Strong and Weak Policy Relations

Michael J. May
Dept of Info Science Engineering
Kinneret College on the Sea of Galilee
DN Emek Hayarden 15132, Israel
mjmay@kinneret.ac.il

Carl A. Gunter
Dept of Computer Science
UIUC
Urbana, IL, 61791, USA
cgunter@cs.illinois.edu

Insup Lee, Steve Zdancewic
Dept of Computer and Info Science
University of Pennsylvania
Philadelphia, PA, 19104, USA
{lee,stevez}@cis.upenn.edu

***Abstract*—Access control and privacy policy relations tend to focus on decision outcomes and are very sensitive to defined terms and state. Small changes or updates to a policy language or vocabulary may make two similar policies incomparable. To address this we develop two flexible policy relations derived from bisimulation in process calculi. *Strong licensing* compares the outcome of two policies strictly, similar to strong bisimulation. *Weak licensing* compares the outcome of policies more flexibly by ignoring irrelevant (non-conflicting) differences between outcomes, similar to weak bisimulation. We illustrate the relations using examples from P3P.**

***Index Terms*—policy analysis; privacy policies;**

I. Introduction

The growing complexity of access control policies has led to the development of many policy comparison metrics and tools to aid developers and authors in creating their desired policies. Such metrics take advantage of the allow/forbid nature of access control decisions, enabling them to perform state space exploration of the decisions reached by different policies. Even so, metrics are generally language specific and sensitive to small changes in the terms or underlying vocabulary. In particular, comparing policies with even slightly differing representations of information or considering policy equivalence based on the performance of multiple actions under a policy is difficult with existing tools and techniques.

To that end, we offer policy comparison metrics which are more general and flexible than those based on decision tree based structures or straightforward comparison of policy rules. We introduce flexibility in comparison by adapting concepts from the process calculus literature, borrowing from the notions of strong and weak bisimulation and applying them to the comparison of privacy and access control policies. We call the policy relations we devise *strong* and *weak* licensing since they roughly parallel notions from strong and weak bisimulation respectively.

The rest of this paper is organized as follows. Section II develops the fundamentals of strong and weak licensing and the relations that we derive from them. Section III mentions related work and section IV concludes.[1]

II. Strong and Weak Licensing

Our goal in defining the licensing relations is to enable the evaluation of policy comparison of the form "Is there a way that the policy allows a person to at least achieve the desired outcome in such and such circumstance?" instead of "Does the policy permit/forbid the performance of an action in such and such circumstance?" The former question is subtly different from the latter in that it focusses on the outcomes that the policy allows and compares them against an ideal outcome. If one of the actual outcomes matches the ideal, we conclude that the policy permits, or *licenses* the action under the circumstances. Licensing is a state and transition based comparison metric since it requires an initial state and inspects the reachable states following transitions permitted by the policy.

Focussing on the output behavior of policies enables flexible relations since it lets us abstract away the internal workings of policies and obligations. Furthermore, by relaxing the comparison of output states we can design policy relations which match the intuition as to whether a policy "allows a person to at least achieve" some action(s). As a guide for designing flexible output based relations we draw lessons from the process calculus literature's formulation of bisimulation, the notion that two processes behave similarly from a particular initial state.

An action is *strongly licensed* ($\models$) by a policy if the policy contains a rule which enables the achievement of its precise outcome. An action is *weakly licensed* ($\models^*$) by a policy if the policy contains some rule or series of rules which enables the achievement of the action's outcome modified by some additional actions unrelated to the original action. To maintain generality in developing our relations, we do not specify precisely what kinds of actions are "unrelated" and may be ignored. Such actions are determined on a policy by policy basis.

We use the following variable conventions and families of relations to analyze policies. A policy $\phi = \{e_1, e_2, \ldots\}$ is a set of paragraphs or sentences ("rules") (e) which offer permitted combinations of actions, rights are stored in the state s, and a list of parameters is provided to make a decision from the policy g. State representations are finite and updateable via actions by agents as governed by the policy. Thus, when an actor performs the actions of e with parameters g, it transforms s to produce some resulting state s' where the effects of e have been performed. We denote such a transition $s \xrightarrow{e(g)} s'$. We use bar ($\overline{e}$) for variable series and ϕ^* for the set of all possible

[1]An extended technical report version of this work with more complete examples and proofs is available [10].

ordered series of rules using the rules in ϕ. The empty state (*i.e.,* with no rights) is denoted $s_\emptyset$.

A. Bisimulation

As a background for readers unfamiliar with process calculus relations we provide a brief overview of strong and weak bisimulation. Since an in depth discussion of bisimulation is beyond the scope of this work, we focus only on the aspects which we apply to privacy policy relations in this work. For more in depth study of bisimulation, we refer the reader to the many books and papers on the subject.

Strong and weak bisimulation are run time relations in that they refer to the behavior of processes from an initial state through transitions (a trace) until either a final or a "stuck" state is reached. Simply, for two processes p and q and a series of transitions $e_1, e_2, \ldots$, if we can show that $p \xrightarrow{e_1} p_1 \xrightarrow{e_2} p_2 \ldots$ and $q \xrightarrow{e_1} q_1 \xrightarrow{e_2} q_2 \ldots$, then the two processes are strongly bisimilar. For each step taken by p, q can take the same step. Weak bisimulation is a relaxation of strong bisimulation to allow for the execution of invisible τ operations. For instance, for the traces $p \xrightarrow{e_1} p_1 \xrightarrow{\tau} p_2 \xrightarrow{\tau} p_3 \xrightarrow{e_2} p_4 \ldots$ and $q \xrightarrow{e_1} q_1 \xrightarrow{e_2} q_2 \ldots$, p and q are weakly bisimilar. A τ operation is non-observable to outsiders and so is not considered in the comparison.

B. P3P Example

For this work we use a running example in P3P. P3P enables web sites to publish their privacy practices in a standardized XML format. The most important elements in P3P policy are the "Statement" elements which include the following child elements:

- Data-Group (D): the data covered
- Non-identifiable (i): promises that no identifiable information will be collected
- Purpose (P): set of purposes for which data is collected
- Recipient (R): Indicates who may receive the data collected
- Retention (T): Indicates how long the data will be kept

Comparing two P3P statements e_1 and e_2 requires a data category by data category examination of the policies. A full policy ϕ may contain several statements. Often a simple statement level comparison between two policies is not possible for a few reasons. For instance, since the "Purpose" element is a set of purposes, if e_1 and e_2's "Purpose" elements are not a superset one of the other, the comparison will fail. The retention and recipient elements also contain confounding elements. See Agrawal, *et al.* [1] for a fuller discussion.A trace based comparison mechanism such as the licensing relations may be more fruitful.

We define an action in P3P as the collection of a data item d. Let i be a boolean which is true iff "Non-identifiable" is included in the statement. Then, for a website's statement $e = (D, i, P, R, T)$, a successful collection of data item $d \in D$ under e results in the addition of a 4-tuple (d, P, R, T) to the website's rights (stored in state s) if d is non-identifiable or if d is identifiable and i is false. The 4-tuple means that the website

```
A
1 <DATA-GROUP><DATA REF="#USER.NAME.GIVEN" />
    </DATA-GROUP>
2 <PURPOSE><CONTACT/><TAILORING/></PURPOSE>
3 <RECIPIENT><OURS/></RECIPIENT>
4 <RETENTION><BUSINESS-PRACTICES/></RETENTION>
B
1 <DATA-GROUP><DATA REF="#USER.NAME.GIVEN" />
    </DATA-GROUP>
2 <PURPOSE><TAILORING/><PSEUDO-ANALYSIS/>
    </PURPOSE>
3 <RECIPIENT><OURS/></RECIPIENT>
4 <RETENTION><NO-RETENTION/></RETENTION>
```

Fig. 1. Example P3P policy snippets

has the right to use d for purposes P, disclose it to entities implied by R, and retain it for a time frame implied by T. A website's rights is a collection of 4-tuples $s = \{(d, P, R, T)\}$ for the rights the website has over each data item d.

A state transition represents the collection of d under e: $s_1 \xrightarrow{e(d,b)} s_2$ where b is a boolean indicating whether d is non-identifiable. If the transition is permitted (*i.e.,* $(d \in D) \wedge (b \vee (\neg b \wedge e.i))$) then $s_2 = s_1 \cup (d, e.P, e.R, e.T)$.

We develop $\models$ and $\models^*$ for P3P using the example statement snippets in Figure 1. Note A and B are not comparable using simple syntactical comparison since neither "Purpose" element is a subset of the other.

C. Strong Licensing

First, let us consider a simple case of strong licensing: when a policy enables the effects of e from just an initial state s_1 with parameters g. Then, ϕ *strongly licenses* e at s_1 with g, denoted $\phi \models_{(s_1,g)} e$. Let s_1, s_2 be states:

Definition 1: $\phi \models_{(s_1,g)} e$ iff $s_1 \xrightarrow{e(g)} s_2 \implies \exists e' \in \phi \;.\; s_1 \xrightarrow{e'(g)} s_2$ □

Adapting the relations from a single e to a series $\overline{e} = \{e_1, e_2, \ldots\}$, if ϕ enables the effects of performing the actions in $\overline{e}$ in order from s_1 with a parameters list $\overline{g} = \{g_1, g_2, \ldots\}$ ($|\overline{e}| = |\overline{g}|$) with a series of rules in ϕ of the same length, ϕ *strongly licenses* $\overline{e}$ at s_1 with $\overline{g}$. Let $s_i, 1 \leq i \leq |\overline{e}| + 1$ be the state such that $s_i \xrightarrow{e_i(g_i)} s_{i+1}$:

Definition 2: $\phi \models_{(s_1,\overline{g})} \overline{e}$ iff for $i = 1..|\overline{e}|$, $s_i \xrightarrow{e_i(g_i)} s_{i+1} \implies \exists e' \in \phi \;.\; s_i \xrightarrow{e'(g_i)} s_{i+1}$. □

Although $\models_{(s,\overline{g})}$ parameterizes over the infinite set ϕ^* and may not be decidable in general, it is decidable so long as $\overline{e}$ finite since we need to perform a maximum of $|\phi|$ operations for each $e \in \overline{e}$. Intuitively, strong licensing corresponds to a policy precisely enabling some action(s). It also restricts the relationship between the policy and the rule(s) to be "in lockstep" meaning that for e or each $e \in \overline{e}$, the policy has one rule which enables precisely performing e's behavior with the same parameters.

The complexity of evaluating $\models$ depends on several policy and state dependent variables. Let us denote the complexity of comparing two states as $|S|$. Let us denote the worst case complexity for executing any given rule (*i.e.,* performing all of its checks and state updates) as $rt(e)$. Let $|\phi|$ denote the number of rules in ϕ.

A naive algorithm for evaluating $\phi_1 \models_{(s_1,g_1)} e_1$ is:

1) Evaluate $s_1 \xrightarrow{e_1(g_1)} s_t$.
2) For each $e_i \in \phi_1$:
    a) Evaluate $s_1 \xrightarrow{e_i(g_1)} s_i$
    b) If $s_i = s_t$ Then Quit;

The worst case complexity for the above algorithm is $rt(e) + (|\phi| \times (rt(e) + |S|))$ since we perform one initial evaluation followed by a maximum of $|\phi|$ evaluations and comparisons. For $\overline{e}$, the complexity is multiplied by the length of $\overline{e}$. Pre-filtering rules from ϕ which clearly do not satisfy the transition $s_1 \xrightarrow{e_1(g_1)} s_t$ may reduce the complexity by reducing the number of evaluations and state comparisons in the best case.

Example 1: ($\models$ for P3P)

Alice is willing to disclose her name to a website if the site will only acquire rights to use it for contacting her and tailoring her home page, use it internally, and retain it for as long as is common in the industry. Then, since a name may be identifiable, $g_1 =$("Alice", true), Alice's target addition to the website's rights is $s_1 =$("Alice", {CONTACT, TAILORING}, {OURS}, {BUSINESS-PRACTICES}). Let e_1 be a rule such that $s_\emptyset \xrightarrow{e_1(g_1)} s_1$ Then, $A \models_{(s_\emptyset,g_1)} e_1$ since A's policy precisely grants the given rights. However, $B \not\models_{(s_\emptyset,g_1)} e_1$ since B grants the rights ("Alice", {TAILORING, PSEUDO-ANALYSIS}, {OURS}, {NO-RETENTION}).

Bob is willing to disclose his name for the same reasons as Alice in addition to the right to use the name for anonymous analysis of usage (pseudo-analysis). Then $g_2 =$ ("Bob", true) and $s_2 =$("Bob", {CONTACT, TAILORING, PSEUDO-ANALYSIS}, {OURS}, {BUSINESS-PRACTICES}). Let e_2 be a rule such that $s_\emptyset \xrightarrow{e_2(g_2)} s_2$. As before, $B \not\models_{(s_\emptyset,g_2)} e_2$, but also $A \not\models_{(s_\emptyset,g_2)} e_2$ since A doesn't permit reaching precisely s_2.□

D. Non-conflicting Rules

The above example motivates a more relaxed relation for policies which perform the actions of a rule with some slight modifications. We call the relation noconflict. The intuition for noconflict is that a policy enables an approximation of the actions of a rule or rule series. We mean to define a function:

$$\mathsf{noconflict}_{(s,g)}(e_2, e_1)$$

which is read, "The effects of performing the actions of e_2 at s with g do not conflict with the effects of e_1 under the same conditions". The relation will necessarily be policy or policy language specific since it requires semantic analysis of the state. We offer an example definition for P3P below. The relation need not be reflexive, so $\mathsf{noconflict}_{(s,g)}(e_2, e_1) \not\Rightarrow \mathsf{noconflict}_{(s,g)}(e_1, e_2)$.

We may adapt noconflict to allow a policy to approximate the effects of a rule with a series of rules. Let $\overline{e}$ be a rule series. Let s_1 be the state such that $s \xrightarrow{e(g)} s_1$ and s_2 be the state such that $s \xrightarrow{\overline{e}(g)} s_2$. We restrict the function to a single argument list for the entire series $\overline{e}$ to restrict comparison to similar cases. The function for series is then nearly identical to the one for single rules:

$$\mathsf{noconflict}_{(s,g)}(\overline{e}, e)$$

Applying the function to comparing the effects of series of rules is straightforward.

Example 2: (noconflict for P3P)

A simple definition for noconflict under P3P would be $\mathsf{noconflict}_{(s,g)}(e_2, e_1)$ if $(e_2.d = e_1.d) \wedge (e_2.P \subseteq e_1.P) \wedge (e_2.R \subseteq e_1.R) \wedge (e_2.T \subseteq e_1.T)$. This misses the hierarchical nature of some P3P elements, however. For instance, the retention term NO-RETENTION clearly permits less than BUSINESS-PRACTICES or INDEFINITE. The definition of a partial order over P3P policies is beyond the scope of this work (but see Hayati, *et al.* [6]), so we simply write $t_1 \Rightarrow t_2$ if t_1 is semantically more restrictive than t_2. Generalizing for sets,

$$T_1 \Rightarrow T_2 \text{ if } \forall t_1 \in T_1, \forall t_2 \in T_2, t_1 \Rightarrow t_2$$

Then $\mathsf{noconflict}_{(s_\emptyset,g)}(e_2, e_1)$ if:

$$(e_2.d = e_1.d) \wedge (e_2.P \subseteq e_1.P \vee e_2.P \Rightarrow e_1.P) \wedge (e_2.R \subseteq e_1.R \vee e_2.R \Rightarrow e_1.R) \wedge (e_2.T \subseteq e_1.T \vee e_2.T \Rightarrow e_1.T)$$

□

E. Weak Licensing

Using noconflict we define weak licensing as a more flexible policy relation than $\models$. We define weak licensing in terms of a policy ϕ weakly licensing a rule but the relation can be easily adapted to the case of one rule weakly licensing another.

If ϕ approximates the effects of e at s_1 with g with a rule series which is not conflicting with e, ϕ *weakly licenses* it at s_1 with g, $\phi \models^*_{(s_1,g)} e$. For s_1 we would like to write:

$$\phi \models^*_{(s_1,g)} e \text{ iff } \exists \overline{e} \in \phi^* \,.\, \mathsf{noconflict}_{(s_1,g)}(\overline{e}, e).$$

The problem is that depending on the definition of noconflict and the way state is represented, it may be undecidable since ϕ^* is unbounded. Specific languages and representations may either be decidable or reach a fixed point from given states or parameters, properties which can be evaluated with a model checker in a straightforward manner. To maintain generality and decidability, we instead restrict $\models^*$ to the power set of rules ($pwr(\phi)$):

Definition 3: $\phi \models^*_{(s_1,g)} e$ iff $\exists \overline{e} \in pwr(\phi)$. $\mathsf{noconflict}_{(s_1,g)}(\overline{e}, e)$ □

Adapting $\models^*$ to series of rules is straightforward:

Definition 4: $\phi \models^*_{(s_1,\overline{g})} \overline{e}$ iff $\exists \overline{e_2} \in pwr(\phi)$. $\mathsf{noconflict}_{(s_1,\overline{g})}(\overline{e_2}, \overline{e})$. □

Note that $\models \subseteq \models^*$. The intuition for the limitation is that in deciding whether an action is permitted it is sufficient to try all possible rules once. This imposes the (reasonable) assumption on ϕ that rights are not enabled by repeated performance of the same action, or more specifically: $\forall \overline{e_1} \in \phi^*, \forall s, \forall \overline{g}, s \xrightarrow{\overline{e_1}(\overline{g})} s' \implies \exists \overline{e_2} \in pwr(\phi) \ . \ s \xrightarrow{\overline{e_2}(\overline{g})} s'$.

Weak licensing intuitively means that an action is permitted by a policy. A policy weakly licenses a series $\overline{e}$ when it contains a series of rules which enables an outcome state which does not conflict with the outcome of $\overline{e}$. We do not look at the intermediate states reached by $\overline{e_2}$, only restricting that they use the same parameters list to ensure that the comparison is justified. Since we do not restrict the length of $\overline{g}$, the series do not need to be the same length.

The complexity of evaluating $\models^*$ depends on the complexity of evaluating $\mathsf{noconflict}$, denoted $|nc|$, which may be policy dependent, in addition to the variables defined above. Let $nc(s_1, s_2)$ denote the evaluation of $\mathsf{noconflict}$ between two states. Since $s_1 \xrightarrow{e_1(g_1)} s_t$ may be weakly licensed by a series of rules in ϕ, the naive algorithm for $\phi_1 \models^*_{(s_1,g_1)} e_1$ is:

1) Evaluate $s_1 \xrightarrow{e_1(g_1)} s_t$.
2) For each $\{e_i, e_{i+1}, \ldots, e_{i+n}\} \in pwr(\phi_1)$:
   a) Evaluate $s_1 \xrightarrow{e_i(g_1)} s_i \xrightarrow{e_{i+1}(g_1)} \ldots \xrightarrow{e_{i+n}(g_1)} s_n$
   b) If $nc(s_n, s_t)$ Then Quit;

The worst case complexity for the above algorithm is $rt(e) + (|pwr(\phi)| \times ((|\phi| \times rt(e)) + |nc|))$ since we perform one initial evaluation followed by a maximum of $|pwr(\phi)|$ evaluations. Each evaluation involves evaluating up to $|\phi|$ steps followed by a single check of $\mathsf{noconflict}$. Evaluating $\phi \models^*_{(s,\overline{g})} \overline{e_1}$ requires $|\overline{e}| \times rt(e)$ more running time since only one evaluation of $\mathsf{noconflict}$ is needed. As with $\models$, preselecting likely rules in ϕ can reduce the best case complexity.

Example 3: ($\models^*$ for P3P)

Using the definition of $\mathsf{noconflict}$ in Example 2, $\models^*$ for P3P is as follows. Let Alice and Bob have preferences and let $e_1, g_1, s_1, e_2, g_2, s_2$ be as in Example 1. Since $A \models_{(s_\emptyset,g_1)} e_1$, trivially $A \models^*_{(s_\emptyset,g_1)} e_1$. For Bob's preference, $\mathsf{noconflict}_{(s_\emptyset,g_2)}(A, e_2)$ since $e_2.P$ is a subset of A's purposes, so $A \models^*_{(s_\emptyset,g_2)} e_2$.

For B, $B \not\models^*_{(s_\emptyset,g_1)} e_1$ since B permits the purpose "pseudo-analysis" which e_1 does not contain and $\mathsf{noconflict}$ does not hold for Alice. For Bob's, since "no-retention" is more restrictive than "business-practices": $\mathsf{noconflict}_{(s_\emptyset,g_2)}(B, e_2)$ and therefore $B \models^*_{(s_\emptyset,g_2)} e_2$. □

III. Related Work

There have been many policy comparison metrics proposed for and applied to access control policies. Fisler, *et al.* [4] present Margrave, a framework for policy comparison and change impact analysis Margrave detects changes in the decision tree of an XACML [11] access control policy. LeMay, *et al.* [7] present PolicyMorph, a tool for composing, comparing, and analyzing attribute based access control policies. Like Margrave, it enables exploration of changes in the decision tree caused by policy changes and provides user feedback and suggestions. Lin, *et al.* [8] propose a filtering mechanism for finding access control policies with similar decisions. The decision relations derived in such work is a special case of strong licensing where the outcome of rules is the allow/forbid decision determined by the policy.

Policy languages such as EPAL [2] and XACML are amenable to comparison using $\models$ and $\models^*$ as well. Since both languages allow policies to define their own custom vocabularies, user input indicating which obligations, purposes, and other custom elements are non-conflicting is required to enable comparison using $\mathsf{noconflict}$ and $\models^*$. An efficient algorithm to compare EPAL policies in shown by Backes, *et al.* [3]. Access control policies in the structure proposed by Harrison, *et al.* [5] are directly comparable using $\models$ since the policies include operational descriptions of rule outcomes. May [9] applies the licensing relations discussed here to a custom language designed for modeling legal privacy policies.

IV. Conclusion

Strong and weak licensing are policy comparison relations derived from applying some concepts of bisimulation from process calculi. The relations are more flexible than other policy comparison measures in that they enable comparison while ignoring irrelevant actions.

Strong licensing compares policies strictly, requiring that every action permitted one policy be permitted by another. The comparison is performed by running the policy on an underlying state and examining differences between the outcomes of the two.

Weak licensing compares policies more flexibly, akin to weak bisimulation where invisible τ transitions may be ignored. A policy weakly licenses an action if it permits the outcome of the action, possibly with some additional, irrelevant obligations.

Acknowledgment

This research was supported in part by ONR MURI N00014-07-1-0907, NSF CCR02-08996, and a grant from Kinneret College.

References

[1] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. An XPath-based preference language for P3P. In *WWW '03*, 2003.

[2] P. Ashley, S. Hada, G. Karjoth, and M. Schunter. E-P3P privacy policies and privacy authorization. In *WPES '02*, 2002.

[3] M. Backes, G. Karjoth, W. Bagga, and M. Schunter. Efficient comparison of enterprise privacy policies. In *SAC '04*, pages 375–382, 2004.

[4] K. Fisler, S. Krishnamurthi, L. Meyerovich, and M. Tschantz. Verification and change impact analysis of access-control policies. In *Inter. Conf. on Software Eng. (ICSE)*, May 2005.

[5] M. A. Harrison, W. L. Ruzzo, and J. D. Ullman. Protection in operating systems. *Commun. ACM*, 19(8):461–471, 1976.

[6] K. Hayati and M. Abadi. Language-based enforcement of privacy policies. In *Privacy Enhancing Technologies (PET)*, May 2004.

[7] M. LeMay, O. Fatemieh, and C. A. Gunter. Policymorph: interactive policy transformations for a logical attribute-based access control framework. In *SACMAT '07*, 2007.

[8] D. Lin, P. Rao, E. Bertino, and J. Lobo. An approach to evaluate policy similarity. In *SACMAT '07*, 2007.

[9] M. J. May. *Privacy APIs: Formal Models For Analyzing Legal Privacy Requirements*. PhD thesis, University of Pennsylvania, Philadelphia, 2008.

[10] M.J. May, C. Gunter, I. Lee, and S. Zdancewic. Strong and weak policy relations. Tech Report MS-CIS-09-10, U. of Pennsylvania, 2009.

[11] T. Moses. eXtensible Access Control Markup Language (XACML). Standard Version 2.0, OASIS, February 2005.