Role based Access Control

Take Grant Model
**Objective**

- Define/understand/represent formally
  - Take grant model
  - Role-based Access Control model
- Analyze/deduce (in TG or RBAC models)
  - stealing of permissions
  - Conspiracy
  - Static/Dynamic separation of duty
- Understand key issue related to secure interoperation
Access control in organizations is based on “roles that individual users take on as part of the organization”

- Access depends on function, not identity
- Example:

  Allison is bookkeeper for Math Dept. She has access to financial records. If she leaves and Betty is hired as the new bookkeeper, Betty now has access to those records. The role of “bookkeeper” dictates access, not the identity of the individual.
RBAC

![Diagram showing RBAC concept]

- Users: $u_1, u_2, \ldots, u_n$
- Permissions: $o_1, o_2, \ldots, o_m$

**Total number Of assignments Possible?**

A role-based access control (RBAC) model is presented. The diagram illustrates the relationships between users, roles, and permissions, showing how users can be assigned to roles, and roles can be assigned permissions. The question of whether the total number of assignments is possible is addressed.
RBAC (NIST Standard)

What model entity would relate to the traditional notion of subject?

Total number of subjects possible?

Role vs Group?
Core RBAC (relations)

- Permissions $= 2^{\text{Operations} \times \text{Objects}}$
- UA $\subseteq$ Users $\times$ Roles
- PA $\subseteq$ Permissions $\times$ Roles
  - \textit{assigned\_users}: Roles $\rightarrow 2^{\text{Users}}$
  - \textit{assigned\_permissions}: Roles $\rightarrow 2^{\text{Permissions}}$
- \textit{Op}(p): set of operations associated with permission p
- \textit{Ob}(p): set of objects associated with permission p
- \textit{user\_sessions}: Users $\rightarrow 2^{\text{Sessions}}$
- \textit{session\_user}: Sessions $\rightarrow$ Users
- \textit{session\_roles}: Sessions $\rightarrow 2^{\text{Roles}}$
  \textit{session\_roles}(s) = \{ r | (\text{session\_user}(s), r) \in \text{UA}) \}
- \textit{avail\_session\_perms}: Sessions $\rightarrow 2^{\text{Permissions}}$
RBAC with Role Hierarchy

Roles

Sessions

Operations

Objects

Permissions

User Sessions (one-to-many)

Role Sessions (many-to-many)

RH (role hierarchy)

UA

PA
RBAC with General Role Hierarchy

- **authorized_users**: Roles → \( 2^{\text{Users}} \)
  \[ \text{authorized_users}(r) = \{u \mid r' \geq r \& (r', u) \in UA\} \]

- **authorized_permissions**: Roles → \( 2^{\text{Permissions}} \)
  \[ \text{authorized_permissions}(r) = \{p \mid r \geq r' \& (p, r') \in PA\} \]

- RH ⊆ Roles x Roles is a partial order
  - called the inheritance relation
  - written as ≥.
  \[ (r_1 \geq r_2) \rightarrow \text{authorized_users}(r_1) \subseteq \text{authorized_users}(r_2) \& \text{authorized_permissions}(r_2) \subseteq \text{authorized_permissions}(r_1) \]

*What do these mean?*
Example

authorized_users(Employee)?
authorized_users(Administrator)?
authorized_permissions(Employee)?
authorized_permissions(Administrator)?
Constrained RBAC

- **Static Separation of Duty**
- **Dynamic Separation of Duty**
- **Role Hierarchy (RH)**
- **User Sessions (one-to-many)**
- **Permissions**

Diagram:
- **Users** connected to **Roles** with **UA** and **PA**
- **Roles** connected to **Operations** and **Objects**
- **User Sessions (one-to-many)** connected to **Roles**
- **Dynamic Separation of Duty**
- **Static Separation of Duty (role hierarchy)**
Static Separation of Duty

- \( SSD \subseteq 2^{\text{Roles}} \times N \)
- In absence of hierarchy
  - Collection of pairs \((RS, n)\) where \(RS\) is a role set, \(n \geq 2\)
    - \(\text{for all } (RS, n) \in SSD, \text{ for all } t \subseteq RS: |t| \geq n \rightarrow \bigcap_{r \in t} \text{assigned_users}(r) = \emptyset\)
- In presence of hierarchy
  - Collection of pairs \((RS, n)\) where \(RS\) is a role set, \(n \geq 2\);
    - \(\text{for all } (RS, n) \in SSD, \text{ for all } t \subseteq RS: |t| \geq n \rightarrow \bigcap_{r \in t} \text{authorized_users}(r) = \emptyset\)
Dynamic Separation of Duty

- \( DSD \subseteq 2^{\text{Roles}} \times \mathbb{N} \)
  - Collection of pairs \((RS, n)\) where \(RS\) is a role set, \(n \geq 2\);
    - A user cannot activate \(n\) or more roles from \(RS\)
  - What is the difference between SSD or DSD containing:
    \((RS, n)\)?

- Consider \((RS, n) = (\{r_1, r_2, r_3\}, 2)\)?
- If SSD – can \(r_1, r_2\) and \(r_3\) be assigned to \(u\)?
- If DSD – can \(r_1, r_2\) and \(r_3\) be assigned to \(u\)?
Can we represent BLP using RBAC?
Advantages of RBAC

- Allows Efficient Security Management
  - Administrative roles, Role hierarchy
- Principle of least privilege allows minimizing damage
- Separation of Duty constraints to prevent fraud
- Allows grouping of objects / users
- Policy-neutral - Provides generality
- Encompasses DAC and MAC policies
RBAC’s Benefits

<table>
<thead>
<tr>
<th>TASK</th>
<th>RBAC</th>
<th>NON-RBAC</th>
<th>DIFFERENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assign existing privileges to new users</td>
<td>6.14</td>
<td>11.39</td>
<td>5.25</td>
</tr>
<tr>
<td>Change existing users’ privileges</td>
<td>9.29</td>
<td>10.24</td>
<td>0.95</td>
</tr>
<tr>
<td>Establish new privileges for existing users</td>
<td>8.86</td>
<td>9.26</td>
<td>0.40</td>
</tr>
<tr>
<td>Termination of privileges</td>
<td>0.81</td>
<td>1.32</td>
<td>0.51</td>
</tr>
</tbody>
</table>
Cost Benefits

- Saves about 7.01 minutes per employee, per year in administrative functions
  - Average IT admin salary - $59.27 per hour
  - The annual cost saving is:
    - $6,924/1000;
    - $692,471/100,000

How do we get this?
Take Grant Model
Take-Grant Protection Model

- System is represented as a directed graph
  - Subject: ●
  - Object: ○ Either: ◯
  - Labeled edge indicates the rights that the source object has on the destination object

- Four graph rewriting rules ("de jure", "by law", "by rights")
  - The graph changes as the protection state changes according to these rules
Take rule

- if $t \in y$, the take rule produces another graph with a transitive edge $\alpha \subseteq \beta$ added.

$x$ takes ($\alpha$ to $y$) from $z$
Grant Rule

- if \( g \in \gamma \), the grant rule produces another graph with a transitive edge \( \alpha \subseteq \beta \) added.

\[
\begin{align*}
x & \quad y \quad z \quad y \\
\text{z grants (\( \alpha \) to \( y \)) to} \quad x
\end{align*}
\]

\[
\begin{align*}
x & \quad y \quad z \quad y \\
\end{align*}
\]
Create and Remove

3. Create rule:

\[ x \xrightarrow{\alpha} y \]

\( x \) creates (\( \alpha \) to new vertex) \( y \)

4. Remove rule:

\[ x \xrightarrow{\beta} y \xrightarrow{\beta - \alpha} y \]

\( x \) removes (\( \alpha \) to) \( y \)
Exercise

- Write a function using HRU operations that implement the
  - Take rule: call it `TG_Take(??)`
  - Grant rule: call it `TG_Grant(??)`
Take-Grant Protection Model: Sharing

- Given $G_0$, can vertex $x$ obtain $\alpha$ rights over $y$?
  - $\text{Can\_share}(\alpha, x, y, G_0)$ is true iff
    - $G_0 \vdash^* G_n$ using the four rules, &
    - There is an $\alpha$ edge from $x$ to $y$ in $G_n$

- $tg$-$path$: $v_0, \ldots, v_n$ with $t$ or $g$ edge between any pair of vertices $v_i, v_{i+1}$
  - Vertices $tg$-$connected$ if $tg$-$path$ between them

- Theorem: Any two subjects with $tg$-$path$ of length 1 can share rights
Any two subjects with *tg-path* of length 1 can share rights

\[
\text{Can}_{\text{share}}(\alpha, x, y, G_0)
\]

- Four possible length 1 *tg*-paths
  - 1. Take rule
  - 2. Grant rule
  - 3. Lemma 3.1?
  - 4. Lemma 3.2?
Any two subjects with \textit{tg-path} of length 1 can share rights

\textbf{Lemma 3.1}

\textbf{Sequence:}
- Create
- Take
- Grant
- Take

Now prove lemma 3.2!
Other definitions

- **Island**: Maximal $tg$-connected subject-only subgraph
  - Can_share all rights in island
  - Proof: Induction from previous theorem

- **Bridge**: $tg$-path between subjects $v_0$ and $v_n$ with edges of the following form:
  - $t \rightarrow *, t \leftarrow *
  - $t \rightarrow * g \rightarrow t \leftarrow *
  - $t \rightarrow *, g \leftarrow, t \leftarrow *
Bridge

By lemma 3.1

By grant

By take
Theorem: Can_share(α, x, y, G₀) (for subjects)

- Subject_can_share(α, x, y, G₀) is true iff if x and y are subjects and
  - there is an α edge from x to y in G₀
  OR if:
  - ∃ a subject s ∈ G₀ with an s-to-y α edge, and
  - ∃ islands I₁, …, Iₙ such that x ∈ I₁, s ∈ Iₙ, and there is a bridge from Iₗ to Iₗ₊₁

![Diagram showing the relationship between subjects and islands with edges and bridges]
What about objects?

Initial, terminal spans

- $x$ *initially spans* to $y$ if $x$ is a subject and there is a $tg$-path between them with $t$ edges ending in a $g$ edge (i.e., $t\rightarrow^*g\rightarrow$)
  - $x$ can grant a right to $y$

- $x$ *terminally spans* to $y$ if $x$ is a subject and there is a $tg$-path between them with $t$ edges (i.e., $t\rightarrow^*$)
  - $x$ can take a right from $y$
Theorem: Can_share(\(\alpha, x, y, G_0\))

Can_share(\(\alpha, x, y, G_0\)) iff there is an \(\alpha\) edge from \(x\) to \(y\) in \(G_0\) or if:

- \(\exists\) a vertex \(s \in G_0\) with an \(s\) to \(y\) \(\alpha\) edge,
- \(\exists\) a subject \(x'\) such that \(x' = x\) or \(x'\) initially spans to \(x\),
- \(\exists\) a subject \(s'\) such that \(s' = s\) or \(s'\) terminally spans to \(s\), and
- \(\exists\) islands \(I_1, \ldots, I_n\) such that \(x' \in I_1\), \(s' \in I_n\) and there is a bridge from \(I_j\) to \(I_{j+1}\)

\(x'\) can grant a right to \(x\)

\(s'\) can take a right from \(s\)
Theorem: Can_share(α,x,y,G₀)

- Corollary: There is an $O(|V| + |E|)$ algorithm to test can_share:
  - Decidable in linear time!!

- Protection state of the rules evolves
  - Following application on rules
  - Thus can characterize what set of states can be generated
One example protection problem

- Sharing through a Trusted Entity
  - Let $p$ and $q$ be two processes
  - Let $b$ be a buffer that they share to communicate
  - Let $s$ be a third party (e.g. operating system) that controls $b$

Witness
- S creates $\{r, w\}$ to new object $b$
- S grants $\{r, w\}$, $b$ to $p$
- S grants $\{r, w\}$, $b$ to $q$
Theft in Take-Grant Model

- $\text{Can\_steal}(\alpha, x, y, G_0)$ is true if there is no $\alpha$ edge from $x$ to $y$ in $G_0$ and $\exists$ sequence $G_1, \ldots, G_n$ s. t.:
  - $\exists$ $\alpha$ edge from $x$ to $y$ in $G_n$,
  - $\exists$ rules $\rho_1, \ldots, \rho_n$ that take $G_{i-1} \vdash \rho_i G_i$, and
  - $\forall v, w \in G_i, 1 \leq i \leq n$, if $\exists$ $\alpha$ edge from $v$ to $y$ in $G_0$ then $\rho_i$ is not “$v$ grants ($\alpha$ to $y$) to $w$”

- Disallows owners of $\alpha$ rights to $y$ from transferring those rights
- Does not disallow them to transfer other rights
- Trojan horse??

- Disallows owners of $\alpha$ rights to $y$ from transferring those rights
- Does not disallow them to transfer other rights
- Trojan horse??

- Disallows owners of $\alpha$ rights to $y$ from transferring those rights
- Does not disallow them to transfer other rights
- Trojan horse??
A witness to theft

- Can u receive (α to w)?
  - u cannot grant (α to w) to anybody
Conspiracy

- Theft indicates cooperation: which subjects are actors in a transfer of rights, and which are not?
- Next question is
  - How many subjects are needed to enable \( \text{Can\_share}(\alpha,x,y,G_0) \)?
- Note that a vertex \( x \)
  - Can pass rights to any vertex to which it initially spans
    - \( (t, *g_{\_}) \)
  - Can take rights from any vertex to which it terminally spans
    - \( (t, *) \)
Conspiracy

- Access set $A(y)$ with focus $y$ ($y$ is subject) is union of
  - set of vertices $y$,
  - vertices to which $y$ initially spans, and
  - vertices to which $y$ terminally spans
- Deletion set $\delta(y,y')$: All $z \in A(y) \cap A(y')$ for which
  - $y$ initially spans to $z$ and $y'$ terminally spans to $z$
  - $y$ terminally spans to $z$ and $y'$ initially spans to $z$
  - $z=y$ & $z=y'$
Conspiracy

- Conspiracy graph $H$ of $G_0$:
  - Represents the paths along which subjects can transfer rights
  - For each subject in $G_0$, there is a corresponding vertex $h(x)$ in $H$
  - if $\delta(y,y')$ not empty, edge from $h(y)$ to $h(y')$
Example: draw the conspiracy graph

How many minimum conspirators involved in \text{Can\_share}(\alpha, x, y, G_0)?
Policy Composition
Problem: *Consistent* Policies

- Policies defined by different organizations
  - Different needs
  - But sometimes subjects/objects overlap
- Can all policies be met?
  - Different categories
    - Build lattice combining them
  - Different security levels
    - Need to be *levels* – thus must be able to order
- What if different DAC and MAC policies need to be integrated?
Secure Interoperability

- Principles of secure interoperation [Gong, 96]
  
  **Principle of autonomy**
  - If an access is permitted within an individual system, it must also be permitted under secure interoperation

  **Principle of security**
  - If an access is not permitted within an individual system, it must not be permitted under secure interoperation

- Interoperation of secure systems can create new security breaches
Secure Interoperability (Example)

\[ F_{12} = \{a, b\} \]

\( F_{12} \) - permitted access between systems 1 and 2

(1) \( F_{12} = \{a, b, d\} \)
Direct access

(2) \( F_{12} = \{c\} \)
Indirect access
Summary

- RBAC is a promising approach
  - Lot of efforts currently expended for this
- Take Grant
  - Restricted model – easy to analyze
    - but usefulness?
- Secure interoperation
  - Growing problem