

# IY5512: Coursework 6: Worked solutions

---

1. *Describe the operation and use of an access control matrix, and explain why operating systems rarely implement access control using this approach.*

An access control matrix is a matrix in which the columns are indexed by objects (i.e. the resources to be protected by the system) and the rows are indexed by the subjects (i.e. the active entities wishing to gain access to the objects). The matrix entries are (sets of) access operations which the subject may perform on the object. This model is the foundation of many theoretical security models.

It is not suitable for direct implementation, for the following reasons.

- The matrix is likely to be extremely sparse, and therefore implementation is inefficient.
- Management of the matrix is likely to be extremely difficult if there are tens of thousands of files and hundreds of users (resulting in millions of matrix entries).

2. *The following three access control mechanisms were discussed in the course notes:*

- *access control matrices;*
- *capabilities;*
- *access control lists.*

*For each access control mechanism, describe the complexity of:*

- *determining authorised access during execution;*
- *adding accesses for a new subject;*
- *deleting all accesses for a particular subject;*
- *determining all subjects which have access to a particular object;*
- *creating a new object to which all subjects have access by default.*

**Access control matrix:**

- determining authorised access during execution: this is simple (just look up the appropriate entry in the table);
- adding accesses for a new subject: not so simple – a new row in the matrix will need to be created and completed;
- deleting all accesses for a subject: relatively simple, since it simply means deleting a row in the access control matrix;
- determining all subjects which have access to a particular object: relatively simple, since it involves checking the relevant column of the matrix;

- ease of creating a new object to which all subjects have access by default: relatively simple, since it simply means creating a column in the access control matrix with all entries 'positive'.

**Capabilities** (i.e. a list for each subject indicating the objects that can be accessed, and that corresponds to a row of an access control matrix):

- determining authorised access during execution: simple;
- adding access for a new subject: simple;
- deleting access by a subject: simple;
- determining all subjects which have access to a particular object: not simple (possibly infeasible), since all capabilities held by every subject will need to be checked;
- creating a new object to which all subjects have access by default: not so simple, since every subject will need to be given a new capability.

**Access control list** (i.e. a list for each object that indicates the subjects that have access to that object, and that corresponds to a column of an access control matrix):

- determining authorised access during execution: reasonable, as long as access to the per-object access control list is straightforward;
- adding access for a new subject: not simple, since every object's access control list will need to be modified (at least all those objects to which the new subject is to be granted access);
- deleting access by a subject: not simple, since every object's access control list will need to be checked, and, if necessary, modified;
- determining all subjects which have access to a particular object: simple – just check the ACL;
- creating a new object to which all subjects have access by default: simple.

3. *What is the difference between the notions of a group and a role? Is there a difference at all?*

This is a tricky one ... The following text is taken from here:

<http://csrc.nist.gov/groups/SNS/rbac/faq.html>

There is a superficial similarity between RBAC roles and traditional groups. As normally implemented, a group is a collection of users, rather than a collection of permissions, and permissions can be associated with both users and the groups to which they belong. The ability to tie permissions directly to users in a group-based mechanism can be regarded as a 'loophole' that makes it difficult to control user-permission relationships. RBAC requires all access through roles, and permissions are connected only to roles, not directly to users. Another aspect of RBAC that distinguishes it from traditional group mechanisms is the concept of a session, which allows activation of a subset of roles assigned to a user. Core RBAC includes those systems with a robust group/ACL mechanism that supports the construction of a many-to-many relation among users and permissions.

This fits with my own view that the difference is mainly about how the notions are implemented, rather than any fundamental difference in the ideas.

4. *Describe an information flow access control model for confidentiality, covering the notions of security labels, sensitivity and clearance (and how they are applied to subjects and objects). How is a partial ordering used to control the flow of information? What security breaches relating to read and write operations does this model address?*

The notion of information flows can be used to construct an access control model. The following access control policy enforces confidentiality requirements.

Every object and subject has a security level (security label). The label on an object indicates the sensitivity of the object, and the label on a subject indicates the clearance of the subject.

The set of security labels is a (partially) ordered set. Information flows must respect the partial ordering, in that information can only flow from entity  $a$  to entity  $b$  if  $a \leq b$  (where  $\leq$  is the partial ordering).

This type of policy prevents information leaks arising from inappropriate read actions. For example, it prevents an unclassified user reading classified information. It also addresses information leaks arising from inappropriate write actions. For example, it prevents Trojan horses downgrading classified information, and prevents classified information being printed to an unclassified printer.

5. *Give a simple example of a case where a problem can arise when constrained RBAC is used in conjunction with hierarchical RBAC.*

Possible issues arise when constrained RBAC is used in conjunction with hierarchical RBAC. This is because a superior inherits the role assignments of all his/her subordinates:

- this seriously complicates checking of static and dynamic constraints;
- it also means that changes in the hierarchy also involve checking all the constraints.

This could become very time-consuming in large systems. A simple example is presented here:

<http://ldapwiki.willeke.com/wiki/RBAC%20Hierarchical>