000, pages 101-114.

[CLSDAA01] M..J. Covington, W. Long, S. Srinivasan, A.K. Dev, M. Ahamad, and G.D. Abowd, "Securing Context-Aware Applications Using Environment Roles", *Proceedings*

of the 6th ACM Symposium on Access Control Models and Technologies, May 2001, pages 10-20.

[CILN02] R. Crook, D. Ince, L. Lin, and B. Nuseibeh, "Security Requirements Engineering: When Anti-Requirements Hit the Fan", (to appear in) Proceedings of IEEE International Requirements Engineering Conference (RE'02), Essen, Germany, September 2002.

[CW87] D.D Clark and D.R Wilson, "A Comparison of Commercial and Military Computer Security Policies", *Proceedings of the IEEE Symposium on Security and Privacy*, 1987, pages 184-194.

[GMPT01] C.K. Georgiadis, I.Mavridis, G. Pangalos, and R.K. Thomas, "Flexible Team-based Access Control Using Contexts", *Proceedings of the 6th ACM Symposium on Access control models and technologies*, May 2001, pages 21-27.

[Handy85] C. Handy, *Understanding Organizations*, Penguin, 1985.

[Mintzberg92] H. Mintzberg, *Structure in Fives: Designing effective organisations,* Prentice Hall, 1992.

[Moffett98] J.D. Moffett, "Control Principles and Role Hierarchies", *Proceedings of the 3rd ACM Workshop on Role-based Access Control*, October 1998, pages 63-69.

[NO95] M. Nyanchama and S. L. Osborn, "Modeling Mandatory Access Control in Role-based Security Systems", *In D.L. Spooner, S.A. Demurjian, and J.E. Dobson, editors, Proceedings of the IFIP WG 11.3 9th Annual Working Conference on Database Security*, Chapman & Hall, 1995, pages 129-144.

[SCFY96] R. Sandhu, E. Coyne, H. Feinstaein, and C.Youmann "Role-Based Access Control Models", *IEEE Computer*, 29(2): 38-47*,* Feb 1996.

[SFK00] R. Sandhu, D. Ferraiolo, and R. Kuhn, "The NIST Model for Role Based Access Control: Towards A Unified Standard", *Proceedings of the 5th ACN Workshop on Role-Based Access Control (RBAC-00*), Berlin Germany, July 26-27 2000, pages 47-64.

[SM98] R. Sandhu and Q. Munawer, "How to do Discretionary Access Control Using Roles", *Proceedings of the 9th ACM Workshop on Role-based Access Control,* October 1998, pages 47-54.

[Spivey92] J.M. Spivey, *The Z - Notation A Reference Manual*, Second Edition, Prentice Hall, 1992.

[Thomas97] R.K. Thomas, "Team-Based Access Control A Primitive for Applying Role Based Access Controls in Collaborative Environments", *Proceedings of the 2nd ACM Workshop on Role-based Access Control,* Fairfax, USA 1997.

[YMB01] W. Yao, K. Moody, and J. Bacon, " A Model of OASIS Role Based Access Control and its Support for Active Security", *SACMAT'01,* Chantilly Virginia, USA, 2001.