

Introduction to Computer Security

Instructor:

Mahadevan Gomathisankaran

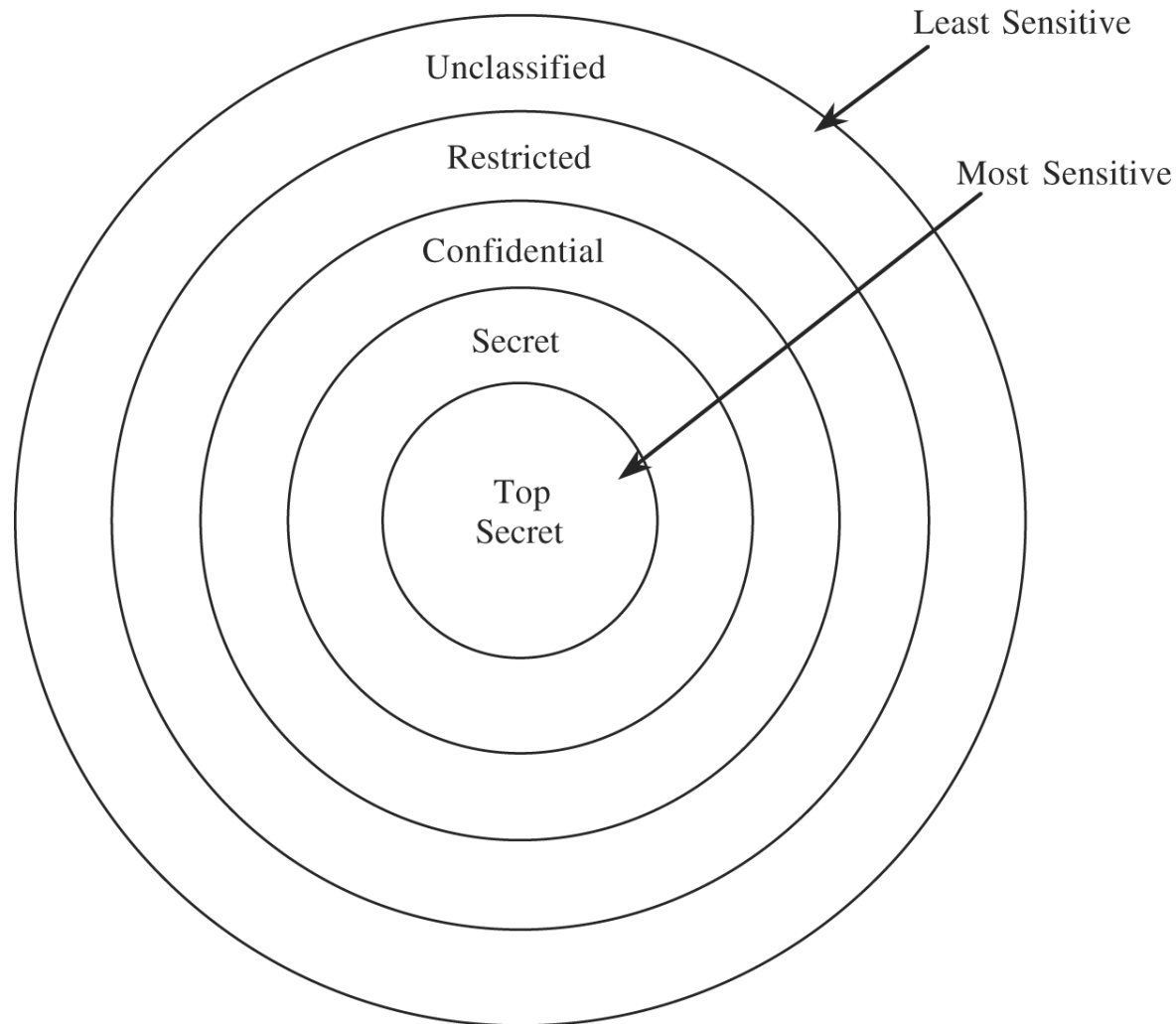
mgomathi@unt.edu

- Project
 - Report and Presentation Due 12/4/2009 5 PM
 - Presentation Schedule posted
 - Interview schedule to be done

- 4 major underpinnings of trusted systems:
 - Policy: Requirements
 - Model: Concrete, verifiable specification
 - Design: Implementation of model (both functionality and construction methods)
 - Trust:
 - Features (system does what it's supposed to do)
 - Assurance (confidence that it enforces the security policy)

- Confidentiality vital to military operations
 - Ideas come from centuries of experience
 - Individual wishes less important than institutional policies
- Foundations of military security policies:
 - Sensitivity levels
 - Information can be unclassified, restricted, confidential, secret, or top secret
 - Subjects cleared for certain sensitivity levels

Military Security Policies – cont'd

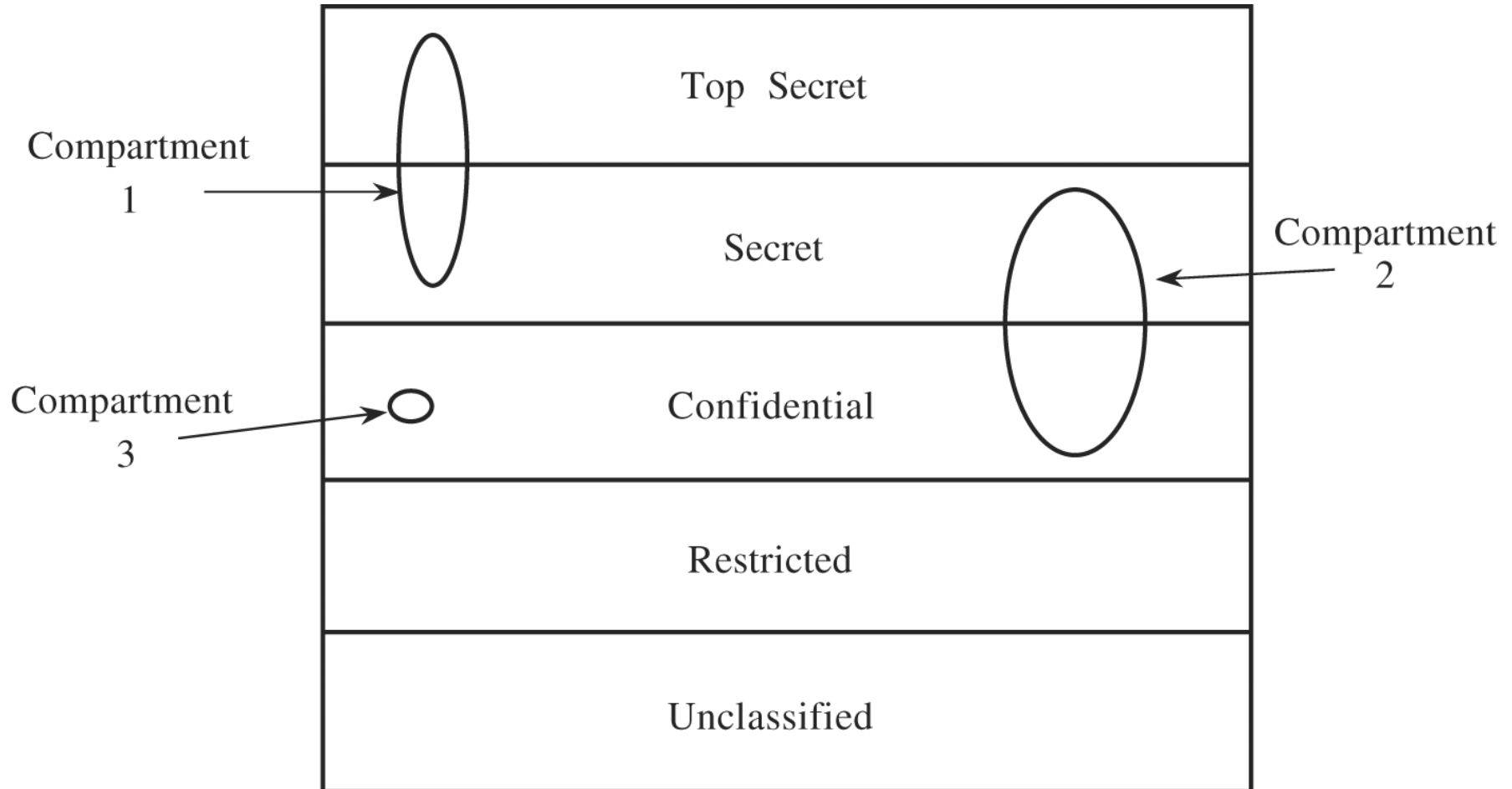


Hierarchy of Sensitivities

Military Policies – Cont'd

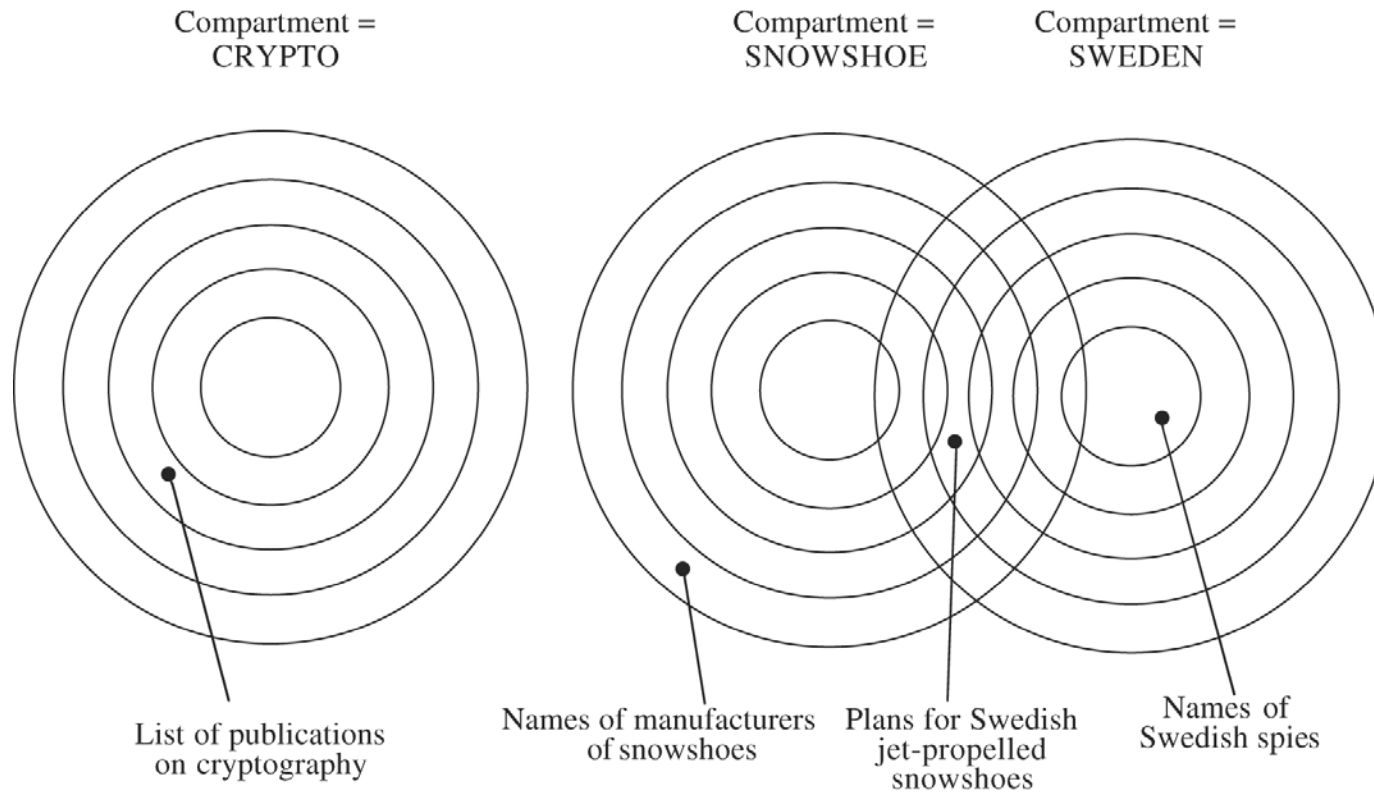
- **Need-to-know rule:** Access to sensitive data is allowed only to subjects who need to know those data to perform their jobs.
- Each piece of classified information may be associated with one or more projects, called **compartments**, describing the subject matter of the information.

Compartment and Sensitivity Levels



Military Policies – Cont'd

- Names are assigned to identify compartments
- A single piece of information can be coded with 0, 1, 2, .. N compartment names, based on # of categories to which it relates.



- Important differences in commercial world:
 - Usually no “security officer” and clearances
 - Integrity often more important than confidentiality (which was the basis for military security models)
 - Prevent unauthorized modification, fraud, and errors

- Basic ideas behind Clark-Wilson model:
 - Well-formed transactions
 - Data manipulated only by specific programs/methods
 - Users access methods, not data
 - Note: Very similar to data abstraction (private data in a class only accessible through approved public methods)
 - Separation of duties Principle:
 - One user should not be doing every transaction
 - Accomplished manually with dual signatures
 - Access triples representation: $\langle \text{userID}, TP_i, \{CDI_j, CDI_k, \dots\} \rangle$
 - User id
 - Transformation procedure
 - 1 or many constrained data items
 - stateless

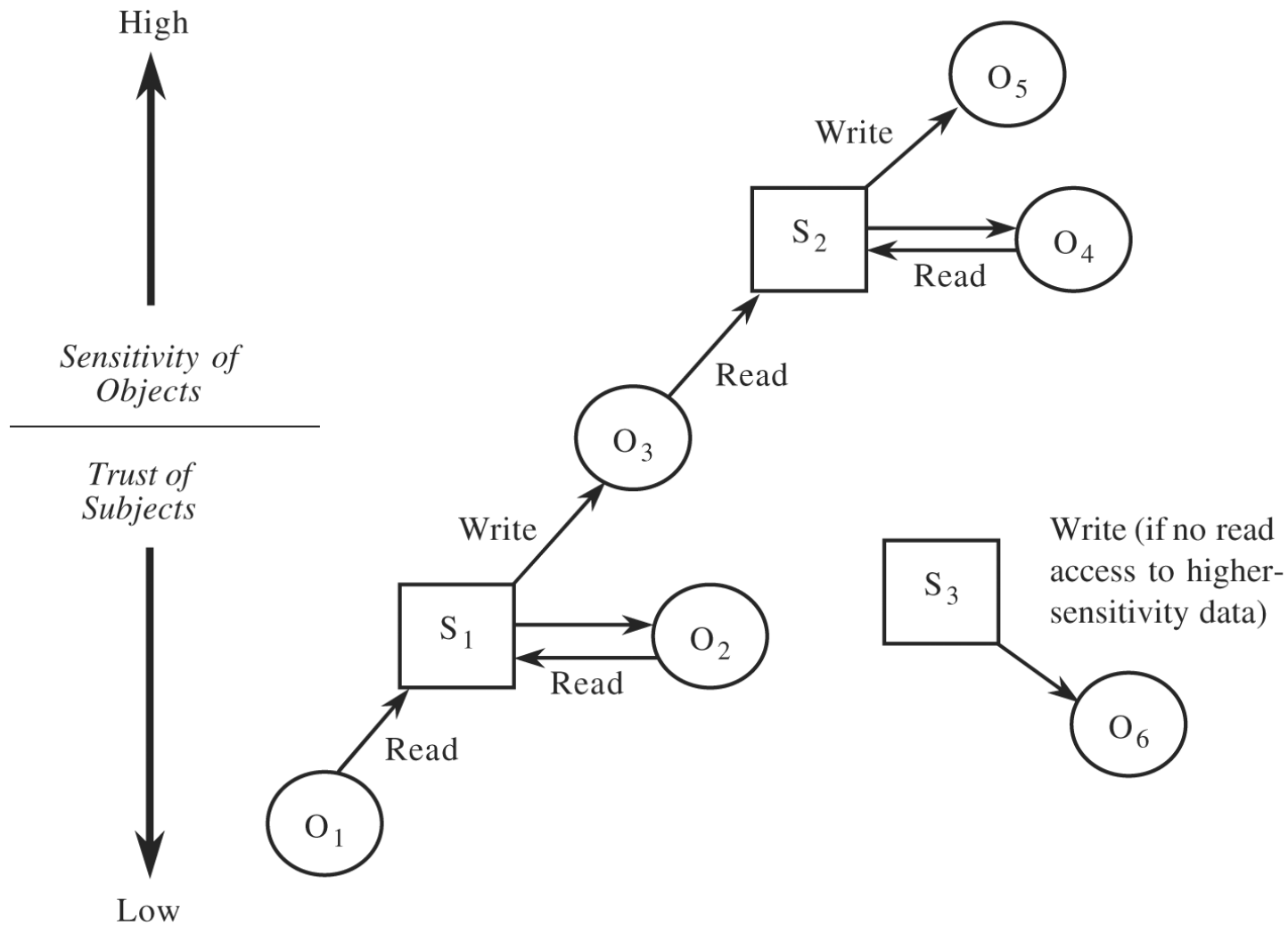
- Goal: Reflect conflict of interest restrictions
- Definitions:
 - Company datasets:
 - All data related to a particular company
 - Conflict of interest classes:
 - Sets of companies which are in competition
 - Objects labeled with company and conflict of interest class
 - Supports “sanitized information” (null conflict of interest class)

Bell-La Padula Confidentiality Model

- Formalization of the military access policy
- Goal: Multi Level Secure (MLS) systems
 - Single systems that can operate on different classifications of data simultaneously
 - “Multi Level” is a *mode of operation* ; others:
 - Dedicated Mode – users cleared for and have need-to-know for everything on system
 - System-High Mode – users cleared for everything on system, and have need-to-know for some info on system
 - Compartmented Mode – users cleared for most restricted data and have need-to-know for info to which he/she has access
 - Bell-La Padula model widely accepted, and forms basis of DoD Trusted Computer System Evaluation Criteria (TCSEC)

- Defined with respect to a lattice
 - Subject S has a clearance $C(S)$ in the lattice
 - Object O has a classification $C(O)$ in the lattice
- Example: Military sensitivity levels/compartments
- Simple Security Property: S may read O only if $C(O) \leq C(S)$
 - Terminology: No “read-ups”
- *-Property: S may write to P only if $C(O) \leq C(P)$ for all objects O which S has read access to
 - Terminology: No “write-downs”

Secure Flow of Information



- Bell-La Padula Model:
 - concentrates on confidentiality,
 - ignores integrity
- Biba Integrity Model
 - focuses on integrity – data classified with “integrity levels”, not secrecy levels
 - dual (counterpart) of Bell-La Padula Model
- **Simple Integrity Property:** Subject S can modify (have write access to) object O only if $I(S) \geq I(O)$ (*no write up*)
- **Integrity *-property:** If subject S has read rights to object O with integrity level $I(O)$, S can have write access to object P only if $I(O) \geq I(P)$ (*no read down*)
 - *Effect: Unreliable data can't corrupt reliable data*

Models for Rights Propagation

- Question: Even if S can't access O now, is it possible that some sequence of operations leads to S gaining access to O ?
 - Example: Bob has read rights to SecretData, but Alice does not. Later, Bob grants read rights to Alice.
- Operations that change Access Control Matrix:
 - Create/Delete Subject
 - Create/Delete Object
 - Enter right r into $A[S,O]$ (i.e., $A[S,O] = A[S,O] \cup \{r\}$)
 - Remove r from $A[S,O]$ (i.e., $A[S,O] = A[S,O] - \{r\}$)

Rights Propagation – Harrison Ruzzo Ullman (HRU) model

- Commands/Conditional Commands are made up of multiple primitive operations and an optional set of conditions:

General Form

command name(O_1, O_2, \dots, O_k)

if $r_1 \in A[S_1, O_1]$ and

$r_2 \in A[S_2, O_2]$ and

...

$r_m \in A[S_m, O_m]$

then

op_1

op_2

...

op_n

Mono-operational Command:

command GrantRead(Grantor, Grantee, File)

if $own \in A[\text{Grantor}, \text{File}]$

then enter r into $A[\text{Grantee}, \text{File}]$

General Command:

command CreateFile(User, Dir, NewF)

if $w \in A[\text{User}, \text{Dir}]$

then create object NewF

enter own into $A[\text{User}, \text{NewF}]$

enter r into $A[\text{User}, \text{NewF}]$

enter w into $A[\text{User}, \text{NewF}]$

Rights Propagation – HRU Results

Definition: A system “leaks right r ” if right r is added to a matrix entry not containing r .

- *Note: Can be a “fake right”, granted only in a “bad configuration”*

Theorem 1: There exists an algorithm that determines whether a mono-operational system can leak right r .

- *Note: Algorithm can take exponential time.*

Theorem 2: No algorithm can decide whether a general system can leak right r .

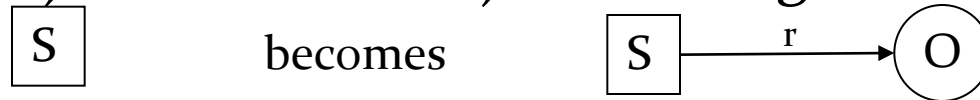
- *Note: No matter how much time is available!*

- Problem: In the HRU model, even decidable questions are inefficient.
 - So: Is there a better representation for rights that allows for more efficiently deciding important questions?
- Take-Grant model represents protection system as a graph, and some questions are efficiently answerable (linear time even!)

Rights Propagation – Take-Grant Model

- Basic operations (note: subjects are also objects!):

- Subject S creates object with rights r



- Subject S revokes right set r from O



- Subject S grants right set r on object P to O



- Subject S takes right set r to object P from O



Take-Grant Model: Results

- Result 1: Given a take-grant graphs, a pair of vertices u and v , and a set of rights r , an algorithm running in $O(n+m)$ time can determine if any set of transitions can lead to an edge connecting u and v with rights r .
 - *Note: Requires that all intermediate transitions be “willing accomplices.”*

- Result 2: Given a take-grant graphs, a pair of vertices u and v , and a set of rights r , an algorithm running in $O(n+m)$ time can determine if u can “steal” rights r to v .
 - *“Stealing” basically rules out granting rights r to another vertex.*