This is joint work/post with Ian Miers, Oded Naor, and Ari Juels

.

Recently, a number of proposals and applications have evolved on Ethereum that require voting; from contracts that use voting-based DAOs for governance or other administrative tasks, to boardroom-like votes to proposals for open source funding. Voting appears to be a natural target for Ethereum and applications. Unfortunately, the space of voting algorithms opens a new class of buying and bribery coordination mechanisms that could prove devastating in permissionless networks, either as credible threats or as actual deployed threats. Some schemes, even ones which assume robust identity, may be even more vulnerable than naive schemes.

Beyond the standard integrity and privacy properties of a voting scheme— that votes are counted correctly and voters remain secret— for many applications we want a scheme that is bribery resistant. In on-chain voting schemes today, this is not the case: users can prove how they voted and thus be paid for their vote. Moreover, this can be done with a smart contract or “Dark DAO”, making it trustless. A natural question is whether technical protections against such an attack are possible in-protocol.

Here we propose a voting scheme that is immune to bribery attacks. Doing so requires a major assumption: that some party/group of parties/ secure hardware is trusted to prevent bribery. If these parties are dishonest, then the scheme still has integrity and privacy, but not bribery-resistance. The key question we are hoping for the feedback of projects involving voting is: are these assumptions acceptable for blockchain based voting?

Interestingly, it appears that as long as trusted execution environments such as SGX exist, we must use them to ensure a scheme is bribrery resistant. With SGX, users can be bribed to put/generate their keys in an SGX enclave that only allows them to vote in certain ways (see the “Dark DAO article”).

To prevent this, we have to ensure the keys are not in the exclusive possession of such an enclave. The only way we see to do so is by using an enclave: attest that keys were honestly generated by code that does not enforce such restrictions.

The general approach to bribery-resistance— what the literature perhaps confusingly calls coercion resistance—is to allow voters to submit as many fake votes as they want, but only allow one vote to be genuine. To anyone but the Election Authority, fake and real votes are indistinguishable. The EA, which may be a single party, a group of parties, or some trusted hardware, can test if a vote is genuine or not. They can then prove to everyone that 1) all genuine votes were included 2) all fake votes were excluded 3) the vote was computed correctly.

Ignoring cryptographic details, our proposed scheme functions as follows:

1. A voter registers by putting cryptographic material, along with an attestation of its correctness, on the blockchain.

2. Users cast votes for a given candidate by submitting a ciphertext, the candidate name, and a zero-knowledge proof to the blockchain. They do so via an anonymous channel like Tor.

3. For each ballot, the EA tests if a ballot is a fake vote or a real one. If real, the EA outputs an additively homomorphic ciphertext with the encryption of 1. If fake, an encryption of 0. For each ballot, it also outputs a proof that the ciphertext and test were computed correctly. These can be checked by anyone

4. The sum of those ciphertexts is computed by everyone. For each candidate, the voting authority then gives the decryption of the sum. The sum is the number of honest votes for that candidate.

The challenge with this scheme is the need for an Election Authority. It needs decryption keys for the ballots. These keys must remain secret, otherwise a briber can use them to test for fake votes by a briber. As such, it cannot be run by a smart contract. This means either selecting a set of parties to operate as the EA or using trusted hardware. In either case, there is no risk that the EA can manipulate the election or violate vote privacy. They can, however, refuse to reveal the election results if they don’t like the outcome.

For smart contracts, there are two key requirements here that may be hard to stomach:

1. The existence of an online, likely stake-bonded, committee during the vote. This is necessary because someone needs to hold secret data that creates an asymmetry between honest players and the attacker, stopping the attacker from gaining full insight into protocol operation.

2. The use of SGX by users/voters for coercion resistance. Users should need use SGX only for registration

, not for voting itself. As stated above, even if SGX is 100% broken, the protocol provides privacy and integrity

, but not coercion resistance (or bribery resistance, as we are interested in). Voters can likely also use a shared/hosted instance of a cloud service that provides SGX, like Azure, using an additional protocol to ensure security.

So, researchers for voting projects in the space: does it seem likely that the above are palatable, and if not let’s have a discussion on the relaxed level of security we want to shoot for without them?

I’m mainly making this post to determine whether it’s worth fully analyzing or implementing such a scheme. Basically:

- How much are projects worried about any kind of bribery attacks?
- How much are projects worried about MPC attacks?
- How much are projects worried about trusted hardware attacks?
- Is requiring SGX during registration a dealbreaker? What about if this can be outsourced to a cloud instance (or e.g. bonded network of cloud instances) securely?
- Is requiring a live staked committee a dealbreaker?
- Are there any projects that want to commit to using such a scheme or working with us to develop it? Please contact me (phil thecirclesignthing linux themostpopularthreeletterTLD) if you have any interest in this.

(obviously, we are tabling short-term practical concerns here like getting licensed by Intel for either an attack or this defense; these are addressed somewhat in the blog post… it’s worth noting that none of the private vote buying attacks strictly require SGX, a minimal MPC can likely be made practical as well)