L4: Example Security Policies

Hui Chen, Ph.D.
Dept. of Engineering & Computer Science
Virginia State University
Petersburg, VA 23806

Acknowledgement

- Many slides are from or are revised from the slides of the author of the textbook
  - Matt Bishop, Introduction to Computer Security, Addison-Wesley Professional, October, 2004, ISBN-13: 978-0-321-24774-5. Introduction to Computer Security @ VSU's Safari Book Online subscription
  - http://nob.cs.ucdavis.edu/book/book-intro/slides/

Overview

- Confidentiality Policy Model
  - Goals of Confidentiality Policy Model
  - Bell-LaPadula Model
- Integrity Policy Model
  - Goals of Integrity Policy Model
  - Biba’s model
  - Clark-Wilson model
- Hybrid Policy Model
  - Chinese Wall Model
  - ORCON and RBAC

Confidentiality Policies

- Goal: prevent the unauthorized disclosure of information
  - Deals with information flow
  - Integrity incidental
- Multi-level security models are best-known examples
  - Bell-LaPadula Model is the basis for many, or most, of these models

Bell-LaPadula Model: Preliminary Version

- Security *levels* arranged in *linear ordering*
  - (highest) Top Secret
  - Secret
  - Confidential
  - (lowest) Unclassified
- Subjects have *security clearance* $L(s)$
- Objects have *security classification* $L(o)$

Example

Security Level	Subject (s)	Object (o)
Top Secret (TS)	Tamara	Personnel Files
Secret (S)	Samuel	E-Mail Files
Confidential (C)	Claire, Clarence	Activity Logs
Unclassified (UC)	Ulaley, Ursula	Telephone Lists

- Let $L(S) = l_s$ be the security clearance of subject *s*
- Let $L(O) = l_o$ be the security classification of object *o*

Reading Information

- Information flows *up*, not *down*
  - "Reads up" disallowed, "reads down" allowed
- Simple Security Condition (Preliminary Version)
  - Subject *s* can read object *o* *iff* $L(o) \le L(s)$ and *s* has permission to read *o*
  - Note: it combines mandatory control (relationship of security levels) and discretionary control (the required permission)
  - Sometimes called "*no reads up*" rule

Example

Security Level	**Subject**	**object**
Top Secret (TS)	Tamara	Personnel Files
Secret (S)	Samuel	E-Mail Files
Confidential (C)	Claire, Clarence	Activity Logs
Unclassified (UC)	Ulaley, Ursula	Telephone Lists

- Tamara *can* read all files (if she has the permission)
- Claire *cannot* read Personnel or E-Mail
- Ulaley *can* only read Telephone Lists (if she has the permission)

Writing Information

- Information flows up, not down
  - "Writes up" allowed, "writes down" disallowed
- *-Property (Preliminary Version)
  - Subject *s* can write object *o iff* $L(s) \leq L(o)$ and *s* has permission to write *o*
  - Note: combines mandatory control (relationship of security levels) and discretionary control (the required permission)
  - Sometimes called "*no writes down*" rule

Example

Security Level	**Subject**	**object**
Top Secret (TS)	Tamara	Personnel Files
Secret (S)	Samuel	E-Mail Files
Confidential (C)	Claire, Clarence	Activity Logs
Unclassified (UC)	Ulaley, Ursula	Telephone Lists

- Tamara *write* read personnel files (if she has the permission)
- Claire *cannot* write Telephone Lists
- Ulaley *can* write all files (if she has the permission)

Basic Security Theorem: Preliminary Version

- If a system is initially in a secure state, and every transition of the system satisfies the simple security condition, preliminary version (i.e., no read up rule), and the *-property, preliminary version (i.e., no write down rule), then every state of the system is secure
- **Proof**: induct on the number of transitions

Review Questions

- What is partial ordering?
- What is total ordering?

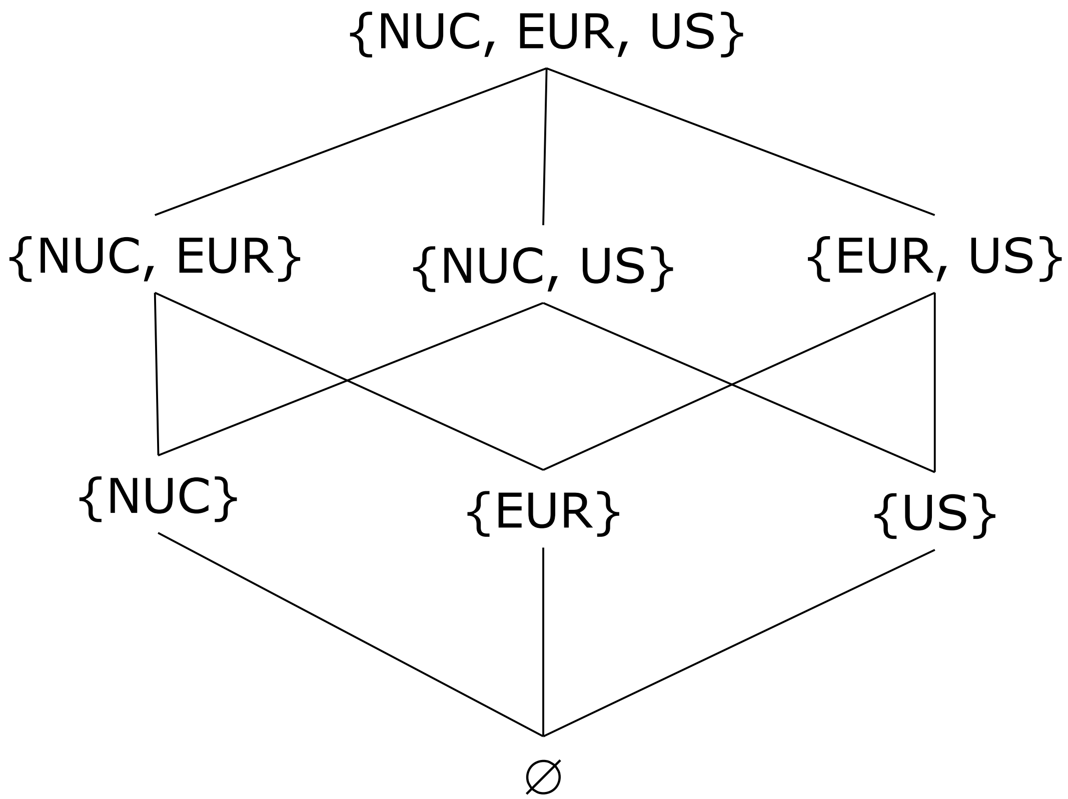
Bell-LaPadula Model

- Expand notion of security level to include categories
  - Arising from the “need-to-know” principle
- Security level is (*clearance*, *category set*)
- Examples
  - (Top Secret, { NUC, EUR, ASI })
  - (Confidential, { EUR, ASI })
  - (Secret, { NUC, ASI })

Levels and Lattices

- (A, C) *dom* (A', C') *iff* $A' \leq A$ and $C' \subseteq C$
  - *dom* reads *dominates*
- Examples
  - (Top Secret, {NUC, ASI}) *dom* (Secret, {NUC})
  - (Secret, {NUC, EUR}) *dom* (Confidential,{NUC, EUR})
  - (Top Secret, {NUC}) ¬*dom* (Confidential, {EUR})
- Let C be set of classifications, K set of categories. Set of security levels $L = C \times K$, *dom* form lattice
  - *Least upper bound:* $lub(L) = (max(A), C)$
  - *Greatest lower bound:* $glb(L) = (min(A), \varnothing)$

Lattice Example



Levels and Ordering

- Security levels *partially ordered*
  - Any pair of security levels may (or may not) be related by *dom*
- “dominates” serves the role of “greater than” in the preliminary version
  - “greater than” is a *total ordering*, though

Reading Information

- Information flows *up*, not *down*
  - “Reads up” disallowed, “reads down” allowed
- Simple Security Condition (Step 2)
  - Subject *s* can read object *o* iff *L*(*s*) *dom* *L*(*o*) and *s* has permission to read *o*
    - Note: combines mandatory control (relationship of security levels) and discretionary control (the required permission)
  - Sometimes called “no reads up” rule

Writing Information

- Information flows up, not down
  - "Writes up" allowed, "writes down" disallowed
- *-Property (Step 2)
  - Subject *s* can write object *o* iff *L*(*o*) *dom* *L*(*s*) and *s* has permission to write *o*
    - Note: combines mandatory control (relationship of security levels) and discretionary control (the required permission)
  - Sometimes called "no writes down" rule

Basic Security Theorem, Step 2

- If a system is initially in a secure state, and every transition of the system satisfies the simple security condition, step 2, and the *-property, step 2, then every state of the system is secure
- **Proof**: induct on the number of transitions
- In actual Basic Security Theorem, discretionary access control treated as third property, and simple security property and *-property phrased to eliminate discretionary part of the definitions — but simpler to express the way done here.

Problem

- At times, a subject must communicate with another subject at a lower level
  - But the model requires no *read up* and no *write down*
- Example
  - Colonel has (Secret, {NUC, EUR}) clearance
  - Major has (Secret, {EUR}) clearance
    - Major can talk to colonel (i.e., "write up" or "read down" is allowed)
    - Colonel cannot talk to major (i.e., "read up" or "write down" is disallowed)
  - But that is *clearly absurd*!

Solution

- Define *maximum security level*, *current security level* for subjects
  - *maxlevel*(*s*) *dom curlevel*(*s*)
- Example
  - Treat Major as an object (Colonel is writing to him/her)
  - Colonel has *maxlevel* (Secret, { NUC, EUR })
  - Colonel sets *curlevel* to (Secret, { EUR })
  - Now *L*(Major) *dom curlevel*(Colonel)
    - Colonel can write to Major without violating "no writes down"

Example: The Data General B2 UNIX System

- Section 5.2.2

Exercise L4-1

- Question 2(a), 2(c), and 2(e) of Exercises 5.5 in page 71 of the textbook
  - Hint:
    - Two different types of access
      - Read
      - Write
    - Can she read the document?
      - Read: no reads up
    - Can she write the document?
      - Write: no writes down

Homework 4

- Question 2(b) and 2(d) of Exercises 5.5 in page 71 of the textbook
  - Hint:
    - Two different types of access
      - Read
      - Write
    - Can she read the document?
      - Read: no reads up
    - Can she write the document?
      - Write: no writes down
    - Show me the figures

Summary for Confidentiality Policies

- Confidentiality Policies
- The Bell-LaPadula Model
  - Permeates all policy modeling in computer security
  - First mathematical model
  - Formed the bases for several computer security standards
    - e.g., U.S. Department of Defense's Trusted Computer System Evaluation Criteria
      - http://csrc.nist.gov/publications/history/dod85.pdf
- Additional Reading
  - DG/UX

Integrity Policies

- Goal: prevent the unauthorized changes of information
- Emphasis on preserving data integrity
  - Lipner identifies 5 requirements (S. Lipner, 1982)

Lipner’s Requirements (1-3)

1. Users will not write their own programs, but will use existing production programs and databases.
2. Programmers will develop and test programs on a non-production system; if they need access to actual data, they will be given production data via a special process, but will use it on their development system.
3. A special process must be followed to install a program from the development system onto the production system.

Lipner’s Requirements (4-5)

4. The special process in requirement 3 must be controlled and audited.
5. The managers and auditors must have access to both the system state and the system logs that are generated.

Principals of Operations

- Separation of duty
  - e.g, developing application, releasing the application to production
- Separation of functions
  - e.g., development systems, production systems
- Auditing
  - Analyzing systems to determine what actions took place and who performed them.
  - e.g., extensive logging, log analysis

Log Data Management and Analysis: A Fertile Field

- New businesses
  - http://www.splunk.com/company
  - https://www.loggly.com/why-loggly/
  - https://logentries.com/about-us/news/
  - http://www.sumologic.com/about-sumo/
  - http://www.logrhythm.com/siem-2.0/logrhythm-security-intelligence/log-management-log-analysis.aspx
  - ……
- New research
  - https://www.usenix.org/legacy/events/byname/slaml.html
  - http://dl.acm.org/citation.cfm?id=2038633&picked=prox
  - ……

Commercial and Military Environments

- Military (or government) environment
  - Clearance (to access specific categories, security levels) controls the ability to access information (in those compartments)
- Commercial environment
  - Rarely grant access based on the basis of *clearance*
  - Based on *needs*
    - If modeled using the Bell-LaPadula model
      - It requires a large number of categories and security levels
      - It becomes difficult to control the categories and security levels
  - Information aggregation

Biba Integrity Model

- Integrity level
  - Program and data
  - Describes one’s confidence on the behavior of a program
    - whether if it will execute correctly
    - whether if one can detect problems with its input and stop executing
  - Describes one’s confidence on accuracy or reliability of data
- Implicitly incorporates the notion of “trust”
  - Trustworthiness is used as a measure of integrity level

Integrity Level and Security Level

- Important point: *integrity levels are **not** security levels*
- *Security labels* limits the flow of information
- *Integrity labels* limits the modification of information

Biba Integrity Model

- Set of subjects S, objects O, integrity levels I
- relation $\leq$, $\subseteq$, and $I \times I$ holds when second dominates first
- min: $I \times I \rightarrow I$ returns lesser of integrity levels
- i: $S \cup O \rightarrow I$ gives integrity level of entity
- Access
  - $\underline{\text{r}}$ (read): $S \times O$ means $s \in S$ can read $o \in O$
  - $\underline{\text{w}}$ (write) (and $\underline{\text{x}}$) defined similarly

Biba's Model

- Similar to Bell-LaPadula model
  - $s \in S$ can read $o \in O$ iff $i(s) \leq i(o)$
  - $s \in S$ can write to $o \in O$ iff $i(o) \leq i(s)$
  - $s_1 \in S$ can execute $s_2 \in S$ iff $i(s_2) \leq i(s_1)$
- Add compartments and discretionary controls to get full dual of Bell-LaPadula model

LOCUS Operating System

- An early design and implementation of a distributed operating system
- A cluster of machines that appears to be one single system
- Locus Computing Corporation (Founded in 1982)
- Further Reading
  - Bruce Walker, Gerald Popek, Robert English, Charles Kline, and Greg Thiel. 1983. The LOCUS distributed operating system. In *Proceedings of the ninth ACM symposium on Operating systems principles* (SOSP '83). ACM, New York, NY, USA, 49-70. DOI=10.1145/800217.806615

LOCUS and Biba

- Goal: prevent untrusted software from altering data or other software
- Approach: make levels of trust explicit
  - *credibility rating* based on estimate of software's trustworthiness (0 untrusted, *n* highly trusted)
  - *trusted file systems* contain software with a single credibility level
  - Process has *risk level* or highest credibility level at which process can execute
  - Must use *run-untrusted* command to run software at lower credibility level

Clark-Wilson Integrity Model

- A radically different model
- Meets the requirements of commercial environment
    - Integrity of the data in the system and the actions performed on the data
- Integrity defined by a set of constraints (or properties)
- Data in a *consistent* or valid state when it satisfies given properties
- *Well-formed transaction* move system from one consistent state to another

Example

- Bank transactions
- Bank
    - *Notations*
        - D today's deposits
        - W withdrawals
        - YB yesterday's balance
        - TB today's balance
    - Integrity constraint: $D + YB - W \equiv TB$

Integrity of Transactions

- Transactions must start with a consistent state and end with a consistent state
- No constraint is violated
- Issue: who examines, certifies transactions done correctly?

Entities

- Data may be or may not be subject to integrity controls
  - CDIs: constrained data items
    - Data subject to integrity controls
  - UDIs: unconstrained data items
    - Data not subject to integrity controls
- Two sets of procedures
  - 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

Recall Lipner’s Five Requirements

Lipner's Requirements (1-3)

1. Users will not write their own programs, but will use existing production programs and databases.
2. Programmers will develop and test programs on a non-production system; if they need access to actual data, they will be given production data via a special process, but will use it on their development system.
3. A special process must be followed to install a program from the development system onto the production system.

Lipner's Requirements (4-5)

4. The special process in requirement 3 must be controlled and audited.
5. The managers and auditors must have access to both the system state and the system logs that are generated.

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)

Summary of Integrity Policy Models

- Integrity policies deal with trust
  - As trust is hard to quantify, these policies are hard to evaluate completely
  - Look for assumptions and trusted users to find possible weak points in their implementation
- Biba based on multilevel integrity
- Clark-Wilson focuses on separation of duty and transactions

A “Chinese Wall”

- Boston Legal, Season 2, Episode 12

  Alan Shore: *Shirley fired you?*

  Melissa Hughes: *Worse. I got transferred to human resources where all the people persons are.*

  Alan Shore: *She’s building a Chinese Wall.*

  Melissa Hughes: *What’s that?*

  Alan Shore: *It’s a legal concept. It keeps anyone in the firm who may be involved in this case from talking to me about it.*

Arising of Conflict of Interest

Problem:

- Anthony counsels Bank of Galactica about investments
- He is also counsels Starbank about investments

- Potential conflict of interest
  - His advice for either bank would affect his advice to the other bank
  - Cannot counsel both banks

Chinese Wall Model

- Refers equally to *confidentiality and integrity*
- Describes policies that involve a *conflict of interest* in business

Organization

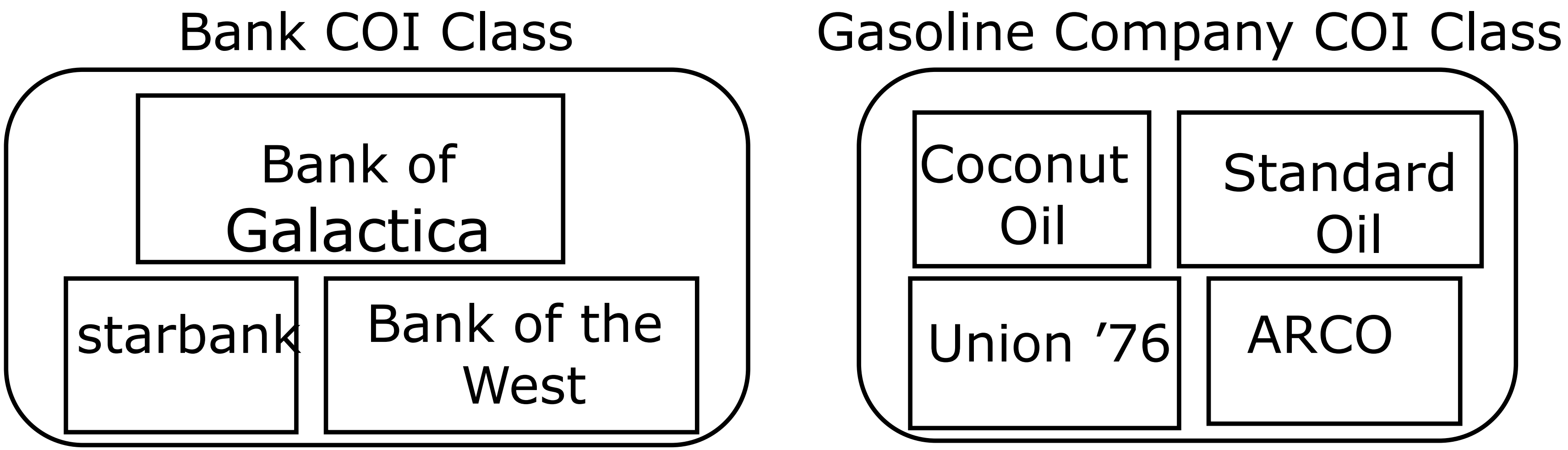
- Entities → Conflict of Interest classes
  - Organize entities into "conflict of interest" classes
- Read (Confidentiality)
  - Control subject accesses to each class
- Write (Integrity)
  - Control writing to all classes to ensure information is not passed along in violation of rules
- Allow sanitized 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
  - *Notation: CD*(*O*)
    - Company dataset that contains object O
- Conflict of interest class (COI)
  - Contains datasets of companies in competition
  - Notation: *COI*(*O*)
    - COI class that contains object O
  - Assume: each object belongs to **exactly one** *COI* class

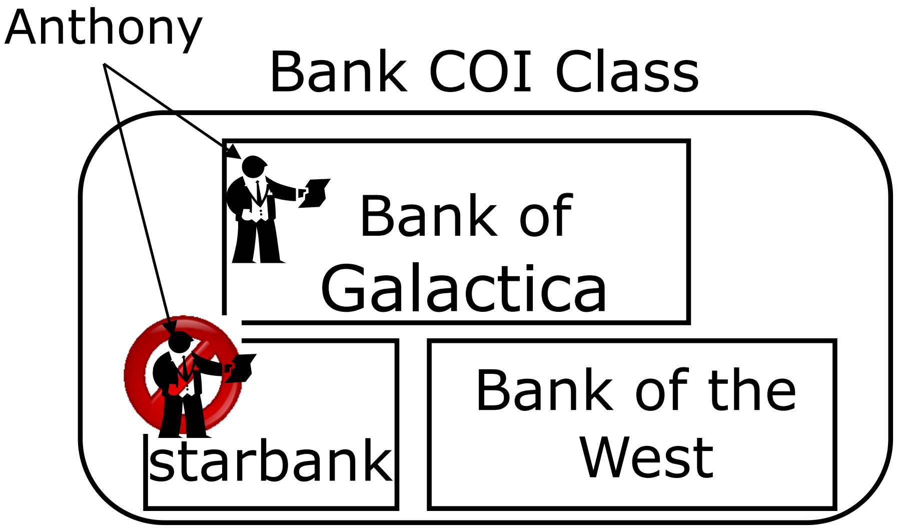
Example: A Chinese Wall Model Database

- Two COI classes
  - Bank COI class: 3 CDs
  - Gasoline Company COI class: 4 CDs



Reading

- 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



Anthony

Bank COI Class

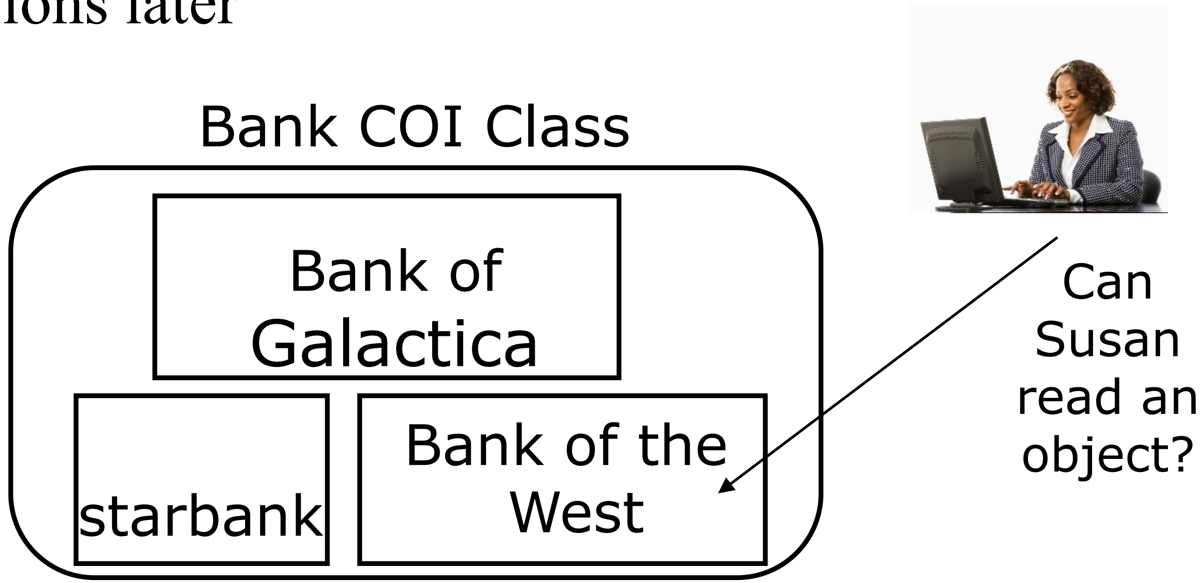
Bank of Galactica

starbank

Bank of the West

Reading

- 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



Can Susan read an object?

CW-Simple Security Condition: Preliminary Version

- Let $PR(S)$ be set of objects that S has already read
- S can read O iff either condition holds:
  - There is an O' such that S has accessed O' and $CD(O') = CD(O)$
    - Meaning S has read *something* in O's dataset
  - For all objects 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
- 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:
  - *O* is a sanitized object

CW-Simple Security Condition

- Let $PR(S)$ be set of objects that S has already read
- S can read O iff either condition holds:
  - There is an O' such that S has accessed O' and $CD(O') = CD(O)$
    - Meaning S has read *something* in O's dataset
  - For all objects 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)
  - O is a sanitized object
- Initially, $PR(s) = \varnothing$, so initial read request granted

Writing

- Anthony, Susan work in **same** trading house
  - Anthony can read Bank of Galactica’s CD, ARCO’s CD
  - Susan can read Starbank’s CD, ARCO’s CD
- Scenario
  - Anthony reads from Bank of Galactica’s CD and writes to ARCO’s CD
  - Then, Susan can read it.
  - Hence, indirectly, Susan can read information from Bank of Galactica s CD, a clear conflict of interest

CW-*-Property

- *S* can write to *O* iff both of the following hold
  - The CW-simple security condition permits *S* to read *O*; and
  - 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

CW-*-Property

- Anthony, Susan work in **same** trading house
  - Anthony can read Bank of Galactica’s CD, ARCO’s CD
  - Susan can read Starbank’s CD, ARCO’s CD
- Can Anthony write to ARCO’s CD?
- Can Susan write to ARCO’s CD?
- Can Anthony write to Bank of Galactica’s CD?
- Can Susan write to Starbank’s CD?

Compare to Bell-LaPadula Model

- 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
    - Chinese-Wall history lets Anna know if she can
    - No way for Bell-LaPadula to capture this
- Access constraints change over time
  - Initially, subjects in Chinese-Wall can read any object
  - Bell-LaPadula constrains set of objects that a subject can access
    - Cannot clear all subjects for all categories, because this violates CW-simple security condition

Compare to Bell-LaPadula

- Bell-LaPadula model cannot emulate the Chinese Wall model faithfully
- However, Chinese Wall model can emulate the Bell-LaPadula model

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 critical
- 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
- Principle 2
- Principle 3
- Principle 4

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. Donot 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
  - Bell-LaPadula mode is a subject of CISS model
- CISS focuses on objects being accessed; Bell-LaPadula 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

- ORginator 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).
- Mandatory Access Control and Discretionary Access Control

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 DAC and MAC

- 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 cannot 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

- 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*: $(\forall s \in S)(\forall t \in T)\ [canexec(s, t) \rightarrow actr(s) \neq \varnothing]$.
  - 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:

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

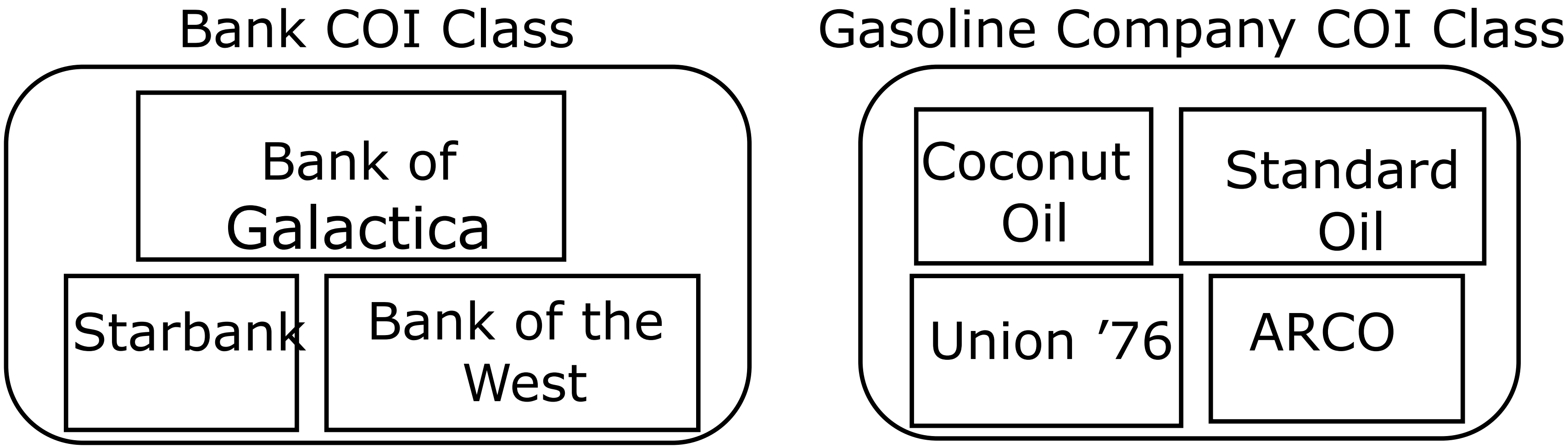
Separation of Duty

- 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:

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

Exercise L4-2

- The exercise is on the Chinese-Wall model
  - A trading house has two COI Classes given below. Susan and Anthony are two employees of the company.
  - Show two examples of authorized states, two examples of unauthorized states using Access Control Matrix, and describe the transitions between them.



Bank COI Class

Bank of Galactica

Starbank

Bank of the West

Gasoline Company COI Class

Coconut Oil

Standard Oil

Union '76

ARCO

Summary of Hybrid Models

- 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