Stockholm University

Intro to InfoSec – Access Control

Stefan Axelsson, Nov. 2020

Today – Access Control

Stockholm University

- Chapter 2.1 in Pfleeger et.al. *Security in Computing*, 5th ed.
- Access Control – The art of keeping peoples fingers out of the cookie jar
  - What/Why
  - How
    - Some different methods

Access Control

Stockholm University

- How do we protect general objects?
- By implementing access controls
- We want someting flexible so that policies can be implemented/changed easily at a fine granularity
  - One user read something, another change same thing, others no access at all
- We talk of
  - Subjects
  - Objects
  - Access modes

Subject, Object, Access mode

Stockholm University

- Subjects
  - Human users, often represented by surrogates: programmes running on behalf of user
- Objects – Things on which action can be performed
  - Files, tables, programmes, memory objects, hardware devices, strings, data fields, network connections, processors
  - Also users (programmes representing a user) – OS can act on user – Allow execution of programme, assigning privileges to user etc.

Subject, Object, Access mode

Stockholm University

- Access modes
  - Any controllable actions of subjects on objects
    - Read, write, modify, delete, execute, create, destroy, copy, export, import etc.
    - You've probably never used a system that implemented the whole list above
      - I can't think of one off the top of my head…
    - And there are more modes than those above – Append for example
      - Note complexity – If you have *read*, *write* and *create*, does it matter if you don't have *copy*?

Effective Policy Implementation

Stockholm University

- Access control is a mechanism
  - But you of course need a policy
    - Who should be allowed to do what?
      - This can quickly become tedious
    - You need higher level policy – Most don't really, but just do something "reasonable"
      - At least when it comes to AC implementation
- MAC/DAC
  - Mandatory AC vs. Discretionary AC
    - Almost all systems DAC…

Effective Policy Implementation

Stockholm University

- For AC policy to work you have to:
  - Check every access
    - If not you can't e.g. revoke a users privilege to access object
  - Enforce least privilege
    - Just as much access as is needed
      - Even if harmless – Security weakness
  - Verify acceptable usage
    - Access is yes/no – But certain objects should be accessed in certain ways
      - Stack – Only push/pop/clear
        - So not only *who*, but also *how*

Effective Policy Implementation

Stockholm University

- Tracking (audit)
  - Also keep track of accesses
    - Do we have accesses that aren't needed
    - Failure? What was happening before
    - Is someone abusing access?
      - IDS – Intrusion Detection System
        - Is someone abusing privileges?
        - Is someone a masquerader?
  - Often needed to make sure policy and mechanism works
    - Run in "try mode" first and log all violations to see how policy needs to change

Effective Policy Implementation

Stockholm University

- Granularity
  - Alice has access to whole computer and all that's on it
    - Useful in some circumstances, but not others
  - Bob has read access to these three bytes
    - Need to do lots of checking for what benefit?
  - Databases often have more fine grained AC
    - *Database system* has access to all
      - But it makes fine grained decisions given internal policy – Names but not salaries etc.
    - Note that OS often has a granular view
      - What's in a network connection? Can't say

Reference monitor

- From Jim Anderson's report in the early 70-ties
  - Codified ancient principles from castles etc.
- HW and SW that is
  - Always invoked; validates every access attempt
  - Immune from tampering
  - Assuredly correct
- It's not a "thing" rather it's the sum of all things in your computer that makes above true
  - It's a concept, not SW/HW
  - Typically spread out all over HW/OS – Not ideal

Implementation Methods

Stockholm University

- How do we implement AC?
  - Access Control Directory
  - Access Control Matrix
  - Access Control List – ACL
  - Capabilities

Methods – Access Control Directory

Stockholm University

- Keep a per user directory of all the files (objects) a user can access
  - Owner has ”control” over who to give access
- Easy to implement – One list per user
  - However…
    - Large list if many users can access common itemes – e.g. subroutine library
      - One entry per user, all up to date
        - Deletion…
    - Revocation
      - Need to check all users for every object
    - Can become complex with pseudonyms

Methods – Access Control Matrix

- One row per subject, and column per object
- Previous problems disapear
  - But take space, since sparse
    - Most subjects no access to most objects
    - Triplet <subject, object, right> takes less space
      - But then it's not so easy to operate on anymore

TABLE 2-8 Access Control Matrix

	Bibliog	Temp	F	Help .txt	C_ Comp	Linker	Clock	Printer
USER A	ORW	ORW	ORW	R	X	X	R	W
USER B	R	—	—	R	X	X	R	W
USER S	RW	—	R	R	X	X	R	W
USER T	—	—	R	X	X	X	R	W
SYS MGR	—	—	—	RW	OX	OX	ORW	O
USER SVCS	—	—	—	O	X	X	R	W

Methods – Access Control List

Stockholm University

- ACL *"akl"* for short
- Take columns from matrix
  - One list for each **object** (directory=subject)
  - Mitigate problems by having wildcards
    - Checked last – You not just user, but also group, and maybe something else
      - You *write*, group *read*, others *execute*
        - Matched against your actual (list) of groups, etc.

Methods – Capabilities

Stockholm University

- User keeps track, OS relieved (to some extent)
- Capability – Unforgable token that grants access
  - Could be communicated and subclassed
    - Kerberos – Network login and AC system used this
- Must be unforeable – OS doesn't hand it out, but keeps it for you and gives you identifier
  - Or (Kerberos) encrypt it
- Distribution must be managed
  - Include subject in ticket so it can be checked
- Used for increased granularity – Typically on top of something
  - Linux – Not just superuser, but now capabilities

Procedure Oriented Access Control

Stockholm University

- Our access methods were coarse
  - Read, write, etc. to whole object
- What if we could specify procedure (code) that runs and checks complex behaviour?
  - ”Smart contracts”
- Often providing an API that’s the only way to access
  - Not write user database but *set*/*update* password, *add user*, *lock account*, etc.
- Increased security but no fast easy access code must run every time
  - In most systems you simulate this when needed

Role-Based Access Control

Stockholm University

- Keeping track of subjects difficult if they change roles
  - It's on Bob's machine, but Bob doesn't work here anymore…
- So instead we have roles to which users are assigned
  - Alice and Bob are *system administrators*
    - Sys admin allowed to add/delete users, remove files etc.
    - Eve *backup admin* only allowed to use tape station
      - But Eve's been promoted – Just add her to the *sys admin* role
- A subject that's seen a lot of study; surprisingly complex…

Summary

Stockholm University

- Subject, Object, Access right
- Most systems do something along these lines but few do anything serious
  - SELinux, Kerberos, Blockchain smart contracts etc.
- Concepts: MAC/DAC, ACL, Capabilities etc.
- Is a fundamental part of all (computer/information) security but often not used well
  - C.f. Edward Snowden – Even NSA couldn't do it properly, and if they couldn't…
    - Actually consequence of 9/11 – Lack of sharing became oversharing