Data Provenance for Accountability Mechanisms and Properties

Benjamin E. Ujcich[1,2] (✉), Adam Bates[3], and William H. Sanders[1,2]

[1] Department of Electrical and Computer Engineering
[2] Information Trust Institute
[3] Department of Computer Science
University of Illinois at Urbana-Champaign, Urbana, Illinois 61801, USA
{ujcich2,batesa,whs}@illinois.edu

Abstract. Data provenance has been proposed as a component of accountable systems design because it attributes system events to system agents and captures how processes use and generate data. However, achieving "accountability" as part of a mechanism design or as a system property remains imprecise since no consensus exists on the term's definition. We synthesize prior accountability formalisms to understand their amenability to data provenance, we show how accountability is related to assessment of responsibility and blame, and we consider future challenges in building provenance-driven accountability systems.

1 Introduction

Accountability is a cross-disciplinary concept that straddles the boundaries between the legal, regulatory, management, and technical domains [1]. Accountability has drawn a resurgent interest recently with the rise of automated decision-making by machine learning algorithms [2–4], where the "right to explanation" of how such decisions are made is starting to become codified into law [5].

However, a precise and technical definition of *accountability*—as a *mechanism* for system design or as a system *property* to be verified or validated—has remained elusive. A recent study [6] of systems incorporating "accountability" found that nearly half of such systems did not explicitly define the term, though informal definitions typically included the ability to assign responsibility and blame, provide integrity, enforce (after-the-fact) compliance, collect evidence, ensure transparency, and provide traceability. We begin with a working definition of an *accountability mechanism* as one that binds a system's states and events to identities of agents or principals within a system, provides validation and compliance mechanisms either during operation or after the fact to ensure correctness of intended behavior or acceptable standards, and ensures the integrity of records about past events when agents misbehave [7].

Defining an *accountability property* as a system invariant is even more complicated. Given the various semantics and end goals of accountability, no single unified and standard accountability formalism exists. This precision gap hinders

the comparability of accountability approaches [6]. Thus, a critical and open research question for designing accountable systems is "How can we reason about a system's accountability properties in a standardized way?"

We posit that *data provenance* can help in accountability by design and in reasoning about accountability properties because provenance captures dependency relationships among data, processes, and agents [8]. To date, however, few accountable system implementations specifically use data provenance [6]. Because no consensus exists on a precise definition of accountability, we argue that the accountability–provenance relationship requires more rigorous formulation. Accountability requires reasoning about causality to assess responsibility [9] and blameworthiness [9–11] among system agents. Provenance captures dependencies with some precision, but standard provenance models' semantics are too imprecise for formal reasoning about causality [12].

We study the accountability–provenance relationship, and we consider what challenges remain. We review the computer science approaches that prior accountability formalisms have taken, particularly for definitions and formal properties. From these formalisms, we show how the W3C PROV data model can serve as an underlying structure for accountability [8,13]. We also put forth what challenges and opportunities remain for provenance and accountability.

2 Approaches to Accountability

Deterrence and Punishment Accountability is complementary to information security's traditional goal of preventing undesirable eventsg [14]. Rather than prevent events, particularly in complex systems in which events' intents are not known before they occur or events' use of data is unclear or not specified, one can *attribute* events to agents and thus bring repercussions for violations, even if punishment happens after such violations occur [15].

Feigenbaum *et al.* [14] propose an accountability formalism that quantifies punishment based on event traces and utility functions. In their formalism, a system changes its state through a set of *events*, where each event attributed to one of a set of principals (*i.e.,* agents). An event trace T orders events temporally (*e.g.,* $\mathsf{T} = \mathsf{e_0 e_1}$ means that event $\mathsf{e_0}$ happens before event $\mathsf{e_1}$), and each event trace has an *expected utility function* $u_i(\mathsf{T})$ that maps an event trace T to a real value for each corresponding principal i in the set of principals. Thus, to reason about accountability, one can model each principal's utility functions in such a case where a *violation*, denoted by event $\mathsf{e_v}$, occurs in the trace.

Responsibility and Blameworthiness Assuming that accountable mechanisms provide attribution by binding system events to the agents that caused them, assessing which agents are "responsible" for a violation and the extent to which agents can be "blamed" is not always straightforward. For instance, suppose an initial event caused by a malicious agent induces benign agents to cause a series of cascading events that they would not otherwise have taken, causing one of

the benign agents to violate a system policy. Is the benign agent considered "responsible" for the violation because it was a direct cause of it? Is the malicious agent considered "blameworthy" because it induced others to act?

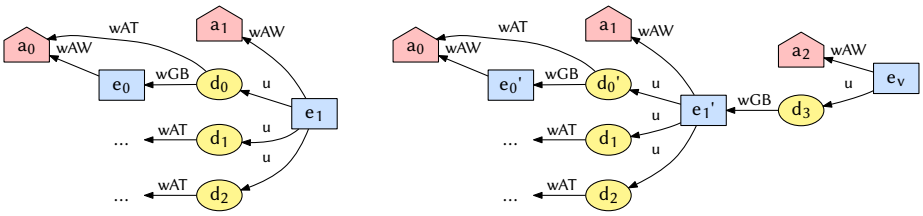
Chockler and Halpern [9] model the extent to which an event $\mathsf{e_0}$ is *responsible* for an event $\mathsf{e_1}$. Responsibility ρ is defined as $\rho = \frac{1}{1+|\Gamma|}, 0 \leq \rho \leq 1$, where Γ is the *contingency set*—the number of events that have to change in order for event $\mathsf{e_0}$ to be a counterfactual cause of event $\mathsf{e_1}$ (*i.e.,* if $\mathsf{e_0}$ did not occur, then $\mathsf{e_1}$ would not have occurred). The extent to which an event is *blameworthy* with respect to another event is based on what the agent that caused the event "knew" (or should have known) about the system's state, which can be modeled as the expected degree of responsibility over a probability distribution of possible events occurring. Feigenbaum *et al.*'s model [14] makes a distinction between *automatic punishment*, in which a violator is punished immediately, and *mediated punishment*, in which a violator temporarily increases his or her utility by causing an event that later leads to a violation.

Judgment Capabilities In issuing judgments against agents after assessing blame, Jagadeesan *et al.* [10] propose several properties that describe the extent to which a system is capable of reasoning about blame, and thus provides accountability: an *upper bound* in which every agent that performed a violation is blamed, a *lower bound* in which every agent who is blamed performed a violation, an *overlap* in which at least one agent who is blamed performed a violation, a notion of *liveness* in which a violation always results in at least one agent's being blamed, and a notion of *blamelessness* in which agents that did not perform a violation want to demonstrate their innocence. An upper bound may falsely blame innocent agents (*i.e.,* false positives), while a lower bound may miss misbehaving agents (*i.e.,* false negatives). Küsters *et al.* [11] propose similar blame properties for *fairness* (*i.e.,* honest agents following a protocol are never blamed) and *completeness* (*i.e.,* if agents do not follow the protocol, then they are blamed).

3 Towards an Accountability Provenance Structure

The W3C PROV data model (W3C PROV-DM) [8,13] graphically represents a system's prior interactions through Agent, Activity (*i.e.,* processes or events), and Entity (*i.e.,* data) nodes that are connected by relations that capture data, process, and responsibility views. From a responsibility view, provenance can provide *attribution* of data to the agents that created them through the wasAttributedTo relation, *association* of events with the agents that were involved through the wasAssociatedWith relation, and *delegation* of tasks from one agent to another through the actedOnBehalfOf relation [8,13]. However, those relations' semantics are defined only at a high level in the PROV data model specification [12,13].

A Motivating Example We revisit the earlier example of a malicious agent and use it as a motivating example of accountability modeling challenges. Figure 1 shows two provenance graphs representing two series of collected events. Figure 1a shows the correct and intended behavior of agents $\mathsf{a_0}$ and $\mathsf{a_1}$. Here, agent



(a) Correct behavior of event trace $\mathsf{T}_{\mathsf{correct}} = \mathsf{e}_0\mathsf{e}_1$.

(b) Violating behavior of event trace $\mathsf{T}_{\mathsf{violation}} = \mathsf{e}'_0\mathsf{e}'_1\mathsf{e}_\mathsf{v}$.

Fig. 1. Two provenance graphs showing agents a_0, a_1, and a_2 interacting with activities e_0, e'_0, e_1, e'_1, and e_v with two event traces $\mathsf{T}_{\mathsf{correct}}$ and $\mathsf{T}_{\mathsf{violation}}$.

a_0 creates an entity d_0 by way of event e_0. Subsequently, agent a_1 uses d_0, as well as entities d_1 and d_2, in event e_1.

Now suppose that a_0 is malicious and intends to cause a system violation that eventually manifests itself in the violating event e_v, as shown in Figure 1b. Agent a_0 generates a corrupted entity d'_0 by way of event e'_0. Agent a_1 uses entities d'_0, d_1, and d_2 in event e'_1, and, as a result of the event's internal processes, generates a corrupted entity d_3. At some time later, agent a_2 reads the corrupted entity d_3, causing a violating event e_v that violates some policy of the system. Suppose that an auditor detects this, either as soon as e_v occurs (*e.g.,* through a reference monitor) or after it has occurred (*e.g.,* through a routine audit).

Event Traces We can represent event traces in the provenance graph structure as paths connecting activities through used and wasGeneratedBy relations. Figure 1 shows traces $\mathsf{T}_{\mathsf{correct}}$ and $\mathsf{T}_{\mathsf{violation}}$. Feigenbaum *et al.*'s model [14] assumes that once violations occur, no actions can undo them. In W3C PROV, we can see that data that are generated by a violating event and subsequently used by later events are also subject to the violation. The malicious agent a_0 derives a higher utility from the trace $\mathsf{e}'_0\mathsf{e}'_1$ than from $\mathsf{e}'_0\mathsf{e}'_1\mathsf{e}_\mathsf{v}$, as otherwise a_0 would not have been incentivized to cause the events leading to the violation.

Responsibility We start with basic approaches to assessing the responsibility of the actions represented in Figure 1 and argue that a more complex model is required. From Figure 1b, we might first assume that agent a_2 was responsible for the violating event e_v, since e_v wasAssociatedWith a_2 according to how the provenance was collected. Under a different approach, we might also surmise that all agents from which paths exist that start at e_v might be responsible to various degrees, since they all contributed to some past events.

However, when we compare events e_1 and e'_1 after obtaining *evidence* of known correct behavior from Figure 1a, we see that 1) e_1 and e'_1 differ by one data value, and 2) e_1 does not generate an entity (*i.e.,* entity d_3), so does not subsequently cause a_2 to use d_3 and cause a violation. Using the Chockler and Halpern model [9], we can say that the degree of responsibility ρ for d'_0 with respect to d_3 is 1. By using evidence of correct behavior as shown in Figure 1a,

we see that the contingency set's size is 0, because the absence of d'_0 alone prevents d_3 from being generated.

Blameworthiness Blameworthiness requires "knowledge of" the agent's state (or expected state) when the event occurred. Relating the causal responsibility notions to those of blameworthiness by judging which agents ought to be punished is nontrivial. For instance, without the context explained in the motivating example, can one say that agent a_0 "knowingly" generated bad data to cause a violation, or that agent a_2 "knowingly" used such data if it was aware of the system's policies? The provenance graphs in Figure 1 do not directly capture that information, but an auditor who has access to additional evidence may be able to infer patterns and derive probability distributions of events' likeliness [9].

4 Challenges and Opportunities

For responsibility, the unintended behavior represented in the provenance graph needs to be compared to correct behavior found in *reference events* [16]. The database community has considered causal responsibility concepts as applied to provenance [16–18], though challenges remain in modeling the semantics in W3C PROV-DM. Modelers could define the intended provenance structure by using graph grammars for pattern matching [19, 20]. For blameworthiness, an auditor could look at patterns as proximate evidence of an agent's "knowledge."

From a modeling perspective, provenance data are as precise as the level of granularity at which they are collected, which may create an incomplete view of the world. In properties of blame [10], external factors that are not captured in the provenance data may be responsible for a violation, but it would not be possible to determine that from the provenance data alone. Another challenge exists in mitigating the *dependency explosion* if long-running activities use and generate many entities that are not causally dependent upon each other, leading to false dependencies and erroneous responsibility and blame.

5 Conclusion

We presented prior accountability formalisms in the context of data provenance to understand provenance's ability to aid accountability mechanisms and properties in systems. We found that accountability formalisms have focused primarily on modeling of deterrence, responsibility, blame, and judgments, and we outlined what challenges remain and opportunities exist for data provenance.

Acknowledgments

The authors would like to thank Jenny Applequist for her editorial assistance. This material is based upon work supported by the Maryland Procurement Office under Contract No. H98230-18-D-0007 and by the National Science Foundation under Grant Nos. CNS-1657534 and CNS-1750024.

References

1. N. Papanikolaou and S. Pearson, "A cross-disciplinary review of the concept of accountability," in *Proceedings of the International Workshop on Trustworthiness, Accountability and Forensics in the Cloud (TAFC)*, 2011.
2. N. Diakopoulos and S. Friedler, "How to hold algorithms accountable," *MIT Technology Review*, Nov. 2016.
3. R. Binns, "Algorithmic accountability and public reason," *Philosophy & Technology*, May 2017.
4. J. Kroll, S. Barocas, E. Felten, J. R. Reidenberg, D. Robinson, and H. Yu, "Accountable algorithms," *University of Pennsylvania Law Review*, vol. 165, 2016.
5. Council of the European Union, "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 (General Data Protection Regulation)," in *Official Journal of the European Union*, vol. L 119, May 2016, pp. 1–88.
6. S. Kacianka, K. Beckers, F. Kelbert, and P. Kumari, "How accountability is implemented and understood in research tools," in *Product-Focused Software Process Improvement*, M. Felderer, D. Méndez Fernández, B. Turhan, M. Kalinowski, F. Sarro, and D. Winkler, Eds. Springer, 2017.
7. A. R. Yumerefendi and J. S. Chase, "The role of accountability in dependable distributed systems," in *Proceedings of HotDep '05*. USENIX Association, 2005.
8. L. Moreau and P. Groth, "Provenance: An introduction to PROV," *Synthesis Lectures on the Semantic Web: Theory and Technology*, vol. 3, no. 4, pp. 1–129, 2013.
9. H. Chockler and J. Y. Halpern, "Responsibility and blame: A structural-model approach," *Journal of Artificial Intelligence Research*, vol. 22, no. 1, pp. 93–115, Jul. 2004.
10. R. Jagadeesan, A. Jeffrey, C. Pitcher, and J. Riely, "Towards a theory of accountability and audit," in *Proceedings of ESORICS '09*, M. Backes and P. Ning, Eds. Springer, 2009, pp. 152–167.
11. R. Küsters, T. Truderung, and A. Vogt, "Accountability: Definition and relationship to verifiability," in *Proceedings of ACM CCS '10*. ACM, 2010, pp. 526–535.
12. J. Cheney, "Causality and the semantics of provenance," in *Proceedings of Developments in Computational Models (DCM '10)*, 2010.
13. World Wide Web Consortium, "PROV-DM: The PROV data model," 2013.
14. J. Feigenbaum, A. D. Jaggard, and R. N. Wright, "Towards a formal model of accountability," in *Proceedings of NSPW '11*. ACM, 2011, pp. 45–56.
15. F. B. Schneider, "Accountability for perfection," *IEEE Security Privacy*, vol. 7, no. 2, pp. 3–4, March 2009.
16. A. Chen, Y. Wu, A. Haeberlen, W. Zhou, and B. T. Loo, "Differential provenance: Better network diagnostics with reference events," in *Proceedings of ACM HotNets '14*. ACM, 2015, pp. 25:1–25:7.
17. A. Meliou, W. Gatterbauer, and D. Suciu, "Bringing provenance to its full potential using causal reasoning," in *Proceedings of TaPP '11*. USENIX Association, 2011.
18. B. Salimi, L. Bertossi, D. Suciu, and G. V. den Broeck, "Quantifying causal effects on query answering in databases," in *Proceedings of TaPP '16*. USENIX Association, 2016.
19. M. Lemay, W. U. Hassan, T. Moyer, N. Schear, and W. Smith, "Automated provenance analytics: A regular grammar based approach with applications in security," in *Proceedings of TaPP '17*. USENIX Association, 2017.
20. W. U. Hassan, M. Lemay, N. Aguse, A. Bates, and T. Moyer, "Towards scalable cluster auditing through grammatical inference over provenance graphs," in *Proceedings of NDSS '18*. Internet Society, 2018.