RBAC-based Secure Interoperation using Constraint Logic Programming

Jinwei Hu, Ruixuan Li, Zhengding Lu

Huazhong University of Science and Technology
Wuhan, China
Outline

- Background
- Building secure Interoperability
- Formalization
- Conflict Resolution
- Conclusion and Future work
Secure Interoperation

Secure interoperability

Security and availability

Resources

policies

Sharing resources

Secure interoperability

Security and availability
Challenges and Solutions

- Challenges
  - Heterogeneity in access control policies
  - Formal analysis

- Solutions
  - RBAC (Role-Based Access Control)
  - CLP (Constraint Logic Programming)
Why Role-Based Access Control (RBAC)?

- being policy-neutral and being able to support a wide range of access control requirements
- widely supported in commodity operating systems and database systems
- deployed in many organizations to model and enforce their access control needs
Why Constraint Logic Programming (CLP)?

- CLP gives rise to a concise, understandable, and easy-to-maintain specification of secure interoperation.
- Expressive enough to capture various requirements of interoperation.
Why CLP? (cont.)

- reasonable performance when evaluating access request against interoperation specification
- special algorithm for efficient evaluation in presence of real-time changes
Our contributions

- Two types of interoperation mechanisms for RBAC-based collaborating environments, formalizing them in CLP
- Rules to detect and resolve conflicts induced by interoperation
Related work

- Manually select role mappings
  - Security administrators manually select intended role mappings
  - Easy to implement
  - Error prone and heavy burden on administrators
Related work (contd.)

- Automatic generation of role mappings
  - Role mappings are generated according to requests and sharing policies
    - less overhead
    - no formal analysis
Outline

- Background
- **Building secure Interoperability**
- Formalization
- Conflict Resolution
- Conclusion and Future work
**Building secure interoperability**

- The domain $\alpha$ issues an interoperation request.
- The domain $\beta$ generates rules for secure interoperability.
- The domain $\beta$ detects and resolves induced conflicts.
The components of secure interoperability

- Permission-based interoperability (PI)
- Role-based interoperability (RI)
- Conflict Resolution
Pl vs. Ri

PI vs. RI

RIPI vs. RI

PI

RI
## PI vs. RI (cont.)

<table>
<thead>
<tr>
<th></th>
<th>PI</th>
<th>RI</th>
</tr>
</thead>
<tbody>
<tr>
<td>level</td>
<td>Permission level</td>
<td>Role level</td>
</tr>
<tr>
<td>Interoperation mechanism</td>
<td>assignment of permissions to foreign roles</td>
<td>mapping foreign roles to local ones</td>
</tr>
<tr>
<td>potential conflicts</td>
<td>permission-based SoD</td>
<td>User-based SoD, role-based SoD, permission-based SoD and cyclic inheritance</td>
</tr>
<tr>
<td>Advantages</td>
<td>fewer induced conflicts and fine-grained control of interoperation</td>
<td>convenient interoperation management</td>
</tr>
<tr>
<td>Shortcomings</td>
<td>less convenient interoperation specification</td>
<td>more variations of conflicts to cover</td>
</tr>
</tbody>
</table>
Outline

- Background
- Building secure Interoperability
- Formalization
- Conflict Resolution
- Conclusion and Future work
Formalization: RBAC

- A n-ary predicate $pred$ with the following intended interpretation represents a relation $R$.

$$
\Pi^* \models_3 \text{pred}(a_1, \ldots, a_n) \text{ iff } (a_1, \ldots, a_n) \in R
$$

$$
\Pi^* \models_3 \text{senior}(r_i, r_j) \text{ if and only if } (r_i, r_j) \in SENIOR
$$

- Authorizations in RBAC models are defined as a relation $\text{AUTHORIZED}$

$$
\text{authorized}(u, p) \leftarrow \text{ura}(u, r_1), \text{active}(u, r_1),
\text{pra}(r_2, p), \text{senior}(r_1, r_2).
$$
Formalization: PI

- Given a permission-based request, a request instance is created.

- A request instance could be in initial state, received state, or effective state.

- A request instance in the received state and the effective state is represented by predicates $p_{rcvd}$ and $p_{eftv}$, respectively.
Formalization: PI (cont.)

- An initial instances turns into received only if the demanded permission is shareable.
  \[ p \text{ rcvd}(\alpha, r_\alpha, \beta, p, m) \leftarrow p\_req(\alpha, r_\alpha, \beta, p, m), \text{shareable}(\beta, \alpha, p). \]

- A received instances turns effective if its effect does not violate any constraints in \( C \) of collaborating domains
  \[ p\_eftv(\alpha, r_\alpha, \beta, p) \leftarrow p\_rcvd(\alpha, r_\alpha, \beta, p, _), \text{consistent}(C). \]
The authorization rule is extended according to the specifications of effective instances:

\[
\text{authorized}(u, p) \leftarrow p\_eftv(\alpha, r_\alpha, \beta, p), \text{ura}(u, r_1), \\
\hspace{1cm} \text{active}(u, r_1), \text{senior}(r_1, r_\alpha).
\]
Formalization: RI

- The literals and rules introduced in this section are based on CLP on finite sets.

Table I

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Constraint</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>∈</td>
<td>membership constraint</td>
<td>$a \in A$</td>
</tr>
<tr>
<td>⊆</td>
<td>subset constraint</td>
<td>$A \subseteq B$</td>
</tr>
<tr>
<td>=</td>
<td>equality constraint</td>
<td>$A = B$</td>
</tr>
<tr>
<td>∉</td>
<td>the negative of ⊆</td>
<td>$A \not\subseteq B$</td>
</tr>
</tbody>
</table>
Each role-based request has four states: initial state, received state, mapped state and effective state. The four states are represented by predicates $r_{\text{req}}$, $r_{\text{rcvd}}$, $r_{\text{mapped}}$ and $r_{\text{eftv}}$, respectively.

When a role-based request arrives, a corresponding predicate, it is defined:

$$r_{\text{rcvd}}(\alpha, r_\alpha, PS_{\text{RIR}}, \beta, M) \leftarrow$$

$$PS_{\text{RIR}} = \{ p : r_{\text{req}}(\alpha, r_\alpha, PS, \beta, M),$$

$$p \in PS, \text{shareable}(\beta, \alpha, p) \}.$$
Formalization: RI (cont.)

- We consider as candidate for mapping a role whose permission set is a subset of the requested permission set whereas there is no senior role that satisfies this condition.

\[ \text{improperRole}(R_i, PS_{RIR}) \leftarrow \]
\[ \text{senior}(R_l, R_i), R_l \neq R_i, PermSet(R_l) \subseteq PS_{RIR}. \]

\[ \text{r_mapped}(\alpha, r_\alpha, \beta, R_i, M) \leftarrow \text{r_rcvd}(\alpha, r_\alpha, PS_{RIR}, \beta, M), \]
\[ PermSet(R_i) \subseteq PS_{RIR}, \]
\[ \text{not improperRole}(R_i, PS_{RIR}). \]
After checking the constraint consistency, role mappings may become effective.

\[
\text{consistent}(\alpha, r_\alpha, \beta, r) \leftarrow \text{not } ci(\alpha, r_\alpha, \beta, r),
\]

\[
\text{not violated}_\text{drpc}(\alpha, r_\alpha, \beta, r),
\]

\[
\text{not violated}_\text{crpc}(\alpha, r_\alpha, \beta, r),
\]

\[
\text{not violated}_\text{cupc}(\alpha, r_\alpha, \beta, r).
\]

\[
r_{eftv}(\alpha, r_\alpha, \beta, r) \leftarrow
\]

\[
r_{mapped}(\alpha, r_\alpha, \beta, r, \_), \text{consistent}(\alpha, r_\alpha, \beta, r).
\]

\[
\text{authorized}(u, p) \leftarrow \text{ura}(u, r_i), \text{active}(u, r_i),
\]

\[
eftv\_\text{path}(r_i, r_j), \text{pra}(r_j, p).
\]
Formalization: RI (cont.)

• However, there is a possibility that part of requested permissions are not shared by role-mappings.

$$improperPerm(\alpha, r_\alpha, \beta, p) \leftarrow$$

$$r_{eftv}(\alpha, r_\alpha, \beta, Rl, _), p \in PermSet(Rl).$$

• To maximize interoperability, a rule which constructs a received permission-based interoperation request is defined below.

$$p_{rcvd}(\alpha, r_\alpha, \beta, p, M) \leftarrow$$

$$r_{rcvd}(\alpha, r_\alpha, PS\_RIR, \beta, M), p \in PermSet(Rj),$$

$$p \in PS\_RIR, not \ improperPerm(\alpha, r_\alpha, \beta, p),$$

$$PermSet(Rj) \not\subseteq PS\_RIR$$

$$PS\_RIR \not\subseteq PermSet(Rj).$$
Outline

- Background
- Building secure Interoperability
- Formalization
- Conflict Resolution
- Conclusion and Future work
Conflict Resolution

Requirements:

- preserves original accesses in an individual domain.
- all interoperation concerned, irrespective of their order in which they become available, are treated equally.
- revokes the least preferable one among all involved in conflicts
Various types of constraints

- Permission-based constraints
  - disjoint role-permission constraint (drpc)
  - conflicting user-permission constraint (cupc)

- Role-based constraints
  - Free of Cyclic Inheritance
  - Static role-based Separation-of-Duty constraint

- ect.
Rules for free of cyclic inheritance

- Take the rules to resolve cyclic inheritance for example.
Basic definitions

- A role connection

\[ \text{rolecon}(r_1, r_2, N) \leftarrow \text{senior}(r_1, r_3), \text{r_mapped}(\alpha, r_3, \beta, r_2, N). \]

- An effective role connection

\[ \text{eftv_rolecon}(r_1, r_2) \leftarrow \text{senior}(r_1, r_3), \text{r_eftv}(\alpha, r_3, \beta, r_2). \]
Basic definitions (cont.)

- **Path**

\[
\text{path}(r, r, \_ ) \leftarrow .
\]

\[
\text{path}(r, r', N) \leftarrow \text{rolecon}(r, r_i, M), \text{path}(r_i, r', N), M \geq N.
\]

- **Effective path**

\[
\text{eftv\_path}(r, r) \leftarrow .
\]

\[
\text{eftv\_path}(r, r') \leftarrow \text{eftv\_rolecon}(r, r_i), \text{eftv\_path}(r_i, r').
\]
Rules for Free of Cyclic Inheritance

- Cyclic inheritance is formed if and only if
  \[ \Pi^* \models \text{eftv}_\text{path}(r_2, r_i) \land \text{eftv}_\text{path}(r_j, r_1) \land \text{senior}(r_1, r_2) \land r_{eftv}(\alpha, r_i, \beta, r_j). \]

- The following rule specifies a general case of cyclic inheritance
  \[ ci(\alpha, r_\alpha, \beta, r) \leftarrow r_{mapped}(\alpha, r_\alpha, \beta, r, N), \]
  \[ \text{path}(r_2, r_\alpha, N), \text{path}(r, r_1, N), \text{senior}(r_1, r_2). \]
Outline

- Background
- Building secure Interoperability
- Formalization
- Conflict Resolution
- Conclusion and Future work
Conclusion

- We present a formalization of RBAC-based secure interoperation in CLP.
- Secure interoperation is enabled at both the role level and the permission level while keeping role hierarchies unchanged.
- A set of conflict resolution rules are in position to safeguard domains.
Future work

- We are in the process of performing various experiments on interoperation specification in CLP.
Thank you!

http://idc.hust.edu.cn