Clark-Wilson Integrity Model

- Integrity defined by a set of constraints
  - Data in a *consistent* or valid state when it satisfies these
- Example: Bank
  - *D* today's deposits, *W* withdrawals, *YB* yesterday's balance, *TB* today's balance
  - Integrity constraint: $D + YB - W$
- *Well-formed transaction* move system from one consistent state to another
- Issue: who examines, certifies transactions done correctly?

Entities

- CDIs: constrained data items
  - Data subject to integrity controls
- UDIs: unconstrained data items
  - Data not subject to integrity controls
- IVPs: integrity verification procedures
  - Procedures that test the CDIs conform to the integrity constraints
- TPs: transaction procedures
  - Procedures that take the system from one valid state to another

Certification Rules 1 and 2

CR1 When any IVP is run, it must ensure all CDIs are in a valid state

CR2 For some associated set of CDIs, a TP must transform those CDIs in a valid state into a (possibly different) valid state

- Defines relation *certified* that associates a set of CDIs with a particular TP
- Example: TP balance, CDIs accounts, in bank example

Enforcement Rules 1 and 2

ER1 The system must maintain the certified relations and must ensure that only TPs certified to run on a CDI manipulate that CDI.

ER2 The system must associate a user with each TP and set of CDIs. The TP may access those CDIs on behalf of the associated user. The TP cannot access that CDI on behalf of a user not associated with that TP and CDI.

- System must maintain, enforce certified relation
- System must also restrict access based on user ID (*allowed* relation)

Users and Rules

CR3 The allowed relations must meet the requirements imposed by the principle of separation of duty.

ER3 The system must authenticate each user attempting to execute a TP

- Type of authentication undefined, and depends on the instantiation
- Authentication *not* required before use of the system, but *is* required before manipulation of CDIs (requires using TPs)

Logging

CR4 All TPs must append enough information to reconstruct the operation to an append-only CDI.

- This CDI is the log
- Auditor needs to be able to determine what happened during reviews of transactions

Handling Untrusted Input

CR5 Any TP that takes as input a UDI may perform only valid transformations, or no transformations, for all possible values of the UDI. The transformation either rejects the UDI or transforms it into a CDI.

- In bank, numbers entered at keyboard are UDIs, so cannot be input to TPs. TPs must validate numbers (to make them a CDI) before using them; if validation fails, TP rejects UDI

Separation of Duty In Model

ER4 Only the certifier of a TP may change the list of entities associated with that TP. No certifier of a TP, or of an entity associated with that TP, may ever have execute permission with respect to that entity.

- Enforces separation of duty with respect to certified and allowed relations

Comparison With Requirements

1. Users can't certify TPs, so CR5 and ER4 enforce this
2. Procedural, so model doesn't directly cover it; but special process corresponds to using TP
   - No technical controls can prevent programmer from developing program on production system; usual control is to delete software tools
3. TP does the installation, trusted personnel do certification

Comparison With Requirements

4. CR4 provides logging; ER3 authenticates trusted personnel doing installation; CR5, ER4 control installation procedure
   - New program UDI before certification, CDI (and TP) after
5. Log is CDI, so appropriate TP can provide managers, auditors access
   - Access to state handled similarly

Comparison to Biba

- Biba
  - No notion of certification rules; trusted subjects ensure actions obey rules
  - Untrusted data examined before being made trusted
- Clark-Wilson
  - Explicit requirements that *actions* must meet
  - Trusted entity must certify *method* to upgrade untrusted data (and not certify the data itself)

Chinese Wall Model

Problem:

- Tony advises American Bank about investments
- He is asked to advise Toyland Bank about investments

- Conflict of interest to accept, because his advice for either bank would affect his advice to the other bank

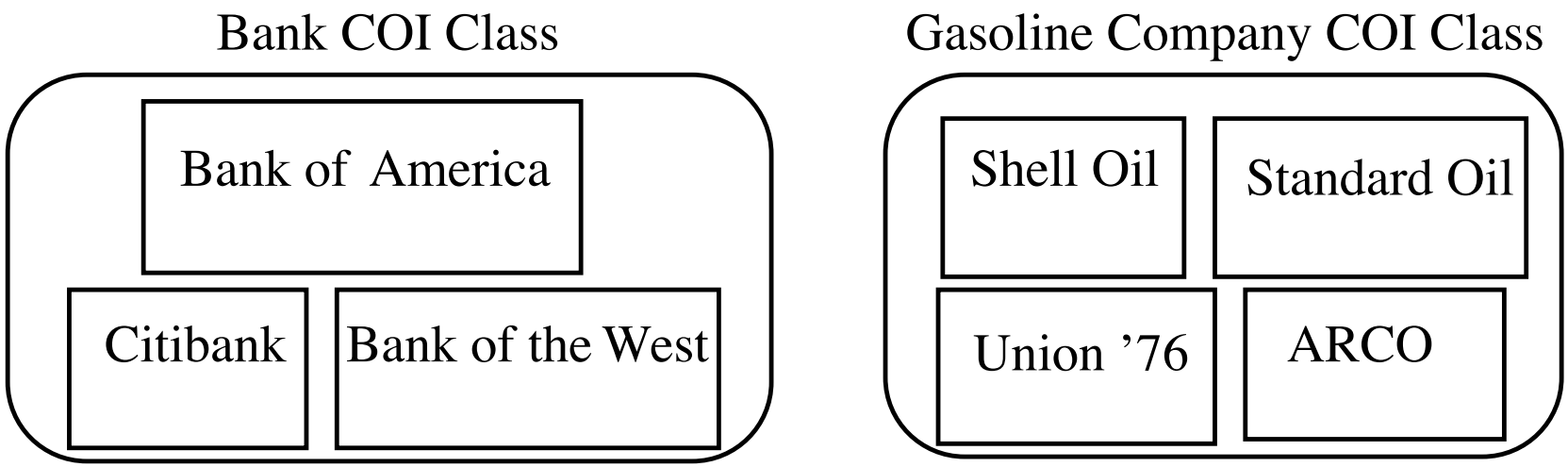
Organization

- Organize entities into “conflict of interest” classes
- Control subject accesses to each class
- Control writing to all classes to ensure information is not passed along in violation of rules
- Allow sanitized data to be viewed by everyone

Definitions

- *Objects*: items of information related to a company
- *Company dataset* (CD): contains objects related to a single company
  - Written *CD*(*O*)
- *Conflict of interest class* (COI): contains datasets of companies in competition
  - Written *COI*(*O*)
  - Assume: each object belongs to exactly one *COI* class

Example


Bank COI Class
Bank of America
Citibank
Bank of the West
Gasoline Company COI Class
Shell Oil
Standard Oil
Union '76
ARCO

Temporal Element

- If Anthony reads any CD in a COI, he can *never* read another CD in that COI
  - Possible that information learned earlier may allow him to make decisions later
  - Let $PR(S)$ be set of objects that S has already read

CW-Simple Security Condition

- *s* can read *o* iff either condition holds:
  1. There is an o' such that *s* has accessed o' and $CD(o') = CD(o)$
     - Meaning *s* has read something in *o*'s dataset
  2. For all $o' \in O$, $o' \in PR(s) \Rightarrow COI(o') \neq COI(o)$
     - Meaning *s* has not read any objects in *o*'s conflict of interest class
- Ignores sanitized data (see below)
- Initially, $PR(s) = \varnothing$, so initial read request granted

Sanitization

- Public information may belong to a CD
  - As is publicly available, no conflicts of interest arise
  - So, should not affect ability of analysts to read
  - Typically, all sensitive data removed from such information before it is released publicly (called *sanitization*)
- Add third condition to CW-Simple Security Condition:
  3. *o* is a sanitized object

Writing

- Anthony, Susan work in same trading house
- Anthony can read Bank 1's CD, Gas' CD
- Susan can read Bank 2's CD, Gas' CD
- If Anthony could write to Gas' CD, Susan can read it
  - Hence, indirectly, she can read information from Bank 1's CD, a clear conflict of interest

CW-*-Property

- *s* can write to *o* iff both of the following hold:
  1. The CW-simple security condition permits *s* to read *o*; and
  2. For all *unsanitized* objects *o′*, if *s* can read *o′*, then $CD(o') = CD(o)$
- Says that s can write to an object if all the (unsanitized) objects it can read are in the same dataset

Formalism

- Goal: figure out how information flows around system
- S set of subjects, O set of objects, $L = C \times D$ set of labels
- $l_1 : O \rightarrow C$ maps objects to their COI classes
- $l_2 : O \rightarrow D$ maps objects to their CDs
- $H(s, o)$ true iff s has *or had* read access to o
- $R(s, o)$: s's request to read o

Axioms

- Axiom 7-1. For all $o, o' \in O$,
  if $l_2(o) = l_2(o')$, then $l_1(o) = l_1(o')$
  - CDs do not span COIs.
- Axiom 7-2. $s \in S$ can read $o \in O$ iff, for all $o' \in O$ such that $H(s, o')$, either
  $l_1(o') \neq l_1(o)$ or $l_2(o') = l_2(o)$
  - s can read o iff o is either in a different COI than every other o' that s has read, or in the same CD as o.

More Axioms

- Axiom 7-3. $\neg H(s, o)$ for all $s \in S$ and $o \in O$ is an initially secure state
  - Description of the initial state, assumed secure
- Axiom 7-4. If for some $s \in S$ and all $o \in O$, $\neg H(s, o)$, then any request $R(s, o)$ is granted
  - If s has read no object, it can read any object

Which Objects Can Be Read?

- Suppose $s \in S$ has read $o \in O$. If s can read $o' \in O$, $o' \neq o$, then $l_1(o') \neq l_1(o)$ or $l_2(o') = l_2(o)$.
  - Says s can read only the objects in a single CD within any COI

Proof

Assume false. Then

$$H(s, o) \wedge H(s, o') \wedge l_1(o') = l_1(o) \wedge l_2(o') \neq l_2(o)$$

Assume s read o first. Then $H(s, o)$ when s read o, so by Axiom 7-2, either $l_1(o') \neq l_1(o)$ or $l_2(o') = l_2(o)$, so

$$(l_1(o') \neq l_1(\mathrm{o}) \vee l_2(o') = l_2(o)) \wedge (l_1(o') = l_1(o) \wedge l_2(o') \neq l_2(o))$$

Rearranging terms,

$$(l_1(o') \neq l_1(o) \wedge l_2(o') \neq l_2(o) \wedge l_1(o') = l_1(o)) \vee$$
$$(l_2(o') = l_2(o) \wedge l_2(o') \neq l_2(o) \wedge l_1(o') = l_1(o))$$

which is obviously false, contradiction.

Lemma

- Suppose a subject $s \in S$ can read an object $o \in O$. Then s can read no o' for which $l_1(o') = l_1(o)$ and $l_2(o') \neq l_2(o)$.
  - So a subject can access at most one CD in each COI class
  - Sketch of proof: Initial case follows from Axioms 7-3, 7-4. If $o' \neq o$, theorem immediately gives lemma.

COIs and Subjects

- Theorem: Let $c \in C$ and $d \in D$. Suppose there are n objects $o_i \in O$, $1 \le i \le n$, such that $l_1(o_i) = d$ for $1 \le i \le n$, and $l_2(o_i) \ne l_2(o_j)$, for $1 \le i, j \le n$, $i \ne j$. Then for all such o, there is an $s \in S$ that can read o iff $n \le |S|$.
  - If a COI has n CDs, you need at least n subjects to access every object
  - Proof sketch: If s can read o, it cannot read any o' in another CD in that COI (Axiom 7-2). As there are n such CDs, there must be at least n subjects to meet the conditions of the theorem.

Sanitized Data

- $v(o)$: sanitized version of object o
  - For purposes of analysis, place them all in a special CD in a COI containing no other CDs
- Axiom 7-5. $l_1(o) = l_1(v(o))$ iff $l_2(o) = l_2(v(o))$

Which Objects Can Be Written?

- Axiom 7-6. $s \in S$ can write to $o \in O$ iff the following hold simultaneously
  1. $H(s, o)$
  2. There is no $o' \in O$ with $H(s, o')$, $l_2(o) \neq l_2(o')$, $l_2(o) \neq l_2(v(o))$, $l_2(o') = l_2(v(o))$.
  - Allow writing iff information cannot leak from one subject to another through a mailbox
  - Note handling for sanitized objects

How Information Flows

- Definition: information may flow from o to o' if there is a subject such that $H(s, o)$ and $H(s, o')$.
  - Intuition: if s can read 2 objects, it can act on that knowledge; so information flows between the objects through the nexus of the subject
  - Write the above situation as (o, o')

Key Result

- Set of all information flows is

  $\{ (o, o') \mid o \in O \wedge o' \in O \wedge l_2(o) = l_2(o') \vee l_2(o) = l_2(v(o)) \}$

- Sketch of proof: Definition gives set of flows:

  $F = \{(o, o') \mid o \in O \wedge o' \in O \wedge \exists\ s \in S$ such that $H(s, o) \wedge H(s, o'))\}$

  Axiom 7-6 excludes the following flows:

  $X = \{ (o, o') \mid o \in O \wedge o' \in O \wedge l_2(o) \neq l_2(o') \wedge l_2(o) \neq l_2(v(o)) \}$

  So, letting F^* be transitive closure of F,

  $F^* - X = \{(o, o') \mid o \in O \wedge o' \in O \wedge \neg(l_2(o) \neq l_2(o') \wedge l_2(o) \neq l_2(v(o))) \}$

  which is equivalent to the claim.

Compare to Bell-LaPadula

- Fundamentally different
  - CW has no security labels, B-LP does
  - CW has notion of past accesses, B-LP does not
- Bell-LaPadula can capture state at any time
  - Each (COI, CD) pair gets security category
  - Two clearances, *S* (sanitized) and *U* (unsanitized)
    - *S dom U*
  - Subjects assigned clearance for compartments without multiple categories corresponding to CDs in same COI class

Compare to Bell-LaPadula

- Bell-LaPadula cannot track changes over time
  - Susan becomes ill, Anna needs to take over
    - C-W history lets Anna know if she can
    - No way for Bell-LaPadula to capture this
- Access constraints change over time
  - Initially, subjects in C-W can read any object
  - Bell-LaPadula constrains set of objects that a subject can access
    - Can't clear all subjects for all categories, because this violates CW-simple security condition

Compare to Clark-Wilson

- Clark-Wilson Model covers integrity, so consider only access control aspects
- If "subjects" and "processes" are interchangeable, a single person could use multiple processes to violate CW-simple security condition
  - Would still comply with Clark-Wilson Model
- If "subject" is a specific person and includes all processes the subject executes, then consistent with Clark-Wilson Model

Clinical Information Systems Security Policy

- Intended for medical records
  - Conflict of interest not critical problem
  - Patient confidentiality, authentication of records and annotators, and integrity are
- Entities:
  - Patient: subject of medical records (or agent)
  - Personal health information: data about patient's health or treatment enabling identification of patient
  - Clinician: health-care professional with access to personal health information while doing job

Assumptions and Principles

- Assumes health information involves 1 person at a time
  - Not always true; OB/GYN involves father as well as mother
- Principles derived from medical ethics of various societies, and from practicing clinicians

Access

- Principle 1: Each medical record has an access control list naming the individuals or groups who may read and append information to the record. The system must restrict access to those identified on the access control list.
  - Idea is that clinicians need access, but no-one else. Auditors get access to copies, so they cannot alter records

Access

- Principle 2: One of the clinicians on the access control list must have the right to add other clinicians to the access control list.
  - Called the *responsible clinician*

Access

- Principle 3: The responsible clinician must notify the patient of the names on the access control list whenever the patient's medical record is opened. Except for situations given in statutes, or in cases of emergency, the responsible clinician must obtain the patient's consent.
  - Patient must consent to all treatment, and must know of violations of security

Access

- Principle 4: The name of the clinician, the date, and the time of the access of a medical record must be recorded. Similar information must be kept for deletions.
  - This is for auditing. Don't delete information; update it (last part is for deletion of records after death, for example, or deletion of information when required by statute). Record information about all accesses.

Creation

- Principle: A clinician may open a record, with the clinician and the patient on the access control list. If a record is opened as a result of a referral, the referring clinician may also be on the access control list.
  - Creating clinician needs access, and patient should get it. If created from a referral, referring clinician needs access to get results of referral.

Deletion

- Principle: Clinical information cannot be deleted from a medical record until the appropriate time has passed.
  - This varies with circumstances.

Confinement

- Principle: Information from one medical record may be appended to a different medical record if and only if the access control list of the second record is a subset of the access control list of the first.
  - This keeps information from leaking to unauthorized users. All users have to be on the access control list.

Aggregation

- Principle: Measures for preventing aggregation of patient data must be effective. In particular, a patient must be notified if anyone is to be added to the access control list for the patient's record and if that person has access to a large number of medical records.
  - Fear here is that a corrupt investigator may obtain access to a large number of records, correlate them, and discover private information about individuals which can then be used for nefarious purposes (such as blackmail)

Enforcement

- Principle: Any computer system that handles medical records must have a subsystem that enforces the preceding principles. The effectiveness of this enforcement must be subject to evaluation by independent auditors.
  - This policy has to be enforced, and the enforcement mechanisms must be auditable (and audited)

Compare to Bell-LaPadula

- Confinement Principle imposes lattice structure on entities in model
  - Similar to Bell-LaPadula
- CISS focuses on objects being accessed; B-LP on the subjects accessing the objects
  - May matter when looking for insiders in the medical environment

Compare to Clark-Wilson

- CDIs are medical records
- TPs are functions updating records, access control lists
- IVPs certify:
    - A person identified as a clinician is a clinician;
    - A clinician validates, or has validated, information in the medical record;
    - When someone is to be notified of an event, such notification occurs; and
    - When someone must give consent, the operation cannot proceed until the consent is obtained
- Auditing (CR4) requirement: make all records append-only, notify patient when access control list changed

ORCON

- Problem: organization creating document wants to control its dissemination
  - Example: Secretary of Agriculture writes a memo for distribution to her immediate subordinates, and she must give permission for it to be disseminated further. This is "originator controlled" (here, the "originator" is a person).

Requirements

- Subject $s \in S$ marks object $o \in O$ as ORCON on behalf of organization X. X allows o to be disclosed to subjects acting on behalf of organization Y with the following restrictions:
  1. o cannot be released to subjects acting on behalf of other organizations without X's permission; and
  2. Any copies of o must have the same restrictions placed on it.

DAC Fails

- Owner can set any desired permissions
  - This makes 2 unenforceable

MAC Fails

- First problem: category explosion
  - Category C contains o, X, Y, and nothing else. If a subject $y \in Y$ wants to read o, $x \in X$ makes a copy o'. Note o' has category C. If y wants to give $z \in Z$ a copy, z must be in Y—by definition, it's not. If x wants to let $w \in W$ see the document, need a new category C' containing o, X, W.
- Second problem: abstraction
  - MAC classification, categories centrally controlled, and access controlled by a centralized policy
  - ORCON controlled locally

Combine Them

- The owner of an object cannot change the access controls of the object.
- When an object is copied, the access control restrictions of that source are copied and bound to the target of the copy.
  - These are MAC (owner can't control them)
- The creator (originator) can alter the access control restrictions on a per-subject and per-object basis.
  - This is DAC (owner can control it)

RBAC

- Access depends on function, not identity
  - Example:
    - Allison, bookkeeper for Math Dept, has access to financial records.
    - She leaves.
    - Betty hired as the new bookkeeper, so she now has access to those records
  - The role of "bookkeeper" dictates access, not the identity of the individual.

Definitions

- Role *r*: collection of job functions
  - *trans*(*r*): set of authorized transactions for *r*
- Active role of subject *s*: role *s* is currently in
  - *actr*(*s*)
- Authorized roles of a subject *s*: set of roles *s* is authorized to assume
  - *authr*(*s*)
- *canexec*(*s*, *t*) iff subject *s* can execute transaction *t* at current time

Axioms

- Let S be the set of subjects and T the set of transactions.
- *Rule of role assignment*:
  $(\forall s \in S)(\forall t \in T)\ [canexec(s, t) \rightarrow actr(s) \neq \varnothing]$.
  - If s can execute a transaction, it has a role
  - This ties transactions to roles
- *Rule of role authorization*:
  $(\forall s \in S)\ [actr(s) \subseteq authr(s)]$.
  - Subject must be authorized to assume an active role (otherwise, any subject could assume any role)

Axiom

- *Rule of transaction authorization*:
  $(\forall s \in S)(\forall t \in T)$
  $[canexec(s, t) \rightarrow t \in trans(actr(s))]$.
  - If a subject s can execute a transaction, then the transaction is an authorized one for the role s has assumed

Containment of Roles

- Trainer can do all transactions that trainee can do (and then some). This means role r contains role r' ($r > r'$). So:

$$(\forall s \in S)[\, r' \in authr(s) \wedge r > r' \rightarrow r \in authr(s) \,]$$

Separation of Duty

- Let r be a role, and let s be a subject such that $r \in auth(s)$. Then the predicate $meauth(r)$ (for mutually exclusive authorizations) is the set of roles that s cannot assume because of the separation of duty requirement.
- Separation of duty:

$$(\forall r_1, r_2 \in R)\ [\ r_2 \in meauth(r_1) \rightarrow$$
$$[\ (\forall s \in S)\ [\ r_1 \in authr(s) \rightarrow r_2 \notin authr(s)\ ]\ ]\ ]$$

Key Points

- Hybrid policies deal with both confidentiality and integrity
  - Different combinations of these
- ORCON model neither MAC nor DAC
  - Actually, a combination
- RBAC model controls access based on functionality