

Synthesising Verified Access Control Systems through Model Checking

Hasan N. Qunoo

School of Computer Science
University of Birmingham

Computer Security Reading Group 2007
Presentation for RW framework proposed by Nan Zhang et. al. [2]

Outline

- 1 Introduction
 - Definitions
 - Motivation
- 2 RW framework
 - RW Overview
 - The RW formalism and its description language
 - Example
- 3 Properties and their *RW* specifications
 - Run-statement
 - Strategies and guessing strategies
 - Properties
 - Case Study
- 4 Conclusion, Future Work and Questions
 - Conclusion & Future Work
 - Questions

Access Control

Access Control

The process of mediating every request to resources and data maintained by a system and determining whether the request should be granted or denied.

Access Control Policy

Regulations that can be **specified** in an appropriate language and then **enforced** by the access control mechanism.

Access Control Model

formal representation of the access control security policy and its working. It allows us to design proof of security properties.

Access Control Policy

Access Control Policy

must fit the services that the system intends to provide.

An Access Control Policy:

- 1 Should provide users *enough* permissions to carry out their actions and achieve their legitimate goals.
- 2 Should *prohibit* malicious goals to be reached.

Modern access control systems complexity make reasoning about them by hand infeasible.

Motivation Example

Consider a *conference paper review system*:

- 1 The chair of the PC assigns papers to PC members for reviewing.
- 2 PC member a can read PC member b 's review for a paper p provided p is not assigned to a .
- 3 If two PC members, a and b , are both assigned paper p for reviewing, a can read b 's review for p only if a has already submitted its own review for p . This is also true for b .
- 4 Having been assigned a paper p , PC member a can give up being reviewer for p before the reviewing is finished.

Using your *imagination* ... *identify* a security attack where a reviewer can be influenced by another reviewer's.

Possible attack

Stategy 1

- 1 c assigns p to both a and b for reviewing. (rule 1)
- 2 Before a submits his review for p , he resigns being reviewer for p . (rule 4)
- 3 a reads b 's review for p . (rule 2)
- 4 c assigns p to a for reviewing again. (rule 1)

Three reasons have caused the problem:

- 1 **Interactions of rules.**
- 2 **Co-operations between agents.**
- 3 **Multi-step actions.**

RW framework

RW framework uses

a *model-checking* algorithm to evaluate the fitness of access control policies. *Why?*

RW framework consists of

- 1 The RW access control formalism.
- 2 The RW access control policy description and specification language.
- 3 The RW model-checking algorithm.
- 4 AcPeg .

The RW formalism and its description language

Definition

Let $\mathbf{L}(P)$ be the set of the propositional logic formulas built from the propositional variables in the set P . An access control system S is a tuple $\langle \Sigma, P, \mathfrak{r}, \mathfrak{w} \rangle$, where Σ is a set of *agents*, P is a set of *propositional variables* and the mappings $\mathfrak{r}, \mathfrak{w} : P \times \mathcal{P}(\Sigma) \rightarrow \mathbf{L}(P)$ specify the immediate access rights of coalitions of agents.

An agent $a \in \Sigma$ is allowed to read and overwrite variable p iff the current state of the system s satisfies the propositional logic formula $\mathfrak{r}(p, \{a\})$ and $\mathfrak{w}(p, \{a\})$, respectively.

Example - a conference paper review (CPR) system

- Whether an agent is a PC member, or is the chair, or is an author of a paper is readable by all the agents. Authorship of papers cannot be changed.
- The PC chair appoints agents to be PC members. A PC member can resign her membership.
- Whether a PC member is a reviewer of a paper is readable by all the PC members except the author(s) of the paper.
- The PC chair can assign a paper to a PC member for reviewing, provided the PC member is not the paper's author.
- A reviewer of a paper can give up being reviewer, unless he has already appointed a sub-reviewer for the paper.

RW formalisation for the CPR system

To model this example in the *RW* formalism:

Two classes, `Agent` and `Paper`, need to be defined. `Agent` and `Paper` are the set of agents and papers respectively. To model relations between them, we need a number of predicates from which the set of propositional variables, P , is built.

For $a, b \in \text{Agent}$, $p \in \text{Paper}$, P includes:

<code>author(p, a)</code>	<code>pcmember(a)</code>
<code>chair(a)</code>	<code>reviewer(p, a)</code>
<code>subreviewer(p, a, b)</code>	

RW formalisation for the CPR system

For each $p \in P$, $a \in \text{Agent}$, $r(p, a)$ and $w(p, a)$ are defined:

using variables in P and logical connectors: \neg (negation), \wedge (conjunction), \vee (disjunction), \rightarrow (implication), \exists (existential quantification) and \forall (universal quantification), as follows. (For two agents a and b , $a = b$ denotes that a and b are the same agent, \top denotes the boolean value true, Σ denotes the set Agents, and ' \Leftarrow ' denotes 'is defined as'.)

Example - a conference paper review (CPR) system

- Whether an agent is a PC member, or is the chair, or is an author of a paper is readable by all the agents. Authorship of papers cannot be changed.

$$r(\text{author}(p, a), x) \Leftarrow \top$$

$$r(\text{pcmember}(a), x) \Leftarrow \top$$

$$r(\text{chair}(a), x) \Leftarrow \top$$

$$w(\text{author}(p, a), x) \Leftarrow \perp$$

- The PC chair appoints agents to be PC members. A PC member can resign her membership.

$$w(\text{pcmember}(a), x) \Leftarrow \text{chair}(x) \vee (\text{pcmember}(x) \wedge a = x)$$

Example - a conference paper review (CPR) system

- Whether a PC member is a reviewer of a paper is readable by all the PC members except the author(s) of the paper.

$$r(\text{reviewer}(p, a), x) \Leftrightarrow \text{pcmember}(x) \wedge \neg \text{author}(p, x)$$

Example - a conference paper review (CPR) system

- The PC chair can assign a paper to a PC member for reviewing, provided the PC member is not the paper's author.
- A reviewer of a paper can give up being reviewer, unless he has already appointed a sub-reviewer for the paper.

$$w(\text{reviewer}(p, a), x) \Leftrightarrow \left(\begin{array}{l} \left(\text{chair}(x) \wedge \text{pcmember}(a) \wedge \neg \text{author}(p, a) \right) \\ \vee \left(\begin{array}{l} \text{pcmember}(x) \wedge x = a \wedge \text{reviewer}(p, x) \\ \wedge \neg (\exists b \in \Sigma \text{ subreviewer}(p, x, b)) \end{array} \right) \end{array} \right)$$

Run-statement

A system described by the program in the description part is only a template. To perform model-checking, a concrete instance based on the template needs to be constructed. This task is done through a run-statement. The syntax of the run-statement is:

```
⟨RunStatement⟩ ::=  
"run for" ⟨NumberClassPair⟩ ("," ⟨NumberClassPair⟩)*  
⟨NumberClassPair⟩ ::= ⟨Integer⟩ ⟨ClassName⟩
```

for example

```
run for 3 Paper, 4 Agent
```

Strategies and guessing strategies

A strategy

is a sequence of actions by which a coalition A of agents achieves a goal. Each action in the sequence is performed by one agent in A . We assume that an agent a performs actions only if he knows that he has permissions.

A guessing strategy

It is not required that the agent knows he can perform each action.

Strategies and guessing strategies - Example

We define an access control system $S\langle \Sigma, P, r, w \rangle$, where $P = \{u, x, y, z\}$ and $\Sigma = \{a\}$. For each $p \in P$, mappings $r(p, a)$ and $w(p, a)$ are summarised as following.

We define an access control system $S\langle \Sigma, P, r, w \rangle$, where $P = \{u, x, y, z\}$ and $\Sigma = \{a\}$. For each $p \in P$, mappings $r(p, a)$ and $w(p, a)$ are summarised as following:

	u	x	y	z
r	\perp	\top	\top	\top
w	\perp	$\neg u$	u	$x \vee y$

Properties

Properties - Definition

Given a set of access control policy, is a goal, and a set of agents, the decision procedure can figure out whether there are strategies available for the agents to achieve the goal, however, without demonstrating what the strategies are.

Properties

A property

is a query, taking the form of

$$\text{check } \{ \mathcal{L} \mid \varphi \rightarrow A : \psi \}$$

where \mathcal{L} defines a number of quantified variables used in φ , ψ and A ; φ (optional) is a list of conditions based on which the goal, defined by ψ , is to be achieved; and A defines a coalition of agents who work together, intending to achieve the goal.

Example

```
check { E a: Agent, p: Paper || ~chair(a)*! -> a:{
reviewer(p,a)}}
```

Case Study - Back to the problem

Can we write a *RW* queries which find the strategy 1.

```
run for 1 Paper, 3 Agent
check {E disj a,b,c: Agent, p: Paper ||
  chair(c)*! & pmember(a)*!
  & ~submittedreview(p,a)!
  & submittedreview(p,b)*!
  & ~author(p,a)*!
  & ~reviewer(p,a)!
  & ~subreviewer(p,b,a)*!
  & ~subreviewer(p,c,a)*!
  & ~subreviewer(p,a,a)*!
  -> {a}:([review(p,b)] AND
  {a,c}:({submittedreview(p,a)}))}
```

Case Study - Back to the problem

RW find the following strategy

```
[a=1 b=2 c=3 p=1]      Strategy: 4.1
Coalition: [1]
if (review(1,2) is true) by 1 {
  skip;
} else {
  skip;
}
Coalition: [1, 3]
set reviewer(1,1) to true by 3;
set submittedreview(1,1) to true by 1;
skip;
```

Conclusion

- Access Control Policies are error-prone, hard to get right and might introduce a security holes.
- Such holes can be identified by the imagination of designers.
- Using *model-checking* to evaluate accesscontrol policy has advantages over the other approaches:
 - Because a model, M , is built based on the policy, it enables us to understand the policy as a whole. Any change made on a single rule or caused by adding/deleting a rule will be reflected on M . This makes the study of interactions of rules easier.
 - Model-checking's ability to perform temporal reasoning is suitable for exploring possible consequences of multi-step actions.

Future Work

- 1 In the *RW* language, there is no way to express mutual exclusions between the values of different variables. For example:

an agent can not be a student and a
lecturer at the same time
or

$$\text{Chair}(x) \rightarrow \neg \exists p, \text{author}(p, x)$$

- 2 In the *RW* language, there is no way to express inheritance between roles. In the context of the *RW* formalism inheritance means that one variable's value being true implies another's being true. For example:

$$\text{Chair}(x) \rightarrow \text{pmember}(x)$$

Thank you for listening!

Your comments would be much appreciated

For Further Reading I



M. R. Huth and M. D. Ryan

Logic in Computer Science: Modelling and Reasoning about Systems.

Second Edition with major revisions and updates.

Cambridge University Press, 2004. 427 pages.



Nan Zhang, Dimitar P. Guelev, and Mark Ryan.

Synthesising Verified Access Control Systems through Model Checking. .

Journal of Computer Security (in print).

For Further Reading II



Nan Zhang, Mark D. Ryan and Dimitar Guelev

Evaluating Access Control Policies Through Model Checking.

Eighth Information Security Conference (ISC'05). Lecture Notes in Computer Science volume 3650, pages 446-460, Springer-Verlag, 2005.