Some useful Information

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.

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 $\rightarrow 2^{\text{Roles}}$

$session\_roles(s) = \{r \mid (\text{session\_user}(s), r) \in \text{UA}\}$

avail_session_perms: Sessions $\rightarrow 2^{\text{Permissions}}$

RBAC with general Role hierarchy

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

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

(Note that for any role $r \geq r$ – so all role assigned *to r* are also authorized t*o r*)

authorized_permissions: Roles$\rightarrow$ 2Permissions

- $authorized\_permissions(\text{r}) = \{p \mid r \geq r' \;\&(p, r') \in PA\}$

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, $n \geq 2$;

$for\ all\ (RS, n) \in SSD, for\ all\ t \in RS: |t| \geq n \rightarrow \cap_{r \in t}\ assigned\_users(\text{r}) = \varnothing$

In presence of hierarchy

Collection of pairs (RS, n) where RS is a role set, n $\geq$ 2;

$for\ all\ (RS, n) \in SSD, for\ all\ t \in RS: |t| \geq n \rightarrow \cap_{r \in t}\ authorized\_uers(r) = \varnothing$