Expressing and Reasoning about Conflicting Norms in Cybersecurity: Poster

Jiaming Jiang, Nirav Ajmeri, Rada Y. Chirkova, Jon Doyle, Munindar P. Singh
Department of Computer Science
North Carolina State University
Raleigh, NC, United States

ABSTRACT

Secure collaboration requires the collaborating parties to apply the right policies for their interaction. We adopt a notion of conditional, directed norms as a way to capture the standards of correctness for a collaboration. How can we handle conflicting norms? We describe an approach based on knowledge of what norm dominates what norm in what situation. Our approach adapts answer-set programming to compute stable sets of norms with respect to their computed conflicts and dominance. It assesses agent compliance with respect to those stable sets. We demonstrate our approach on a healthcare scenario.

CCS Concepts

•**Computing methodologies** → **Multi-agent systems;**

Keywords

Normative System, Dominance Relation, Norm

1. INTRODUCTION

We address the hard cybersecurity problem of achieving secure collaboration. Specifically, we focus on the aspect of expressing and reasoning about user requirements by providing a high-level modeling language along with decision procedures and a tool for reasoning about expressions in that language.

Secure collaboration is often framed in terms of policies to be applied by the various collaborators. We step behind these decision-making policies to understand the standards of correctness by which the policies (and resulting behaviors) of the participants can be judged. Specifically, we adopt the idea of social norms from Singh [2] as a way to characterize the expectations that autonomous parties have of one another in an implementation-independent manner. Such norms may be elicited from stakeholders when creating a new system.

A particular problem we deal with is that the norms may conflict with each other in that it would not be possible for all of them to be satisfied on the same enactment of the system. However, quite often, we have some clarity on which norm may be more important than which norm in which situation. Given such knowledge, it may be possible to find solutions (that is, possible enactments of the system) that do not violate a norm unless that norm itself is dominated (in the situation where it is violated) by another norm and that second norm is satisfied. In effect, in this broader sense of compliance, an enactment may satisfy a hierarchy of norms without satisfying each of the members of that hierarchy.

The relevant previous security approaches concern access control. The newer approaches support exception handling in access control. For example, Marinovic et al.'s [1] approach handles cases where a user overrides access control based on urgency. In such a case the user becomes subject to new restrictions, such as additional monitoring or logging. Our approach is about whether the user should override access control.

HotSoS '16 April 19-21, 2016, Pittsburgh, PA, USA

ACM ISBN 978-1-4503-4277-3/16/04.

DOI: http://dx.doi.org/10.1145/2898375.2898395

2. APPROACH

Accordingly, we propose an approach for formalizing norms and dominance relations between them. This approach enables expressing requirements for secure collaboration and provides a formal basis for (1) evaluating compliance of the interactions of the autonomous parties with respect to those requirements and for (2) determining if the requirements are mutually consistent. We illustrate the approach via the following example from healthcare.

EXAMPLE 1. *Alice, a child, suddenly takes ill. Society expects Alice's guardian, Bob, to take her immediately to an emergency room. However, after Bob takes Alice to an emergency room, the attending physician, Dave, does not have access to Alice's medical records, which is held by Alice's pediatrician, Carol. By privacy laws, Carol cannot share Alice's certain protected health information without Bob's consent. But as this is an emergency situation, Carol shares Alice's records with Dave for treatment.* ■

2.1 Principals and Norms

We adopt the following background concepts from Singh [2]. A *principal* is an autonomous entity able to participate in normative relationships. A principal may be an *individual* or an *organization*. We consider the activity of principals within some interval $\mathcal{T}$ of times. Each organization defines the *roles* that principals take in the organization and the organizational *norms* that create directed relationships between two principals, called the *subject* and *object* of the norm. The organization forms the *context* of the norms it establishes. In Example 1, the principals are the five named individuals, and the two organizations: society and the hospital. The roles of principals include Bob being the guardian of Alice, Carol being Alice's pediatrician, and Dave being Alice's care giver after Alice being sent to the emergency.

In our ASP tool, each norm—also called a *general norm* for clarity—is represented by a tuple (A, C, D, E), where A denotes

the antecedent, C the consequent, D the deadline of A, and E the organization establishing the norm. The subject and object properties of a norm are specified in A and C and instantiated later on as described below. Singh [2] has identified various norm types. For concreteness and brevity, we consider *commitment*, *authorization*, and *prohibition* and do not treat the others.

Example 1 includes one norm of each of the three types and they are formalized as follows in our approach. In the formalization, we adopt a linear timeline from 1 to T_{max}. The predicate names that end with a capital *R* are used to denote roles. We omit the time binding of roles for simplicity, assuming each role is effective throughout the whole timeline. For example, *guardianR(C,G)* means G is C's guardian. What *guardCom* is saying is that if a principal C falls sick at time T_1, then C's guardian, G, is committed to take C to C's pediatrician, P, at time T_2, which is any time within 3 time units after T_1 and the norm is defined in *society*.

1. **guardCom**: (sick(C, T_1) $\wedge$ guardianR(C, G) $\wedge$ pediatricianR(C, P), bring(G, D, P, T_2), 3, society).
2. **shareAut**: (treat(E, C, T_1) $\wedge$ emPhysician $\wedge$ noConsent(P, G, C, T_1) $\wedge$ pediatricianR(C, P) $\wedge$ caregiver($E, C, T1$) $\wedge$ guardianR(C, G), sharePHI(P, C, T_2), 5, hospital).
3. **sharePro**: (noConsent(P, G, C, T_1) $\wedge$ pediatricianR(C, P) $\wedge$ caregiver(E, C, T_1) $\wedge$ guardianR(C, G), sharePHI(P, C, T_2), T_{max}, hospital).

Moreover, our approach creates a *norm instance* of a general norm at the time at which the antecedent or the consequent of the norm instance becomes true for the subject and object in question. When the antecedent becomes true, we say the norm instance becomes *detached*. We summarize a norm instance as a tuple (N, S, O, T), where N is the general norm the instance instantiates, S and O denote the specific subject and object principals involved, and T denotes the time at which the instance becomes detached. Once detached, a norm instance can be satisfied or violated following Singh's specification [2].

Accountability maps to the directionality in the structure of norms. The organization establishing the norm always holds the subject accountable. Thus, in Example 1, if Bob violates the norm governing his behavior as a guardian, society can hold Bob accountable for the violation. If Carol violates the norm concerning her patients' privacy, the hospital can hold her accountable for the violation.

2.2 Dominance Among Norms

Conflict detection and dominance relations apply to norm instances rather than to general norms. We say two detached norm instances are in *conflict* if they cannot be satisfied simultaneously. The process is determining whether two consequents are consistent or contradictory is a more subtle and situation dependent case and we would not discuss it here for simplicity. For example, *shareAut* and *sharePro* conflict with each other after Alice is sent to the emergency room because their consequents both include Carol sharing Alice's PHI with Dave. The specific conflict detection is achieved using the tool by the following formalization:

```
authorization(N1) ∧ prohibition(N2) ∧
con(N1) ∧ ¬con(N2) ∧
detached(NI1,T) ∧ detached(NI2,T) ∧
isInstOf(NI1,N1) ∧ isInstOf(NI2,N2)
→ conflicting(NI1,NI2,T),
```

which means if a norm instance NI_1's general norm is of type authorization, another norm instance NI_2's general norm is of type prohibition, the two instances are both detached at time T and their consequents are logically equivalent, then they are in conflict at time T.

We use dominance relations to resolve conflicts. When one norm instance dominates another, the satisfaction of the dominant instance vitiates violation of the dominated one. The general approach is selecting from the set of all currently detached norm instances a set of *nondominated* norm instances, that is, a (typically maximal) subset of the currently detached norm instances none of which is dominated by a conflicting norm instance in the same set. Figure 1 describes how our approach can be realized computationally in Example 1. When two norm instances NI_1 and NI_2 conflict and one of them dominates the other, then the dominated instance, NI_2, is *suppressed*, which means it cannot be violated.

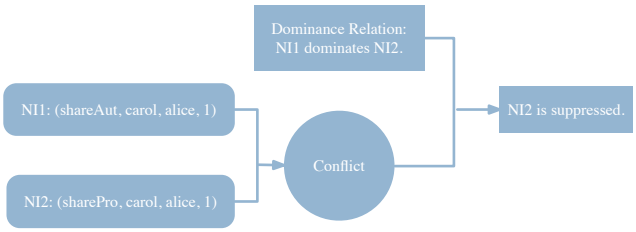

Figure 1: Dominance Relation

Since the healthcare scenario is emergent and the purpose if for treatment, we say the instance of *shareAut* dominating the instance of *sharePro*. Thus, *sharePro* are suppressed and we have a set of compliant and non-dominated norm instances. Formally, our tool describes the dominance relation as following:

```
conflicting(NI1,NI2,T) ∧
dominates(NI1,NI2,T) ∧
→ suppressed(NI2,T);

caregiverR(E,C,T) ∧
emPhysician(E) ∧ treat(E,C,T) ∧
isInstOf(NI1, shareAut) ∧ object(NI1,C) ∧
isInstOf(NI2, sharePro) ∧ object(NI2,C)
→ dominates(NI1,NI2,T).
```

3. DISCUSSION

Dealing formally about norms as implementation-independent requirements can lead to early detection of errors and to the design of better computational mechanisms for secure collaboration.

The next step of our research is conducting a human subject study aiming to compare the expressiveness and learnability of our ASP tool with other existing approaches. We would also extend our results and approaches to incorporate sanctions and to formalize specific examples with a multi-level hierarchy of norms.

Acknowledgments

We thank the US Department of Defense for support through the Science of Security Lablet grant to NC State University.

4. REFERENCES

[1] S. Marinovic, N. Dulay, and M. Sloman. Rumpole: An introspective break-glass access control language. 17(1):2:1–2:31, Aug. 2014.

[2] M. P. Singh. Norms as a basis for governing sociotechnical systems. 5(1):21:1–21:23, Dec. 2013.