CIS 551 / TCOM 401

Computer and Network Security

Spring 2007

Lecture 5

Announcements

- Reminder:
  - Send in project groups by the 25th
  - If you haven't started on the project -- start now.

- Some of today's slides are adapted from slides by John Mitchell

Recap from last time

- We've been studying Access Control Mechanisms
  - Access control lists
  - Capabilities
  - Unix/Windows OS access control
  - Stack inspection
- Today:
  - Discretionary access control (DAC)
  - Mandatory access control (MAC)
  - Information-flow security

Access Control

- *Discretionary*: The individual user may, at his own discretion, determine who is authorized to access the objects he creates.

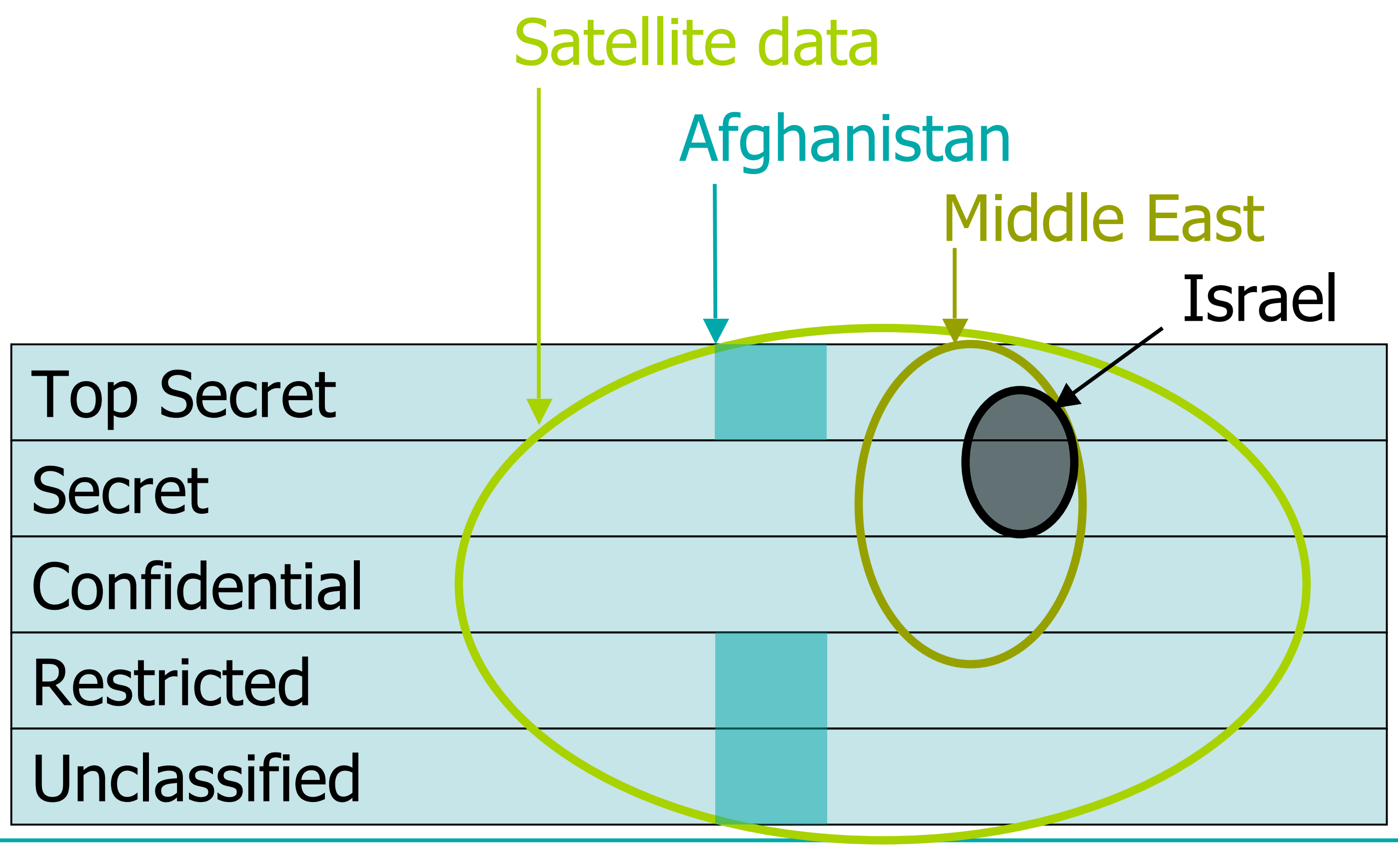
- *Mandatory*: The creator of an object does not necessarily have the ability to determine who has authorized access to it.
  - Typically policy is governed by some central authority
  - The policy on an object in the system depends on what object/information was used to create the object.
  - Examples?

Multilevel Security

- Multiple levels of confidentiality ratings
- Military security policy
    - Classification involves sensitivity levels, compartments
    - Do not let classified information leak to unclassified files
- Group individuals and resources
  – Use some form of hierarchy to organize policy
- Trivial example: Public ≤ Secret
- *Information flow*
  – Regulate how information is used throughout entire system
  – A document generated from both Public and Secret information must be rated Secret.
  – Intuition: "Secret" information should not flow to "Public" locations.

Military security policy

- Sensitivity levels
- Compartments



Military security policy

- Classification of personnel and data
  - Class D = $\langle$rank, compartment$\rangle$
- Dominance relation
  - $D_1 \le D_2$ iff $rank_1 \le rank_2$
    and $compartment_1 \subseteq compartment_2$
  - Example: $\langle$Restricted, Israel$\rangle$ $\le$ $\langle$Secret, Middle East$\rangle$
- Applies to
  - Subjects – users or processes: C(S) = "clearance of S"
  - Objects – documents or resources: C(O) = "classification of O"

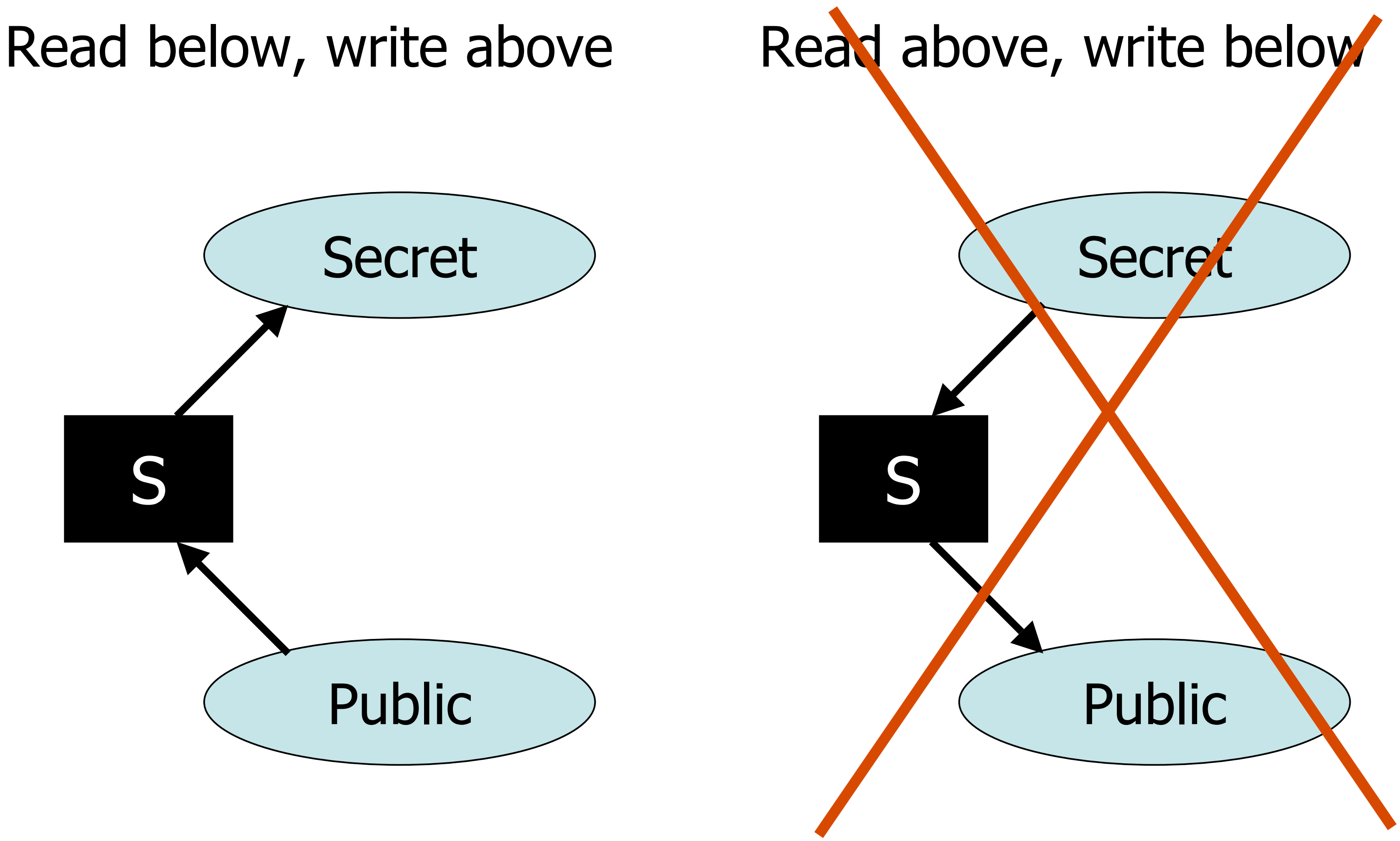
Bell-LaPadula Confidentiality Model

- “No read up, no write down.”
  - Subjects are assigned clearance levels drawn from the lattice of security labels.

    C(S) = "clearance of the subject S"
  - A principal may read objects with lower (or equal) security label.
    - Read: $C(O) \leq C(S)$
  - A principal may write objects with higher (or equal) security label.
    - Write: $C(S) \leq C(O)$

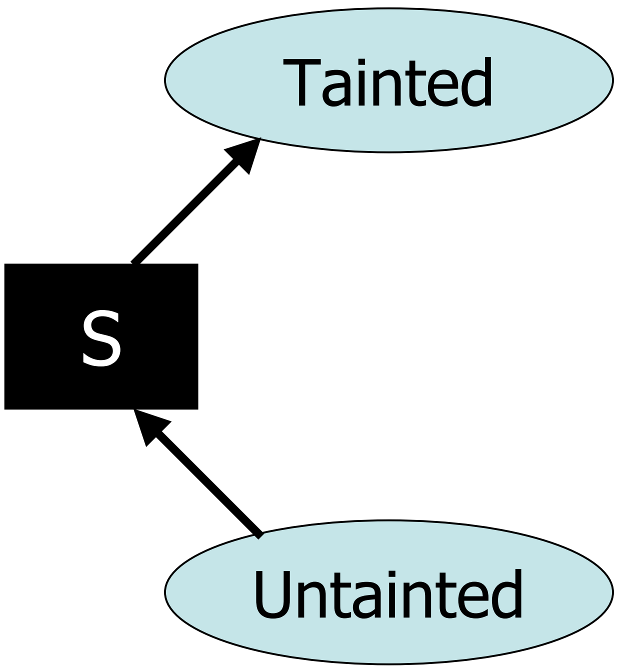
- Example:
  A user with Secret clearance can:
  - Read objects with label Public and Secret
  - Write/create objects with label Secret

Picture: Confidentiality

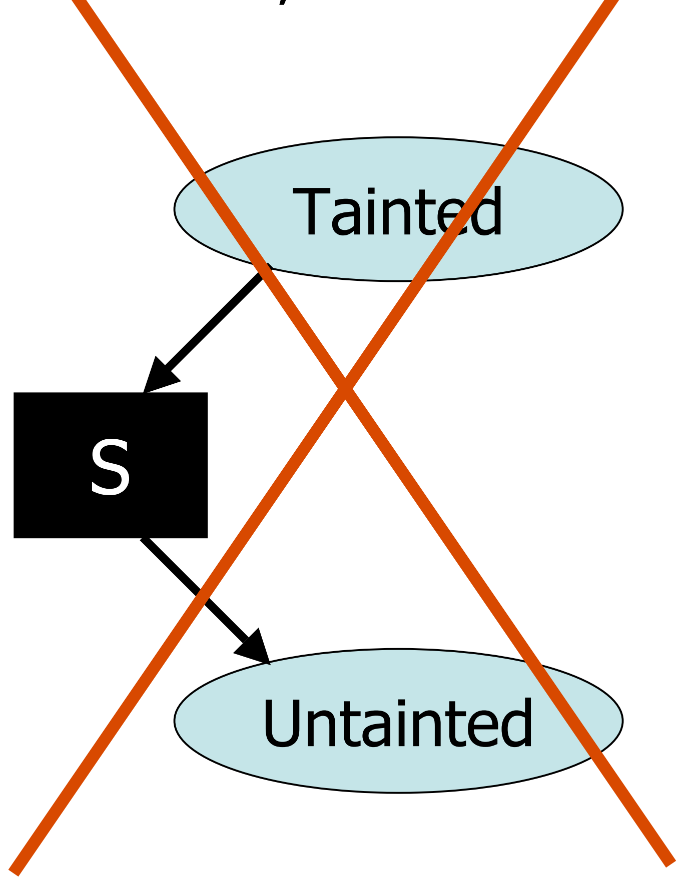


Picture: Integrity

Read below, write above



Read above, write below



Multilevel Security Policies

- In general, security levels form a "join semi-lattice"
  - There is an ordering $\leq$ on security levels
  - For any pair of labels L1 and L2 there is an "join" operation:

    $L1 \oplus L2$ is a label in the lattice such that:
    (1) $L1 \leq L1 \oplus L2$ and $L2 \leq L1 \oplus L2$ "upper bound"
    (2) If $L1 \leq L3$ and $L2 \leq L3$ then $L1 \oplus L2 \leq L3$ "least bound"

- For example: Public $\oplus$ Secret = Secret
- Labeling rules:
  - Classification is a function C : Object $\rightarrow$ Lattice
  - If some object O is "created from" objects $O_1,\ldots,O_n$ then $C(O) = C(O_1) \oplus \ldots \oplus C(O_n)$

Implementing Multilevel Security

- Dynamic:
  - Tag all values in memory with their security level
  - Operations propagate security levels
  - Must be sure that tags can't be modified
  - Expensive, and approximate
- Classic result: Information-flow policies cannot be enforced purely by a reference monitor!
  - Problem arises from implicit flows
- Static:
  - Program analysis
  - May be more precise
  - May have less overhead

Information Flows through Software

Explicit Flows:

```
int{Secret} X = f();
int{Public} Y = 0;

Y = X;
```

Implicit Flows:

```
int{Secret} X = f();
int{Public} Y = 0;
int{Public} Z = 0;
int{Public} W = 0;

if (X > 0) then {
  Y = 1;
} else {
  Z = 1;
}
W = 3;
```

Perl's Solution (for Integrity)

- The problem: need to track the source of data
- Examples: Format string, SQL injection, etc.

```
$arg = shift;
system ("echo $arg");
```

- Give this program the argument `"; rm *"`
- Perl offers a *taint checking* mode
  - Tracks the source of data (trusted vs. tainted)
  - Ensure that tainted data is not used in system calls
  - Tainted data can be converted to trusted data by pattern matching
  - Doesn't check implicit flows

SELinux

- Security-enhanced Linux system (NSA)
  - Enforce separation of information based on confidentiality and integrity requirements
  - Mandatory access control incorporated into the major subsystems of the kernel
    - Limit tampering and bypassing of application security mechanisms
    - Confine damage caused by malicious applications

http://www.nsa.gov/selinux/

SELinux Security Policy Abstractions

- Security-Encanced Linux
  - Built by NSA
- Type enforcement
  - Each process has an associated domain
  - Each object has an associated type (label)
  - Configuration files specify
    - How domains are allowed to access types
    - Allowable interactions and transitions between domains
- Role-based access control
  - Each process has an associated role
    - Separate system and user processes
  - Configuration files specify
    - Set of domains that may be entered by each role

Two Other MAC Policies

- "Chinese Wall" policy: [Brewer & Nash '89]
  - Object labels are classified into "conflict classes"
  - If subject accesses one object with label L1 in a conflict class, all access to objects labeled with other labels in the conflict class are denied.
  - Policy changes dynamically

- "Separation of Duties":
  - Division of responsibilities among subjects
  - Example: Bank auditor cannot issue checks.

Covert Channels & Information Hiding

- A covert channel is a means by which two components of a system that are not permitted to communicate do so anyway by affecting a shared resource.

- Information hiding: Two components of the system that are permitted to communicate about one set of things, exchange information about disallowed topics by encoding contraband information in the legitimate traffic.

- Not that hard to leak a small amount of data
  - A 64 bit encryption key is not that hard to transmit
  - Even possible to encode relatively large amounts of data!

- Example channels / information hiding strategies
  - Program behavior
  - Adjust the formatting of output:
    use the “\t” character for “1” and 8 spaces for “0”
  - Vary timing behavior based on key
  - Use "low order" bits to send signals
  - Power consumption
  - Grabbing/releasing a lock on a shared resource

Watermarking Basic Idea

- Pictures, Video, and Sound
  - Human perception is imperfect
  - There are a lot of “least significant bits”
  - Modifying the least significant bits doesn’t change the picture much

(R,G,B) = (182,54,89)

(R,G,B) = (182,54,90)

- Encode a signal in the least significant bits.

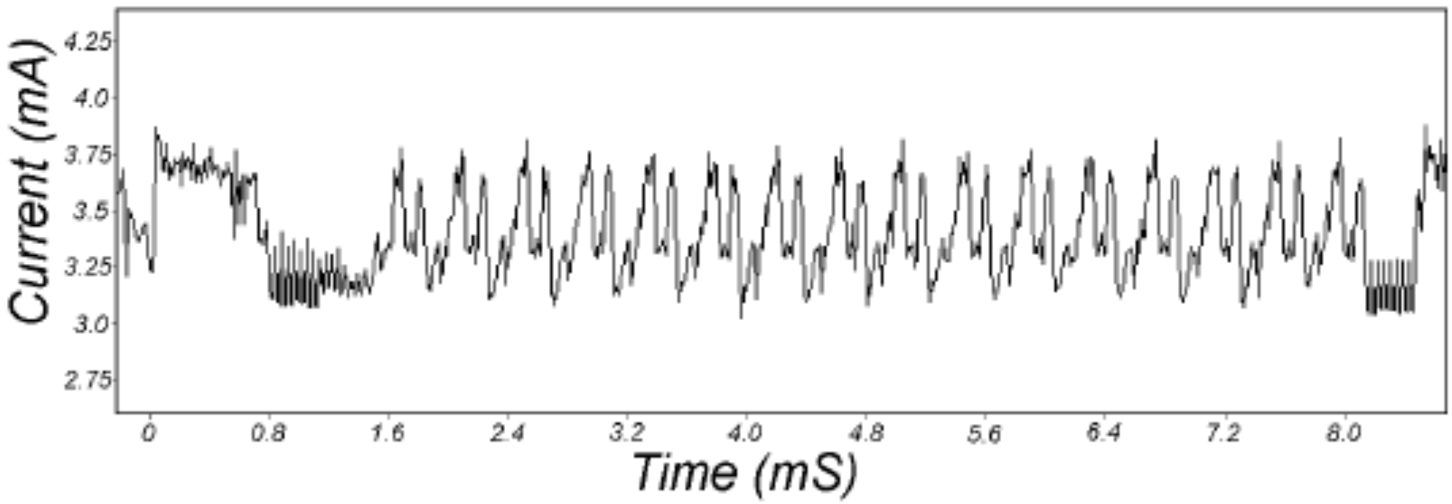
Watermarking Example

Original Image

Watermarked Image

Differential Power Analysis

- Read the value of a DES password off of a smartcard by watching power consumption!



- This figure shows simple power analysis of DES encryption. The 16 rounds are clearly visible.

TEMPEST Security

- Transient Electromagnetic Pulse Emanation Standard
  - (Or?) Temporary Emanation and Spurious Transmission
  - Emission security (Van Eck phreaking)
  - computer monitors and other devices give off electromagnetic radiation
  - With the right antenna and receiver, these emanations can be intercepted from a remote location, and then be redisplayed (in the case of a monitor screen) or recorded and replayed (such as with a printer or keyboard).
- Policy is set in National Communications Security Committee Directive 4
- Guidelines for preventing EM reception
  - Shield the device (expensive)
  - Shield a location (inconvenient?)

Defenses for Covert Channels

- Well specified security policies at the human level
- Auditing mechanisms at the human level
  - Justify prosecution if the attacker is caught
- Code review
  - This is a form of audit
- Automated program analysis
  - Type systems that let programmers specify confidentiality labels li
  - Transform programs so that both branches of a conditional statement take the same amount of time
  - Disallow branches on "secret" information

Countermeasures

- Against timing attacks:
  - Make all operations run in same amount of time
    - Hard to implement!
    - Can’t design platform-independent algorithms
    - All operations take as long as slowest one
  - Add random delays
    - Can take more samples to remove randomness
- Against power analysis attacks:
  - Make all operations take the same amount of power
    - Again, hard to implement
  - Add randomness