1-1-2009

Strong and Weak Policy Relations

Michael J. May
Kinneret College on the Sea of Galilee

Carl A. Gunter
University of Illinois at Urbana-Champaign

Insup Lee
University of Pennsylvania, lee@cis.upenn.edu

Stephan A. Zdancewic
University of Pennsylvania, stevez@cis.upenn.edu

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-09-10.

Strong and Weak Policy Relations

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 and EPAL.

Keywords
policy analysis, privacy policies

Comments
University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-09-10.

Strong and Weak Policy Relations*

Michael J. May Carl A. Gunter Insup Lee Steve Zdancewic

Technical Report (MS-CIS-09-10)
University of Pennsylvania

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 and EPAL.

Keywords: policy analysis; privacy policies;

1 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.

We develop the licensing relations at a high level, independent of any specific policy language or representation. Since the relations are state based, however, for comparison we require some state representation over which policies operate. To concretize and show the usefulness of the licensing relations we apply them to W3C's Platform for Privacy Preferences (P3P) [13] language and the Enterprise Policy Authorization Language (EPAL) [2], privacy policy languages which have been widely deployed and studied.

The rest of this paper is organized as follows. Section 2 develops the fundamentals of strong and weak licensing and the relations that we derive from them. We introduce a case study using licensing to examine sample privacy policies in P3P and EPAL in section 3. Section 4 mentions related work and section 5 concludes.

*Michael J. May (mjmay@kinneret.ac.il), Kinneret College on the Sea of Galilee, DN Emek Hayarden 15132, Israel. Carl A. Gunter (cgunter@cs.illinois.edu), UIUC, Urbana, IL 61791, USA. Insup Lee and Steve Zdancewic ({lee, stevez}@cis.upenn.edu), University of Pennsylvania, Philadelphia, PA 19104. This is an extended version of a paper in the proceedings of the 2009 IEEE International Symposium on Policies for Distributed Systems and Networks (POLICY 2009). *Last update: August 9, 2009*

2 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. As an example application of the relations we develop in Sections 2.4 and 3 we shows examples of "unrelated" actions in P3P and an EPAL policy. The examples give guidance for what kinds of actions may likely be ignored in other languages and policy representations.

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

2.1 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.

2.2 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:

```
A                                        B
<data-group base="">                     <data-group base="">
  <data ref="#user.name.given" />          <data ref="#user.name.given" />
</data-group>                            </data-group>
<purpose>                                <purpose>
  <contact /> <tailoring />                <tailoring /> <pseudo-analysis />
</purpose>                               </purpose>
<recipient><ours /></recipient>          <recipient><ours /></recipient>
<retention>                              <retention>
  <business-practices />                   <no-retention />
</retention>                             </retention>
```

Figure 1: Example P3P policy snippets

- 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 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.

2.3 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 $s_1 \xrightarrow{e(g)} s_2 \implies \exists e' \in \phi \;.\; s_1 \xrightarrow{e'(g)} s_2$ □

Generalizing for all possible parameters at s_1:

$$\phi \models_{(s_1)} e \text{ iff } \forall g, \phi \models_{(s_1,g)} e$$

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 $\overline{e}$ is 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 naïve 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:

$$s_1 = (\text{"Alice"}, \{\textsc{Contact}, \textsc{Tailoring}\}, \{\textsc{ours}\}, \{\textsc{business-practices}\}).$$

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:

$$s_2 = (\text{"Bob"}, \{\textsc{Contact}, \textsc{Tailoring}, \textsc{Pseudo-analysis}\}, \{\textsc{ours}\}, \{\textsc{business-practices}\}).$$

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. □

2.4 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:

$$\mathsf{noconflict}_{(s,g)}(e_2, e_1) \text{ if } (e_2.d = e_1.d) \wedge (e_2.P \subseteq e_1.P) \wedge (e_2.R \subseteq e_1.R) \wedge (e_2.T \subseteq e_1.T).$$

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.* [7]), 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:

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

□

2.5 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 precisely:

$$\forall \overline{e_1} \in \phi^*, \forall s, \forall \overline{g}, s \xrightarrow{\overline{e_1}(\overline{g})} s' \implies \exists \overline{e_2} \in pwr(\phi) \ . \ s \xrightarrow{\overline{e_2}(\overline{g})} s'$$

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 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 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 naïve 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 noconflict. Evaluating $\phi \models^*_{(s,\overline{g})} \overline{e_1}$ requires $|\overline{e}| \times rt(e)$ more running time since only one evaluation of 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 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 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 $B \models^*_{(s_\emptyset, g_2)} e_2$. □

3 Application

Strong and weak licensing enable more flexible comparisons between different versions of policies, including versions which use slightly dissimilar vocabularies. In this section we show two similar policies written in different languages and show how they can be compared using strong and weak licensing. Doing so requires us to define noconflict for the respective languages as well as a unified notion of state.

The EPAL privacy policy language is more generic than P3P and is meant for use in the composition of enterprise policies. Each policy has two parts: the *vocabulary* which defines terms such as users, data, purposes, actions, and obligations and the *rules* which are rulings (allow or forbid) based on the combination of terms in the vocabulary. The two languages also differ in that EPAL policies are meant to be enforced inside a company while P3P is meant to be a policy summary aimed at consumers. Despite differences in the policy format, we may compare two similarly structured policies P3P and EPAL policies using licensing.

Comparing an EPAL and P3P policy requires the definition of appropriate purpose elements, action elements (for disclosure), and retention obligations in the EPAL policy vocabulary. The parameters to an EPAL policy are a 4-tuple $g = (u, d, p, a)$ where u is the role of the user, d is the category of data, p is the purpose of the action, and a is a description of the action. We may represent the rights approved by an EPAL policy as a set of 5-tuples $s = \{(u, d, p, a, O)\}$ where u, d, p, a are as above and O is a list of obligations associated with the action. See Ashley, *et al.* [3] for the details of EPAL rule evaluation. Since P3P does not differentiate roles at a company, we drop the u term when comparing to P3P, $s = (d, p, a, O)$ if any user has the given rights.

We compare two policies in a location based services scenario. The policies define obligations and rules for some aspects of a wireless mapping service company. The device of a subscriber to the mapping service sends a stream of location information to the mapping service while the device is powered on. The policies describe the following business practices:

1. It allows messages and customized maps to be sent to a user
2. It allows transfer of location information about the user provided that
   a. only city-level accurate information is given and
   b. the company receiving the data has a privacy policy similar to the provider company
3. It allows location information to be stored by the company for 24 hours at most
4. The provider must give customers full access to all data stored about them.

Writing the policy in P3P involves the use of additional P3P elements not discussed above (`Access`) as well as the definition of location objects in both languages. Due to space considerations we use only policy snippets below and post the full policy documents in appendices[1]. For ease of reference we show the argument lists (g_1, g_2, g_3, g_4) and states $(s_1, s_2, s_3, s_4, s_5)$ used in the following example in Figure 2 instead of in the text.

[1] They are available online as well in full: P3P: [`www2.kinneret.ac.il/mjmay/policies/P3P.xml`], EPAL vocabulary: [`www2.kinneret.ac.il/mjmay/policies/EPALVocab.xml`], EPAL policy: [`www2.kinneret.ac.il/mjmay/policies/EPALPol.xml`]

g_1 =(“Alice”, “Boise”, {}, “Transfer to ABC Co.”)
g_2 =(“Boise”, false)
g_3 =(“Alice”, “Boise”, {}, “Transfer to ABC Co.”, false)
g_4 =(“Bob”, “Boise”, “Internal”, “Store”, false)
g_5 =(“Claire”, “Boise”, “Services”, “SendContent”, false)

s_1 =(“Boise”, {}, “Transfer”, ChkOtherPolicy)
s_2 =(“Boise”, {Tailoring}, {ours, same}, {business-practices})
s_3 = (“Boise”, {}, “Transfer”, “ChkOtherPolicy”)
s_4 = s_3∪(“Boise”, “Internal”, “Store”, “24HrRetain”)
s_5 = s_4∪(“Boise”, “Services”, “SendContent”)

Figure 2: Arguments and states in example

Example 4 (Transfer) Let us consider aspects 2a, 2b, and 3 of the above policy. The P3P snippet which mentions those aspects is as follows. For space considerations we show only two lines of the full data-group element.

```
<data-group base="">
 <DATA ref="#location.civil.street"/>
 <DATA ref="#location.civil.city"/>
</data-group>
<purpose>
 <Tailoring/>
</purpose>
<recipient>
 <ours/><same/>
</recipient>
<retention>
 <business-practices/>
</retention>
```

The parallel EPAL policy rules are:

```
<rule id="Transfer" ruling="allow">
 <user-category refid="Manager"/>
 <data-category refid="Location"/>
 <purpose refid="Root"/>
 <action refid="Transfer"/>
 <condition refid="CityOnly"/>
 <obligation refid="ChkOtherPolicy"/>
</rule>
<rule id="24HrRetain" ruling="allow">
 <user-category refid="Worker"/>
 <data-category refid="Location"/>
 <purpose refid="Internal"/>
 <action refid="Store"/>
 <obligation refid="24HourRetain"/>
</rule>
<rule id="DeliverData" ruling="allow">
 <user-category refid="Root"/>
 <data-category refid="Location"/>
```

```
 <purpose refid="Services"/>
 <action refid="SendContent"/>
</rule>
```

Let us denote the above policies as ϕ_P and ϕ_E respectively. Policy ϕ_P has only a single statement, so there is only one rule e_P. Policy ϕ_E contains three rules e_{E1}, e_{E2}, and e_{E3} respectively. The EPAL policy includes custom user, data category, purpose, condition, action, and obligation elements as defined in the vocabulary and policy file. The intention of each term is largely intuitive. The purpose "Root" refers to *all possible purposes*. The condition "CityOnly" checks that the data provided is only at city level accuracy. The obligation "ChkOtherPolicy" requires the mapping company to verify that the privacy policy of the data recipient is compatible with its own, an obligation indicated by the tag "same" in P3P's "Recipient" element.

Consider the action generated by the arguments g_1 in Figure 2 to the rights of the mapping company. Provided that Alice has the role Manager, it is permitted by e_{E1} since the information is at city accuracy. The result is $s_\emptyset \xrightarrow{e_{E1}(g_1)} s_1$ as shown in Figure 2. The obligation "ChkOtherPolicy" is satisfied once Alice has verified that ABC Co.'s privacy policy is satisfactory.

Comparing the action to the P3P policy, we must first adapt the parameters to the P3P format, yielding g_2 since the city alone is unlikely to be identifiable information. The result is $s_\emptyset \xrightarrow{e_P(g_2)} s_2$.

Defining noconflict between ϕ_E and ϕ_P requires some policy specific definitions. First, we note the parallel between the (custom) EPAL action "Transfer" and the P3P element "Recipient." The imposition of the "ChkOtherPolicy" obligation in the EPAL policy is similar to the obligation imposed by "same" in P3P. Second, we note the parallel between the (custom) EPAL action "Store" and the P3P element "Retention." The obligation "24HourRetain" requires that the mapping company delete the information after 24 hours which is the company policy (and therefore equivalent to "business-practices" in the "Retention"). Third, the inclusion of the element "ours" in the P3P policy means the mapping company may itself use the data collected, a condition satisfied by the presence of other rules in the EPAL policy (shown in the online policy). Fourth, the P3P element "tailoring" permits customization of user data, a service reflected in e_{E3}.

With the above observations, we first define a unified parameter format $g = (u, d, p, a, i)$ which includes all of the parameters for ϕ_E and ϕ_P. The parameter list is then g_3. We may show that $\mathsf{noconflict}_{(s_\emptyset, g_3)}(s_2, s_1)$ since s_2 permits the actions of s_1 while granting other rights and imposing obligations unrelated to the information transfer. We may therefore conclude $\phi_P \models^*_{(s_\emptyset, g_3)} s_2$. This matches the intuitive conclusion that following ϕ_P is effectively the same as following ϕ_E.

It is not the case that $\mathsf{noconflict}_{(s_\emptyset, g_3)}(s_1, s_2)$, however, since s_1 permits less than s_2. However, using g_4 (where Bob is a Worker) and g_5 (where Claire is an employee), ϕ_E permits:

$$s_\emptyset \xrightarrow{e_{E1}(g_3)} s_3 \xrightarrow{e_{E2}(g_4)} s_4 \xrightarrow{e_{E3}(g_5)} s_5$$

By the argument above:

$$\mathsf{noconflict}_{(s_\emptyset, \{g_3, g_4, g_5\})}(\{e_{E1}, e_{E2}, e_{E3}\}, e_P)$$

and therefore $\phi_E \models^*_{(s_\emptyset, \{g_3, g_4, g_5\})} e_P$. □

4 Related Work

There have been many policy comparison metrics proposed for and applied to access control policies. Fisler, *et al.* [5] present Margrave, a framework for policy comparison and change impact analysis Margrave detects changes in the decision tree of an XACML [12] access control policy. LeMay, *et al.* [9] 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.* [10] 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 [3] 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 noconflict and $\models^*$. An efficient algorithm to compare EPAL policies in shown by Backes, *et al.* [4]. Karjoth, *et al.* [8] offers a technique to support an EPAL-P3P policy relationship similar to the one described here through automatic translation.

Access control policies in the structure proposed by Harrison, *et al.* [6] are directly comparable using $\models$ since the policies include operational descriptions of rule outcomes. May [11] applies the licensing relations discussed here to a custom language designed for modeling legal privacy policies.

5 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.

The relations can be easily adapted for use in many policy languages provided that a suitable definition of non-conflicting obligations is provided. For languages which do not have a fixed set of atomic obligations, the relations need to be customized for each policy.

Acknowledgment

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

A P3P Sample Policy

```
<?xml version="1.0"?>
<!--P3P Policy for POLICY2009
Meant to model a policy that a location based services provider would offer its customers.
Compare to EPAL Policy.-->

<policies xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.w3.org/2002/01/p3pv1"
      xmlns:p3p="http://www.w3.org/2002/01/P3Pv1">

<!--Data schema for location based service-->

<DATASCHEMA xmlns="http://www.w3.org/2002/01/P3Pv1">

<datastruct name="location.datetime" short-description="Date/Time observed"
   structref="http://www.w3.org/TR/P3P/base#date">
</datastruct>
```

```
<datastruct name="location.gps.latitude" short-description="Latitude">
<categories><location/></categories>
</datastruct>

<datastruct name="location.gps.longitude" short-description="Longitude">
<categories><location/></categories>
</datastruct>

<datastruct name="location.civil.roomnumber" short-description="Room Number">
<categories><location/></categories>
</datastruct>

<datastruct name="location.civil.building" short-description="Building name">
<categories><location/></categories>
</datastruct>

<datastruct name="location.civil.street" short-description="Street address"
   structref="http://www.w3.org/TR/P3P/base#postal.street">
<categories><location/></categories>
</datastruct>

<datastruct name="location.civil.city" short-description="City"
   structref="http://www.w3.org/TR/P3P/base#postal.city">
<categories><location/></categories>
</datastruct>

<datastruct name="location.civil.county" short-description="County">
<categories><location/></categories>
</datastruct>

<datastruct name="location.civil.postalcode" short-description="Postal code"
    structref="http://www.w3.org/TR/P3P/base#postal.postalcode">
<categories><location/></categories>
</datastruct>

<datastruct name="location.civil.stateprov" short-description="State/Province"
   structref="http://www.w3.org/TR/P3P/base#postal.stateprov">
<categories><location/></categories>
</datastruct>

<datastruct name="location.civil.country" short-description="Country"
   structref="http://www.w3.org/TR/P3P/base#postal.country">
<categories><location/></categories>
</datastruct>

</DATASCHEMA>

<policy discuri="http://www.example.com/English-privacy.txt" name="AdLoc">

<entity>
<data-group>
```

```
<DATA ref="#business.name">The Testing Company</DATA>
<DATA ref="#business.contact-info.postal.street">123 Testing Avenue</DATA>
<DATA ref="#business.contact-info.postal.city">Test</DATA>
<DATA ref="#business.contact-info.postal.stateprov">PA</DATA>
<DATA ref="#business.contact-info.postal.country">USA</DATA>
<DATA ref="#business.contact-info.postal.postalcode">10048</DATA>
<DATA ref="#business.organization">Testing Company</DATA>
<DATA ref="#business.contact-info.online.uri">http://www.adloc.example.com</DATA>
</data-group>
</entity>

<access>
<all/>
</access>

<statement>
<consequence>This policy defines obligations and rules for some aspects
   of a location based service.
(1) It allows messages to be sent to a user but requires the user to be paid five
      cents for any advertising message
(2) It allows transfer of location information about the user provided that the
      information isn't more accurate than city information and that company
      receiving the data has a privacy policy that is compliant with some standard.
(3) It allows location information to be stored by the company for 24 hours at most.
</consequence>

<purpose>
<Contact/>
<Tailoring/>
<Pseudo-analysis/>
<Pseudo-decision/>
</purpose>

<recipient>
<ours/><same/>
</recipient>

<retention>
<business-practices/>
</retention>

<data-group base="">
<DATA ref="#user.name.given"/>
<DATA ref="#user.name.prefix"/>
<DATA ref="#user.name.family"/>
<DATA ref="#user.login.id"/>

<DATA ref="#location.datetime"/>
<DATA ref="#location.gps.latitude"/>
<DATA ref="#location.gps.longitude"/>
<DATA ref="#location.civil.roomnumber"/>
```

```
<DATA ref="#location.civil.building"/>
<DATA ref="#location.civil.street"/>
<DATA ref="#location.civil.city"/>
<DATA ref="#location.civil.county"/>
<DATA ref="#location.civil.postalcode"/>
<DATA ref="#location.civil.stateprov"/>
<DATA ref="#location.civil.country"/>
</data-group>

</statement>

</policy>
</policies>
```

B EPAL Sample Vocabulary

```
<?xml version="1.0"?>

<!-- Vocabulary in the EPAL language aimed at location based services.
Compare with AdlocP3P.xml policy.-->

<!--Vocabulary for Location Based Services - Edited 2 March 2009-->
<epal-vocabulary version="1.2" xmlns="http://www.research.ibm.com/privacy/epal"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.research.ibm.com/privacy/epal epal.xsd
                     http://www.w3.org/2001/XMLSchema xs-dummy.xsd ">

  <vocabulary-information id="AdLocVocab">
    <short-description language="en">LBS Vocabulary</short-description>
    <long-description language="en">This is a vocabulary in EPAL that
    will be used for describing things a location based services provider
    would need.</long-description>
    <issuer>
      <name>LBS Company</name>
      <organization>Testing Company</organization>
      <e-mail>mjmay@kinneret.ac.il</e-mail>
      <address>123 Testing Avenue</address>
      <country>USA</country>
    </issuer>
    <location>http://www.lbs.example.com</location>
    <version-info end-date="2010-07-23T12:00:00"
      last-modified="2004-07-23T15:17:00"
      revision-number="" start-date="2004-07-23T15:17:00" test="true"/>
  </vocabulary-information>

  <!--User categories for the policy-->
  <user-category id="Worker">
    <short-description language="en">Worker</short-description>
    <long-description language="en">Worker user type with
       limited powers</long-description>
```

```
</user-category>

<user-category id="Manager">
  <short-description language="en">Manager</short-description>
  <long-description language="en">Manager with more powers</long-description>
</user-category>

<!--Data categories-->
<data-category id="Location">
  <short-description language="en">Location information</short-description>
  <long-description language="en">Data about the location of a
     customer</long-description>
</data-category>

<!--Purposes-->
<purpose id="Advertising" parent="Contact">
  <short-description language="en">Advertising</short-description>
  <long-description language="en">Includes all forms of targeted
     advertising</long-description>
</purpose>

<purpose id="Billing" parent="Contact">
   <short-description language="en">Billing</short-description>
   <long-description language="en">Contact with a customer for the purposes
      of billing</long-description>
</purpose>

<purpose id="Services" parent="Contact">
   <short-description language="en">Services</short-description>
   <long-description language="en">Used for providing services to the
      customer as detailed in customer agreement, (i.e. Local Maps,
      Messaging)</long-description>
</purpose>

<purpose id="Contact" parent="Root">
  <short-description language="en">Contact</short-description>
  <long-description language="en">Parent for all methods of actively contacting
     the consumer</long-description>
</purpose>

<purpose id="CustomerService" parent="Internal">
  <short-description language="en">Customer Service</short-description>
  <long-description language="en">Use of data to provide customer service
     to the user</long-description>
</purpose>

<purpose id="Internal" parent="Root">
  <short-description language="en">Internal usage</short-description>
  <long-description language="en">Use of the data for purely internal
     functions</long-description>
</purpose>
```

```
  <purpose id="Root">
    <short-description language="en">Parent of all purposes.  This purpose
    includes all other purposes  and therefore is only allowed if the data
    may be used for everything.</short-description>
  </purpose>

  <!--Actions-->
  <action id="SendMessage">
    <short-description language="en">Send a message</short-description>
    <long-description language="en">Send a message to the user</long-description>
  </action>

  <action id="SendContent">
     <short-description language="en">Send content</short-description>
     <long-description language="en">Send content to the user as per the
        service contract</long-description>
  </action>

  <action id="Store">
    <short-description language="en">Store information</short-description>
    <long-description language="en">Store the information about the user in
       the database</long-description>
  </action>

  <action id="Transfer">
    <short-description language="en">Transfer information</short-description>
    <long-description language="en">Transfer a user's location information to
       a third party source</long-description>
  </action>

  <action id="SendMap">
    <short-description language="en">Send a map</short-description>
    <long-description language="en">Send a map to the user based on the location
       generated from a device</long-description>
  </action>

<!--Containers for data-->
<container id="LocationContainer">
 <short-description language="en">Location of a user</short-description>
 <long-description language="en">Location information about a user, including
  userID and time of day</long-description>

 <attribute id="LocUserID" maxOccurs="1" minOccurs="1"
  simpleType="http://www.w3.org/2001/XMLSchema#string" auditable="true">
 <short-description language="en">The userID of the subject</short-description>
 </attribute>

 <attribute id="lRoomNum" maxOccurs="1" minOccurs="0"
  simpleType="http://www.w3.org/2001/XMLSchema#string" auditable="true">
 <short-description language="en">The room number of the
```

```
 subject</short-description>
</attribute>

<attribute id="lBuilding" maxOccurs="1" minOccurs="0"
 simpleType="http://www.w3.org/2001/XMLSchema#string" auditable="true">
<short-description language="en">The building name or number</short-description>
</attribute>

<attribute id="lAddress" maxOccurs="1" minOccurs="0"
 simpleType="http://www.w3.org/2001/XMLSchema#string" auditable="true">
<short-description language="en">The subject's current street address</short-description>
</attribute>

<attribute id="lPostalCode" maxOccurs="1" minOccurs="0"
 simpleType="http://www.w3.org/2001/XMLSchema#string" auditable="true">
<short-description language="en">The subject's current
 postal code</short-description>
</attribute>

<attribute id="lCity" maxOccurs="1" minOccurs="0"
 simpleType="http://www.w3.org/2001/XMLSchema#string" auditable="true">
<short-description language="en">The subject's current city</short-description>
</attribute>

<attribute id="lStateProv" maxOccurs="1" minOccurs="0"
 simpleType="http://www.w3.org/2001/XMLSchema#string" auditable="true">
<short-description language="en">The subject's current state
 or province</short-description>
</attribute>

<attribute id="lCounty" maxOccurs="1" minOccurs="0"
 simpleType="http://www.w3.org/2001/XMLSchema#string" auditable="true">
<short-description language="en">The subject's current county or
 township</short-description>
</attribute>

<attribute id="lCountry" maxOccurs="1" minOccurs="0"
 simpleType="http://www.w3.org/2001/XMLSchema#string" auditable="true">
<short-description language="en">The subject's current country</short-description>
</attribute>

<attribute id="lCreation" maxOccurs="1" minOccurs="1"
 simpleType="http://www.w3.org/2001/XMLSchema#dateTime">
<short-description language="en">The date and time of the location
 data's creation</short-description>
</attribute>

<attribute id="lLatitude" maxOccurs="1" minOccurs="1"
 simpleType="http://www.w3.org/2001/XMLSchema#string" auditable="true">
<short-description language="en">The latitude of the subject</short-description>
</attribute>
```

```
 <attribute id="lLongitude" maxOccurs="1" minOccurs="1"
  simpleType="http://www.w3.org/2001/XMLSchema#string" auditable="true">
 <short-description language="en">The longitude of the subject</short-description>
 </attribute>
</container>

<container id="UserInfo">
   <short-description language="en">Customer information</short-description>
   <long-description language="en">Full customer information available</long-description>

   <attribute id="UserID" maxOccurs="1" minOccurs="1"
      simpleType="http://www.w3.org/2001/XMLSchema#string" auditable="true">
   <short-description language="en">The UserID subject</short-description>
   </attribute>

   <attribute id="FullName" maxOccurs="1" minOccurs="1"
      simpleType="http://www.w3.org/2001/XMLSchema#string" auditable="true">
   <short-description language="en">The full name of the subject</short-description>
   </attribute>

   <attribute id="DeviceInfo" maxOccurs="1" minOccurs="1"
      simpleType="http://www.w3.org/2001/XMLSchema#string" auditable="true">
   <short-description language="en">Information about the mobile device
      of the user</short-description>
   </attribute>
</container>

<!--Obligations-->
<obligation id="24HourRetain">
   <short-description language="en">Retain the data for only 24 hours</short-description>
   <long-description language="en">Data may be retained for 24 hours after
   collection, but must be deleted afterwards</long-description>
</obligation>

<obligation id="GrantAccesss">
   <short-description language="en">Customer must be allowed full access
   to the data as stored.</short-description>
</obligation>

<obligation id="GetConsent">
   <short-description language="en">Customer can give specific informed
   consent before this action may be done</short-description>
</obligation>

<obligation id="CheckOutsidePolicy">
   <short-description language="en">Check that the policies of an outside
   company for compliance with our level of privacy rules</short-description>
</obligation>
</epal-vocabulary>
```

C EPAL Sample Policy

```
<?xml version="1.0"?>

<!--Policy in the EPAL language aimed at location based services.
Compare with P3P.xml policy.-->

<!--EPAL policy for location based services example-->
<epal-policy default-ruling="deny"
 version="1.2" xmlns="http://www.research.ibm.com/privacy/epal"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.research.ibm.com/privacy/epal epal.xsd
 http://www.w3.org/2001/XMLSchema xs-dummy.xsd ">
  <policy-information id="AdLocPolicy">
   <short-description language="en">Policy controlling messages sent, location
   based services rendered, retention, and transfer of location data
   </short-description>
   <long-description language="en">This policy defines obligations and rules
   for some aspects of a location based service.
   (1) It allows messages to be sent to a user as well as customized maps
   (2) It allows transfer of location information about the user provided that
       (A) only city-level accurate information is given and
       (B) that the company receiving the data has a privacy policy that is
       compliant with the provider company
   (3) It allows location information to be stored by the company for 24 hours
       at most
   (4) It obligates the provider to give customers full access to all data
       stored about them.
   </long-description>

   <issuer>
     <name>LBS Company</name>
     <organization>The Testing Company</organization>
     <e-mail>mjmay@kinnneret.ac.il</e-mail>
     <address>123 Testing Avenue</address>
     <country>USA</country>
   </issuer>

   <location>http://www.lbs.example.com</location>
   <version-info end-date="2010-07-26T12:00:00"
      last-modified="2004-07-26T12:19:00"
      start-date="2004-07-26T12:19:00" test="true"/>
  </policy-information>

  <epal-vocabulary-ref id="AdLocEPALVocab"
     location="http://www2.kinneret.ac.il/mjmay/policies/EPALVocab.xml"/>

  <!--Conditions.-->
  <condition id="CityOnly">
   <short-description language="en">The location information must only be of
   city level granularity.  That means that the information for all fields of
```

```
finer grain than city must be left blank</short-description>
<predicate refid="http://www.research.ibm.com/privacy/epal#and">
<!--Room number field is blank-->
<predicate refid="http://www.research.ibm.com/privacy/epal#string-equal">
 <function refid="http://www.research.ibm.com/privacy/epal#string-bag-to-value">
  <attribute-reference container-refid="LocationContainer"
     attribute-refid="RoomNum"/>
 </function>
  <attribute-value simpleType="http://www.w3.org/2001/XMLSchema#string">""
  </attribute-value>
</predicate>

<!--Building field is blank-->
 <predicate refid="http://www.research.ibm.com/privacy/epal#string-equal">
  <function refid="http://www.research.ibm.com/privacy/epal#string-bag-to-value">
   <attribute-reference container-refid="LocationContainer"
      attribute-refid="Building"/>
  </function>
   <attribute-value simpleType="http://www.w3.org/2001/XMLSchema#string">""
   </attribute-value>
</predicate>

 <!--Address field is blank-->
  <predicate refid="http://www.research.ibm.com/privacy/epal#string-equal">
   <function refid="http://www.research.ibm.com/privacy/epal#string-bag-to-value">
    <attribute-reference container-refid="LocationContainer"
       attribute-refid="Address"/>
   </function>
    <attribute-value simpleType="http://www.w3.org/2001/XMLSchema#string">""
    </attribute-value>
  </predicate>

 <!--Postal code field is blank-->
  <predicate refid="http://www.research.ibm.com/privacy/epal#string-equal">
   <function refid="http://www.research.ibm.com/privacy/epal#string-bag-to-value">
    <attribute-reference container-refid="LocationContainer"
       attribute-refid="PostalCode"/>
   </function>
    <attribute-value simpleType="http://www.w3.org/2001/XMLSchema#string">""
    </attribute-value>
  </predicate>

  <!--Latitude field is blank-->
   <predicate refid="http://www.research.ibm.com/privacy/epal#string-equal">
    <function refid="http://www.research.ibm.com/privacy/epal#string-bag-to-value">
     <attribute-reference container-refid="LocationContainer"
        attribute-refid="Latitude"/>
    </function>
     <attribute-value simpleType="http://www.w3.org/2001/XMLSchema#string">""
     </attribute-value>
  </predicate>
```

```
   <!--Longitude field is blank-->
    <predicate refid="http://www.research.ibm.com/privacy/epal#string-equal">
     <function refid="http://www.research.ibm.com/privacy/epal#string-bag-to-value">
      <attribute-reference container-refid="LocationContainer"
         attribute-refid="Longitude"/>
     </function>
      <attribute-value simpleType="http://www.w3.org/2001/XMLSchema#string">""
      </attribute-value>
   </predicate>
  </predicate>
</condition>

<!--Rules-->
<!--Allows messages and maps to be sent to the subjects as per the contract-->
<rule id="DeliverData" ruling="allow">
 <short-description language="en">Deliver content and data to the
    user</short-description>
 <long-description language="en">Allows the delivery of content and data
    to the user as per the contract</long-description>
 <user-category refid="Root"/>
 <data-category refid="Location"/>
 <purpose refid="Services"/>
 <action refid="SendContent"/>
</rule>

<!--Allows advertising messages to be sent to the subjects if the user
   has given consent-->
<rule id="SendAd" ruling="allow">
 <short-description language="en">Send an advertising message</short-description>
 <long-description language="en">Allows the sending of advertising
 messages if the user has opted in</long-description>
 <user-category refid="Root"/>
 <data-category refid="Location"/>
 <purpose refid="Advertising"/>
 <action refid="SendMessage"/>
 <obligation refid="GetConsent"/>
</rule>

<!--Allows customer service messages to be sent by the workers without obligation-->
<rule id="SendCustomerService" ruling="allow">
 <short-description language="en">Send customer service notice</short-description>
 <long-description language="en">Allows the sending of customer service
    information</long-description>
 <user-category refid="Worker"/>
 <data-category refid="Location"/>
 <purpose refid="CustomerService"/>
 <action refid="SendMessage"/>
</rule>

<!--Allows the transfer of location information to outside parties provided that
```

```
      only city level information is provided-->
  <rule id="Transfer" ruling="allow">
   <short-description language="en">Transfer location information</short-description>
   <long-description language="en">Allows the transfer of location
      information by a manager only if the data is reduced to city-level
      accuracy and the receiving company has a policy that has been
      checked</long-description>
      <user-category refid="Manager"/>
      <data-category refid="Location"/>
      <purpose refid="Root"/>
      <action refid="Transfer"/>
      <condition refid="CityOnly"/>
      <obligation refid="ChkOtherPolicy"/>
  </rule>

 <!--Allows data collection so long as the subject is granted access to it-->
 <rule id="GrantAccess" ruling="allow">
    <short-description language="en">Collection only if access
       granted</short-description>
    <long-description language="en">Allows the collection of data only if
        the subject is allowed access to it</long-description>
    <user-category refid="Worker" />
    <data-category refid="Location" />
    <purpose refid="root" />
    <action refid="Store "/>
    <obligation refid="GrantAccess" />
 </rule>

 <!--Allows data storage for only 24 hours-->
 <rule id="24HrRetain" ruling="allow">
  <short-description language="en">Retention for only 24 hours</short-description>
  <long-description language="en">Allows the retention of data only for
     24 hours, after which data must be destroyed</long-description>
  <user-category refid="Worker"/>
  <data-category refid="Location"/>
  <purpose refid="Internal"/>
  <action refid="Store"/>
  <obligation refid="24HourRetain"/>
 </rule>
</epal-policy>
```

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, C. Powers, and M. Schunter. Enterprise Privacy Authorization Langugae (EPAL 1.2). W3C Member Submission, Nov 2003.

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

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

[5] 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.

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

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

[8] G. Karjoth, M. Schunter, and E. Van Herreweghen. Translating privacy practices into privacy promises–how to promise what you can keep. In *Fourth IEEE Inter. Workshop on Policies for Distributed Systems and Networks*, 2003.

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

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

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

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

[13] W3C. The Platform for Privacy Preferences 1.1 (P3P1.1). Working draft, World Wide Web Consortium, Feb 2006.