Hybrid Models

ECS 153 Spring Quarter 2021

Module 12

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

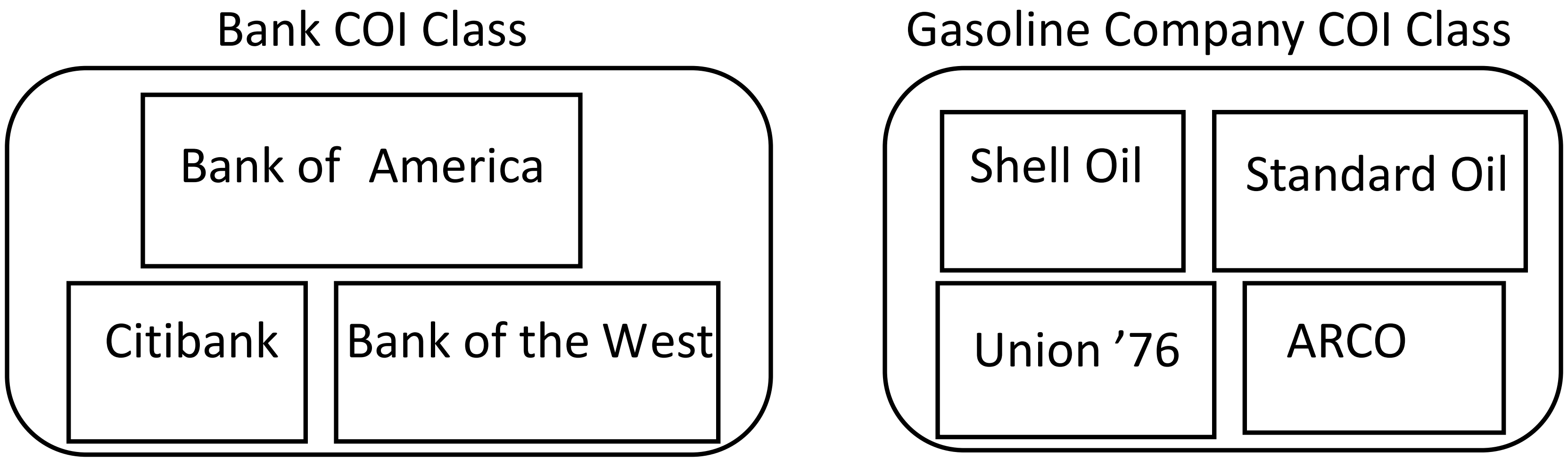
Organization

- 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, $PR(s) = \emptyset$, 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

Aggressive Chinese Wall Model

- Assumption of Chinese Wall Model: COI classes are actually related to business, and those are partitions
  - Continuing bank and oil company example, the latter may invest in some companies, placing them in competition with banks
  - One bank may only handle savings, and another a brokerage house, so they are not in competition
- More formally: Chinese Wall model assumes the elements of O can be partitioned into COIs, and thence into CDs
  - Define CIR to be the conflict of interest relation induced by a COI
  - For $o, o' \in O$, if o, o' are in the same COI, then $(o, o') \in CIR$

The Problem

- Not true in practice!
  - That is, in practice *CIR* does not partition the objects, and so not an equivalence class
  - Example: a company is not in conflict with itself, so $(o, o) \notin CIR$
  - Example: company *c* has its own private savings unit; *b* bank that does both savings and investments; oil company *g* does investments. So $(c, b) \in CIR$ and $(b, g) \in CIR$, but clearly $(c, g) \notin CIR$

The Solution

- Generalize *CIR* to define COIs not based on business classes, so *GCIR* is the reflexive, transitive closure of *CIR*
- To create it:
  - For all $o \in O$, add (o, o) to *CIR*
  - Take the transitive closure of this
- Then $(o, o')\in GICR$ iff there is an indirect information flow path between *o* and *o*’
  - Recall $(o, o') \in CIR$ iff there is a direct information flow path between *o*, *o*’
- Now replace the COIs induced by *CIR* with generalized COIs induced by *GCIR*

Compare to Bell-LaPadula

- Fundamentally different
  - CW has no security labels, Bell-LaPadula does
  - CW has notion of past accesses, Bell-LaPadula 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

Originator Controlled Access Control

- 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)

Digital Rights Management (DRM)

- The persistent control of digital content
- Several elements:
  - Content: information being protected
  - License: token describing the uses allowed for the content
  - Grant: part of a license giving specific authorizations to one or more entities, and (possibly) conditions constraining the use of the grant
  - Issuer: entity issuing the license
  - Principal: identification of an entity, used in a license to identify to whom the license applies
  - Device: mechanism used to view the content

Example: Movie Distribution by Downloading

- Content: movie itself
- License: token binding payying the movie to the specific downloaded copy
- Grant: movie can be played on some specific set of equipment provided the equipment is located in a geographical area
- Issuer: movie studio
- Principal: user who downloaded the movie
- Device: set of equipment used to play the movie; it manages the licenses, principle, and any copies of the movie

Relationships

Elements related, and the relationship must satisfy all of:

1. The system must implement controls on the use of the content, constraining what users can do with the content
   - Encrypting the content and providing keys to authorized viewers fails this, as the users can distribute the keys indiscriminently
2. The rules that constrain the users of the content must be associated with the content, not the users
3. The controls and rules must persist throughout the life of the content, regardless of how it is distributed and to whom it is distributed

Conditions

- Stated using a rights expression language
- Example: Microsoft’s ReadyPlay uses a language supporting temporal constraints such as
  - Allowing the content to be viewed over a specific period of time
  - Allowing a validity period for the license
  - Allowing constraints on copying, transferring, converting the content
  - Allowing geographical constraints
  - Allowing availability constraints (for example, content can’t be played when being broadcast)

Example: Microsoft PlayReady DRM

Setup

- Content is enciphered using AES
- Key made available to a license server, encrypted content to a distribution server

Play

- Client downloads content, requests license
- License server authenticates client; on success, constructs license and sends it
- Client checks the constraints and, if playback allowed, uses the key in the license to decipher content

Example: Apple's FairPlay DRM

Set up system to play using iTunes

- iTunes generates globally unique number, sends it to Apple's servers
- Servers add it to list of systems authorized to play music for that user
  - At most 5 systems at a time can be authorized

Obtain content using iTunes

- Content enciphers by AES with a master key
- Master key locked with a randomly generated user key from iTunes
- iTunes sends user key to Apple server; stored there and in iTunes, encrypted

Example: Apple’s FairPlay DRM

Play content using iTunes

- iTunes decrypts user key
- iTunes uses user key to decrypt master key
- iTunes uses master key to decrypt content
- Note it need not contact Apple servers for authorization

Authorize new system

- Apple server sends that system all user keys stored on server

Example: Apple's FairPlay DRM

Deauthorize system

- System deletes all locally stored user keys
- Notifies Apple servers to delete globally unique number from list of authorized computers

Copying content to another system

- Cannot be decrypted without user key, which is not copied

Role-Based Access Control

- 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*: $(\forall s \in S)(\forall t \in T)\ [canexec(s, t) \rightarrow actr(s) \neq \emptyset]$.
  - 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:
  $$(\forall s \in S)[\, r' \in authr(s) \land r > r' \rightarrow r \in authr(s) \,]$$

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:

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

Role Engineering

- *Role engineering*: defining roles and determining needed permissions
- Often used when two organizations using RBAC merge
  - Roles in one organization rarely overlap with roles in other
  - Job functions often do overlap
- *Role mining*: analyzing existing roles, permission assignments to determine optimal assignment of permissions to roles
  - *NP*-complete, but in practice optimal solutions can be approximated or produced