Some useful Information

Mapping of Turing machine to protection system

- All Tape Symbols, States $\Rightarrow$ rights
- Tape cell $\Rightarrow$ subject
- Cell s_i has A $\Rightarrow$ s_i has ***A*** rights on itself
- Cell s_k $\Rightarrow$ s_k has ***end*** rights on itself
- State p, head at s_i $\Rightarrow$ s_i has ***p*** rights on itself
- Distinguished right ***own***: s_i ***owns*** s_{i+1} for $1 \le i < k$

Bell-Lapadula Rules

Let $L(S) = l_s$ be the security clearance of subject S, and let $L(O) = l_o$ be the security classification of object O. For all security classifications l_i, $i = 0, ..., k - 1$, $l_i < l_{i+1}$.

Simple Security Condition, Preliminary Version: S can read O if and only if $l_o \le l_s$ and S has discretionary read access to O.

***-Property** (Star Property), Preliminary Version: S can write O if and only if $l_s \le l_o$ and S has discretionary write access to O.

Biba Rules

Biba's Model: Strict Integrity Policy (dual of Bell-LaPadula)

- s can read $o \leftrightarrow i(s) \le i(o)$ (no read-down)
- s can write $o \leftrightarrow i(o) \le i(s)$ (no write-up)
- s_1 can execute $s_2 \leftrightarrow i(s_2) \le i(s_1)$

Low-Water-Mark Policy

- s can **write** $o \leftrightarrow i(o) \le i(s)$ (prevents writing to higher level)
- s **reads** $o \rightarrow i'(s) = min(i(s), i(o))$ (drops subject's level)
- s_1 can **execute** $s_2 \leftrightarrow i(s_2) \le i(s_1)$ (prevents executing higher level objects)

Chinese Wall Rules

CW-Simple Security Condition: S can read O if and only if any of the following holds.

- There is an object O' such that S has accessed O' and CD(O') = CD(O).
- For all objects O', O' $\in$ PR(S) $\Rightarrow$ COI(O') $\neq$ COI(O).
- O is a sanitized object.

(O' $\in$ PR(s) indicates O' has been previously read by s)

CW-*-Property: A subject S may write to an object O if and only if both of the following conditions hold.

- The CW-simple security condition permits S to read O.
- For all unsanitized objects O', S can read O' $\Rightarrow$ CD(O') = CD(O).

Clark-Wilson Certification and Enforcement Rules

Certification rule 1 (CR1): When any IVP is run, it must ensure that all CDIs are in a valid state.

Certification rule 2 (CR2): For some associated set of CDIs, a TP must transform those CDIs in a valid state into a (possibly different) valid state.

Enforcement rule 1 (ER1): The system must maintain the certified relations, and must ensure that only TPs certified to run on a CDI manipulate that CDI.

Enforcement rule 2 (ER2): The system must associate a user with each TP and set of CDIs. The TP may access those CDIs on behalf of the associated user. If the user is not associated with a particular TP and CDI, then the TP cannot access that CDI on behalf of that user.

Certification rule 3 (CR3): The allowed relations must meet the requirements imposed by the principle of separation of duty.

Enforcement rule 3 (ER3): The system must authenticate each user attempting to execute a TP.

Certification rule 4 (CR4): All TPs must append enough information to reconstruct the operation to an append-only CDI.

Certification rule 5 (CR5): Any TP that takes as input a UDI may 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.

Enforcement rule 4 (ER4): Only the certifier of a TP may change the list of entities associated with that TP. No certifier of a TP, or of an entity associated with that TP, may ever have execute permission with respect to that entity.

Lipner's Requiements

1. Users will not write their own programs, but will use existing production programs and databases.
2. Programmers will develop and test programs on a non-production system; if they need access to actual data, they will be given production data via a special process, but will use it on their development system.
3. A special process must be followed to install a program from the development system onto the production system.
4. The special process in requirement 3 must be controlled and audited.
5. The managers and auditors must have access to both the system state and the system logs that are generated.

Core RBAC

Permissions = $2^{\text{Operations x Objects}}$
UA $\subseteq$ Users x Roles
PA $\subseteq$ Permissions x Roles
assigned_users: Roles $\rightarrow$ 2Users
assigned_permissions: Roles $\rightarrow$ 2Permissions
Op(p): set of operations associated with permission p
Ob(p): set of objects associated with permission p
user_sessions: Users $\rightarrow 2^{\text{Sessions}}$
session_user: Sessions $\rightarrow$ Users
session_roles: Sessions $\rightarrow 2^{\text{Roles}}$
$\quad$ *session_roles*(s) = $\{r \mid (\text{session\_user}(s), r) \in \text{UA}\}$
avail_session_perms: Sessions $\rightarrow 2^{\text{Permissions}}$

RBAC with general Role hierarchy

authorized_users: Roles$\rightarrow 2^{\text{Users}}$

- *authorized_users*(r) = $\{u \mid r' \geq r \;\&(r', u) \in UA\}$

authorized_permissions: Roles$\rightarrow$ 2Permissions

- *authorized_permissions*(r) = $\{p \mid r \geq r' \;\&(p, r') \in PA\}$

RH $\subseteq$ Roles x Roles is a partial order, called the inheritance relation & written as $\geq$.
$(r_1 \geq r_2) \rightarrow$ *authorized_users*(r_1) $\subseteq$ *authorized_users*(r_2) &
authorized_permisssions(r_2) $\subseteq$ *authorized_permisssions*(r_1)

Static SoD

SSD $\subseteq 2^{\text{Roles}}$ x N

In absence of hierarchy

Collection of pairs (RS, n) where RS is a role set, $n \geq 2$;
for all $(RS, n) \in SSD$, *for all* $t \in RS$: $|t| \geq n \rightarrow \cap_{r \in t}$ *assigned_users*(r)= $\varnothing$

In presence of hierarchy

Collection of pairs (RS, n) where RS is a role set, n $\geq$ 2;
for all $(RS, n) \in SSD$, *for all* $t \in RS$: $|t| \geq n \rightarrow \cap_{r \in t}$ *authorized_uers*(r)= $\varnothing$