Automated Symbolic Analysis of Security Policies

Anh Truong
Fondazione Bruno Kessler, Trento, Italia
University of Trento, Italia

Plan of the Presentation

- Context
- Security Problem
- ASASP: an Automated Analysis Tool for Access Control Policies
- New Heuristics:
  - Forward Useful Actions
  - Ordering the Actions
- Conclusion

Context

- Access Control: the process of
  - mediating requests to resources of a system
  - determining if a request should be granted/denied

  $\Longrightarrow$ crucial for system security
- Access Control policies specify which user can access which resource (and how)
- The design and management of access control are difficult, especially in large systems
  - Models (e.g., Role-Based Access Control)

Context

- Access Control: the process of
  - mediating requests to resources of a system
  - determining if a request should be granted/denied

$\Longrightarrow$ crucial for system security

- Access Control policies specify which user can access which resource (and how)
- The design and management of access control are difficult, especially in large systems
  - Models (e.g., Role-Based Access Control)

Context

- Access Control: the process of
  - mediating requests to resources of a system
  - determining if a request should be granted/denied

  $\Longrightarrow$ crucial for system security

- Access Control policies specify which user can access which resource (and how)
- The design and management of access control are difficult, especially in large systems
  - Models (e.g., Role-Based Access Control)

Simplest Access Control Model

User	Permission
Alice	GrantTenure
Alice	AssignGrades
Alice	ReceiveHBenefits
Alice	UseGym
Bob	GrantTenure
Bob	AssignGrades
Bob	UseGym
Charlie	AssignGrades
Charlie	ReceiveHBenefits
Charlie	UseGym
David	AssignHWScores
David	Register4Courses
David	UseGym
Eve	ReceiveHBenefits
Eve	UseGym
Fred	Register4Courses
Fred	UseGym
Greg	UseGym

Role-Based Access Control (RBAC)

Idea: decompose subject-object relationship using roles:

User Assignment (UA)

User	Role
Alice	PCMember
Bob	Faculty
Charlie	Faculty
David	TA
David	Student
Eve	UEmployee
Fred	Student
Greg	UMember

Permission Assignment (PA)

Role	Permission
PCMember	GrantTenure
PCMember	AssignGrades
PCMember	ReceiveHBenefits
PCMember	UseGym
Faculty	AssignGrades
Faculty	ReceiveHBenefits
Faculty	UseGym
TA	AssignHWScores
TA	Register4Courses
TA	UseGym
UEmployee	ReceiveHBenefits
UEmployee	UseGym
Student	Register4Courses
Student	UseGym
UMember	UseGym

Role-Based Access Control (RBAC)

Idea: decompose subject-object relationship using roles:

User Assignment (UA)

User	Role
Alice	PCMember
Bob	Faculty
Charlie	Faculty
David	TA
David	Student
Eve	UEmployee
Fred	Student
Greg	UMember

Permission Assignment (PA)

Role	Permission
PCMember	GrantTenure
PCMember	AssignGrades
PCMember	ReceiveHBenefits
PCMember	UseGym
Faculty	AssignGrades
Faculty	ReceiveHBenefits
Faculty	UseGym
TA	AssignHWScores
TA	Register4Courses
TA	UseGym
UEmployee	ReceiveHBenefits
UEmployee	UseGym
Student	Register4Courses
Student	UseGym
UMember	UseGym

Role-Based Access Control (RBAC)

Idea: decompose subject-object relationship using roles:

User Assignment (UA)

User	Role
Alice	PCMember
Bob	Faculty
Charlie	Faculty
David	TA
David	Student
Eve	UEmployee
Fred	Student
Greg	UMember

Permission Assignment (PA)

Role	Permission
PCMember	GrantTenure
PCMember	AssignGrades
PCMember	ReceiveHBenefits
PCMember	UseGym
Faculty	AssignGrades
Faculty	ReceiveHBenefits
Faculty	UseGym
TA	AssignHWScores
TA	Register4Courses
TA	UseGym
UEmployee	ReceiveHBenefits
UEmployee	UseGym
Student	Register4Courses
Student	UseGym
UMember	UseGym

Role-Based Access Control (RBAC)

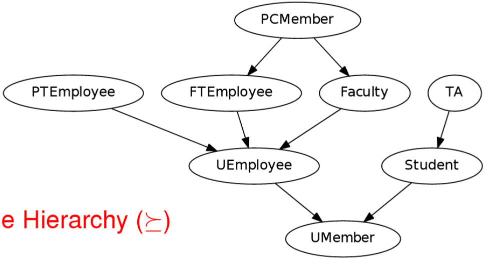
The use of role hierarchies leads to a compact RBAC policies

User Assignment (UA)

User	Role
Alice	PCMember
Bob	Faculty
Charlie	Faculty
David	TA
David	Student
Eve	UEmployee
Fred	Student
Greg	UMember

Permission Assignment (PA)

Role	Permission
PCMember	GrantTenure
Faculty	AssignGrades
TA	AssignHWScores
UEmployee	ReceiveHBenefits
Student	Register4Courses
UMember	UseGym



Role Hierarchy ($\succeq$)

Administrative RBAC

- (A)RBAC model simplifies specification and administration of access control policies
- Idea: specify how RBAC policies are changed by administrative actions
- Our focus: ARBAC97
    - Administrative actions can only modify User Assignment

ARBAC97: URA97 sub-model

- Role Assignment:
  $Faculty \ : \ \langle +\{Student\}, -\{TA\} \rangle \Longrightarrow \oplus PTEmpl.$

  $PCMember \ : \ \langle +\{PTEmpl.\}, \emptyset \rangle \Longrightarrow \oplus Faculty$

- Role Revocation:
  $Faculty \ : \ \langle +\{Student\}, \emptyset \rangle \Longrightarrow \ominus Student$

- Mutually Exclusive Roles (MER):
  MER(TA, PTEmployee)

  $Faculty \ : \ \langle +\{Student\}, \emptyset \rangle \Longrightarrow \oplus TA$

  Assign Fred to *TA*? No

User	Role
Alice	PCMember
Bob	Faculty
Charlie	Faculty
David	TA
David	Student
Eve	UEmployee
Fred	Student
Greg	UMember

ARBAC97: URA97 sub-model

- Role Assignment:
  $Faculty \ : \ \langle +\{Student\}, -\{TA\}\rangle \Longrightarrow \oplus PTEmpl.$

  $PCMember \ : \ \langle +\{PTEmpl.\}, \emptyset\rangle \Longrightarrow \oplus Faculty$

- Role Revocation:
  $Faculty \ : \ \langle +\{Student\}, \emptyset\rangle \Longrightarrow \ominus Student$

- Mutually Exclusive Roles (MER):
  MER(TA, PTEmployee)

  $Faculty \ : \ \langle +\{Student\}, \emptyset\rangle \Longrightarrow \oplus TA$

  Assign Fred to *TA*? No

User	Role
Alice	PCMember
Bob	Faculty
Charlie	Faculty
David	TA
David	Student
Eve	UEmployee
Fred	Student
Greg	UMember

ARBAC97: URA97 sub-model

- Role Assignment:
  $Faculty \ : \ \langle +\{Student\}, -\{TA\}\rangle \Longrightarrow \oplus PTEmpl.$

  $PCMember \ : \ \langle +\{PTEmpl.\}, \emptyset\rangle \Longrightarrow \oplus Faculty$

- Role Revocation:
  $Faculty \ : \ \langle +\{Student\}, \emptyset\rangle \Longrightarrow \ominus Student$

- Mutually Exclusive Roles (MER):
  MER(TA, PTEmployee)

  $Faculty \ : \ \langle +\{Student\}, \emptyset\rangle \Longrightarrow \oplus TA$

  Assign Fred to *TA*? No

User	Role
Alice	PCMember
Bob	Faculty
Charlie	Faculty
David	TA
David	Student
Eve	UEmployee
Fred	Student
Fred	PTEmployee
Greg	UMember

ARBAC97: URA97 sub-model

- Role Assignment:
  $Faculty : \langle +\{Student\}, -\{TA\} \rangle \Longrightarrow \oplus PTEmpl.$

  $PCMember : \langle +\{PTEmpl.\}, \emptyset \rangle \Longrightarrow \oplus Faculty$

- Role Revocation:
  $Faculty : \langle +\{Student\}, \emptyset \rangle \Longrightarrow \ominus Student$

- Mutually Exclusive Roles (MER):
  MER(TA, PTEmployee)

  $Faculty : \langle +\{Student\}, \emptyset \rangle \Longrightarrow \oplus TA$

  Assign Fred to *TA*? No

User	Role
Alice	PCMember
Bob	Faculty
Charlie	Faculty
David	TA
David	Student
Eve	UEmployee
Fred	Student
Fred	PTEmployee
Greg	UMember

- Role Assignment:
  $Faculty \ : \ \langle +\{Student\}, -\{TA\} \rangle \Longrightarrow \oplus PTEmpl.$

  $PCMember \ : \ \langle +\{PTEmpl.\}, \emptyset \rangle \Longrightarrow \oplus Faculty$

- Role Revocation:
  $Faculty \ : \ \langle +\{Student\}, \emptyset \rangle \Longrightarrow \ominus Student$

- Mutually Exclusive Roles (MER):
  MER(TA, PTEmployee)

$Faculty \ : \ \langle +\{Student\}, \emptyset \rangle \Longrightarrow \oplus TA$

Assign Fred to *TA*? No

User	Role
Alice	PCMember
Bob	Faculty
Charlie	Faculty
David	TA
David	Student
Eve	UEmployee
Fred	Student
Fred	PTEmployee
Greg	UMember
Fred	Faculty

ARBAC97: URA97 sub-model

- Role Assignment:
  Faculty : $\langle +\{\mathit{Student}\}, -\{\mathit{TA}\}\rangle \Longrightarrow \oplus \mathit{PTEmpl.}$

  PCMember : $\langle +\{\mathit{PTEmpl.}\}, \emptyset\rangle \Longrightarrow \oplus \mathit{Faculty}$

- Role Revocation:
  Faculty : $\langle +\{\mathit{Student}\}, \emptyset\rangle \Longrightarrow \ominus \mathit{Student}$

- Mutually Exclusive Roles (MER):
  MER(TA, PTEmployee)

  Faculty : $\langle +\{\mathit{Student}\}, \emptyset\rangle \Longrightarrow \oplus \mathit{TA}$

  Assign Fred to *TA*? No

User	Role
Alice	PCMember
Bob	Faculty
Charlie	Faculty
David	TA
David	Student
Eve	UEmployee
Fred	Student
Fred	PTEmployee
Greg	UMember
Fred	Faculty

- Role Assignment:
  $Faculty \;:\; \langle +\{Student\}, -\{TA\}\rangle \Longrightarrow \oplus PTEmpl.$

  $PCMember \;:\; \langle +\{PTEmpl.\}, \emptyset\rangle \Longrightarrow \oplus Faculty$

- Role Revocation:
  $Faculty \;:\; \langle +\{Student\}, \emptyset\rangle \Longrightarrow \ominus Student$

- Mutually Exclusive Roles (MER):
  MER(TA, PTEmployee)

  $Faculty \;:\; \langle +\{Student\}, \emptyset\rangle \Longrightarrow \oplus TA$

  Assign Fred to *TA*? No

User	Role
Alice	PCMember
Bob	Faculty
Charlie	Faculty
David	TA
David	Student
Eve	UEmployee
Fred	Student
Fred	PTEmployee
Greg	UMember
Fred	Faculty

ARBAC97: URA97 sub-model

- Role Assignment:
  $Faculty \; : \; \langle +\{Student\}, -\{TA\} \rangle \Longrightarrow \oplus PTEmpl.$

  $PCMember \; : \; \langle +\{PTEmpl.\}, \emptyset \rangle \Longrightarrow \oplus Faculty$

- Role Revocation:
  $Faculty \; : \; \langle +\{Student\}, \emptyset \rangle \Longrightarrow \ominus Student$

- Mutually Exclusive Roles (MER):
  MER(TA, PTEmployee)

  $Faculty \; : \; \langle +\{Student\}, \emptyset \rangle \Longrightarrow \oplus TA$

  Assign Fred to *TA*? No

User	Role
Alice	PCMember
Bob	Faculty
Charlie	Faculty
David	TA
Eve	UEmployee
Fred	Student
Fred	PTEmployee
Greg	UMember
Fred	Faculty

ARBAC97: URA97 sub-model

- Role Assignment:
  $Faculty \; : \; \langle +\{Student\}, -\{TA\} \rangle \Longrightarrow \oplus PTEmpl.$

  $PCMember \; : \; \langle +\{PTEmpl.\}, \emptyset \rangle \Longrightarrow \oplus Faculty$

- Role Revocation:
  $Faculty \; : \; \langle +\{Student\}, \emptyset \rangle \Longrightarrow \ominus Student$

- Mutually Exclusive Roles (MER):
  MER(TA, PTEmployee)

  $Faculty \; : \; \langle +\{Student\}, \emptyset \rangle \Longrightarrow \oplus TA$

  Assign Fred to *TA*? No

User	Role
Alice	PCMember
Bob	Faculty
Charlie	Faculty
David	TA
Eve	UEmployee
Fred	Student
Fred	PTEmployee
Greg	UMember
Fred	Faculty

ARBAC97: URA97 sub-model

- Role Assignment:
  $Faculty \;:\; \langle +\{Student\}, -\{TA\} \rangle \Longrightarrow \oplus PTEmpl.$

  $PCMember \;:\; \langle +\{PTEmpl.\}, \emptyset \rangle \Longrightarrow \oplus Faculty$

- Role Revocation:
  $Faculty \;:\; \langle +\{Student\}, \emptyset \rangle \Longrightarrow \ominus Student$

- Mutually Exclusive Roles (MER):
  MER(TA, PTEmployee)

  $Faculty \;:\; \langle +\{Student\}, \emptyset \rangle \Longrightarrow \oplus TA$

  Assign Fred to *TA*? No

User	Role
Alice	PCMember
Bob	Faculty
Charlie	Faculty
David	TA
Eve	UEmployee
Fred	Student
Fred	PTEmployee
Greg	UMember
Fred	Faculty

ARBAC97: URA97 sub-model

- Role Assignment:
  $Faculty \; : \; \langle +\{Student\}, -\{TA\} \rangle \Longrightarrow \oplus PTEmpl.$

  $PCMember \; : \; \langle +\{PTEmpl.\}, \emptyset \rangle \Longrightarrow \oplus Faculty$

- Role Revocation:
  $Faculty \; : \; \langle +\{Student\}, \emptyset \rangle \Longrightarrow \ominus Student$

- Mutually Exclusive Roles (MER):
  MER(TA, PTEmployee)

  $Faculty \; : \; \langle +\{Student\}, \emptyset \rangle \Longrightarrow \oplus TA$

  Assign Fred to *TA*? No

User	Role
Alice	PCMember
Bob	Faculty
Charlie	Faculty
David	TA
Eve	UEmployee
Fred	Student
Fred	PTEmployee
Greg	UMember
Fred	Faculty

Analysis of Access Control Policies

...

User	Role
Alice	PCMember
Bob	Faculty
Fred	Student

...

$Faculty : \langle +\{Student\}, -\{TA\} \rangle \Rightarrow \oplus PTEmpl.$

User	Role
Alice	PCMember
Bob	Faculty
Fred	Student
Fred	PTEmployee

...

$PCMem. : \langle +\{PTEmpl.\} +, \emptyset \rangle \Rightarrow \oplus Faculty$

User	Role
Alice	PCMember
Bob	Faculty
Fred	Student
Fred	PTEmployee
Fred	Faculty

...

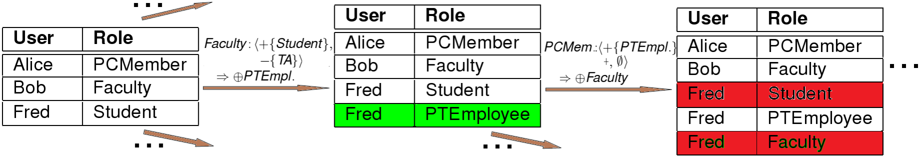
- **Problem:** Starting from an initial RBAC policy and using the administrative actions in the ARBAC policy, is there a way to assign roles Student and Faculty to Fred? Yes: conflict

  $\Longrightarrow$ Need for conflict analysis

- In large systems (e.g., Dresdner bank: 40,000 users and 1,300 roles), analysis of access control policies can be very difficult.
- To predict the effects of changes on policies of real-world complexity by manual inspection is unfeasible.

  $\Longrightarrow$ **Automated support needed!**

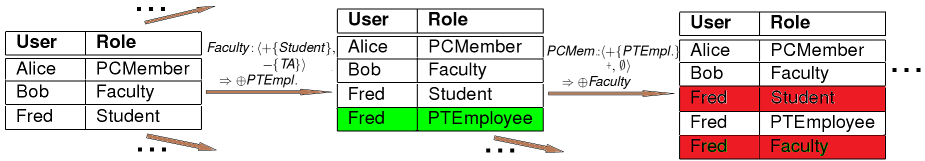
Analysis of Access Control Policies



- **Problem:** Starting from an initial RBAC policy and using the administrative actions in the ARBAC policy, is there a way to assign roles Student and Faculty to Fred? Yes: conflict
  $\Longrightarrow$Need for conflict analysis
- In large systems (e.g., Dresdner bank: 40,000 users and 1,300 roles), analysis of access control policies can be very difficult.
- To predict the effects of changes on policies of real-world complexity by manual inspection is unfeasible.

  $\Longrightarrow$ **Automated support needed!**

Analysis of Access Control Policies



- **Problem:** Starting from an initial RBAC policy and using the administrative actions in the ARBAC policy, is there a way to assign roles Student and Faculty to Fred? Yes: conflict
  $\Longrightarrow$Need for conflict analysis
- In large systems (e.g., Dresdner bank: 40,000 users and 1,300 roles), analysis of access control policies can be very difficult.
- To predict the effects of changes on policies of real-world complexity by manual inspection is unfeasible.
  $\Longrightarrow$ **Automated support needed!**

Analysis of Access Control Policies

User	Role
Alice	PCMember
Bob	Faculty
Fred	Student

$\xrightarrow[\Rightarrow \oplus PTEmpl.]{Faculty : \langle +\{Student\}, -\{TA\}\rangle}$

User	Role
Alice	PCMember
Bob	Faculty
Fred	Student
Fred	PTEmployee

$\xrightarrow[\Rightarrow \oplus Faculty]{PCMem. : \langle +\{PTEmpl.\} +, \emptyset\rangle}$

User	Role
Alice	PCMember
Bob	Faculty
Fred	Student
Fred	PTEmployee
Fred	Faculty

...

- **Problem:** Starting from an initial RBAC policy and using the administrative actions in the ARBAC policy, is there a way to assign roles Student and Faculty to Fred? Yes: conflict
  $\Longrightarrow$Need for conflict analysis
- In large systems (e.g., Dresdner bank: 40,000 users and 1,300 roles), analysis of access control policies can be very difficult.
- To predict the effects of changes on policies of real-world complexity by manual inspection is unfeasible.

$\Longrightarrow$ **Automated support needed!**

Analysis of Access Control Policies

User	Role
Alice	PCMember
Bob	Faculty
Fred	Student

$\xrightarrow{Faculty : \langle +\{Student\}, -\{TA\}\rangle \Rightarrow \oplus PTEmpl.}$

User	Role
Alice	PCMember
Bob	Faculty
Fred	Student
Fred	PTEmployee

$\xrightarrow{PCMem.:\langle +\{PTEmpl.\} +, \emptyset\rangle \Rightarrow \oplus Faculty}$

User	Role
Alice	PCMember
Bob	Faculty
Fred	Student
Fred	PTEmployee
Fred	Faculty

...

- **Problem:** Starting from an initial RBAC policy and using the administrative actions in the ARBAC policy, is there a way to assign roles Student and Faculty to Fred? Yes: conflict

  $\Longrightarrow$ Need for conflict analysis
- In large systems (e.g., Dresdner bank: 40,000 users and 1,300 roles), analysis of access control policies can be very difficult.
- To predict the effects of changes on policies of real-world complexity by manual inspection is unfeasible.

  $\Longrightarrow$ **Automated support needed!**

ASASP: **A**utomated **S**ymbolic **A**nalysis of **S**ecurity **P**olicies Tool

URA97: User-role Reachability Problem

- Given
  - initial RBAC policy $\langle U, R, UA_0 \rangle$
  - a set administrative actions $\psi = \langle \mathit{role\_assignment}, \mathit{role\_revocation} \rangle$
- Establish if a user u ($\in U$) can be assigned to a role r ($\in R$) by applying a sequence of administrative actions in ψ

URA97: User-role Reachability Problem

- Given
  - initial RBAC policy $\langle U, R, UA_0 \rangle$
  - a set administrative actions $\psi = \langle \mathit{role\_assignment}, \mathit{role\_revocation} \rangle$
- Establish if a user u ($\in U$) can be assigned to a role r ($\in R$) by applying a sequence of administrative actions in ψ

Other Security analysis problems (e.g., Role containment, Weakest precondition...) can be reduced to User-role reachability problem

$\Longrightarrow$ User-role reachability problem is core problem in security analysis.

Available Analysis Tools

Features	RBAC-PAT	MOHAWK	PMS	VAC	ASASP
MER constraints	✗	✗	✗	✗	✓
Unknown number of Users	✗	✗	✗	✓	✓
Non-Separate Administration	✗	✗	✓	✓	✓
$\|C_a\| > 1$	✗	✗	✗	✗	✓

Our Approach: Using a Decidable Fragment of (Many-sorted) First-order Logic

- Sorts: *User*, *Role*
- Predicate symbols: $ua : User \times Role$
- Defining UA_0:

$$\forall u, r.(ua(u,r) \Leftrightarrow \left(\begin{array}{ll} (u = u_1 \wedge r = Role\,1) & \vee \\ (u = u_2 \wedge r = Role\,2) & \vee \\ (u = u_3 \wedge r = Role\,3) & \vee \\ \vdots & \end{array} \right))$$

- **MER Constraints:** No user can be *TA* and *PTEmployee* at the same time:

$$\forall u.\neg(ua(u, TA) \wedge ua(u, PTEmployee))$$

- **Goal :** There exists a user who is member of a certain role:

$$\exists u, r.(ua(u,r) \wedge r = Student)$$

Our Approach: Using a Decidable Fragment of (Many-sorted) First-order Logic

- $UEmpl. \ : \ \langle +\{Student\}, -\{TA\}\rangle \Longrightarrow \oplus PTEmpl.$

$$\exists u_a, r_a.(ua(u_a, r_a) \wedge r_a = UEmployee) \wedge$$
$$\exists u. \left(\begin{array}{l} ua(u, Student) \wedge \forall r_2.(r_2 = TA \Rightarrow \neg ua(u, r_2)) \wedge \\ \forall x, y.(ua'(x, y) \Leftrightarrow ((x = u \wedge y = PTEmployee) \vee ua(x, y))) \end{array} \right)$$

- $UEmpl. \ : \ \langle \{Student\}, \emptyset\rangle \Longrightarrow \ominus Student$

$$\exists u_a, r_a.(ua(u_a, r_a) \wedge r_a = UEmployee) \wedge$$
$$\exists u. \left(\begin{array}{l} \exists r_1.(ua(u, r_1) \wedge r_1 = Student) \wedge \\ \forall x, y.(ua'(x, y) \Leftrightarrow (\neg(x = u \wedge y = Student) \wedge ua(x, y))) \end{array} \right)$$

Our Approach

Given symbolic representation of

- T_{RBAC} = theory specifying RBAC policies
- $I(ua)$ = initial RBAC policy
- $G(ua)$ = a goal (e.g., user u is a member of role r)
- $\tau(ua, ua')$ = administrative actions in ψ

Our Approach

Given symbolic representation of

- T_{RBAC} = theory specifying RBAC policies
- $I(ua)$ = initial RBAC policy
- $G(ua)$ = a goal (e.g., user u is a member of role r)
- $\tau(ua, ua')$ = administrative actions in ψ

Run a symbolic backward reachability procedure

- $R_0(ua) := G(ua)$
- $R_{i+1}(ua) := \exists ua'.(R_i(ua') \wedge \tau(ua, ua'))$ (pre-image) for i ≥ 0

Our Approach

Run a symbolic backward reachability procedure

- $R_0(ua) := G(ua)$
- $R_{i+1}(ua) := \exists ua'.(R_i(ua') \wedge \tau(ua, ua'))$ (pre-image) for i ≥ 0

- **Initial States**
- **Goal States**
- **Backward Reachable States**

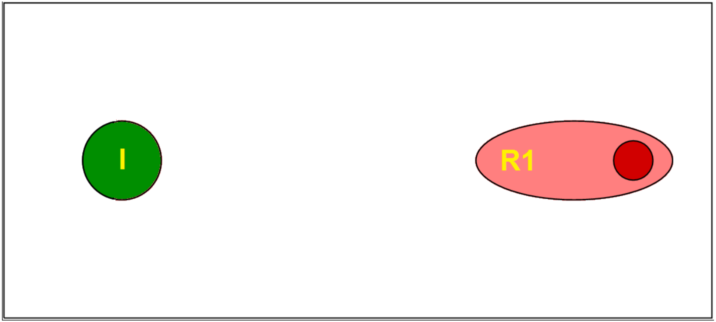
Safety check:
$R_0 \wedge I$

Our Approach

Run a symbolic backward reachability procedure

- $R_0(ua) := G(ua)$
- $R_{i+1}(ua) := \exists ua'.(R_i(ua') \wedge \tau(ua, ua'))$ (pre-image) for i ≥ 0

Initial States **Goal States** **Backward Reachable States**



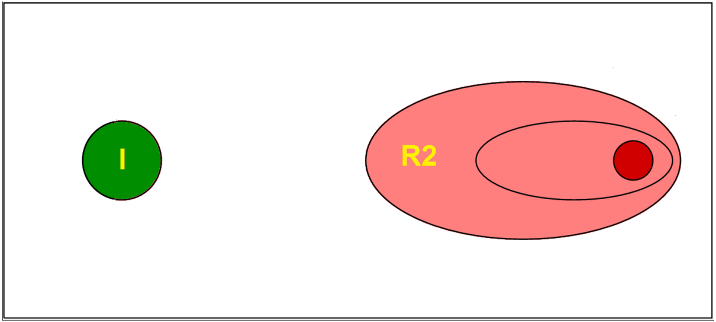
Safety check:
$R_1 \wedge I$

Our Approach

Run a symbolic backward reachability procedure

- $R_0(ua) := G(ua)$
- $R_{i+1}(ua) := \exists ua'.(R_i(ua') \wedge \tau(ua, ua'))$ (pre-image) for i $\geq$ 0

Initial States **Goal States** **Backward Reachable States**



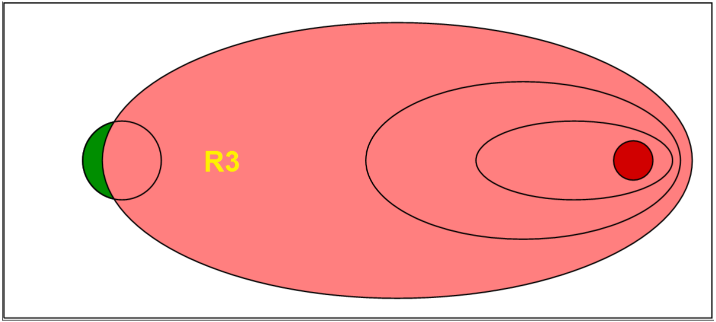
Safety check:
$R_2 \wedge I$

Our Approach

Run a symbolic backward reachability procedure

- $R_0(ua) := G(ua)$
- $R_{i+1}(ua) := \exists ua'.(R_i(ua') \wedge \tau(ua, ua'))$ (pre-image) for i $\geq$ 0

Initial States **Goal States** **Backward Reachable States**



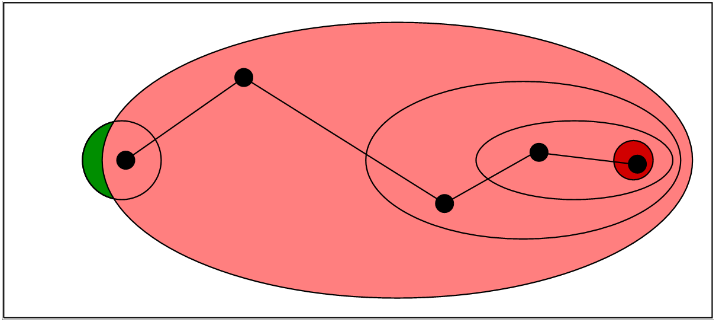
Safety check:
$R_3 \wedge I$

Our Approach

Run a symbolic backward reachability procedure

- $R_0(ua) := G(ua)$
- $R_{i+1}(ua) := \exists ua'.(R_i(ua') \wedge \tau(ua, ua'))$ (pre-image) for i ≥ 0

- **Initial States**
- **Goal States**
- **Backward Reachable States**

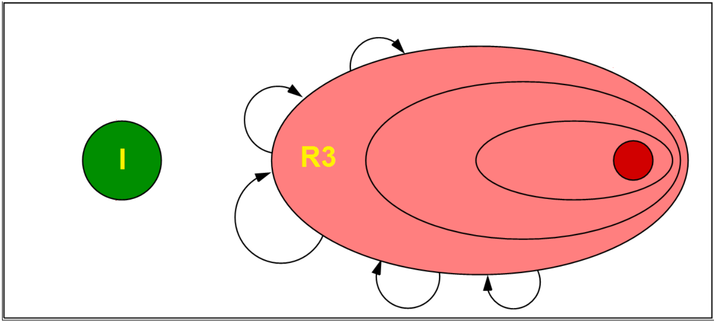


Our Approach

Run a symbolic backward reachability procedure

- $R_0(ua) := G(ua)$
- $R_{i+1}(ua) := \exists ua'.(R_i(ua') \wedge \tau(ua, ua'))$ (pre-image) for i ≥ 0

Initial States **Goal States** **Backward Reachable States**



Fix-point check
$R_4 \Rightarrow R_3$

ASASP

- An automated analysis tool for Administrative RBAC policies
- Scalability: Heuristics
  - Useful actions
  - Increasingly precise approximations of large policies
  - Reuse of previous computation states
- Initial results: ASASP outperforms MOHAWK and RBAC-PAT on their benchmarks
- Very recently, new tools VAC and PMS with their benchmarks are introduced
  - ASASP seems to have bad behaviors with these benchmarks

$\Rightarrow$ need further heuristics

- An automated analysis tool for Administrative RBAC policies
- Scalability: Heuristics
  - Useful actions
  - Increasingly precise approximations of large policies
  - Reuse of previous computation states
- Initial results: ASASP outperforms MOHAWK and RBAC-PAT on their benchmarks
- Very recently, new tools VAC and PMS with their benchmarks are introduced
  - ASASP seems to have bad behaviors with these benchmarks

$\Rightarrow$ need further heuristics

ASASP

- An automated analysis tool for Administrative RBAC policies
- Scalability: Heuristics
  - Useful actions
  - Increasingly precise approximations of large policies
  - Reuse of previous computation states
- Initial results: ASASP outperforms MOHAWK and RBAC-PAT on their benchmarks
- Very recently, new tools VAC and PMS with their benchmarks are introduced
  - ASASP seems to have bad behaviors with these benchmarks

$\Rightarrow$ need further heuristics

- An automated analysis tool for Administrative RBAC policies
- Scalability: Heuristics
  - Useful actions
  - Increasingly precise approximations of large policies
  - Reuse of previous computation states
- Initial results: ASASP outperforms MOHAWK and RBAC-PAT on their benchmarks
- Very recently, new tools VAC and PMS with their benchmarks are introduced
  - ASASP seems to have bad behaviors with these benchmarks

$\Rightarrow$ need further heuristics

ASASP with new Heuristics

- Forward Useful Actions
- Ordering the Actions

Recall: Useful Actions

- Let ψ be administrative actions and R_g a set of roles:
  - An action $\tau \in \psi$ is *0-useful* iff its target role is in R_g
  - τ is *k-useful* (for $k > 0$) iff it is:
    - $(k-1)$-useful or,
    - its target role occurs (possibly negated) in the simple pre-condition of a $(k-1)$-useful action

Recall: Useful Actions

- Given ψ:
  - $r_a : \langle +\{r_1\}, \emptyset\rangle \Longrightarrow \oplus r_2$
  - $r_a : \langle +\{r_2\}, \emptyset\rangle \Longrightarrow \oplus r_3$
  - $r_a : \langle +\{r_2\}, -\{r_4\}\rangle \Longrightarrow \oplus r_5$
  - $r_a : \langle +\{r_1\}, \emptyset\rangle \Longrightarrow \ominus r_1$
  - $r_a : \langle +\{r_2\}, \emptyset\rangle \Longrightarrow \ominus r_2$
  - Goal: r_5

1. $\psi^{\leq 0} := \{r_a : \langle +\{r_2\}, -\{r_4\}\rangle \Longrightarrow \oplus r_5\}$
2. $\psi^{\leq 1} := \psi^{\leq 0} \cup \{r_a : \langle +\{r_1\}, \emptyset\rangle \Longrightarrow \oplus r_2\}$
3. $\psi^{\leq 2} := \psi^{\leq 1} \cup \{r_a : \langle +\{r_1\}, \emptyset\rangle \Longrightarrow \ominus r_1\}$
4. Stop since fix-point reached: $\psi^{\leq k} = \psi^{\leq 2}$ for $k > 2$

Recall: Useful Actions

- Given ψ:
  - $r_a : \langle +\{r_1\}, \emptyset \rangle \Longrightarrow \oplus r_2$
  - $r_a : \langle +\{r_2\}, \emptyset \rangle \Longrightarrow \oplus r_3$
  - $r_a : \langle +\{r_2\}, -\{r_4\} \rangle \Longrightarrow \oplus r_5$
  - $r_a : \langle +\{r_1\}, \emptyset \rangle \Longrightarrow \ominus r_1$
  - $r_a : \langle +\{r_2\}, \emptyset \rangle \Longrightarrow \ominus r_2$
  - Goal: r_5

1. $\psi^{\leq 0} := \{r_a : \langle +\{r_2\}, -\{r_4\} \rangle \Longrightarrow \oplus r_5\}$
2. $\psi^{\leq 1} := \psi^{\leq 0} \cup \{r_a : \langle +\{r_1\}, \emptyset \rangle \Longrightarrow \oplus r_2\}$
3. $\psi^{\leq 2} := \psi^{\leq 1} \cup \{r_a : \langle +\{r_1\}, \emptyset \rangle \Longrightarrow \ominus r_1\}$
4. Stop since fix-point reached: $\psi^{\leq k} = \psi^{\leq 2}$ for $k > 2$

Recall: Useful Actions

- Given ψ:
  - $r_a \ : \ \langle +\{r_1\}, \emptyset \rangle \Longrightarrow \oplus r_2$
  - $r_a \ : \ \langle +\{r_2\}, \emptyset \rangle \Longrightarrow \oplus r_3$
  - $r_a \ : \ \langle +\{r_2\}, -\{r_4\} \rangle \Longrightarrow \oplus r_5$
  - $r_a \ : \ \langle +\{r_1\}, \emptyset \rangle \Longrightarrow \ominus r_1$
  - $r_a \ : \ \langle +\{r_2\}, \emptyset \rangle \Longrightarrow \ominus r_2$
  - Goal: r_5

1. $\psi^{\leq 0} := \{r_a \ : \ \langle +\{r_2\}, -\{r_4\} \rangle \Longrightarrow \oplus r_5\}$
2. $\psi^{\leq 1} := \psi^{\leq 0} \cup \{r_a \ : \ \langle +\{r_1\}, \emptyset \rangle \Longrightarrow \oplus r_2\}$
3. $\psi^{\leq 2} := \psi^{\leq 1} \cup \{r_a \ : \ \langle +\{r_1\}, \emptyset \rangle \Longrightarrow \ominus r_1\}$
4. Stop since fix-point reached: $\psi^{\leq k} = \psi^{\leq 2}$ for $k > 2$

Recall: Useful Actions

- Given ψ:
  - $r_a \; : \; \langle +\{r_1\}, \emptyset \rangle \Longrightarrow \oplus r_2$
  - $r_a \; : \; \langle +\{r_2\}, \emptyset \rangle \Longrightarrow \oplus r_3$
  - $r_a \; : \; \langle +\{r_2\}, -\{r_4\} \rangle \Longrightarrow \oplus r_5$
  - $r_a \; : \; \langle +\{r_1\}, \emptyset \rangle \Longrightarrow \ominus r_1$
  - $r_a \; : \; \langle +\{r_2\}, \emptyset \rangle \Longrightarrow \ominus r_2$
  - Goal: r_5

1. $\psi^{\leq 0} := \{r_a \; : \; \langle +\{r_2\}, -\{r_4\} \rangle \Longrightarrow \oplus r_5\}$
2. $\psi^{\leq 1} := \psi^{\leq 0} \cup \{r_a \; : \; \langle +\{r_1\}, \emptyset \rangle \Longrightarrow \oplus r_2\}$
3. $\psi^{\leq 2} := \psi^{\leq 1} \cup \{r_a \; : \; \langle +\{r_1\}, \emptyset \rangle \Longrightarrow \ominus r_1\}$
4. Stop since fix-point reached: $\psi^{\leq k} = \psi^{\leq 2}$ for $k > 2$

New Heuristics: Forward Useful Actions

- backward vs. forward useful actions
- Let ψ be administrative actions and R_i a set of roles presenting in UA_0:
  - $\tau \in \psi$ is forward 0-*useful* iff its pre-condition is a subset of R_i
  - τ is forward k-*useful* (for $k > 0$) iff it is:
    - $(k-1)$-useful or,
    - its pre-condition is a subset of $R_i = R_i \cup \{r|$ r is the target role of a $(k-1)$-useful action$\}$

New Heuristics: Forward Useful Actions

- backward vs. forward useful actions
- Let ψ be administrative actions and R_i a set of roles presenting in UA_0:
  - $\tau \in \psi$ is forward 0-*useful* iff its pre-condition is a subset of R_i
  - τ is forward k-*useful* (for $k > 0$) iff it is:
    - $(k-1)$-useful or,
    - its pre-condition is a subset of $R_i = R_i \cup \{r |$ r is the target role of a $(k-1)$-useful action$\}$

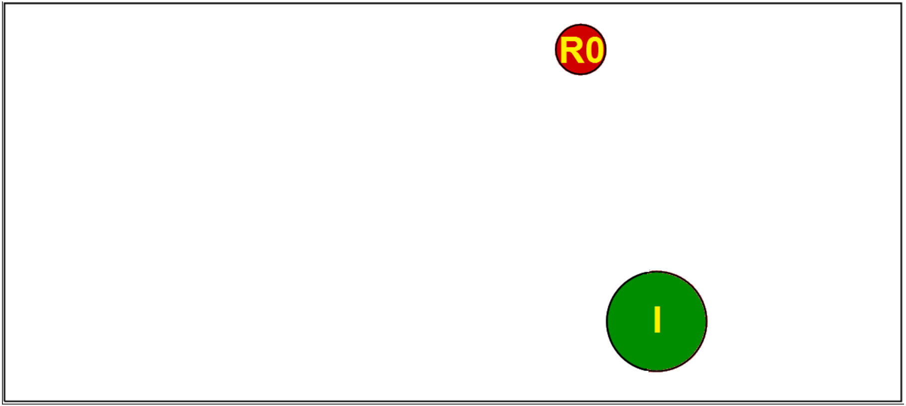
New Heuristics: Forward Useful Actions

- backward vs. forward useful actions
- Let ψ be administrative actions and R_i a set of roles presenting in UA_0:
  - $\tau \in \psi$ is forward 0-*useful* iff its pre-condition is a subset of R_i
  - τ is forward k-*useful* (for $k > 0$) iff it is:
    - $(k-1)$-useful or,
    - its pre-condition is a subset of $R_i = R_i \cup \{r|$ r is the target role of a $(k-1)$-useful action$\}$

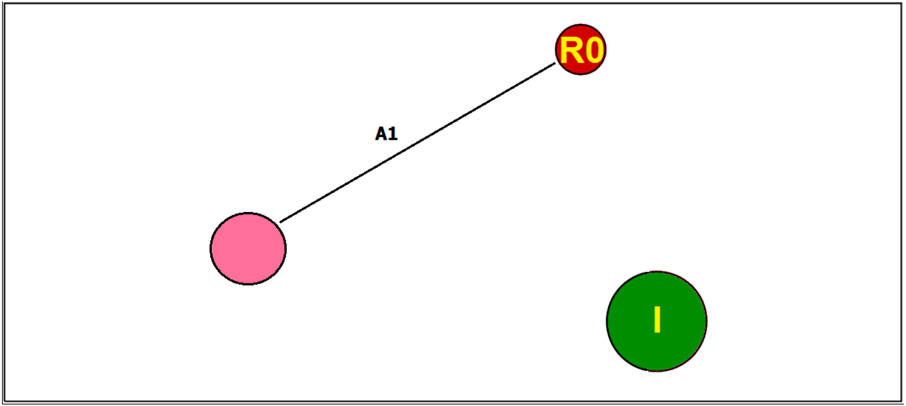
New Heuristics: Integrating to ASASP

- Compute ψ_B by using backward useful actions
- Compute ψ_F by using forward useful actions
- Solve the user-role reachability with the set $\psi' = \psi_B \cap \psi_F$ of actions

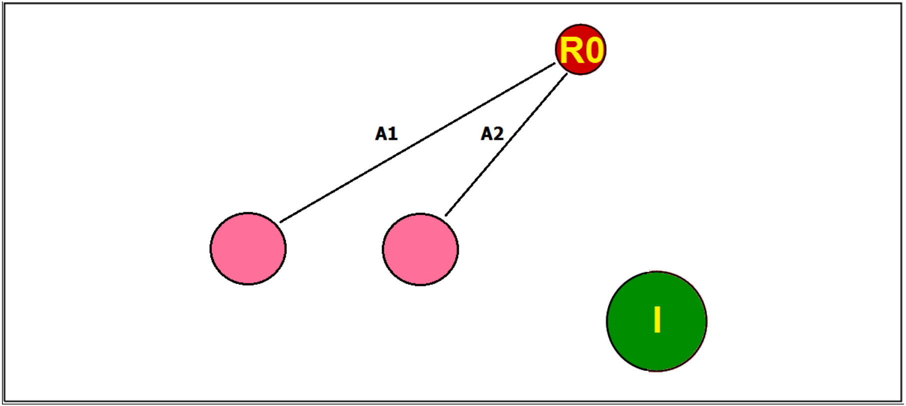
New Heuristics: Ordering the Actions



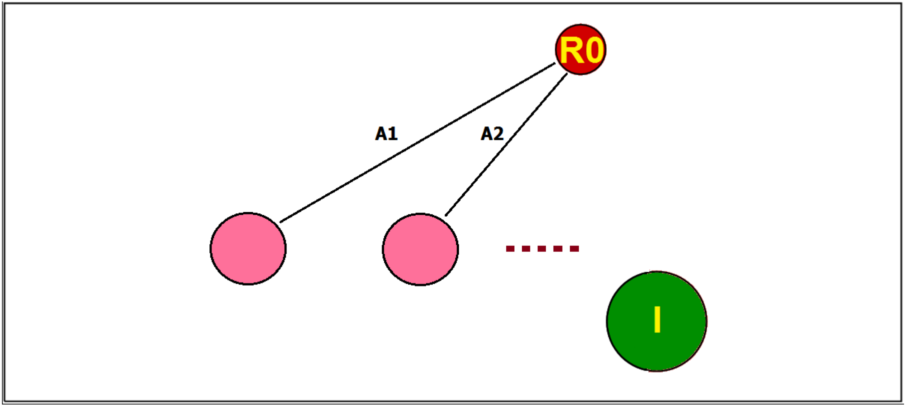
New Heuristics: Ordering the Actions



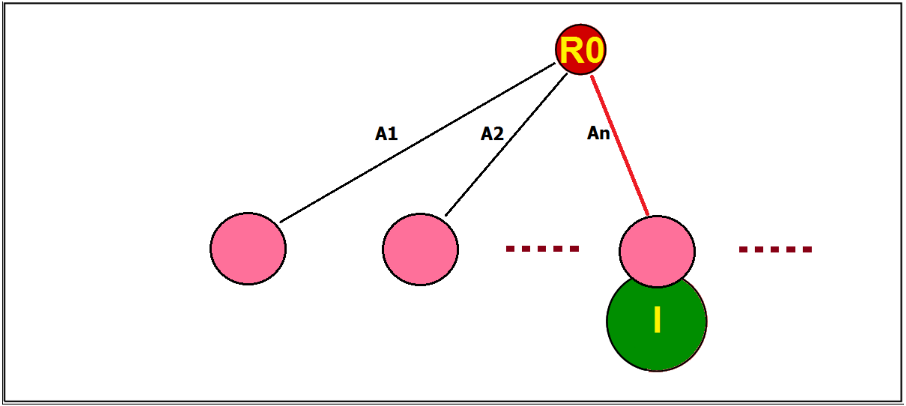
New Heuristics: Ordering the Actions



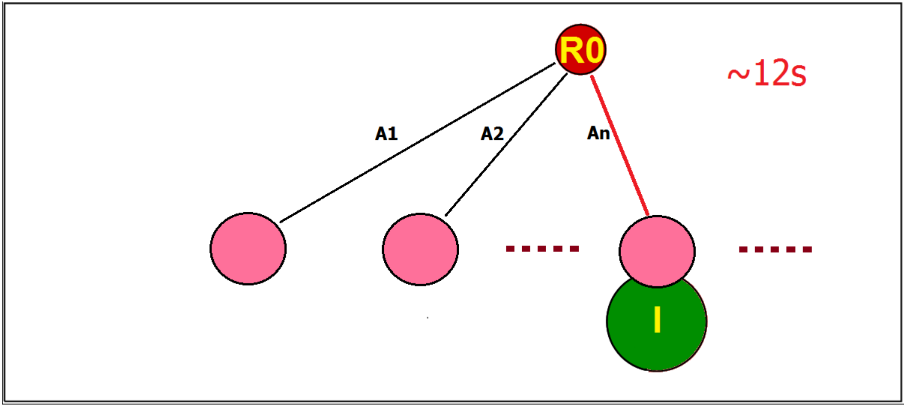
New Heuristics: Ordering the Actions



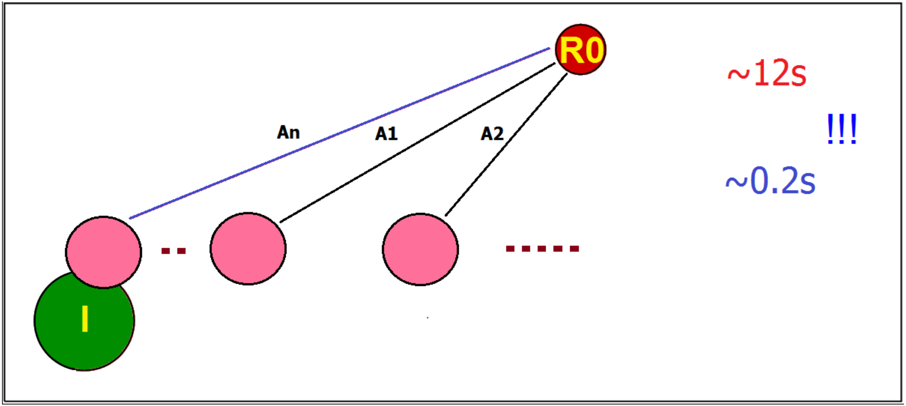
New Heuristics: Ordering the Actions



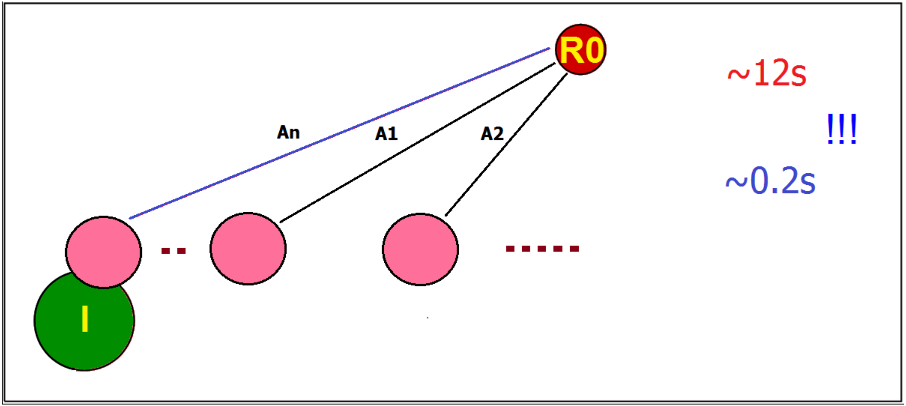
New Heuristics: Ordering the Actions



New Heuristics: Ordering the Actions



New Heuristics: Ordering the Actions



$\Rightarrow$ Ordering Actions in ψ

New Heuristics: Ordering the Actions

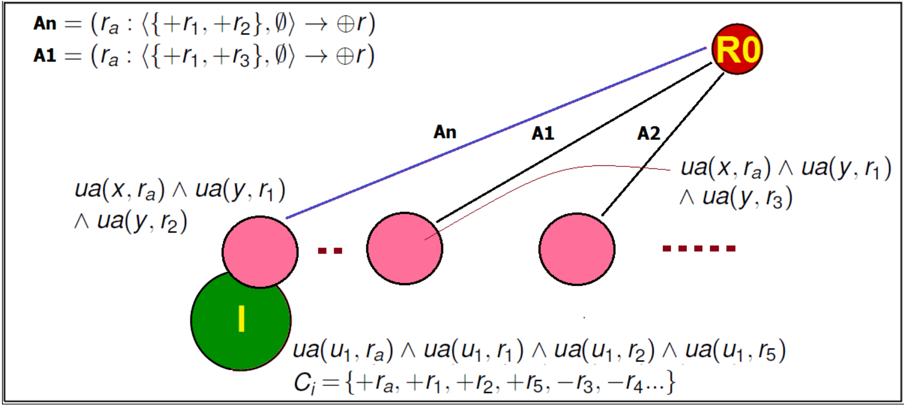
- Consider the “difference” between two sets of states
- Define

$$Diff(C_1, C_2) = (P_1 \setminus P_2) \cup (N_1 \setminus N_2)$$

  where
  $C_1 = P_1|N_1$, $C_2 = P_2|N_2$ are pre-conditions,
  P_1, P_2 (N_1, N_2) are sets of roles of the form $+r$ ($-r$, resp.)

- Example: let $C_1 = \{+r_1, +r_2| - r_4\}$ and $C_2 = \{+r_1, +r_3| - r_4, -r_2\}$
  - $Diff(C_1, C_2) = \{+r_2\}$

New Heuristics: Ordering the Actions



- Consider the "difference" between two sets of states
- Define

$$Diff(C_1, C_2) = (P_1 \setminus P_2) \cup (N_1 \setminus N_2)$$

where
$C_1 = P_1|N_1$, $C_2 = P_2|N_2$ are pre-conditions,
P_1, P_2 (N_1, N_2) are sets of roles of the form $+r$ ($-r$, resp.)

New Heuristics: Ordering the Actions

- Consider the "difference" between two sets of states
- Define

$$Diff(C_1, C_2) = (P_1 \setminus P_2) \cup (N_1 \setminus N_2)$$

  where
  $C_1 = P_1 | N_1$, $C_2 = P_2 | N_2$ are pre-conditions,
  P_1, P_2 (N_1, N_2) are sets of roles of the form $+r$ ($-r$, resp.)

- Example: let $C_1 = \{+r_1, +r_2 | -r_4\}$ and $C_2 = \{+r_1, +r_3 | -r_4, -r_2\}$
  - $Diff(C_1, C_2) = \{+r_2\}$

New Heuristics: Ordering the Actions

- Consider the "difference" between two sets of states
- Define

$$Diff(C_1, C_2) = (P_1 \setminus P_2) \cup (N_1 \setminus N_2)$$

  where
  $C_1 = P_1|N_1$, $C_2 = P_2|N_2$ are pre-conditions,
  P_1, P_2 (N_1, N_2) are sets of roles of the form $+r$ ($-r$, resp.)

- Example: let $C_1 = \{+r_1, +r_2| - r_4\}$ and $C_2 = \{+r_1, +r_3| - r_4, -r_2\}$
  - $Diff(C_1, C_2) = \{+r_2\}$

New Heuristics: Ordering the Actions

- Let ψ be the set of actions and C_i represent the "pre-condition" of initial states UA_0 (i.e., all roles in UA_0 are in C_i)

1. For each $\tau = (C_a : C \rightarrow \otimes r) \in \psi$:
   2. If $true \in C_a$ and $true \in C$:
      3. set τ be the highest order in ψ'
   4. Else:
      5. Calculate $Diff(C_a \cup C, C_i)$ for τ
6. Order the actions by cardinality of their Diff (from lower value to higher one)
   7. If $|Diff_{\tau_1}| = |Diff_{\tau 2}|$ where $\tau_1 = (C_{a1} : C_1 \rightarrow \otimes r_1)$ and $\tau_2 = (C_{a2} : C_2 \rightarrow \otimes r_2)$:
      8. τ_1 has higher order if $|C_{a1} \cup C_1| < |C_{a2} \cup C_2|$ and vice versa

New Heuristics: Ordering the Actions

- Let ψ be the set of actions and C_i represent the "pre-condition" of initial states UA_0 (i.e., all roles in UA_0 are in C_i)

1. For each $\tau = (C_a : C \rightarrow \otimes r) \in \psi$:
   2. If $true \in C_a$ and $true \in C$:
      3. set τ be the highest order in ψ'
   4. Else:
      5. Calculate $Diff(C_a \cup C, C_i)$ for τ
6. Order the actions by cardinality of their Diff (from lower value to higher one)
   7. If $|Diff_{\tau_1}| = |Diff_{\tau 2}|$ where $\tau_1 = (C_{a1} : C_1 \rightarrow \otimes r_1)$ and $\tau_2 = (C_{a2} : C_2 \rightarrow \otimes r_2)$:
      8. τ_1 has higher order if $|C_{a1} \cup C_1| < |C_{a2} \cup C_2|$ and vice versa

ASASP: Experimental Setting

- Experiments:
  - Data sets: 4 packages from MOHAWK, VAC, PMS
  - randomly generated test cases inspired by real case studies widely adopted by the community such as: a Hospital, a University, and an European Bank
- MOHAWK performs better than RBAC-PAT (RBAC-PAT does not scale up to handle these benchmarks)

ASASP: Experimental Setting

- Experiments:
  - Data sets: 4 packages from MOHAWK, VAC, PMS
  - randomly generated test cases inspired by real case studies widely adopted by the community such as: a Hospital, a University, and an European Bank
- MOHAWK performs better than RBAC-PAT (RBAC-PAT does not scale up to handle these benchmarks)

MOHAWK's Testcases: Separate Administration Assumption I

Test suite	# Roles ◇ #Rules	MOHAWK Time	VAC Time	VAC # Rules	PMS *Fwd* Time	PMS *Prll* Time	ASASP Time	ASASP # Rules
Test suite 1	3 ◇ 15	0.42	0.19	1	0.35	0.41	**0.09**	2
	5 ◇ 25	0.50	0.32	1	0.36	0.44	**0.11**	2
	20 ◇ 100	0.60	0.31	1	0.30	0.35	**0.10**	2
	40 ◇ 200	0.94	0.66	1	0.48	0.53	**0.32**	2
	200 ◇ 1000	2.65	0.91	1	0.44	0.52	**0.28**	2
	500 ◇ 2500	4.87	1.57	1	0.92	1.06	**0.73**	2
	4000 ◇ 20000	16.90	1.89	1	33.51	22.33	**1.24**	2
	20000 ◇ 80000	51.56	2.52	1	*TO*	*TO*	**1.17**	2
	30000 ◇ 120000	65.54	4.32	1	*TO*	*TO*	**1.68**	2
	40000 ◇ 200000	131.14	9.84	1	*TO*	*TO*	**2.25**	2

MOHAWK's Testcases: Separate Administration Assumption II

Test suite	# Roles ⋄ #Rules	MOHAWK	VAC		PMS Fwd	PMS Prll	ASASP	
		Time	Time	# Rules	Time	Time	Time	# Rules
Test suite 2	3 ⋄ 15	0.40	0.21	1	0.31	0.33	**0.12**	2
	5 ⋄ 25	0.50	0.29	1	0.35	0.38	**0.21**	2
	20 ⋄ 100	0.54	0.14	1	0.34	0.41	**0.10**	2
	40 ⋄ 200	1.21	0.51	1	0.57	0.54	**0.16**	2
	200 ⋄ 1000	2.54	0.73	1	0.49	0.61	**0.14**	2
	500 ⋄ 2500	5.02	1.02	1	1.14	0.73	**0.43**	2
	4000 ⋄ 20000	12.31	1.33	1	26.16	19.38	**1.08**	2
	20000 ⋄ 80000	24.42	4.75	1	*TO*	*TO*	**1.01**	2
	30000 ⋄ 120000	94.85	6.77	1	*TO*	*TO*	**1.09**	2
	40000 ⋄ 200000	140.89	9.89	1	*TO*	*TO*	**1.49**	2

MOHAWK's Testcases: Separate Administration Assumption III

Test suite	# Roles ◇ #Rules	MOHAWK	VAC		PMS		ASASP	
					Fwd	*Prll*		
		Time	Time	# Rules	Time	Time	Time	# Rules
Test suite 3	3 ◇ 15	0.41	0.12	1	0.32	0.39	**0.09**	2
	5 ◇ 25	0.49	0.17	1	0.50	0.43	**0.08**	2
	20 ◇ 100	0.77	0.21	1	0.36	0.42	**0.14**	2
	40 ◇ 200	0.87	0.57	1	0.38	0.47	**0.17**	2
	200 ◇ 1000	5.93	1.93	1	0.82	0.98	**0.51**	2
	500, 2500	3.78	0.93	1	0.64	0.86	**0.12**	2
	4000 ◇ 20000	14.05	4.01	1	18.43	13.29	**1.12**	2
	20000 ◇ 80000	30.29	3.56	1	*TO*	*TO*	**2.65**	2
	30000 ◇ 120000	109.16	9.13	1	*TO*	*TO*	**1.89**	2
	40000 ◇ 200000	154.12	9.92	1	*TO*	*TO*	**2.15**	2

VAC's Testcases: Separate Administration Assumption

Test case	# Roles ⋄ #Rules	MOHAWK	VAC		PMS		ASASP	
					Fwd	*Prll*		
		Time	Time	# Rules	Time	Time	Time	# Rules
Bank1	531 ⋄ 5126	*Err*	**0.36**	0	*TO*	*TO*	42.67	576
Bank2	531 ⋄ 5126	*Err*	**0.48**	0	*TO*	*TO*	48.81	584
Bank3	531 ⋄ 5126	*Err*	**0.76**	2	*TO*	*Err*	38.63	497
Bank4	531 ⋄ 5126	*Err*	**1.97**	5	*TO*	*TO*	5.71	566

VAC's Testcases: Non-Separate Administration Assumption

Test case	# Roles ⋄ # Rules	VAC		PMS		ASASP	
				Fwd	*Prll*		
		Time	# Rules	Time	Time	Time	# Rules
Hospital1	13 ⋄ 37	**0.06**	5	0.71	*Err*	1.02	15
Hospital2	13 ⋄ 37	**0.09**	5	0.87	$3m15.71$	1.14	13
Hospital3	13 ⋄ 37	**0.29**	2	0.85	0.49	0.42	4
Hospital4	13 ⋄ 37	**0.47**	4	0.62	0.26	2.47	12
University1	32 ⋄ 449	**0.09**	7	0.89	*Err*	1.91	17
University2	32 ⋄ 449	0.68	8	0.67	0.56	**0.48**	2
University3	32 ⋄ 449	**0.06**	5	*TO*	*Err*	8.15	40
University4	32 ⋄ 449	1.85	12	**0.62**	*TO*	2.19	18

PMS's Testcases: Non-Separate Administration Assumption

Test case	# Roles ◇ # Rules	VAC		PMS		ASASP	
				Fwd	*Prll*		
		Time	# Rules	Time	Time	Time	# Rules
Test 1	40 ◇ 487	16.06	3	0.63	**0.48**	1.31	2
Test 2	40 ◇ 450	0.19	0	0.67	0.45	**0.18**	0
Test 3	40 ◇ 462	8.12	3	0.52	0.53	**0.41**	2
Test 4	40 ◇ 446	7.81	3	0.55	42.38	**0.39**	2
Test 5	40 ◇ 480	45.37	47	0.95	**0.51**	2.31	9
Test 6	40 ◇ 479	25.63	13	0.75	**0.46**	1.79	4
Test 7	40 ◇ 467	$1m3.26$	101	3.72	2.16	**1.68**	2
Test 8	40 ◇ 484	$1m10.64$	65	4.18	$2m11.86$	**2.34**	8
Test 9	40 ◇ 463	$1m26.08$	89	4.92	$6m18.84$	**2.79**	11
Test 10	40 ◇ 481	27.14	38	**0.35**	0.53	2.65	5

Conclusion

- Security analysis of Access Control Policies
- ASASP: solve the user-role reachability problems for ARBAC policies.
- New heuristics for ASASP
  - Backward Useful Actions
  - Ordering Actions
- Current works:
  - Solve user-role reachability problems for Administrative Temporal RBAC policies
  - An incremental version of the approach
  - Proposed pre-processing role hierarchies strategies

Thank you for your attention!