Access Control

principals, objects and their operations

Overview of Today's Lecture:

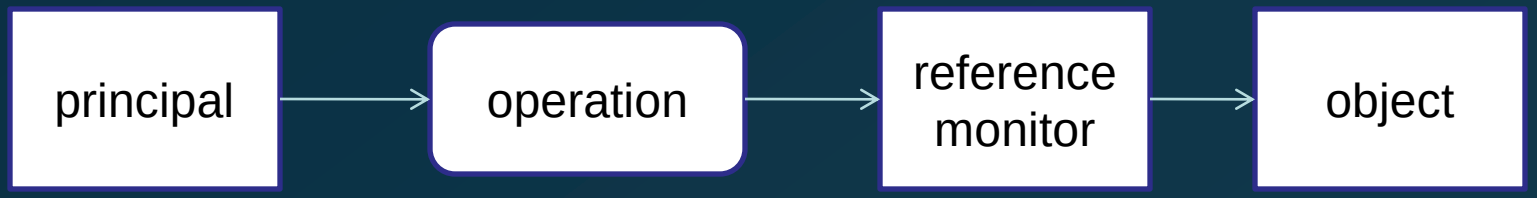
- Authentication and Authorisation
- Access Operations
- Ownership
- Access Control Structures
  - Access Control Matrix
  - Capabilities
  - Access Control List
- Intermediate Controls

Background:

- Logged on
- Protect your files
- Some are private and some public
- Language needed to express this
- Mechanisms needed to enforce it

Authentication and Authorisation:

- *Subject/Principal* - an active entity
- *Object* - being accessed
- *Access operation*
- *Reference monitor* – grants or denies access



continued…

- Access Control - 2 steps
  - *Authentication* – who requested access?
  - *Authorisation* – who is allowed to access *x*?

- *Subject* – operates on behalf of principals (human users)
- *Principal* – a name associated with a subject

Principal vs. Subject:

- Principal -

  “An entity that can be granted access to objects or can make statements affecting access control decisions”

  e.g. user identity in an OS

  - when discussing security policies

- Subject –

  “An active entity within an IT system”

  e.g. process running under a user identity

  - when discussing operational systems enforcing policies

Subject vs. Object:

- *Object* – files or resources (memory, printers, etc…)
- Not a clear distinction between the two

Subjects and *Objects* merely distinguish between the active and passive party in an access request

Two options of focusing control:

- what a subject is allowed to do
- what may be done with an object

Access Operations:

- from reading and writing to method calls
- various systems use different access operations
- sometimes similar operations have different meanings

Access Modes

- *Observe* – look at contents of an object
- *Alter* – change contents of an object

Access Operations:

Access Rights – Bell-LaPadula model

- *execute, read, append, write*
- operates on files only

Access Attributes – Multics OS

- distinguishes between data and directory access attributes
- write = append (Bell-LaPadula)

Continued…

Unix

- *read* – reading from a file / list contents of dir
- *write* – writing to a file / create, rename file in dir
- *execute* – executing a (program) file/ search dir

Windows - (*standard permissions*)

- read control
- delete
- write DACL (modify access control list)
- write owner (modify owner of a resource)
- synchronise

Ownership:

Who is in charge of setting security policies?

- *Owner* can be defined for each resource
- *Owner* decides who gets access (*discretionary policy)*

or

- A system wide policy (*mandatory* policy)

Most OSs support the concept of ownership

Access Control Structures:

- Help express access control policy
- A way to check that policy is captured correctly

- Access Control Matrix
- Capabilities
- Access Control Lists

Access Control Matrix:

- Access rights defined individually for each combination of subject and object
- An abstract concept
- Not very suitable direct implementation
- Not very scalable

	Marks.doc	Edit.exe	Game.exe
Alice	-	{execute}	{execute, read}
Bill	{read, write}	{execute}	{execute, read, write}

Capabilities:

- Access rights kept with subject or object
- Every subject is given a *capability*

Capability – an un-forgeable token specifying the subject's access rights

- Corresponds to a row in a an access control matrix

e.g.

Alice's capability: edit.exe: execute, game.exe: execute, read;

Capabilities:

- Typically associated with discretionary access control
- Subject can pass on its capabilities
- Not a widely used security mechanism
- Difficult to get an overview of permissions of an object
- Difficult to revoke capability

Access Control List (ACL):

- Access rights to an object stored with the object itself
- Corresponds to the column of access control matrix

e.g. *ACL for edit.exe: Alice: execute; Bill: execute;*

- Management of individual subjects cumbersome
- *Groups* – derive access rights from user's group
- In Unix – *user*, *group* and *others*

continued…

- Good for managing access to objects
- Overview of permissions given to users difficult

Summary

- Managing access control - complex in large systems
- Tedious and error prone
- Subject - or Object-only based access control limited

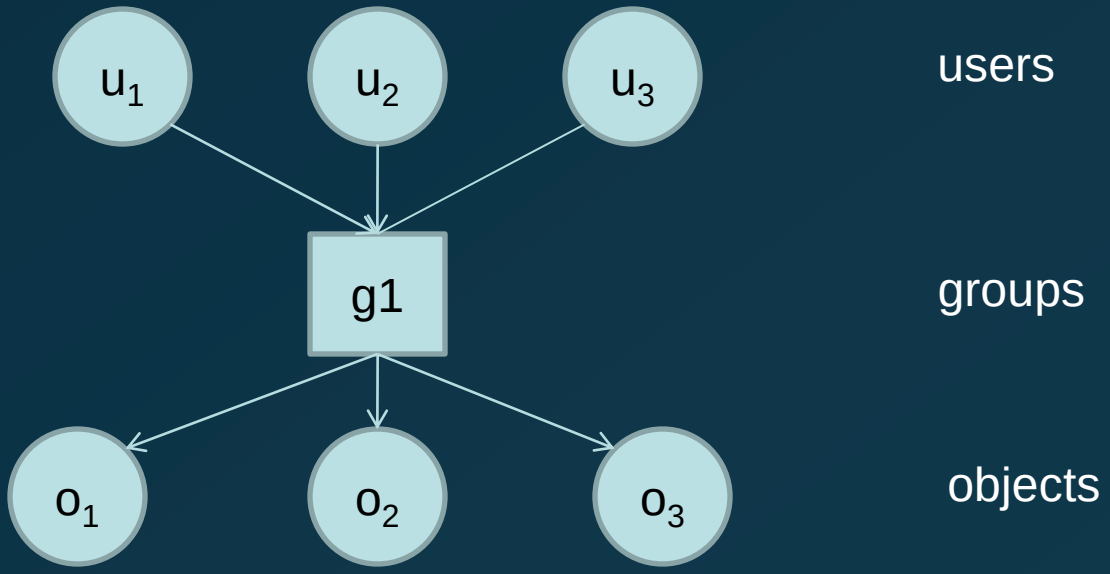
Intermediate Controls:

- Problems of complexity solved by indirection

- Groups
- Negative Permissions
- Privileges
- Role-Based Access Control
- Protection Rings

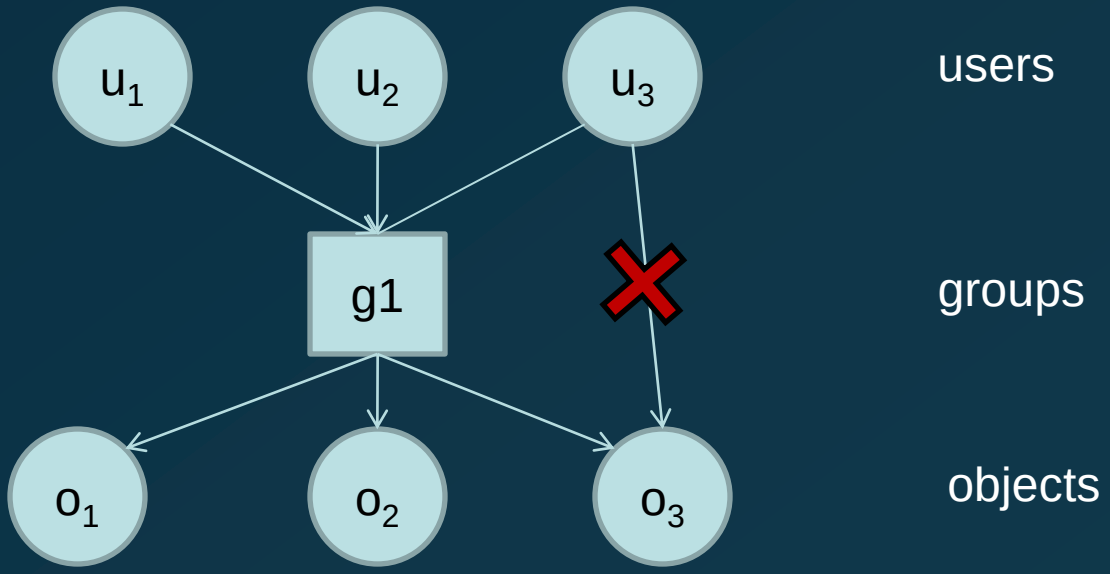
Groups:

- Users with similar access rights collected in groups
- Groups are given permissions to access objects



Negative Permission:

- An access operation a user is not allowed to perform
- *Policy conflict* – negative permission contradicts the positive one – resolved by reference monitor



Privileges:

- Collection of rights to execute certain operations
- An intermediate layer between subjects and operations
- Associated with operating system functions
- Activities such as administration, backup, network access

Role-Based Access Control (RBAC):

- Privileges come predefined with OS
- *A Role* - Collection of application specific operations
- Subjects derive access rights from the role they perform
- RBAC focuses on users and jobs they perform

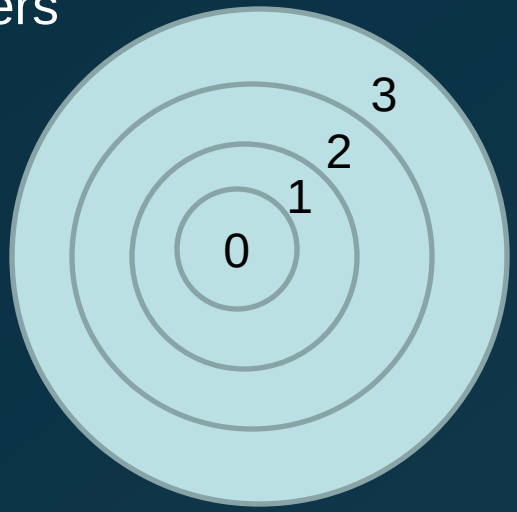
continued…

Layers (between subject and objects):

- Roles – collection of procedures, assigned to users
- Procedures – high level access control methods
- Datatypes – each object of certain data type

Protection Rings:

- Hardware based access control
- Each subject and object assigned a number depending on importance
- Decision made by comparing subject's and object's numbers



0 – operating system kernel

1 – operating system

2 – utilities

3 – user processes

Summary:

- Access Control
- Its structures

Next Lecture

Enforcing Access Control

End