The Clark Wilson Security Model

George W. Dinolt
CS4605
Background

- Based on Commercial Policies
- Importance is more on Integrity of computations
- Support prevention of and disclosure of fraud
- Support prevention of errors in calculations/data entry/data reporting.
- Claim: Need additional mechanisms to support Integrity Policies
Commercial Policies Concepts

- Fundamental idea is the Well Formed Transaction
- Users/Programs only manipulate data in specified ways that preserve integrity of the data
- Separation of Duties, people creating procedures are not allowed to execute them on live data
Differences with DoD Model

- Data is associated with the set of programs that can be used to manipulate it (not a security level)
- Access decisions are based on the fact that Users are given access to particular programs that manipulate particular data items
- Users are grouped by the duties (programs) they are to perform
Mandatory Commercial Policy

- Users Cannot change the programs that they can execute
- Users Cannot change the data associated with particular programs
- System/Application administrators responsible for assigning
Commercial Policy Properties

- Identify and Authenticate Users
- Ensure that specific data items can only be manipulated by a specific set of programs
- The programs meet the “well formed transaction” rules
- Maintain a log that contains program, user name, data files accessed
- Integrity properties are always enforced
- Protection Mechanisms cannot be changed
**Integrity Model Terms**

**CDI**: Set of **Constrained Data Items**, the elements that are to be protected.

**UDI**: The set of **Unconstrained Data Items**.

**IVP**: Set of **Integrity Verification Procedures**, functions that determine whether a particular data collection of CDI’s satisfy a particular integrity constraint.
Terms Continued

$\mathcal{T}P$: Set of Transform Procedures, each transform procedure is a function from a set of $CDI$'s to a set of $CDI$'s. The goal is that if the original set of $CDI$'s satisfy the appropriate $IVP$ then the transformed $CDI$'s will also. $\mathcal{T}P$s must be treated as atomic transactions.

$UserID$: The names of the set of users that can use the system
Certification Properties

**C1:** 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, provided the initial state was valid. For each TP and each set of CDIs that it may manipulate, the security officer must specify a “relation”, which defines that execution. A relation is thus of the form:

$$(TP_i, (CDI_{a}, CDI_{b}, CDI_{c}, \ldots))$$

where the list of CDIs defines a particular set of arguments for which the TP has been certified.
Enforcement Properties

**E1:** The system must maintain the list of relations specified in rule **C2**, and must ensure that the only manipulation of any CDI is by a TP, where the TP is operating on the CDI as specified in some relation.

**E2:** The system must maintain a list of relations of the form

\[(UserID, TP_i, (CDI_a, CDI_b, CDI_c, \ldots))\]

which list the data objects that TP may reference on behalf of that user. It must ensure that only executions described in one of the relations are performed.

**E3:** The system must authenticate the identity of each user attempting to execute a TP
C3: The list of relations in E2 must be certified to meet the separation of duty requirements

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 an input value must be certified to perform only valid transformations or else no transformations, for any possible value of the UDI. The transformation should take the input from the UDI to a CDI or the UDI is rejected. Typically this is an edit program.
Mandatory Policy

E4: Only the agent permitted to certify entities may change the list of such entities associated with other entities: specifically, those associated with a $\mathcal{T}P$. An agent that can certify an entity may not have any execute rights with respect to that entity.
There are sets role, subject and tran and functions:

\[ AR : subject \rightarrow role \ \{ \text{the active role of subjects} \} \]

\[ RA : subject \rightarrow 2^{role} \ \{ \text{the authorized roles of subjects} \} \]

\[ TA : role \rightarrow 2^{tran} \ \{ \text{transactions authorized for a role} \} \]

\[ exec : subject \times tran \rightarrow bool \ \{ \text{true if subject can execute transaction} \} \]
RBAC Rules

- Role Assignment:
  \[ \forall s: subject, t: tran : (\text{exec}(s, t) \Rightarrow RA(s) \neq \phi) \]

- Role Authorization:
  \[ \forall s: subject : AR(s) \in RA(s) \]

- Transaction authorization:
  \[ \forall s: subject, t: tran : (\text{exec}(s, t) \Rightarrow t \in TA(RA(s))) \]

- Object access: there are additional sets object, and modes
  \[ access : role \times tran \times object \times mode \rightarrow bool \]
  \[ \forall s: subject, t: tran : o: object : (\text{exec}(s, t) \Rightarrow access(AR(s), t, o, x)) \]
RBAC / CW Comparison

- RBAC has *subject*, CW has *UserID*
- RBAC has *tran*, CW has *TP*
- RBAC has *transaction authorization*, CW assigns users to *TP*
- RBAC has *transaction authorization* and *Object access*, CW has *TP* bound to *CDI*.