Reachability Analysis For Role Based Administration of User Attributes

Xin Jin, Ram Krishnan, Ravi Sandhu

Institute of Cyber Security (ICS)
University of Texas at San Antonio (UTSA)
San Antonio, Texas, USA

April 22, 2014
Content

Background and Motivation

Related Work

Contributions of Our Paper

Conclusion and Future Work
Background and Motivation

What is User Attributes

- Professor
- Computer Science
- Teaches (CS423)
- Office (2805)
- Manages (Sec Lab)
- Grad Committee
Related research with User Attributes

- Attribute based access control (ABAC): Jin et al (DBSEC 12), Wang et al (FSME 04), Hu et al (NIST draft model 2013), Chadwick et al (WETICE 06), XACML 3.0 (06), Pirretti et al (CCS 06), Li et al (Oakland 02)

- Attribute based encryption (ABE): Goyal et al (CCS 06), Bethencourt et al (Oakland 07), Ostrovsky et al (CCS 07), Rouselakis et al (CCS 13), Liu et al (CCS 13)

- Identity management: Chadwick et al (Computer 09)

- Usage control: UCON_{ABC} by Park et al (TISSEC 04)
In each organization, certain administrators have to **assign** user attributes values when the user is provisioned and **modify** user attributes values thereafter.

Attributes of the same user **constrain** each other. Administration rules are specified to regulate attribute modifications.

**Example Rule**

The `clearance` attribute of users can be assigned to “topsecret” if: “officer” ∈ role(u) ∧ clearance(u) == “secret” ∧ work-type(u) == “full-time”.
Motivation for Reachability Problem

Example Authorization Policy

\[ \text{read}(sub, obj) \rightarrow \neg (\text{clearance}(u) == \text{"topsecret"} \land \text{work-type}(u) == \text{"part-time"}) \]

Questions

Given a predefined administrative rules, will Alice ever be able to access \( obj \) in the future? It is equivalent to ask whether Alice’s attribute can reach conditions which satisfies the authorization policy.
Background and Motivation

Attributes Assignment Constraints

- **Rule 1**: assign clearance(u) to “topsecret” **IF**:
  
  “officer” ∈ role(u) ∧ clearance(u) == “secret” ∧ work-type(u) == “full-time”.

- **Rule 2**: assign work-type(u) to “part-time” **IF** “officer” ∈ role(u).

Transition by Rule 1

From rule 1, it seems that the user will never get access to obj.

Transition by Rule 2

“officer” ∈ role(Alice), clearance(Alice) == “topsecret”,
work-type(Alice) == “full-time”

→ “officer” ∈ role(Alice), clearance(Alice) == “topsecret”,
work-type(Alice) == “part-time”.
Given a **large set** of administration rules, it is hard to tell whether user attributes can reach certain values as expected.

Constraints (Crampton *et al* (SACMAT 03), Ahn *et al* (TISSEC), Bijon *et al* (PASSAT 13)) can be deployed on user attributes assignment. It prevent values to be assigned. Reachability is still important. Help understand what each assignment enables indirectly and also help design constraints.

**Reachability analysis** help solves this problem by determining whether user attributes can reach certain value based on given policies.
Related Work

- **The Harrison Ruzzo Ullman (HRU) model**: Safety problem regarding leakage of a specific right. Others are TAM, ATAM by Sandhu *et al* (Oakland 92).

- **Role Based Trust Management (RT)**: safety analysis on trust relationships: Li *et al* (Oakland 02, 03)

- **ARBAC97 Related**: Safety analysis on role administration rules: Stoller *et al* (CCS 07, ESORICS 10, CSFW 06, SACMAT 09), Alberti *et al* (ASIACCS 2011), Armando *et al* (DBSEC 2012), Li *et al* (SACMAT 04)

- **Others**: policy mis-configuration detection, model checking, policy analysis, etc.
Related Work

Limitations

▶ Analysis on only rules for one user attribute—role, and is for RBAC authorization policy, i.e., role represents permissions.
▶ There is connection between those work and reachability analysis for attributes. But it is not intuitive and has not been studied.
▶ Attribute reachability is beyond the safety analysis of role as defined in related work.
Our Contributions

- Formally define user attribute administration as state transition system.
- Define two kinds of reachability problems in the context of attribute administration Model.
- Provide formal proof for problem complexity. Most problems are in PSPACE-complete.
- Discover practical restrictions on policies and design polynomial time solvable algorithms.
Contributions of Our Paper

Attributes, State, State Transition and Rules

- assign(ar, Alice, clearance, classified)
  - clearance(Alice) = unclassified
    - role(Alice) = {employee}
  - clearance(Alice) = classified
    - role(Alice) = {employee}
- add(ar, Alice, role, manager)
  - clearance(Alice) = unclassified
    - role(Alice) = {employee, manager}
- assign(ar, Alice, clearance, topsecret)
  - clearance(Alice) = topsecret
    - role(Alice) = {employee, manager}
- delete(ar, Alice, role, manager)
  - clearance(Alice) = topsecret
    - role(Alice) = {employee}
Contributions of Our Paper

State Transition Rules

User attributes changes as guided by some models. We take a restricted version of the Generalized User-Role Assignment Model (GURA) (Jin et al WSRAS12) here. It is simple while the reachability problem is not obvious.

\[ can_{add} \subseteq AR \times C \times SCOPE_{sua} \]
\[ can_{delete} \subseteq AR \times C \times SCOPE_{sua} \]
\[ can_{assign} \subseteq AR \times C \times SCOPE_{aua} \]

\textit{sua}: a set-valued attribute, \textit{aua}: an atomic-valued attribute, \textit{AR}: administrative role, \textit{C}: preconditions on attributes of users.

\begin{itemize}
  \item \textbf{if} \( \langle hr, \text{clearance}(u) = \text{secret} \land \text{employee} \in \text{role}(u), \text{manager} \rangle \in \text{can}_{\text{add}}_{\text{role}} \)
  \item \textbf{then} \textit{add}(hr, Alice, role, manager) is allowed if clearance(Alice) == secret \land employee \in \text{role}(Alice).
\end{itemize}
Contributions of Our Paper

The \textit{rGURA}_0 Schemes

For preconditions in each \texttt{can\_assign}_{aua} relation:

\[ \varphi ::= \neg \varphi \mid \varphi \land \varphi \mid aua(u) = avalue \]
\[ avalue ::= aval_1 \mid aval_2 \mid \ldots \mid aval_n \]

where \text{SCOPE}_{aua} = \{aval_1, aval_2, \ldots, aval_n\}.

For preconditions in each \texttt{can\_add}_{sua} and \texttt{can\_delete}_{sua} relations:

\[ \varphi ::= \neg \varphi \mid \varphi \land \varphi \mid svalue \in sua(u) \]
\[ svalue ::= sval_1 \mid sval_2 \mid \ldots \mid sval_m \]

where \text{SCOPE}_{sua} = \{sval_1, sval_2, \ldots, sval_m\}.
Contributions of Our Paper

Example \( r\text{GURA}_0 \) Instance

\[
UA = \{\text{clearance, dept, role, project}\} \\
U = \{\text{Alice}\} \\
AR = \{ar_1, ar_2\}
\]

- **can\_assign\_dept**: \( \langle ar_1, \text{dept}(u) = \text{finance, IT} \rangle \)
- **can\_add\_role**: \( \langle ar_2, \text{employee} \in \text{role}(u) \land \neg(\text{manager} \in \text{role}(u)), \text{leader} \rangle \)
- **can\_delete\_project**: \( \langle ar_1, \text{prj}_1 \in \text{project}(u) \land \neg(\text{prj}_2 \in \text{project}(u)), \text{prj}_3 \rangle, \langle ar, \neg(\text{prj}_1 \in \text{project}(u)) \land \neg(\text{prj}_2 \in \text{project}(u)), \text{prj}_4 \rangle \)
Contributions of Our Paper

The \( \text{rGURA}_1 \) Schemes

For preconditions in all relations:

\[
\varphi ::= \neg \varphi \mid \varphi \land \varphi \mid \text{aua}(u) = \text{avalue} \mid \text{svalue} \in \text{sua}(u)
\]

Example \( \text{rGURA}_1 \) instance:

UA = \{clearance, dept, role, project\}
U = \{Alice\}
AR = \{ar_1, ar_2\}

\[\text{can-assign}_{\text{dept}}: \langle \text{ar}, \text{dept}(u) = \text{finance} \land \neg (\text{prj}_1 \in \text{project}(u)) \land \neg (\text{prj}_2 \in \text{project}(u)) \land \text{employee} \in \text{role}(u) \land \neg (\text{manager} \in \text{role}(u)), \text{IT} \rangle\]
Contributions of Our Paper

Reachability Problem Example

Given an initial state:

- \( \text{att}_1(\text{Alice}) = 1 \)
  \( \text{att}_2(\text{Alice}) = \{1,2\} \)

- \( \text{att}_1(\text{Alice}) = 4 \)
  \( \text{att}_2(\text{Alice}) = \{1,2,3\} \)

- \( \text{att}_1(\text{Alice}) = 12 \)
  \( \text{att}_2(\text{Alice}) = \{1,2,3,5\} \)

- \( \text{att}_1(\text{Alice}) = 10 \)
  \( \text{att}_2(\text{Alice}) = \{3\} \)

- \( \text{att}_1(\text{Alice}) = 3 \)
  \( \text{att}_2(\text{Alice}) = \{1,2\} \)

- \( \text{att}_1(\text{Alice}) = 3 \)
  \( \text{att}_2(\text{Alice}) = \{1\} \)

- \( \text{att}_1(\text{Alice}) = 5 \)
  \( \text{att}_2(\text{Alice}) = \{1,2,3\} \)

Any target state such that a query is satisfied?

World-Leading Research with Real-World Impact
Contributions of Our Paper

Two types of Reachability Problems (RP)

A query is a state or subset of a state. Given an initial state and a query of the following types:

- \(RP_{\leq}\): All attributes should be the same.
- \(RP_{\geq}\): For set-valued attribute, the target state may contain additional values.

Example:
Initial state: \(\text{att}_1(\text{Alice}) = 1, \text{att}_2(\text{Alice}) = \{1, 2\}\)
Query: \(\text{att}_1(\text{Alice}) = 1, \text{att}_2(\text{Alice}) = \{1, 3\}\)
Target States that satisfy the query:
- \(RP_{\leq}\): \(\text{att}_1(\text{Alice}) = 1, \text{att}_2(\text{Alice}) = \{1, 3\}\)
- \(RP_{\geq}\): \(\text{att}_1(\text{Alice}) = 1, \text{att}_2(\text{Alice}) = \{1, 3, 4\}\) OR \(\text{att}_1(\text{Alice}) = 1, \text{att}_2(\text{Alice}) = \{1, 3, 5\}\) OR \(\text{att}_1(\text{Alice}) = 1, \text{att}_2(\text{Alice}) = \{1, 3, 6\}\)
Contributions of Our Paper

Content of Analysis

We use \([\text{rGURA}_x]-[\text{atomic, set}, \text{Restriction}]\) denote a specialized rGURA scheme.

- The subscript \(x\) takes a value of 0 or 1.
- \text{Restriction} represents possible combinations of SR, \(\overline{D}\) and \(\overline{N}\).

Example

\([\text{rGURA}_1-\text{atomic, } \overline{N}]\) denotes an rGURA\(_1\) scheme where only atomic-valued attributes are defined and the administrative rules satisfy \(\overline{N}\).
Contributions of Our Paper

Analysis Results

- **rGURA₀**
  - [rGURA₀]
    - $RP \supseteq: C_1$
    - $RP =: C_1$
  - Only set-valued attributes
  - [rGURA₀-set]
    - $RP \supseteq: Th₂$
    - $RP =: Th₃$
  - Only atomic-valued attributes
  - [rGURA₀-atomic]
    - $RP =: Th₁$

- **rGURA₁**
  - [rGURA₁]
    - $RP \supseteq: C_4$
    - $RP =: C_4$
  - Only set-valued attributes
  - [rGURA₁-atomic]
    - $RP =: C₂$
    - No Negation
  - [rGURA₁-atomic, $N$]
    - $RP =: Th₄$
    - No Negation
  - [rGURA₁-set, $N$]
    - $RP \supseteq: Th₅$
    - $RP =: Th₆$
    - Single Rule, No Deletion
  - [rGURA₁-set, SR, $D$]
    - $RP \supseteq: Th₇$
    - $RP =: Th₈$
    - Solvable in Polynomial time (P)

$N$: No Negation  
SR: Single Rule  
$D$: No Deletion

PSPACE Complete

World-Leading Research with Real-World Impact
Lemma 1: All problems are within PSPACE.
Non-deterministic Turing Machine can simulate the algorithm. Polynomial space is needed. Thus, it is NPSPACE (NPSPACE = PSPACE).
Contributions of Our Paper

Result 2

\[ \text{RP}\_\_\_ \text{in rGURA}_0\text{-set} \text{ is a reduction from ARBAC97 analysis problem as proved in CSFW06 by S. Stoller.} \]

\[ \text{RP}\_\_\_ \text{in rGURA}_0\text{-atomic} \text{ is equivalent to path search problem in directed graph.} \]
Contributions of Our Paper

Result 3

RP\_\_ in [rGURA\_1-set, \overline{N}] can be solved by policy traversal.
RP\_\_ in [rGURA\_1-atomic, \overline{N}] is a reduction from SAS planning problem in AI.
## Our contribution

- Motivate user attributes reachability analysis.
- Define reachability problems based on a restricted version of GURA model.
- Formal proof and polynomial time solvable algorithm design.

## Interesting future work

- Heuristic algorithm to solve the general case $RP_{\preceq}$ and $RP_{\succeq}$ in $[rGURA_1]$.
- Bring Authorization Policy into consideration.
- Bring ABAC into consideration such as subject attributes and its constraints.
Thanks
Any Questions?