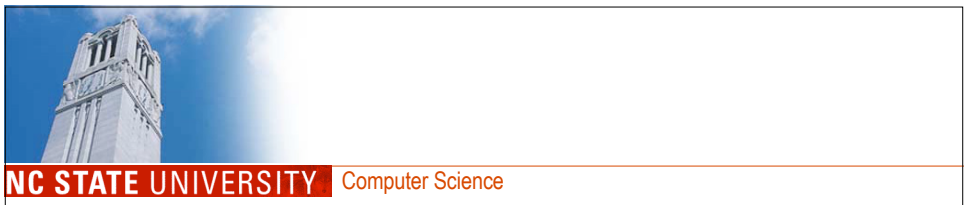


NC STATE UNIVERSITY Computer Science

CSC 405
Introduction to Computer Security

Topic 5. Trusted Operating Systems
-- Part I

Secure v.s. Trusted

- Secure
  - Something either is or is not secure
  - Property of *presenter*
  - *Asserted* based on product characteristics
  - *Absolute*: not qualified as to how, where, when or by whom used
  - A *goal*
- Trusted
  - *Graded*: degree of trustworthiness
  - Property of *receiver*
  - *Judged* based on evidence and analysis
  - *Relative*: viewed in context of use
  - A *characteristic*

Trusted Operating Systems

- Policy
  - A set of rules that lay out what is to be secured and why
- Model
  - Representation of policy
  - Facilitate the analysis of the policy
- Design
  - Intended functionality
  - Implementation
- Trust
  - Features: necessary functionality to enforce the security policy
  - Assurance: confidence that the OS will enforce the policy

Security Policies

- Military security policy
- Commercial security policies
  - Clark-Wilson policy
  - Separation of duty
  - Chinese wall security policy

Military Security Policy

- Classification of information
  - Ranks of levels: unclassified, restricted, confidential, secret, top secret
- Need-to-know
  - Access to sensitive data is allowed only to subjects who need to know them to perform their jobs
  - Compartment
    - Projects describing the subject matter of the information
- Security class (of information)
  - Determined by both rank and compartment
  - Example: <secret, {crypto}>

Military Security Policy (Cont'd)

- Clearance
  - A clearance is an indication that a person is trusted to access information up to
    - A certain level of sensitivity, and
    - Certain categories of information
  - Denoted in the same way as security classes
- A subject can read an object only if
  - The clearance level is at least as high as that of the information, and
  - The subject has a need-to-know about all compartments for which the information is classified
  - Dominate relation

Clark-Wilson Commercial Security Policy

- In commercial environment, preventing unauthorized data modification is usually paramount
  - No user of the system, even if authorized, may be permitted to modify data items in such a way that assets or accounting records of the company are lost or corrupted
- Introduce the notion of well-formed transactions

Clark-Wilson Commercial Security Policy

- Example
  - A purchasing agent creates an order, and sends the order to the supplier and the receiving department
  - The supplier ships the goods. The receiving department receives and checks the delivery, and signs the delivery form, which then goes to the accounting department
  - The supplier sends an invoice to the accounting department. An accountant compares the invoice with the order and the delivery form, and issues a check to the supplier

Clark-Wilson Commercial Security Policy

- Well-formed transactions
  - Date items must satisfy some integrity constraints
    - Constrained Data Items (CDIs)
  - Perform the steps in order
  - Perform exactly the steps listed
  - Authenticate the individuals who perform the steps
  - Can manipulate data only through trusted code!
    - Called Transformation Procedures (TPs)
- Policy defined in terms of access triples
  - <userID, TP, {CDIs}>
  - Combine a transformation procedure, one or more constrained data items, and the identification of authorized user

Separation of Duty

- Required division of responsibilities
  - Also a part of Clark-Wilson model
- Examples
  - Different people must issue orders, receiver goods, and write checks
  - Any check with amount over $100,000 must have two separate authorization signatures from designated officials

Clark-Wilson Model (Cont'd)

- Mechanisms are needed to ensure
  - A data item can be manipulated only by a specific set of programs
  - Programs must be inspected for proper construction; controls must be provided on the ability to install and modify these programs
  - Each user must be permitted to use only certain sets of programs
  - Assignment of people to programs must be controlled and inspected

Differences from Other Models

- A data item is not associated with a particular security level, but rather with a set of TPs
- A user is not given read/write access to data items, but rather permissions to execute certain programs

The Clarke-Wilson Model for Integrity (1)

- Unconstrained Data Items (UDIs)
  - data with low integrity
- Constrained Data Items (CDIs)
  - data items within the system to which the integrity model must apply
- Integrity Verification Procedures (IVPs)
  - confirm that all of the CDIs in the system conform to the integrity specification
- Transformation Procedures (TPs)
  - well-formed transactions

The Clarke-Wilson Model for Integrity (2)

- C1: (Certification) All IVPs must properly ensure that all CDIs are in a valid state at the time the IVP is run
- C2: All TPs must be certified to be valid. That is, they must take a CDI to a valid final state, given that it is in a valid final state to begin with. For each TP, the security officer must specify the set of CDIs that the TP has been certified

The Clarke-Wilson Model for Integrity (3)

- E1: (Enforcement) The system must ensure that only TPs can access CDIs and any TP can only access the CDIs it is certified for
- E2: The system must maintain a relation of the form, (UserID, TPi, (CDIa, CDIb, CDIc,…). A user can only execute TPs that it is allowed to access

The Clarke-Wilson Model for Integrity (4)

- C3: The relation in E2 must be certified to meet the separation of duty requirement
- E3: The system must authenticate the identity of each user attempting to execute a TP

The Clarke-Wilson Model for Integrity (5)

- C4: All TPs must be certified to write to an append-only CDI (the log) all information necessary to permit the nature of the operation to be reconstructed
- C5: Any TP that takes a UDI as input must be certified to 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

The Clarke-Wilson Model for Integrity (6)

- E4: Only the agent permitted to certify entities may do so. An agent that can certify entity (TP or CDI) may not have any execute rights with respect to that entity.

The Chinese Wall Security Policy

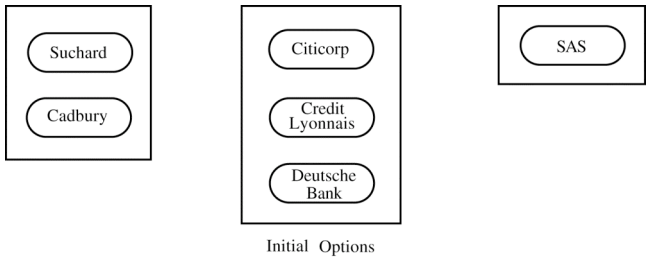
- Data are stored in a hierarchical arranged system
  - the lowest level consists of individual data items
  - the intermediate level group data items into company data sets
  - the highest level group company datasets whose corporation are in competition

Simple Security Rule in Chinese Wall Policy

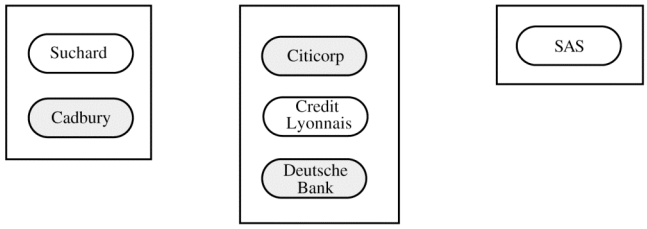
- Access is only granted if the object requested
  - is in the same company dataset as an object already accessed by that subject, i.e., within the Wall,

    or
  - belongs to an entirely different conflict of interest class.

Example -- Chinese Wall Security Policy



Initial Options



After Selecting Suchard and Credit Lyonnais

Security Models

- Multi-level security models
  - Bell La Padula model -- multi-level confidentiality
  - Biba model -- multi-level integrity
  - Models for determining the system security policy
- Models for analysis purposes
  - Graham-Denning model
  - Harrison-Ruzzo-Ullman (HRU) model
  - Take-grant systems
  - Models for understanding the properties of the protection system

Discretionary Access Control (DAC) Revisited

- DAC allows access rights to be propagated at subject's discretion
  - often has the notion of owner of an object
  - used in UNIX, Windows, etc.
- "A means of restricting access to objects based on the identity and need-to-know of users and/or groups to which the object belongs. Controls are discretionary in the sense that a subject with a certain access permission is capable of passing that permission (directly or indirectly) to any other subject."

Mandatory Access Control

- Mandatory access controls (MAC) restrict the access of subjects to objects based on a system-wide policy
  - Denying users full control over the access to resources that they create
  - The system security policy (as set by the administrator) entirely determines the access rights granted

Multi-Level Security (MLS)

- The capability of a computer system to
  - Carry information with different sensitivities
  - Permit simultaneous access by users with different security clearances and needs-to-know, and
  - Prevent users from obtaining access to information for which they lack authorization
- Typically use MAC
- Primary Security Goal: Confidentiality

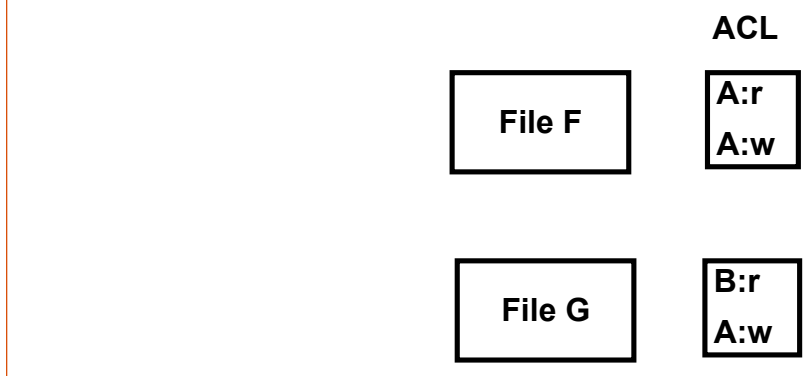
Why MAC is Necessary -- Inherent Weakness of DAC

- Unrestricted DAC allows information from an object which can be read to any other object which can be written by a subject
  - Do not provide multi-level security
- Suppose our users are trusted not to do this deliberately. It is still possible for Trojan Horses to copy information from one object to another

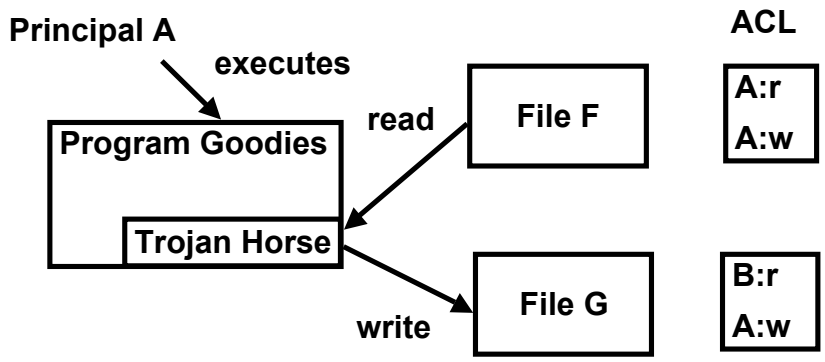
Trojan Horse Revisited

- A Trojan Horse is rogue software installed, perhaps unwittingly, by duly authorized users
- A Trojan Horse does what a user expects it to do, but in addition exploits the user's legitimate privileges to cause a security breach

Trojan Horse Example



Trojan Horse Example



- Principal B can read contents of file F copied to file G

Trojan Horse Revisited (Cont'd)

- Viruses and logic bombs are examples of Trojan Horses
- It is possible to embed Trojan Horses in hardware and firmware
- It is possible to embed Trojan Horses in critical system software such as compilers and Database Management Systems

Bell-LaPadula Model: A MAC Model

- Introduced in 1973
- Air Force was concerned with security in time-sharing systems
  - Many OS bugs
  - Accidental misuse
- Basic idea
  - Information should not flow downward
- Main Objective
  - Enable one to show that a computer system can securely process classified information

Basic Idea

- There are security classifications or security levels
- Example
  - Top Secret
  - Secret
  - Confidential
  - Unclassified
- In this case, Top Secret > Secret > Confidential > Unclassified

A Simplified Version of the Model

- A computer system is modeled as a state-transition system
  - In each state, each subject is given a level (clearance), and each object is given a level (security class)
- A state is secure if it satisfies
  - Simple Security Condition (no read up):
    - S can read O iff $L(S) \geq L(O)$
  - The Star Property (no write down)
    - S can write O iff $L(S) \leq L(O)$

Star-Property

- Applies to subjects (principals) not to users
- Users are trusted (must be trusted) not to disclose secret information outside of the computer system
- Subjects are not trusted because they may have Trojan Horses embedded in the code they execute
- Star-property prevents overt leakage of information and does not address the covert channel problem

Issues with BLP

- Deal only with confidentiality
  - does not deal with integrity at all
- Does not deal with information flow through covert channels
- The approach of defining a secure system to be one in which every reachable state is secure is flawed

Covert Channels Revisited

- Security objective of MLS in general, BLP in particular
  - high-classified information cannot flow to low-cleared users
- Overt channels of information flow
  - read/write an object
- Covert channels of information flow
  - communication channel based on the use of system resources not normally intended for communication between the subjects (processes) in the system

Covert Channels Revisited (Cont'd)

- Covert channels cannot be blocked by *-property
- It is generally very difficult, if not impossible, to block all cover channels
- One can try to limit the bandwidth of covert channels
- Military requires cryptographic components be implemented in hardware
  - to avoid Trojan horse leaking keys through covert channels

More on MLS: Security Levels

- Used as attributes of both subjects & objects
  - clearance & classification
- Typical military security levels:
  - top secret ≥ secret ≥ confidential ≥ unclassified
- Typical commercial security levels
  - restricted ≥ proprietary ≥ sensitive ≥ public

Security Categories

- Also known as compartments
- Typical military security categories
  - army, navy, air force
  - nato, nasa, noforn
- Typical commercial security categories
  - Sales, R&D, HR
  - Dept A, Dept B, Dept C

Security Labels

- Labels = Levels × P (Categories)
- Define an ordering relationship among Labels
  - (e1, C1) ≤ (e2, C2) iff. e1 ≤e2 and C1 ⊆ C2
- This ordering relation is a partial order
  - reflexive, transitive, anti-symmetric
  - e.g., ⊆
- All security labels form a lattice
  - Least upper bound
  - Greatest lower bound

The Need-To-Know Principle

- Even if someone has all the necessary official approvals (such as a security clearance) to access certain information, they should not be given access to such information unless they have a *need to know*:
  - That is, unless access to the specific information necessary for the conduct of one's official duties.
- More easily implemented using DAC

Need for Multi-Level Integrity Protection

- Bell-LaPadula and other information-flow based security definitions address confidentiality, what about integrity
- What does integrity mean?
  - System integrity: system behave as expected
  - Data integrity: data not changed in "incorrect" ways
- One difference between confidentiality & integrity
  - A subject cannot leak a piece of confidential information without reading it, but can introduce low-integrity information without reading any
    - Some trust has to be placed on subjects for integrity

Integrity Defined (Biba)

- A subsystem possesses the property of integrity if it can be trusted to adhere to a well-defined code of behavior
- How to guarantee integrity?
  - The subsystem needs to be initially determined (by some external agency) to perform properly.
    - e.g., using program verification technique
  - Ensure that subsystem cannot be corrupted to perform in a manner contrary to the original determination.

Biba: Integrity Levels

- Each subject (program) has an integrity level
  - reflects confidence in the program executing correctly (what does "correctly" mean?)
- Each object has an integrity level
  - reflects degree of confidence in the data
    - quality of info in an object vs. importance of an object
- Integrity levels are totally ordered
- Integrity levels different from security levels
  - a highly sensitive data may have low integrity (e.g., information collected by spy)

Five Mandatory Policies in Biba

- Strict integrity policy
- Subject low-water mark policy
- Object low-water mark policy
- Low-water mark Integrity Audit Policy
- Ring policy

Strict Integrity Policy

- Rules:
  - s can read o iff $i(s) \leq i(o)$
    - no read down
    - stops indirect sabotage by contaminated data
  - s can write to o iff $i(o) \leq i(s)$
    - no write up
    - stops directly malicious modification
- Ensures no information path from low-integrity object to high-integrity object
  - Why is this desirable?

Subject Integrity Levels

- What does it mean that a subject is trusted to execute correctly at integrity level i1?
- Three possibilities:
  - Generate information at level i1 from any data
  - Generate information at level i1 when reading data of integrity level i1 or higher
  - Generate information at any level $i \leq i1$ when reading data of integrity level i or higher

Object Integrity Levels

- An object integrity level may be based on
  - Quality of information (levels may change)
  - Importance of the object (levels do not change)
- Intuitively, quality integrity level should be at least as high as importance integrity level
- Quality integrity level may be higher than importance integrity level

Subject Low-Water Policy

- Subject's integrity level decreases as reading lower integrity data
- The reading rule is relaxed:
  - When s reads o, the integrity level of s is set to min[i(s), i(o)]
  - Can read down, but lower integrity level
  - If the integrity levels are not totally ordered, then glb[i(s), i(o)]
- Ensures that there is no information path from low integrity data to high integrity data

Object Low-Water Mark Policy

- The writing rule is relaxed:
  - When s writes o, the integrity level of o is set to min[i(s),i(o)]
  - Implies that object integrity level represents quality rather than importance
- Also ensures that there is no information path from a low integrity object to a high integrity object

Low-Water Mark Integrity Audit Policy

- The integrity levels of subjects and objects both change to reflect the contamination
  - After s observes o, the integrity level of s is lowered to min(i(s), i(o))
  - After s modifies o, the integrity level of o is lowered to min(i(s), i(o))

The Ring Policy

- Integrity levels of subjects and objects are fixed
- Rules
  - Any subject can read any object
  - s can write to o iff $i(o) \leq i(s)$
- Intuitions
  - subjects are trusted to process inputs correctly, and to generate outputs of a certain integrity level

Summary of Biba's Policies

- Different policies assume different kinds of trust in subjects
  - The ring model assumes subjects can correctly process inputs and generate data of a certain integrity level
  - The low-water mark models assume subjects do not introduce low integrity information themselves, but may be contaminated by the source
  - The strict integrity model assumes subjects may be contaminated by the source and can only generate data of a certain integrity level

Key Difference between Confidentiality and Integrity

- For confidentiality, no trust needs to be placed on subjects
  - Theoretically, no subject needs to be trusted for confidentiality; however, one does need trusted subjects in BLP to make system realistic
- For integrity, one has to trust subjects
  - Therefore, one has to justify such trust

Graham-Denning Model

- A formal system of protection rules
- Components of the model
  - A set of subjects S
  - A set of objects O
  - A set of rights R
    - Each right can be transferable or non-transferable
  - An access control matrix A
  - Each object has a subject owner
  - Each subject has a designated subject as the controller

Graham-Denning Model (Cont'd)

- Eight primitive protection rights
  - Phrased as commands issued by subjects,with effects on other subjects or objects
  - Create object o
  - Create subject s
  - Delete object o
  - Delete subject s
  - Read access right of s on o
  - Delete access right of s on o
  - Grant access right r to s on o
  - Transfer access right r or r* to s on o

Harrison-Ruzzo-Ullman (HRU) Model

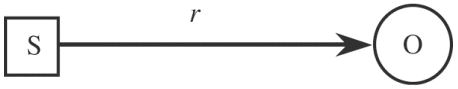
- A variation on the Graham-Denning model
- Used to answer the following type of questions
  - Will user X ever have access to object Y, given a configuration of access controls?
- Differences from Graham-Denning model
  - Each subject is an object, too
  - Six primitive operations
    - Create object o
    - Create subject s
    - Destroy subject s
    - Destroy object o
    - Enter right r into A[s, o]
    - Delete right r from A[s, o]

HRU Model (Cont'd)

- Expressive
  - These HRU operations are adequate to model protection systems
- Two important results
  - In the modeled system, in which commands are restricted to a single operation each, it is possible to decide whether a given subject can ever obtain a particular right to an object
  - If commands are not restricted to one operation each, it is not always decidable whether a given protection system can confer a given right

Take-Grant Systems

- Four primitive operations
  - Create
  - Revoke
  - Take
  - Grant
  - Take and grant are new operations
- Has a graph representation



Take-Grant Systems (Cont'd)