Which E-Voting Problems Do We Need to Solve?

Vanessa Teague[1,2(✉)]

[1] Thinking Cybersecurity Pty. Ltd., Melbourne, Australia
vanessa.teague@anu.edu.au
[2] The Australian National University, Canberra, Australia

Secure e-voting sounds like a cryptography problem. There are private inputs, complex computations to be done on them, things to be verified, and authorities to be partially trusted. The cryptography literature is full of mathematically beautiful schemes for efficiently running electronic elections under various trust models and with various verifiability and privacy properties.

But nearly thirty years after the first voting-specific cryptography papers were written, some parts of the problem are solved while others seem as unachievable as ever. The more we learn about voting as a practical problem in security, the harder it seems.

First, we discovered that there are specific properties just for voting: *receipt freeness* [BT94]—the impossibility of proving your inputs even if you want to—is different from privacy, and necessary to avoid vote-buying and coercion. *Forced-randomisation* is a specific attack that makes sense in some voting systems, and could have a political impact if deployed against politically-biased classes of voters. Elections also need *public verifiability*, in which not only the participants, but any observer, can verify the accuracy of the computation without trusting authorities. Voting is not just a class of specific functions to be computed by (standard) MPC.

Second, decades after Ken Thompson's "Reflections on Trusting Trust" Turing Award lecture, we are still not good at checking what a computer is actually doing (oddly enough). For voting, this really matters: can we run a trustworthy electoral process using an unscrutinisable voting device? (Honestly, I wonder why this doesn't matter more in other contexts too.) There are surprising and clever techniques for allowing real humans to challenge and verify computations done by a computer [Ben06, RBH+09, AN06]. There is useful work on formalising the process in which a human can verify an electronic computation [KZZ17]. However, they are both practically and intellectually difficult for even the most diligent real human. Most practical systems use something simpler such as code voting, which has much stronger trust assumptions but is much easier to use—some even have multiple steps to allow voters to signal whether their verification succeeded [ZCC+13]. Also, important practical studies [KHRV19] demonstrate that the accuracy of fraud detection is much higher for simple schemes that people can easily understand. Nevertheless *cast-as-intended verification*, in which a voter verifies that their electronic vote matches their intention, is probably the hardest part of the voting problem. People do not even check plain-paper printouts well enough to give decent confidence [BMM+20]. Cryptographic veri-

T. Malkin and C. Peikert (Eds.): CRYPTO 2021, LNCS 12825, pp. 3–7, 2021.
https://doi.org/10.1007/978-3-030-84242-0_1

fication is harder—if people are deliberately deceived in their verification instructions, or just confused, then their verification is unsound. So practical, usable, cast-as-intended verification is likely to remain an active area of research.

Third, incentives really matter: not only, "Who has an incentive to conduct a challenge properly?" but also, "Which administrator has an incentive to implement a truly transparent and verifiable election system, when they are more likely to keep their job by sweeping problems under the rug?" In the vVote end-to-end verifiable pollsite voting project I worked on [CRST15], the electoral authorities in Victoria were reluctant to give voters any cast-as-intended verification instructions at all —the cast-as-intended protocol existed, but it only slowed the process down and ran the risk of exposing problems in a system that would otherwise be trusted. Unless this incentive is reversed, by requiring election outcomes to be supported by evidence, this behaviour will not change.

Fourth, there is no particular correlation between trustworthiness and trust, for electronic processes. Many criticisms of end-to-end verifiability in the research literature highlight the problem that people may not trust something they do not understand. That is a valid criticism and a genuine problem, but so is the opposite problem: too much trust in things that do not deserve it. Arguably the long US history of trusting the untrustworthy, particularly paperless DREs (direct-recording electronic voting machines), has caused a situation in which trust has completely broken down due to a historical lack of evidence supporting election results. Although most US jurisdictions have now returned to using paper, trust has not returned as quickly as improved processes. A little bit of healthy skepticism—and quicker scrapping of untrustworthy machines—might have been a better way of building long-term trust.

Fifth (at last we get to something related to cryptography), precise security definitions and implementation correctness really matter. The two cryptographic errors in the Swisspost/iVote/Scytl e-voting system [HLPT20] were misalignments of a primitive's properties with its protocol assumptions. In the case of the shuffle proof, a trapdoor commitment scheme was used in a protocol that was proven secure only under the assumption that the trapdoor was not known to the prover. In the case of the noninteractive ZKPs for equality of discrete logs, the problem was adaptive vs static security—a statically secure primitive was used in a protocol in which the adversary could adapt the input. It might be tempting to dismiss these errors as a consequence of inadequately reviewed software, and hence irrelevant to the research community, but the same problem had been identified earlier in Helios (by its designers: [BPW12]). The Civitas system [CCM08], based on Juels, Catalano and Jakobsson [JCJ10], had an equivalent problem: the use of plaintext equivalence tests (with distributed trust) in a context where Plaintext Equivalence proofs (with public verifiability) were required [MPT20]. There is no mistake in the JCJ proof, nor is there a mistake in the security proof of the PETs they refer to, but there is a misalignment between the property that is proven of the primitive, and the property that is assumed by the protocol proof. This misalignment breaks the main security goals of the system, as well as several followup works. It is hard to see how for-

mal methods—even very sophisticated ones—could catch this kind of problem without a human looking very closely. Of course, this could happen in any system (not just in voting), but it is frightening how long things that completely undermined the core security properties went undetected, even in good quality systems that had been open for years. It is hard to see how a system based on cryptography alone could be robust against these kinds of mistakes.

Sixth, every democratic country is different (which is lucky for some of us). In Australia, participation is compulsory; in Switzerland, it is important to maintain privacy over who participated. Some countries take the secret ballot very seriously, others not so much. Some countries have a tolerable public key infrastructure, others don't. And elections may consist of numerous referenda very frequently, detailed preferences to be expressed every few years, or something else. A technical solution that works well in one country may not even meet the basic requirements elsewhere.

It is humbling that probably the best advance in recent times has come not from cryptography but from statistics—Risk Limiting Audits (RLAs) [LS12] use random sampling of paper ballots to guarantee an upper bound on the probability of accepting a wrong election result (this probability is called the *risk limit*). Ballots keep being sampled until either the risk limit is reached or the administrators decide to conduct a full hand count. The precise statistics are no easier for ordinary people to understand than cryptography is, but a lot of people see value in randomly selecting some ballots and observing the error rate. However, there are serious details related to cryptography here too. For example, random ballot samples require publicly verifiable pseudorandom number generation—if it is predictable, the audit is completely meaningless. This is a problem cryptographers can help with: the idea of an RLA as a publicly-verifiable computational process has yet to be adequately formalised and proven secure.

There is Practical Progress in (Some Parts of) the World

The Swiss Internet Voting Rethink. I would not say that the Swiss Internet voting *system* is a great example, but that the Swiss Federal Chancellery's *process* of engaging a large number of experts in an open, public analysis in order to help rewrite their regulations, is a great example other countries could follow.[1] I have no idea what their conclusion will be. Perhaps Internet voting will be discontinued, or further restricted, or replaced with verifiable pollsite e-voting. Perhaps Swiss Internet voting will remain in a perpetual state of experimentation, analysis and limited trust (perhaps that would be a fine outcome), but the decision will be based on evidence.

Open Source Commercial Projects such as Microsoft's ElectionGuard and VotingWorks. Do not underestimate the impact of a supported, open,

[1] CoI statement: I have received money from this process. Nevertheless the fact that they pay people like us to help them improve their legislation indicates that they are making decisions in a better way than most other authorities.

library that everyone can easily use. The ideas have been in the literature for a long time, but they are being produced for the first time in a way that administrators can easily buy and incorporate into transparent elections. These projects focus on the pollsite e-voting case where there are good practical solutions.

Research Challenges for Cryptographers

We do not have an end-to-end verifiable system with receipt freeness for remote voting, even one 'usable' to the standards appropriate for the IACR. Cast-as-intended verification can use a Benaloh-challenge (like Helios) or plain ciphertext-opening (like the Estonian e-voting system). Coercion-resistance can be achieved with JCJ-style fake-able voting tokens. However, we still don't have a good solution that provides both cast-as-intended verification and receipt freeness in a remote setting, except with the introduction of some much stronger trust assumptions. Nor can we add privacy from the client without greatly complicating the voting process. The fact that we haven't even solved this problem, in principle, for highly sophisticated users, shows how far we have to go to make online voting practical, without substantially stronger trust assumptions than I would want in my democracy.

I think the interesting practical research advances are to be made in paper-based cast-as-intended verification enhanced with some cryptography to allow voters to verify what happened to the paper after they submitted it, either in a polling place or by post. There are some interesting early designs in this space, but anyone who can design a system with three or more of: privacy from the voting device, usable verification, receipt freeness, and intuitive public verifiability (to a reasonable risk-limit), will make a substantial contribution to democracy.

Acknowledgement. I would like to thank all my coauthors over the years for making voting research so interesting and rewarding.

References

[AN06] Adida, B., Neff, C.A.: Ballot casting assurance. Proc. Electron. Voting Technol. Workshop (EVT) **6**, 7 (2006)

[Ben06] Benaloh, J.: Simple verifiable elections. EVT **6**, 1–10 (2006)

[BMM+20] Bernhard, M., et al.: Can voters detect malicious manipulation of ballot marking devices? In: 2020 IEEE Symposium on Security and Privacy (SP), pp. 679–694. IEEE (2020)

[BPW12] Bernhard, D., Pereira, O., Warinschi, B.: How not to prove yourself: pitfalls of the fiat-shamir heuristic and applications to helios. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 626–643. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34961-4_38

[BT94] Benaloh, J., Tuinstra, D.: Receipt-free secret-ballot elections. In: Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, pp. 544–553 (1994)

[CCM08] Clarkson, M.R., Chong, S., Myers, A.C.: Civitas: toward a secure voting system. In: 2008 IEEE Symposium on Security and Privacy (sp 2008), pp. 354–368. IEEE (2008)

[CRST15] Culnane, C., Ryan, P.Y.A., Schneider, S., Teague, V.: vVote: a verifiable voting system. ACM Trans. Inf. Syst. Secur. (TISSEC) **18**(1), 1–30 (2015)

[HLPT20] Haines, T., Lewis, S.J., Pereira, O., Teague, V.: How not to prove your election outcome. In: 2020 IEEE Symposium on Security and Privacy (SP), pp. 644–660. IEEE (2020)

[JCJ10] Juels, A., Catalano, D., Jakobsson, M.: Coercion-resistant electronic elections. In: Chaum, D., et al. (eds.) Towards Trustworthy Elections. LNCS, vol. 6000, pp. 37–63. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12980-3_2

[KHRV19] Kulyk, O., Henzel, J., Renaud, K., Volkamer, M.: Comparing challenge-based and code-based internet voting verification implementations. In: Lamas, D., Loizides, F., Nacke, L., Petrie, H., Winckler, M., Zaphiris, P. (eds.) INTERACT 2019. LNCS, vol. 11746, pp. 519–538. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-29381-9_32

[KZZ17] Kiayias, A., Zacharias, T., Zhang, B.: Ceremonies for end-to-end verifiable elections. In: Fehr, S. (ed.) PKC 2017. LNCS, vol. 10175, pp. 305–334. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54388-7_11

[LS12] Lindeman, M., Stark, P.B.: A gentle introduction to risk-limiting audits. IEEE Secur. Priv. **10**(5), 42–49 (2012)

[MPT20] McMurtry, E., Pereira, O., Teague, V.: When is a test not a proof? In: Chen, L., Li, N., Liang, K., Schneider, S. (eds.) ESORICS 2020. LNCS, vol. 12309, pp. 23–41. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59013-0_2

[RBH+09] Ryan, P.Y.A., Bismark, D., Heather, J., Schneider, S., Xia, Z.: Prêt à voter: a voter-verifiable voting system. IEEE Trans. Inf. Forensics Secur. **4**(4), 662–673 (2009)

[ZCC+13] Zagórski, F., Carback, R.T., Chaum, D., Clark, J., Essex, A., Vora, P.L.: Remotegrity: design and use of an end-to-end verifiable remote voting system. In: Jacobson, M., Locasto, M., Mohassel, P., Safavi-Naini, R. (eds.) ACNS 2013. LNCS, vol. 7954, pp. 441–457. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38980-1_28