Advanced Systems Security: Integrity

Trent Jaeger
Systems and Internet Infrastructure Security (SIIS) Lab
Computer Science and Engineering Department
Pennsylvania State University
Choosing Security Goals

- Can we use the ideas for secrecy goals for integrity?
  - Kind of…
Least Privilege

• The protection mechanism should force every process to operate with the minimum privileges needed to perform its task.

• Due to Saltzer and Schroeder (of Multics project)

• One of many “design principles” in their paper “The Protection of Information in Computer Systems” (1975)

• Others
  ‣ Principle of Psychological Acceptability
  ‣ Principle of Fail Safe Defaults
Least Privilege

- How to compute least privilege?
  - Aim: Determines the permissions required for the program to run effectively

- Run the program and see what permissions are used
  - Proposed for a system called Systrace
  - SELinux audit2allow: take denied permissions and add them to policy
  - AppArmor Profile Wizard: Build an approximate profile statically and
Least Privilege

• Is a good goal because...

• Is a poor goal because...

• Can we use it to verify a policy is secure?
Least Privilege

• Is a good goal because…
  ‣ Unnecessary permissions lead to problems (confused deputy)
  ‣ Accounts for function

• Is a poor goal because…
  ‣ Task permissions may conflict with security
  ‣ How do we know when a permission is necessary, but makes the system insecure?

• Can we use it to verify a policy is secure?
  ‣ No. *It defines a policy based on function, not security.*
Information Flow for Integrity

• Another approach looks at the authorized flow of information among processes via objects
Idealized Security

• **Biba Integrity**
  - **Integrity requirement**: Do not depend on data from lower integrity principals
  - Only permit information to flow from high integrity to lower integrity
  - E.g., Can only read a file if your integrity level is dominated by the file’s
Biba and BLP

- Should you use the same policy for BLP and Biba?
- Should you use the same labels for BLP and Biba?
Practical vs. Ideal

- Do these idealized approaches based on information flow enable practical realization of OS enforcement?
- Secrecy is possible in some environments
  - Implemented in a paper world, previously
- Integrity has not been realized in practice
  - Many processes provide high integrity services to others
- Result: Depend on many applications to manage information flows
Assured Guards

- What do we do if a system needs an information flow from low integrity to high?
  - E.g., reading from a network socket
- Not authorized by Biba
  - Unless subject is fully assured to upgrade to high integrity or discard low integrity data
    - Called a guard
- Must be verified to be correct (like a reference monitor)
  - Why?
LOMAC

• Subjects and objects have an integrity label
  ‣ Level and category in a lattice policy

• When subject reads an object of a lower integrity label in lattice
  ‣ Subject’s label is lowered to that of object
  ‣ Define subject’s label in terms of objects accessed

• When subject writes to an object of a higher integrity label in lattice
  ‣ Write is denied
  ‣ Read is still allowed
Biba vs LOMAC

• What is allowed and what is the resultant label?
  ‣ Lattice A $\rightarrow$ B $\rightarrow$ C

• Subject at A reads object at C
  ‣ Biba?
  ‣ LOMAC?

• Subject at C writes object at A
  ‣ Biba?
  ‣ LOMAC?

• Subject at C reads from object at A
Self-Revocation

- Can cause revocation of own access to objects in LOMAC

![Diagram of process steps]

**Step 1:** initial state.
- Level 2
- `ps` to `grep`

**Step 2:** `ps` reads file.
- Level 2
- `ps` to `pipe` to `grep`
  - File: `/proc/327`
  - Level 1

**Step 3:** demotion.
- Level 1
- `ps` to `pipe` to `grep`

**Step 4:** pipe write denied.
- Level 1
- `ps` to `pipe` to `grep`
Avoid Self-Revocation

• What could you do to avoid self-revocation in some cases?
Avoid Self-Revocation

• What could you do to avoid self-revocation in some cases?
  ‣ Add “floors” for subjects
    • Cannot open any objects below floor label (like Biba)
  ‣ This was done in the IX system
    • An MLS UNIX system by McElroy and Reeds
Information Flow

• Is a good goal because…

• Is a poor goal because…

• Can we use it to verify a policy is correct?
Information Flow

• Is a good goal because…
  ‣ No false negatives – an attack requires an illegal information flow
  ‣ Can define data and functional security requirements

• Is a poor goal because…
  ‣ Function may conflict with security
  ‣ How do we know when a permission is illegal, but is necessary for functional requirements?

• Can we use it to verify a policy is correct?
  ‣ Yes. *It defines a policy based on security. But what about exceptions?*
Clark-Wilson Integrity Model

• Goal: define integrity in terms of commercial terms rather than military (MLS/Biba)

• Insights?
Clark-Wilson Integrity Model

• Model consists of a set of certification and enforcement rules governing integrity

• Own terms
  ‣ CDI – Constrained Data Items (High integrity data)
  ‣ UDI – Unconstrained Data Items (Low integrity data)
  ‣ IVP – Integrity Verification Procedure (certify CDIs)
  ‣ TP – Transformation Procedures (High integrity programs)
Clark-Wilson Integrity Model

• Model consists of a set of certification and enforcement rules governing integrity
  ‣ C1—When an IVP is executed, it must ensure the CDIs are valid.
  ‣ C2—For some associated set of CDIs, a TP must transform those CDIs from one valid state to another.
  ‣ C3—Allowed relations must meet the requirements of “separation of duty.”
  ‣ C4—All TPs must append to a log enough information to reconstruct the operation.
  ‣ C5—Any TP that takes a UDI as input may only perform valid transactions for all possible values of the UDI. The TP will either accept (convert to CDI) or reject the UDI.
Clark-Wilson Integrity Model

- Model consists of a set of certification and enforcement rules governing integrity
  - E1—System must maintain a list of certified relations and ensure only TPs certified to run on a CDI change that CDI.
  - E2—System must associate a user with each TP and set of CDIs.
  - E3—System must authenticate every user attempting a TP.
  - E4—Only the certifier of a TP may change the list of entities associated with that TP.
Clark-Wilson Integrity Model

• How does it work?

• Certify TPs and IVPs
  ‣ IVPs certify CDIs and TPs modify them
  ‣ TPs must also be able to handle an UDIs they receive securely

• Run the system
  ‣ Authenticated users can modify a CDI if and only if:
    • They can access TP and CDI and
    • TP is authorized to change CDI
Clark-Wilson Results

- Was this used?
  - Not so much
  - Why do you think it may be difficult to apply in practice?

- Are the information flows authorized different than Biba or LOMAC?
Impact on Labeling and Transition

- How should security goals impact labeling and transitions?
- **Labeling** – assignment of objects to labels
  - What are the requirements for various secrecy/integrity levels?
- **Transitions** – conditions that determine when a label will change
  - What are the requirements for causing a label change?
- What experience do we have?
Labeling

• How do we choose the label of a subject or object when created?
  ▸ Traditional: same label as creator

• Alternatives
  ▸ Based on certification
    • E.g., code hashes for integrity verification
  ▸ Based on location of created object
    • Store this with garbage, so make it garbage
  ▸ Based on creator preference
    • Trust the creator to decide (within a range)
Transitions

• When do we change the label of a subject or object? And to what?
  ▸ Traditional: don’t change

• Alternatives
  ▸ Based on operations performed by subject or on object
    • Low-water mark integrity; High-water mark secrecy
    • There are access control models based on these
  ▸ Based on operations of others
    • E.g., When another process receives untrusted input, downgrade the whole system
Take Away

- To build a secure system, we need to define a mandatory protection system
  - But, what should guide the design?

- Security goals
  - Usually serves as a rough guide

- Types of security goals – biased toward security or function
  - Functional: least privilege; Security: information flow

- Need to develop approaches to design goals for entire mandatory protection system – from function and security