Access Control

Friday, March 16, 2018 12:54 PM

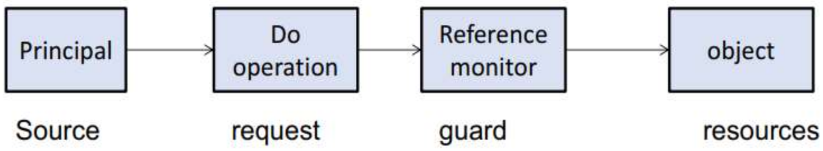
Security boundary - what binds the protection mechanism

Network security - information flow across layers
System security - objects can be data, need to be careful of who can rwx

*Different application domains -> different requirements

- Selective restriction

Access Control Model



- Reference monitor grants/ denies access
  - Enforcer
- Principal = human user, Subject = processes operating on behalf of principals

Ownership

- Owner decides = Discretionary access control
- System-wide policy = Mandatory access control (everyone must follow)

Access

- r: read, w: write, x: execute, s: execute as owner, o: owner

Access Control Matrix

- Principals vs. objects
- Very large to be explicitly stored
- Access Control List (ACL) - by object
  - Access rights to object stored as LL
  - Unix file system
- Capabilities - by subject as LL
- Group subjects/ objects and define access rights on the group

Intermediate Control

- Group, privilege, role-based

Group/ Privileges

- Owner, group, world
- Owner may not have all permissions (e.g. submitting homework - cannot read)

Role-based access control

- Least privilege principle

Protection rings

- Lower ring = higher privilege
- Unix: superuser + user

Bell-LaPadula (BLP Model)

- Higher level = higher security (opposite from protection ring)
- Confidentiality (no information flowing down)
- No read-up, no write-down
- **Implications of write up?

Biba

- Integrity (no information flowing up)
- No write up, no read down

Model with both properties = subject can only r/w to same level (not practical)

Unix

- File System Permission
  - Owner, group, other/ world
- Principals = user-identity (UID) and group-identity (GID)
  - Information stored in pw file (/etc/passwd)
  - * = hash of password, meant to be secure and everyone can read
- Subjects = processes (ps -alx)
- Superuser, UID = 0, no security checks
- Check if owner, then group, then other
  - Owner/ superuser can change the permission bits
- Searchpath
  - Prioritise current directory?
  - Specify the full path to prevent invocation of malicious programmes

Controlled invocation

- Set of operations/ programs with superuser privilege
- Real/ effective UID, privilege escalation
  - s = set SUID, enabled
- Process files follow subject
- Create processes to change specific parts of file (e.g. editprofile)
  - Process can access sensitive information (e.g. employee)
  - Bridge programme, interface (can only be built by root)
  - Process temporarily elevated to superuser (root)
  - May be exploited for privilege escalation (attack)
  - **Important for secure programming and software security
- SavedUID, RealUID, EffectiveUIP
  - Temp savedUID allows temporary degrading of privilege
  - Privileged UID stored in temp