Reasoning about Dynamic Policies

Riccardo Pucella (Northeastern University)

Joint work with:

Vicky Weissman (Cornell University)

Policies

A policy is a statement that says what is and what is not permitted

- Many different frameworks and logics for analyzing and reasoning about policies
  - First-order logics
  - Modal logics

The basic problem:

In situation s, is Joe permitted to perform action a?

Static versus Dynamic Setup

All the analysis frameworks assume an essentially static setup

- The environment can change
- The policies are fixed

Many cases of interest require reasoning about different policies in a single model

Example: Policy Change

Reasoning about a scenario that begins under one set of policies and completes under a modified version

- Initial policy: “A student cannot pass her thesis defense unless she has fulfilled her minor requirement”
- At some point, changed to: “A student cannot pass her preliminary exam unless she has fulfilled her minor requirement”
- Since passing the preliminary exam is a prerequisite to passing the thesis defense, the first policy can presumably be dropped

Example: Policy Change

Reasoning about a scenario that begins under one set of policies and completes under a modified version

- Initial policy: “A student cannot pass her thesis defense unless she has fulfilled her minor requirement”
- At some point, changed to: “A student cannot pass her preliminary exam unless she has fulfilled her minor requirement”
- Since passing the preliminary exam is a prerequisite to passing the thesis defense, the first policy can presumably be dropped

What happens if Alice passed her preliminary exam under the first policy, and defended under the second?

Example: Policy Comparison

Reasoning about different policies within the same system

- An interpretation of the principle of least privilege is: “An agent has only those permissions that are necessary to do her job”
- Can be written down, for a particular system, as a specific set of policies
- There may be easier or more efficient ways to capture this as a set of policies

How do you specify that two sets of policies are equivalent?

Our Approach

We want to understand the issues involved in reasoning about dynamically changing policies

Case study:

- Pick a framework for reasoning about (static) policies
- Extend to handle policy changes
- Understand the extension and generalize to other approaches

Our Approach

We take as a starting point a logic for reasoning about actions and permissions in state transition systems

- The Dynamic Logic of Permissions [Meyden 1998]
- Captures a number of cases of interest
- Lot of work on the underlying formalism
  - propositional dynamic logic (PDL)

System Model

General state-based system representation:

- Set of states
- Set of possible transitions between states

Permissions associated with transitions:

- A possible transition can be allowed (green)
- A possible transition can be forbidden (red)
- A policy set P is a set of green transitions
- By default, other possible transitions are red

Traces

A trace is a sequence of state transitions

Given a policy set P:

- A trace is P-green if all its transitions are in P
- A trace is P-red otherwise

The Dynamic Logic of Permissions

Introduced to reason about legal systems [Meyden 1998]

- PDL-based logic for reasoning about permissions in state transition systems

Syntax for actions:

- Primitive actions $a_1, a_2, \ldots$
- Close under sequencing $;$, choice $\cup$, and iteration *

We will associate sets of traces in the system with actions

Formulas

- Primitive propositions $\Phi_0 = \{p_1, p_2, \dots\}$
  - Primitive facts about the current state
- Boolean operators
- PDL operator $\langle\alpha\rangle\varphi$
- Permission operators $\mathsf{Perm}(\alpha)\varphi$, $\mathsf{FreePerm}(\alpha)\varphi$
  - $\mathsf{Perm}(\alpha)\varphi$: there is a green trace of α that leads to φ true
  - $\mathsf{FreePerm}(\alpha)\varphi$: all traces of α leading to φ true are green

Semantics

A Kripke structure $M = (S, \pi, \tau)$ is used to model the system:

- S: set of states
- π: truth value of primitive propositions at states of S
- τ: transitions corresponding to primitive actions

Also need a policy set P:

- Specifies which transitions are green

Truth of a Formula

We define the relation $(M, s, P) \models \varphi$

- Formula φ is true at state s of M (under policy P)

Standard rules from PDL:

$(M, s, P) \models p$ if $\pi(s)(p) = \mathbf{true}$

$(M, s, P) \models \varphi \wedge \psi$ if $(M, s, P) \models \varphi$, $(M, s, P) \models \psi$

$(M, s, P) \models \neg\varphi$ if $(M, s, P) \not\models \varphi$

$(M, s, P) \models \langle\alpha\rangle\varphi$ if $\exists\sigma \in \tau_s(\alpha)$, $(M, \sigma[f], P) \models \varphi$

where $\tau_s(\alpha)$ is the set of traces from s corresponding to action α

Truth of Permission Operators

Permission operators:

$(M, s, P) \models \mathsf{Perm}(\alpha)\varphi$
if $\exists \sigma \in \tau_s(\alpha)$, σ P-green, $(M, \sigma[f], P) \models \varphi$

$(M, s, P) \models \mathsf{FreePerm}(\alpha)\varphi$
if $\forall \sigma \in \tau_s(\alpha)$ s.t. $(M, \sigma[f], P) \models \varphi$, σ is P-green

A Logic for Dynamic Policies

We extend DLP to deal with changing policies

Identify sublanguage Φ_p of *propositional formulas*

- Set of primitive propositions in Φ_0 closed under negation and conjunction
- We use ρ to range over propositional formulas

New Operators

New operator $\mathsf{Grant}(\rho_1, \rho_2)\varphi$

- "If we assume all transitions from ρ_1 to ρ_2 are green, φ holds"

New operator $\mathsf{Revoke}(\rho_1, \rho_2)\varphi$

- "If we assume all transitions from ρ_1 to ρ_2 are red, φ holds"

Grant and Revoke take propositional formulas:

- Cannot grant a permission based on other permissions

Semantics

Same models as DLP:

- Kripke structure M, policy set P

Semantics of new operators:

$(M, s, P) \models \mathsf{Grant}(\rho_1, \rho_2)\varphi$ if $(M, s, P \cup P^{\rho_1,\rho_2}) \models \varphi$

$(M, s, P) \models \mathsf{Revoke}(\rho_1, \rho_2)\varphi$ if $(M, s, P \setminus P^{\rho_1,\rho_2}) \models \varphi$

where

$$P^{\rho_1,\rho_2} = \{s_1 s_2 \mid (M, s_1, P) \models \rho_1, (M, s_2, P) \models \rho_2\}$$

Example: Policy Change

Let $\Sigma = a_1 \cup \cdots \cup a_n$ be all possible actions

Let $complete$ be a formula true at a target bad state

Let NewPol(φ) be Grant$(\rho_1, \rho_1') \ldots$ Revoke$(\rho_k, \rho_k') \ldots \varphi$

- Change to the policy as a sequence of grants and revokes

Falling between the cracks as a result of policy change:

$$(M, s_{init}, P_{init}) \models \mathsf{Perm}(\Sigma^*)(\neg\mathsf{Perm}(\Sigma^*)complete \wedge \mathsf{NewPol}(\mathsf{Perm}(\Sigma^*)complete)$$

Example: Policy Comparison

Let $A = \{\alpha_1, \dots\}$ be actions with respect to which we want to establish equivalence of policies P_1 and P_2

Let $\mathsf{Pol}_i(\varphi)$ be $\mathsf{Grant}(\rho_{i,1}, \rho'_{i,1}) \dots \mathsf{Grant}(\rho_{i,k_i}, \rho'_{i,k_i})\varphi$

- Policies as sequences of grants

P_1 and P_2 are equivalent with respect to A and some initial state if for all $\alpha \in A$ and all φ,

$$(M, s_{init}, \varnothing) \models \mathsf{Pol}_1(\mathsf{Perm}(\alpha)\varphi) \Leftrightarrow \mathsf{Pol}_2(\mathsf{Perm}(\alpha)\varphi)$$

Axiomatization

Capture properties of the operators and their interaction

Five groups of axioms:

- Axioms for propositional reasoning
- Axioms for actions [Segerberg 1977]
- Axioms for permissions [Meyden 1998]
- Axioms for grants and revocations
- Axioms for interaction of grants and revocations

Theorem: Axiomatization is sound and complete with respect to Kripke structure with policy sets

Grants

Most axioms for Grant describe "set properties", e.g.

- $\mathsf{Grant}(\mathit{false}, \rho)\varphi \Leftrightarrow \varphi$
- $\mathsf{Grant}(\rho_1, \rho_2)\mathsf{Grant}(\rho_3, \rho_4)\varphi \Leftrightarrow$
  $\mathsf{Grant}(\rho_3, \rho_4)\mathsf{Grant}(\rho_1, \rho_2)\varphi$
- From $\rho_3 \Rightarrow \rho_1$ and $\rho_4 \Rightarrow \rho_2$ infer
  $\mathsf{Grant}(\rho_1, \rho_2)\mathsf{Grant}(\rho_3, \rho_4)\varphi \Leftrightarrow \mathsf{Grant}(\rho_1, \rho_2)\varphi$

Grants

Other axioms describe interaction with operators, e.g.

- $\mathsf{Grant}(\rho_1, \rho_2)(\varphi \wedge \psi) \Leftrightarrow$
  $\mathsf{Grant}(\rho_1, \rho_2)\varphi \wedge \mathsf{Grant}(\rho_1, \rho_2)\psi$
- $\mathsf{Grant}(\rho_1, \rho_2)\mathsf{Perm}(\alpha)\varphi \Leftrightarrow$
  $\mathsf{Grant}(\rho_1, \rho_2)\mathsf{Perm}(\alpha)\mathsf{Grant}(\rho_1, \rho_2)\varphi$

Most interesting axiom:

- $\mathsf{Grant}(\rho_1, \rho_2)(\rho_1 \wedge \langle a \rangle \rho_2) \Rightarrow \mathsf{Grant}(\rho_1, \rho_2)(\mathsf{Perm}(a)\rho_2)$
  for primitive actions a

Revocations

Axioms for revocations are similar to axioms for grants

- Same “set properties”
- Same interaction with operators

Only exception, the interesting axiom:

- $\mathsf{Revoke}(\rho_1, \rho_2)(\rho_1 \wedge [a]\rho_2) \Rightarrow$
  $\mathsf{Revoke}(\rho_1, \rho_2)(\neg\mathsf{Perm}(a)\rho_2)$
  for primitive actions a

Interaction of Grants and Revocations

- $\mathsf{Grant}(\rho_1, \rho_2)\mathsf{Revoke}(\rho_3, \rho_4)\varphi \Leftrightarrow$
  $\mathsf{Revoke}(\rho_3, \rho_4)\mathsf{Grant}(\rho_1, \rho_2 \wedge \neg\rho_4)$
  $\mathsf{Grant}(\rho_1 \wedge \neg\rho_3, \rho_2)\varphi$
  - Granting some permissions P_1 and then revoking other permissions P_2 is equivalent to first revoking the permissions P_2, and then granting the permissions in P_1 that would not have been revoked by P_2

Consequence: if $\rho_1 \Rightarrow \rho_3$ and $\rho_2 \Rightarrow \rho_4$ are tautologies, then $\mathsf{Grant}(\rho_1, \rho_2)\mathsf{Revoke}(\rho_3, \rho_4)\varphi$ is equivalent to $\mathsf{Revoke}(\rho_3, \rho_4)\varphi$

Complexity

How difficult is it to reason in this logic?

Model-checking:

- Deciding $(M, s) \models \varphi$ can be done in time polynomial in $|\varphi| + |M| + |P|$

Validity checking:

- Deciding whether formula φ is true in every state of every model
  - Lower bound EXPTIME
  - Upper bound NEXPTIME
  - Conjecture: problem is EXPTIME-complete

Summary

- We showed how to extend a logic for reasoning about static policies with the ability to deal with dynamic policy changes
- We derive a sound and complete axiomatization capturing the interaction of granting and revoking permissions
- Despite the expressiveness, the decision problem is decidable
  - We conjecture no harder than PDL