# Security Policies

- Security policies describe how a secure system should behave
- Policy says what should happen, <u>not</u> how you achieve that
- Generally, if you don't have a clear policy, you don't have a secure system
  - Since you don't really know what you're trying to do

# Informal Security Policies

- “Users should only be able to access their own files, in most cases.”
- “Only authorized users should be able to log in.”
- “System executables should only be altered by system administrators.”
- The general idea is pretty clear
- But it can be hard to determine if a system meets these goals

# Formal Security Policies

- Typically expressed in a mathematical security policy language
- Tending towards precision
  - Allowing formal reasoning about the system and policy
- Often matched to a particular policy model
  - E.g., Bell-La Padula model
- Hard to express many sensible policies in formal ways
  - And hard to reason about them usefully

# Some Important Security Policies

- Bell-La Padula
- Biba integrity policy

# Bell-La Padula Model

- Probably best-known computer security model
- Corresponds to military classifications
- Combines mandatory and discretionary access control
- Two parts:
  - Clearances
  - Classifications

# Clearances

- Subjects (people, programs, etc.) have a *clearance*
- Clearance describes how trusted the subject is
- E.g., *unclassified*, *confidential*, *secret*, *top secret*

# Classifications

- Each object (file, database entry, etc.) has a *classification*
- The classification describes how sensitive the object is
- Using same categories as clearances
- Informally, only people with the same (or higher) clearance should be able to access objects of a particular classification

# Goal of Bell-La Padula Model

- Prevent any subject from ever getting read access to data at higher classification levels than subject’s clearance
  - I.e., don’t let untrusted people see your secrets
- Concerned not just with objects
- Also concerned with the objects’ contents
- Includes discretionary access control
  - Which we won’t cover in lecture

# Bell-La Padula Simple Security Condition

- *Subject S can read object O iff* $l_O \leq l_S$
- Simple enough:
  - If S isn't granted top secret clearance, S can't read top secret objects
- Are we done?

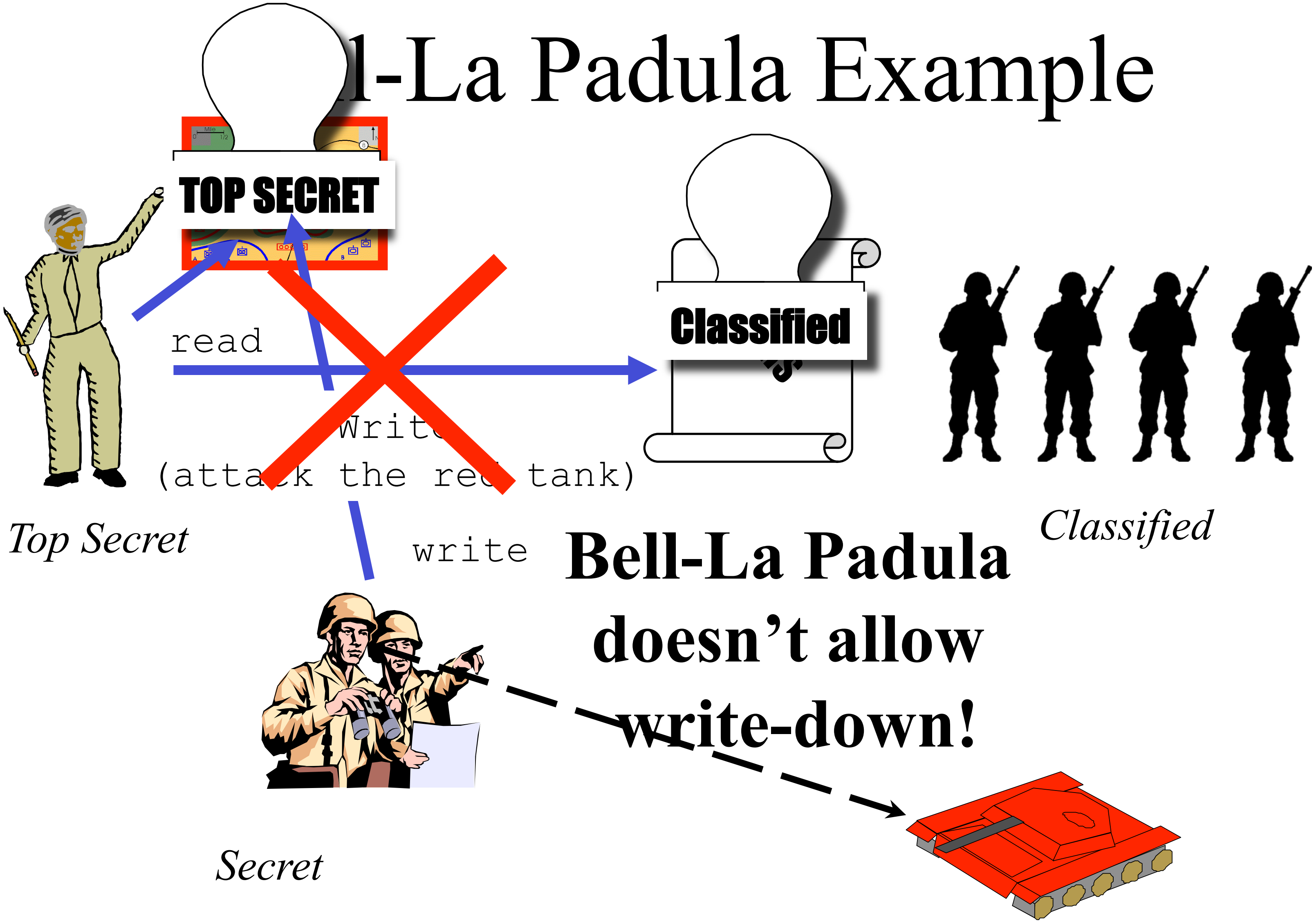
# Why Aren’t We Done?

- Remember, we really care about the information in an object
- A subject with top secret clearance can read a top secret object
- If careless, he could write that information to a confidential object
- Then someone with confidential clearance can read top secret information

# The Bell-La Padula *-Property

- *S can write O iff* $l_S \leq l_O$
- Prevents *write-down*
  - Privileged subjects writing high-classification information to low-classification objects
  - E.g., a top secret user can't write to a confidential data file
- Can be proven that a system meeting these properties is "secure"

# l-La Padula Example



**Bell-La Padula doesn’t allow write-down!**

# So How Do You Really Use The System?

- There have to be mechanisms for reclassification
  - Usually requiring explicit operation
- Danger that reclassification process will be done incautiously
- Real systems also use classes of information

# Integrity Security Policies

- Designed to ensure that information is not improperly changed
- Often the key issue for commercial systems
- Secrecy is nice, but not losing track of your inventory is crucial

# Example: Biba Integrity Policy

- Subject set S, object set O
- Set of ordered integrity levels I
- Subjects and objects have integrity levels
- Subjects at high integrity levels are less likely to screw up data
  - E.g., trusted users or carefully audited programs
- Data at a high integrity level is less likely to be screwed up
  - Probably because it badly needs not to be screwed up

# Biba Integrity Policy Rules

- $s$ can write to $o$ iff $i(o) \leq i(s)$
- $s_1$ can execute $s_2$ iff $i(s_2) \leq i(s_1)$
- A subject $s$ can read object $o$ iff $i(s) \leq i(o)$
- Why do we need the read rule?

# Hybrid Models

- Sometimes the issue is keeping things carefully separated
- E.g., a brokerage that handles accounts for several competing businesses
- Microsoft might not like the same analyst working on their account and IBM’s
- There are issues of both confidentiality and integrity here
- Example – Chinese Wall model

# The Realities of Discretionary Access Control

- Most users never change the defaults on anything
  - Unless the defaults prevent them from doing something they want to do
- Most users don't think about or understand access control
- Probably not wise to rely on it to protect information you care about
  - Unless you're the one setting it
  - And you know what you're doing

# The Problems With Security Policies

- Hard to define properly
  - How do you determine what to allow and disallow?
- Hard to go from policy to the mechanisms that actually implement it
- Hard to understand implications of policy
- Defining and implementing policies is a lot of work