Motivation and overview

We are proposing an ambitious, prestigious project that has the potential for major social impact. We are open to members of the ZCash community to join our team.

Security issues with electronic voting are well explored - the consensus among the security community is that standard approaches are never going to be sufficiently reliable. However, blockchain-based systems have proven to be secure enough to handle billions of dollars' worth of value. We believe that a simple to use blockchain-based system could be a significant improvement on the existing systems of electronic voting and in time could even replace paper ballots in some use cases.

The ZCash blockchain in particular has the potential to revolutionize electronic voting. In time, perhaps even voting in general. Since the beginning of election technology, the two key properties we may want in an election - verifiability and ballot secrecy - have been in tension. Verifiability - the property, that a voter can verify that their vote has been counted correctly - is a crucial component of election integrity. Ballot secrecy means that people other than the voter do not know whom for and if a vote was cast. So far, one typically needed to trade one to improve the other. But ZK-Snarks allow for both.

A voting system could be adapted from a cryptocurrency one in a reasonably straightforward way - by providing voters with cryptocurrency tokens, that they would then send to their candidate of choice. A standard blockchain, of the type we might see in Bitcoin, would be a suitable system for an open election – one in which everyone simply declares publically who they wish to vote for. An election like this has verifiability and so would be extremely difficult to forge. Having everyone declare their voting intent on a public website would have a similar effect. On the other hand, if we would like the votes to be anonymous we could use a typical, anonymous paper ballot. However, there would then be no way for a voter to verify that their vote was counted correctly.

	Paper ballot voting	Open election/ bitcoin blockchain	ZCash blockchain
Ballot secrecy	yes	no	yes
Verifiability	no	yes	yes

The setup we propose is a fork of ZCash, separate from the actual ZCash blockchain and with some modifications to the code. We would have separate voter addresses and candidate addresses, both types shielded. The functionality would be limited to voter addresses receiving the initial voting token, then sending it to a candidate address of their choice in a shielded transaction. (Splitting a token between several candidate addresses may be acceptable, otherwise transactions with less than a full token can be considered invalid and discarded on the receiving side.) This is all possible thanks to the fact that ZK-Snarks use separate sending and receiving keys. After the election, an election committee opens the candidate addresses, perhaps with distributed keys, and proves how many votes each candidate received by publishing the contents of these addresses. Anyone can verify that votes are valid. Voters can verify that their votes were counted correctly.

We propose to produce a research/white paper, a series of videos that explain the system and its benefits to various types of audience, and a Proof of Concept system that includes a fork of ZCash and a simple interface that we will then use to conduct a test election. Given enough funding, we would also like to either modify an existing wallet or create a wallet-like application that will serve as a ballot. This will only be feasible given a successful implementation of Sapling, since our system requires an efficient use of z-addresses.

The basic use case for this system is a small to medium-size election wherever a paper ballot vote is not feasible - many organisations use electronic voting today because their voters are geographically separated and resources are limited. However, in time, and with enough improvements to transaction efficiency, we think that this blockchain-based system could rival the paper ballot.

Technical approach

Our project is divided into 6 stages (see the schedule section), the execution of many of them can overlap, but they are largely dependent on one another. The inclusion of all stages will depend on the funding, and the details of the proposed schedule are likely to be modified depending on the findings along the way.

We do not rely on external software, however we would prefer to base stage 5 - wallet modification on existing wallet that supports shielded transactions. We also might use interface of existing electronic voting systems such as Helios.

We expect to modify the existing ZCash code, and the particular modifications will depend on the findings in the research phase. They will, however, include limiting the existing functionality: we will eliminate block reward (so that there is only a set total number of votes) and the memo field (to limit the possibility of voters deanonymizing themselves), and create the following types of z-addresses with a subset of their existing functionality:

1. Voting authority address, perhaps with a key split between a group of election committee members using Shamir Secrets. The viewing keys can be published at the conclusion of the election. This address will be responsible for distributing the voting tokens.
2. Candidate addresses for receiving the tokens. They would have no spending key, perhaps with keys split between a group of election committee members using Shamir Secrets. Viewing keys to be published at the conclusion of the election.
3. Voter addresses, whose receiving keys will either be eliminated after receiving the initial voting token or only allow for receiving tokens from one address. They have personal keys that stay secret.

Once we fork the blockchain itself, we will create a simple interface. We expect to either use the modified ZCash CLI or replace it with a Java CLI client using ZCash API. Such client would work across operating systems, which is a priority for this project. Alternatively, we may create a Helios-like browser front end, as long as we can store keys locally. This would have the advantage of being familiar to our test audience.

Given enough funding, we would like to further develop an easy to use interface, perhaps perfecting a Java client, or based on an existing wallet that supports z-addresses. This will be feasible post-Sapling, provided appropriate z-addresses wallets appear.

Our team has little experience with the ZCash code, so we have scheduled extra time for code analysis. We decided to price this project at $35/h, a reasonably competitive developer salary in Poland.

Security considerations

This project would be a significant improvement over existing approaches to electronic voting in terms of security.

The weakest point when it comes to security is voter registration. At the moment, we are envisioning additional time between registration and voting to verify the voters by other means, perhaps contacting them in person and verifying fingerprints. If implemented at large scale, this system could use electronic id cards that include a key of the kind currently in use in Estonia.

We are hoping to create a system that is straightforward to use. This would not only help with adoption, but reduce security risk. Ideally, a voter could simply download a client, register, and when it comes to a vote select the name of a candidate from a list and send.

A problem we haven't settled on a solution for is: how do we stop people from creating addresses themselves and using them as relay for votes. In a classic blockchain, one could only accept transactions between registered addresses, but it may not be possible to make that distinction in ZCash. However, we would like to eliminate t-addresses entirely to prevent accidental deanonymization of voters.

In the pre-proposal phase, a question of picking the time to appropriately conclude the election was raised. What happens if a transaction was created before a deadline, but included in the blockchain after? At the moment, we would like to test this system in an organisation with about 2000 voters, so we propose to wait an additional 10 minutes which should be more than enough even in the worst-case scenario for every transaction made before the deadline to go through.

We are including a security audit in the schedule and budget.

Schedule

The ZCash Election project could be divided into 6 phases, which can partially overlap in execution. Some of these phases can be eliminated from the final project in case of insufficient funding. As a research project it focuses initially on a preparation of a White Paper - crucial for all further stages. Our team believes that we need to prepare additional materials that will adequately explain the advantages and setup of this system. We would like to prepare a set of short videos that would explain it to laymen as well as developers and election professionals.

Next would come code analysis and development phases. The projected schedule is based on Researcher/Developer Workday, as we cannot provide more adequate timeline without knowing the funding situation. The calculation assumes that each development phase finishes with working POC rather than a not ready, final product. To an extent, the first 3 stages can be conducted concurrently.

Phases:

1. **White paper *(50 workdays)***
    a. Writing a draft of the paper
    b. Consultations with the election systems and blockchain communities
    c. Corrections
2. **Short education videos *(15 workdays)***

    a. A series of videos including a system explainer, project motivation, a promotional clip, (optional) documentary about a PoC vote.
    b. Script writing, pre-production, photos and assembly.
    c. *Remark*: Calculation is based on an assumption that it will be made by a professional.

3. **Elections Engine - a fork of ZCash *(60 workdays)***
    a. Existing code analysis in terms of figuring out the places where the code should be changed or completely replaced to deliver the functionality descripted by the white paper.
    b. Short technical project document.
    c. Implementation
    d. Testing, documentation and sample elections.
    e. *Remark*: Phase 3 may start concurrently with 2.
4. **Simple UI or CLI modification *(30 workdays)***
    a. Preparation of a list of minimal CLI command number which are essential for voter based on results from previous phase.
    b. Current CLI modifications if needed.
    c. Helios Voting code analysis - our test group is used to this interface. Providing them something with similar look might help to reduce voter's problems during election.
    d. Short technical project document.
    e. Implementation of our sample UI.
    f. Testing, documentation and sample elections.
5. **Wallet modification *(45 workdays)***
    a. Existing code analysis in terms of figuring out the places where the code should be changed or completely replaced to deliver the functionality descripted by the white paper.
    b. Implementation
    c. Testing, documentation and again sample elections.
6. **Security audit**
    a. Scope will depend on outputs of the project.

Our basic offer is to deliver phases 1 to 3 and finish the project with test elections done by group of around 30 voters, who will be supported live by our team to successfully go through ZCash CLI. However, we think, that it would be more interesting to include phase 4, which could lead us to real world elections done by a few hundred users spread around the world. Phase 5 depends on a successful implementation of Sapling and perhaps an emergence of a lightweight zcash wallet. Phase 6 assumes the employment of an independent specialist for code and document analysis.

Evaluation plan

We think that best way to evaluate such possibly changeable research and development project is to mark milestones of each phase. That could simplifie project tracking as we are not proposing calendar based schedule. However, our project consist of different type of etaps: research, educational and development. For each of them we propose different milestones:

White Paper	1. First draft passed for opinion to established reviewers. 2. Document published on Github. Two weeks time for independent reviewers to ask questions and give suggestions. 3. Final publication.
Short education videos	1. Promotion clip 2. System explainer, project motivation clip 3. (optional) short documentary about a PoC vote
All development phases	1. Short technical project document published on Github. One week time for independent reviewers to ask questions and give suggestions. 2. Final publication. 3. Code delivery 4. Real life voting demo

Budget and justification

Project scope alternatives are provided in the “Schedule” section. We assumed 35$/h as wage for each person involved in project, regardless of whether a team member or commercially hired specialist. This is a competitive developer salary in Poland. There is one and only exception in phase 6 - Audit, where we are currently envisioning a fixed effort compensation. That led to the following calculations:

1. White Paper/compensation for research:	14000$
2. Short educational movie - cost:	4200$
3. Elections Engine - modification of ZCash - cost:	16800$

4. Simple UI - CLI modification - cost:	8400$
5. Wallet modification - cost:	12600$
6. Audit - cost:	5000$

Funding options:

Basic project (1-3) - cost	35000$
Extended voting (1-4) - cost	43400$
Extended voting with Audit (1-4,6) - cost	48400$
Total project (1-6) - cost	61000$

Funding can be granted on completion of each phase. However, it’d be helpful if we got some advance payment at the beginning of each phase to facilitate the execution.

Our total project is calculated below typical market workload for open elections systems. 200 workdays to deliver it seems feasible when allowing for a further research phase and our intent to deliver a proof of concept rather than an out-of-the-box working system. The proposed wage allows us to hire professionals on the market in case of insufficient workforce.

Team background and qualifications

All current team members are based in Warsaw, Poland.

Dr Ewa Infeld (research, team leader)

Research in combinatorial probability, cryptography, network models, anonymity systems. Completed a postdoctoral fellowship at Ryerson University’s Department of Mathematics. Stellar lecturing record that may be relevant for the videos. Education: PhD in Mathematics from Dartmouth, two MScs from Cambridge University and London School of Economics. Active in the privacy and security community.

Anna Olchowik (developer)

Currently working for medical AI startup RowAnalytics as a software developer. 5 years of experience in game development. Former research fellow at IIMCB Warsaw (bioinformatics). Experience as a lecturer at a Poland’s first programming bootcamp and as a university lecturer might be useful in a video phase of the project. Graduated from the University of Edinburgh.

Bartosz Owczarek (developer)

Software developer with 10 years of experience, currently working at a major technology corporation. Also skilled in system integration, mobile and cable networks, project management. MSc focused on ITC and Telecommunications Systems Management from Warsaw University of Technology.

Email address(es) for direct contact

evainfeld@riseup.net

ania.olchowik@gmail.com

bjowczarek@gmail.com