Overview

- A trusted OS must provide (minimally) memory protection, file protection, access control, and user authentication
  - Authentication deferred to later...
  - Policies/models, design, assurance

"Trust" vs. "Security"

- A "trusted" system meets the stated or expected security requirements
  - May or may not be "secure"...
  - May be degrees of trust...

Security policies/models

- What model to use?
  - "Military security policy"
    - Primarily concerned with secrecy
    - Information ranked at sensitivity level within a hierarchy (e.g.: unclassified, secret, top secret), and also placed within (one or multiple) "compartments"
  - "Classification" of data = (rank; compartments)
  - Compartments no longer hierarchical...
Military security policy

- Primarily concerned with secrecy
- Information ranked at sensitivity level within a hierarchy (e.g.: unclassified, secret, top secret), and also placed within (one or multiple) “compartments”
- “Classification” of data = (rank; compartments)

Military policy

- Subjects given “clearance”
  - Expressed as (rank; compartments)
  - “Need to know” basis
  - Subject with clearance \((r, C)\) can access object with classification \((r', C')\) only if \(r \geq r'\) and \(C' \subseteq C\)
- Assumes a central “security officer” controlling designations

Commercial security policy

- Typically more relaxed than military security
- May also be more concerned with integrity and/or availability

Example: Clark-Wilson model

- Transactions are the basic operation
  - Not subjects/objects
  - The system should always remain in a "consistent state"
    - A well-formed transaction leaves the system in a consistent state
  - Security policy determined in terms of access triples: allowed transactions, associated data items, and user(s) authorized to perform the transaction
Separation of duty

• In Clark-Wilson model, no way to “pass state” from one access triple to another
• Might allow the same person to perform multiple, related transactions
• Better to augment policy to ensure that different users are in charge of related transactions
• No one user has “too much” power

“Chinese Wall” security policy

• Prevents conflicts of interest
• Objects, company groups, conflict classes
• Subject disallowed from reading from two company groups within same conflict class
• Permissions change dynamically...

Mechanisms for enforcing policy

• The precision of a mechanism is a measure of how overly-restrictive the mechanism is with respect to the policy
• I.e., due to preventing things that are allowed
• Unfortunately, it is impossible (in general) to develop a “maximally-precise” mechanism for an arbitrary policy

Multilevel security

• “Lattice-based” schemes
• Bell-LaPadula model
• Identifies allowable communication flows
• Concerned primarily with ensuring secrecy
• Ensures “confinement” of concurrent processes
Bell-LaPadula model I

- Lattice-based security classes
- Subjects have security clearance
- Objects have security classification
- Dominance relation between security classes

Bell-LaPadula model II

- Simple security condition: S can read O if and only if \( l_O \leq l_S \)
  - We have seen this already...
- *-property: S can write O if and only if \( l_S \leq l_O \)
  - Why?
  - "Read down; write up"
  - Information flows upward

Basic security theorem

- If a system begins in a secure state, and always preserves the simple security condition and the *-property, then the system will always remain in a secure state
  - I.e., information never flows down...

Communicating down...

- How to communicate from a higher security level to a lower one?
  - Max. security level vs. current security level
  - Maximum security level must always dominate the current security level
  - Reduce security level to write down...
    - Security theorem no longer holds
    - Must rely on users to be security-conscious
Commercial vs. military systems

- The Bell-LaPadula model does not work as well for commercial systems
- Users given access to data as needed
  - Discretionary access control vs. mandatory access control
  - Would require large number of categories and classifications
- Decentralized handling of “security clearances”

Biba model

- Concerned with integrity
  - “Dual” of Bell-LaPadula model
- Ordered integrity levels
  - The higher the level, the more confidence
    - More confidence that a program will act correctly
    - More confidence that a subject will act appropriately
    - More confidence that data is trustworthy
  - Note that integrity levels may be independent of security classifications
    - Confidentiality vs. trustworthiness
    - Information flow vs. information modification

Biba model

- Simple integrity condition: $S$ can read $O$ if and only if $I_S \leq I_O$
- (Integrity) *-property: $S$ can write $O$ if and only if $I_O \leq I_S$
  - Why?
  - The information obtained from a subject cannot be more trustworthy than the subject itself
  - “Read up; write down”
  - Information flows downward

Security theorem

- If there is an information transfer path from $o_1$ to $o_n$, then $i(o_n) \leq i(o_1)$
  - Informally: information transfer does not increase the trustworthiness of the data
  - Note: nothing about secrecy...
OS design

• Best if security is built-in from the beginning
• Desired security policies impact key design decisions
• “Shoehorning” security at the end likely to lead to poor design...
• Keep in mind security principles

Security features

• Identification/authentication (later)
• Access control
  • Mandatory access control: individual users do not have control over access rights to objects
  • Discretionary access control: users given ability to control access rights (at least to some extent)
  • Can combine both: access allowed only if MAC and DAC both allow access

• Complete mediation
  • All access to all resources must be mediated
• Object reuse protection
  • Must be careful when de-allocating memory for files or programs...
• Trusted path
  • System must be authenticated to the user
  • E.g., password-entry in Windows

• Accountability/audit
  • Ability to trace faults and recover from them
  • Intrusion detection...

Security features

- Security kernel
  - All accesses mediated via kernel
  - Easier to protect security mechanisms if they are isolated
  - Easier to verify (and potentially fix) problems in a small, localized portion of code
  - But, may downgrade performance...

Reference monitor

- Controls access to objects
- Within security kernel
- Must be tamperproof, always invoked when access to an object is requested, and small enough to be analyzed and tested...

Trusted computing base (TCB)

- TCB includes everything needed to enforce security policy
  - Even if all non-TCB components changed arbitrarily, no security violation
  - Includes hardware, notion of (basic) processes and files, protected memory, and inter-process communication
  - Must be run in some protected state
  - Not the full-blown OS!

Some common flaws

- I/O processing
  - Performed by low-level routines which may be outside the security kernel
  - More complex code; more heavily optimized (possibly at the expense of security)
  - Incomplete mediation
  - Generality, system config problems
Assurance

- Testing
  - Can only demonstrate existence of a problem
  - Testing only observed behavior is limited
  - “Tiger team” analysis may be better
- Formal verification
  - Only for simplest portions of code...motivation to keep security kernel simple
  - Complex...
  - Only as good as the conditions verified

Validation

- Assures that all required components have been implemented
- Open source...
- Third-party evaluation
  - US "orange book"

“Orange book”

- D – no requirements
- C1/C2/B1 – “basic” security features
- B2 – proof of security of underlying model, and specification of trusted computing base
- B3/A1 – more precise description and formal designs of trusted computing base

“Orange book”

- C1
  - Cooperating users at same level of sensitivity
  - Access control; users can protect their own data
  - Discretionary access control
- C2
  - Finer granularity of control
  - Better audit functions; each individual access to each object can be tracked
“Orange book”

- B1
  - Non-discretionary access control; subjects and (most) objects assigned a security level
  - Bell-LaPadula model + DAC to further limit access

“Orange book”

- B2
  - Design and implementation go through more thorough review/testing based on verifiable top-level design
  - Independent modules
  - Principle of least privilege
  - Access control for all subjects/objects, including devices
  - Analysis of covert channels

“Orange book”

- B3
  - Security functions small enough for extensive testing/review and tamperproof
  - A1
    - Verifiable design
    - Formal model and proof of consistency
    - Formal analysis of covert channels

What killed the Orange Book?

- Poor performance
- Expensive and slow
- Poor user interface