CS590U
Access Control: Theory and Practice

Lecture 5 (January 24)

Noninterference and Nondeducibility

Security Policies and Security Models

J.A.Goguen and J.Meseguer

Oakland'1982

Distinction Between Models and Policies

- A model describes the system
  - e.g., a high level specification or an abstract machine description of what the system does
  - this paper uses a state transition systems with focus on operations and outputs
- A security policy
  - defines the security requirements for a given system
- Verification shows that a policy is satisfied by a system

Four Stages in Defining Security

1. Determine the security needs of a given community
2. Express those needs as a formal requirement
3. Model the system which that community is (or will be) using
4. Verify that systems in the model satisfies the requirement

- Maybe switch steps 2 & 3, as the formal security requirement will be based on the model; maybe an iterative process.

An Abstract System Model

- S: set of states
- U: set of subjects
- SC: set of state commands
- Out: set of all possible outputs
- do: $S \times U \times SC \rightarrow S$
  - do(s,u,c)=s’ means that at state s, when u performs command c, the resulting state is s’
- out: $S \times U \rightarrow Out$
  - out(s,u) gives the output that u sees at state s
- $s_0 \in S$ initial state

The Additional Capability Component

- Capt: set of capability tables
- CC: set of capability commands
- out: $S \times Capt \times U \rightarrow Out$
- do: $S \times Capt \times U \times SC \rightarrow S$
- cdo: $Capt \times U \times CC \rightarrow Capt$
  - decides how the capability table is updated
- s_0, t_0: initial state and capability table

Summary of the Modeling Aspect

- The system is modeled as a state-transitional system
- Changes state by subjects executing commands
- Each state has an output for each subject
- Implicit assumptions:
  - Initial state of the system does not contain any sensitive information
  - Information comes into the system by commands
  - Only way to get information is through outputs

Security Policies

- A security policy is a set of noninterference assertions
- Definition of noninterference: Given two group of users G and G', we say G does not interfere with G' if for any sequence of commands w, what users in G' can observe after executing w is the same as what users in G can observe after executing $P_G(w)$, which is w with command initiated by users in G removed.
- Similar in spirit to the notion of zero-knowledge in cryptography
  - if what one can see with high inputs is the same as what one sees without high inputs, no high information is leaked

Examples in the Paper

- Example 2: Multilevel Security (with total ordering):
  - given two security levels x and y such that $x > y$, the set of users whose security level is at least x is non-interfering with the set of users whose security level is dominated by y
- Questions:
  - what if security levels are partially ordered?
  - how to compare with the Bell-LaPadula model?

Usage Examples

- Information flow within a programs
  - certain input channels are noninterfering with certain output channels
- Safety in automated trust negotiation
  - how to say that a negotiator’s behavior does not leak information about its sensitive attributes to entities not authorized to know that information

Comparisons of the BLP work & the Noninterference work

- Differences in model
  - modeling internal structure (objects) or the interface (input & output)
- Differences in formulating security policies
  - BLP is about information flow between objects, and noninterference is about information between subjects
  - BLP specifies access control requirement

Comparisons of BLP & Noninterference

- Precise comparisons are difficult to make because of the fact that different system models are used
- In general, BLP is weaker than noninterference as it does not stop covert channels
- Noninterference is weaker than BLP in that it allows a low user to copy one high-level file to another high-level file
- In both cases, noninterference seems closer to intuition of security

Evaluation of The Non-Interference Policy

- The notion of noninterference is elegant and natural
  - focuses on policy objective, rather than mechanism, such as BLP
- The model is useful for some applications, but may be difficult to apply to real world systems
  - e.g., how to model a system that BLP intends to model, with files storing sensitive information?
- Mostly concerned with deterministic systems
- May be too restrictive

A Model of Information

David Sutherland

System Model

- A system is described by an abstract state machine (similar to the noninterference paper)
  - a set of states
  - a set of possible initial states
  - a set of state transformations
- A possible execution sequence consists of
  - an initial state
  - a sequence of transformations applied to the system

Information

- Consider each possible execution sequence as a possible world.
  - the system is one world
- An information function is one that maps each possible world to a value
- Given a set W of all possible worlds, knowing no information, the current world w could be any one in W. Knowing that f1(w)=x, then one knows only those in W such that f1()=x is possible.

Information Flow From f1 and f2

- Given a set W of possible worlds and two functions f1 and f2, we say that information flows from f1 to f2 if and only if there exists some possible world w and some value z in the range of f2 such that
  - $\forall w' \, (f1(w)=f1(w') \rightarrow f2(w')? \, z)$

Proposition

- Proposition: Given W, f1, f2, information does not flow from f1 to f2 if and only if the function f1 ´ f2 is onto.
- Corollary: The information flow relation is symmetric
- Nondeducibility: a system is nondeducibility secure if information does from flow from high inputs to low outputs

Example: Stream Cipher

- Two high users & one low user
  - high user A generates a message
  - high user B generates a random string at a constant rate
  - the XOR of them (if A generates nothing, then 0 is used) is send to the low user
- This is nondeducibility secure
- This is NOT noninterference secure

Another Example

- A high user and a low user
  - the high user can write to a file
    - one letter at a time
  - the low user can try to read the n'th character in a file
    - if file is shorter than n, or if the the n'th character is blank, returns a random letter
    - otherwise, return the letter
- The system is nondeducible secure

Relationships Between Nondeducibility & Noninterference

- For deterministic systems with just one high user and one low user, a system is noninterference secure if and only if it is nondeducibility secure.
  - nondeducibility implies noninterference: no high input is also a possible world
  - noninterference implies nondeducbility: every possible world is equivalent to the one with no high-level input

Limitations of Nondeducibility & Noninterference

- Nondeducability may be too weak
  - Allows probabilistic reasoning
  - The stream cipher example is still nondeducibility secure even if high level user B generates 0 each time with 99% probability
- Noninterference may be too strong
  - as demonstrated by the stream cipher example

End of Lecture 5

- Next lecture
  - Denning’s work on information flow
  - The confinement problem
  - Covert channel