SecureSystem

1. When you parse an instruction from input file, create an InstructionObject( with fields representing the instruction type(READ, WRITE, BAD), subject name, object name, value if any).
2. Pass instructionObject to ReferenceMonitor.
3. Create two new subjects: Lyle of security level LOW, and Hal of security level HIGH. Store these subjects into the state and inform the reference monitor about them.
4. Create two new objects: LObj of security level LOW, and HObj of security level HIGH. Store these in the ObjectManager, telling the ReferenceMonitor about their levels. The initial value of each should be 0.
5. Read successive instructions from the input file and execute them, following the Bell and LaPadula constraints on reading and writing. You should have methods executeRead and executeWrite within your ReferenceMonitor class that check access requests and perform the appropriate update (if any) on the state, following the instruction semantics outlined above.
6. After each instruction, call printState to display the state change, if any, from the instruction execution.

InstructionObject Class

1. InstructionObject( with fields representing the instruction type(READ, WRITE, BAD), subject name, object name, value if any).

ReferenceMonitor Class

1. Labels are maintained. Can’t be changed (Strong Tranq.)
2. Maintaining two mappings from subject/obj. names to security labels.
3. When a subject request an action (R/W) the action is submitted to the reference monitor. Which decides to preformor or not based on BLP props. (simple security, \* property).
4. Reference ObjectManager to perform appropriate action. Otherwise no object accessed.
5. Always return a int val to the subject val of obj read if legal, or 0.

ObjectManager

1. Preform simple accesses to Objects
2. Store the Objects
3. Does not know about labels or security.
4. Local to ReferenceMonitor

SecurityLevel

1. defines “dominates” relation
2. assume levels are linearly ordered
3. define 2 constants levels HIGH, LOW, such that HIGH dominates LOW