Formal Definition of Accountability and its Applications

Ralf Küsters
University of Trier, Germany
kuesters@uni-trier.de

Tomasz Truderung
University of Trier, Germany
truderung@uni-trier.de

Andreas Vogt
University of Applied Sciences and Arts
Northwestern Switzerland
andreas.vogt@fhnw.ch

I. Motivation

Many security-critical systems, such as those for electronic voting, contract signing, and auctions, involve the use of *(semi-)trusted* parties, such as notaries and authorities. It is crucial that such parties can be held accountable in case they misbehave, as this is a strong, in some cases maybe the main incentive for such parties to follow the protocol. To achieve accountability, a system must provide a solid evidence of a misbehaviour, when one occurs.

Accountability is also a foundation for another important property: *recoverability*. The goal of recoverability is to guarantee that the system produces satisfying results even if some parties misbehave in significant ways (that, otherwise, would invalidate the result). This may require strict procedures for re-running parts of the protocol, as well as built-in mechanisms to pin-point and exclude misbehaving parties.

II. Formal Definition of Accountability

In [3], we propose a general, model-independent, formal definition of accountability. This definition is applicable to a wide range of cryptographic tasks and protocols, yet it allows one to precisely capture the level of accountability a system provides. This is demonstrated in a series of case studies, (see below), in which we apply our definition to protocols for three important cryptographic tasks: contract-signing, voting, and auctions. Our analysis of these protocols reveals some subtleties and unexpected weaknesses.

To define accountability, we assume that there is an agent J who is supposed to blame protocol participants in case of misbehavior. The agent J, which we sometimes refer to as a *judge*, can be a "regular" protocol participant or an (external) arbiter, possibly provided with additional information by other protocol participants (J does not necessarily trust these other protocol participants, as they may be dishonest and provide J with bogus information).

In order to understand the subtleness of accountability, it is instructive to first look at a simple (flawed) definition of accountability, and its possible interpretations, inspired by informal statements about accountability in the literature. Such a definition consists of two conditions:

(i) (*fairness*) J never blames protocol participants who are honest, i.e., run their honest program.[1]

(ii) (*completeness*) If in a protocol run participants "misbehave", then J blames those participants.

While the fairness condition is clear and convincing, this is not the case for the completeness condition. First, the question is what "misbehavior" means. It could be interpreted as behavior that does not correspond to any honest behavior, as specified by the protocol. However, this interpretation is much too strong: no protocol would satisfy it, because this includes misbehavior that is impossible to be observed by any other party. Moreover, this would also include misbehavior that is completely "harmless" and "irrelevant". For example, if, in addition to the messages a party A is supposed to send to another party B, A also sends to B some harmless, unrelated message, then B can observe this misbehavior, but cannot convince J of this fact. This example also shows that interpreting "misbehavior" as dishonest behavior observable by honest parties, and hence, misbehavior that, at least to some extent, affects these parties, does not work either. In fact, a completeness condition based on this notion of "observable misbehavior" would again deem basically all non-trivial protocols not accountable. More importantly, this completeness condition misses the main point: misbehavior that cannot be observed by any honest party may still be very relevant and harmful. We therefore advocate an interpretation that circles around the desired and clearly specified *goals* of a protocol.

On the intuitive level (see [3] for more details), our definition of accountability reads as follows:

(i) (*fairness*) J never blames protocol participants who are honest, i.e., run their honest programs,

(ii) (*completeness*, goal centered) If, in a run, the goal of the protocol is not met—due to the misbehavior of one or more protocol participants—then we J blames those participants who misbehaved, or at least some of them (see below).

The goal of the protocol is domain specific and left open as a parameter of the definition. It makes our definition flexible and applicable to many different domains. For example, for voting protocols a desired goal is that the published result of the election corresponds to the actual votes cast by the voters. The completeness condition now guarantees that, if this is not the case (a fact that must be due to the misbehavior of one or more protocol participants), then one or more participants are held accountable by J; by the fairness condition they

[1] In the cryptographic setting, we allow honest parties to be blamed, but only with negligible probability.

are *rightly* held accountable. In case of auctions, a desired goal is that the announced winner is in fact the winner of the auction and the announced price is correct; if this is not so, by the completeness condition some participant(s) who misbehaved will be blamed.

The completeness condition stated above leaves open who exactly should be blamed. One could think that it is desirable that the judge, whenever a desired goal of a protocol is not met, blames *all* misbehaving parties. This, as explained above, is usually not possible (e.g., if the deviation from the protocol consists in sending a harmless and unrelated message). So, this sets the bar too high for practically every protocol and one needs to relax this too strong requirement.

Therefore, we postulate that proper (that is, strong enough and achievable) level of accountability is captured by *individual accountability* which requires that, whenever the goal of a protocol is not met, at least one misbehaving party is blamed *individually*. Being able to rightly blame individual parties is important in practice, since only this might have actual consequences for a misbehaving party.

While individual accountability is highly desirable, our case studies show that protocols often fail to achieve it. For some protocols, the best one can do in some situations is to blame a group of participants, without specifying which ones amongst them misbehaved. This is too weak, because in such a case it is difficult to take any punitive actions against the misbehaving participants, as it is not clear who exactly is to blame. Still, in order to be able to study the level of accountability provided by protocols which do not provide individual accountability, in our formal definition of accountability [3], we provide a mechanism to precisely specify which groups of participants are blamed and when.

III. Verifiability

It turns out that accountability is closely related to verifiability. Verifiability is a property often studied in the context of e-voting protocols. Informally, verifiability requires that protocol participant (voters in the case of e-votig) can check that the protocol produces correct output (the result of the election is correct).

We show that verifiability can be interpreted as a restricted form of accountability: while for verifiability we only require that the protocol participants can tell if the result is correct or not (i.e. if the goal of is achieved or not), for accountability we require that if the result is not correct, then, *additionally*, misbehaving parties must be singled out. While, given our definitions, this relationship is easy to see, in the literature, accountability and verifiability have not been formally connected before.

We believe that accountability, and more specifically individual accountability, is the property protocol designers should aim for, not just verifiability, which on its own is often too weak a property in practice: If a protocol participant (rightly) complains that something went wrong, then it should be possible to (rightly) hold specific protocol participants accountable for their misbehavior, and by this, resolve the dispute.

IV. Case Studies

Our formal definition has allowed us to carry out a series of case studies, where we provide accountability results (positive and negative) for many prominent protocols. Our analysis has revealed several, often surprising attacks.

A. E-voting

We have analyzed a series of prominent e-voting protocols: the *Bingo Voting* system [2], a variant of the *Helios* voting system [1], *ThreeBallot*, and *VAV* [7].

In some cases (ThreeBallot, VAV, a variant of Helios), we have demonstrated that the protocols do not provide accountability. We have discovered, for example, that the prominent *TreeBallot* protocol, unlike commonly believed, does not even provide any reasonable level of verifiability, let alone accountability [4]. We have also discovered an attack on verifiability (and thus accountability) applicable to a family of voting protocols, including a variant of the Helios voting system [5].

In the case of the Bingo Voting system, we have shown that, while this protocol provides accountability, it fails to provide individual accountability. For instance, any voter can accuse the voting authorities of some kind of misbehaviour and it is then impossible to determine whether the accusation is justified and who is dishonest: the voter by stating a false accusation or the voting authorities. This is a very serious flaw, as it enables malicious parties to undermine the election process simply by filing unjustified accusations.

B. Auction

W have analyzed an electronic auction protocol by Parkes, Rabin, Shieber, and Thorpe [6] which was explicitly designed to be of practical use. Our analysis demonstrates that the protocol does not provide individual accountability, which results in a very serious flaw that makes this protocol effectively unusable: if two bidders with two different bids claim to be the winner of the auction, then, even if it is clear that one of the two bidders misbehaved, it is impossible blame a specific bidder. It even remains open whether the auctioneer was honest and who actually won the auction.

We have proposed a fix for this problem and proved that, with this fix, the protocol indeed guarantees individual accountability.

V. Conclusions

Accountability is an important requirement in many applications. In those applications formal procedures and systems should enable accountability by providing the necessary technical evidence of a misbehavior, if there is such. We have provided a formal framework to precisely study the level of accountability provided by such systems.

An important lesson from our case studies is that protocol/system designers should aim at individual accountability, as lack of thereof may have severe practical consequences.

Accountability is important even if we believe that the authorities are honest. Without (individual) accountability, as our case studies show, trustworthiness of the system and the authorities may be too easily undermined by dishonest parties.

References

[1] Ben Adida. Helios: Web-based Open-Audit Voting. In Paul C. van Oorschot, editor, *Proceedings of the 17th USENIX Security Symposium*, pages 335–348. USENIX Association, 2008.

[2] J.-M. Bohli, J. Müller-Quade, and S. Röhrich. Bingo Voting: Secure and Coercion-Free Voting Using a Trusted Random Number Generator. In A. Alkassar and M. Volkamer, editors, *E-Voting and Identity (VOTE-ID 2007)*, volume 4896 of *Lecture Notes in Computer Science*, pages 111–124. Springer, 2007.

[3] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Accountability: Definition and Relationship to Verifiability. In *Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS 2010)*, pages 526–535. ACM, 2010.

[4] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Verifiability, Privacy, and Coercion-Resistance: New Insights from a Case Study. In *IEEE Symposium on Security and Privacy (S&P 2011)*, pages 538–553. IEEE Computer Society, 2011.

[5] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Clash Attacks on the Verifiability of E-Voting Systems. In *IEEE Symposium on Security and Privacy (S&P 2012)*, pages 395–409. IEEE Computer Society, 2012.

[6] D. Parkes, M. Rabin, S. Shieber, and C. Thorpe. Practical secrecy-preserving, verifiably correct and trustworthy auctions. In *Proceedings of the Eighth International Conference on Electronic Commerce (ICEC'06)*, pages 70–81, 2006.

[7] R. L. Rivest and W. D. Smith. Three Voting Protocols: ThreeBallot, VAV and Twin. In *USENIX/ACCURATE Electronic Voting Technology (EVT 2007)*, 2007.