1 Design

1.1 Overview

In this paper, I present an "end-to-end verifiable system" built on the Ethereum Blockchain, i.e., a system where a voter can be assured their vote has been fairly counted, only eligible voters are allowed to vote and the tallied results of the election are publicly verifiable.

Although this system has been designed and developed with the idea of a national general election in mind, the protocols and ideas involved could be applied to smaller scale ballots which wish to provide transparency in their audit. Although we wish to minimize trust in a central authority, due to the nature of these type of elections (where there needs to be some degree of voter eligibility verification), we cannot fully decentralize this system as we need to only allow those eligible the rights to vote. Despite needing to verify an individual we still need to ensure that their votes are publicly anonymous, especially given the public transactions underpinning the blockchain concept while providing the ability for an individual to verify that their vote was correctly counted.

The designed schema for this protocol is the following:

1. Ballot creation:
    - The available ballots in the election are designed and decided upon.
    - A smart contract is created and pushed to the blockchain for each ballot containing all of the voting options.
2. Pre-election voter verification:
    - Voter registers with an external voter registrar after providing a valid ID (this could be accomplished using pre-existing government electoral registration protocols).
    - This external registrar generates a *user_id* and *nonce* which can be used by the voter to log in to the system.
    - This *user_id* is then tied to any ballots the voter is eligible for.
3. Voter registration:
    - The voter logs into the system using the received *user_id* and nonce upon which time they are immediately required to change their login credentials.
    - The voter can then register to vote through the online system for each of the ballots they are eligible for.
    - A unique Ethereum address, *voter_address*, is generated and validated (while not being linked to the *user_id*)

   - The *user_address* is added to the ballots smart contract which entitles this address to vote in that ballot.

   - The address is funded with enough Ether for the voter to cast their vote.

4. Voting:

   - When the voter decides to cast their vote they are presented with an interface mirroring the options in the ballot smart contract.

   - Upon the voter selecting their options the contract is funded with the voters selected options.

   - At this point the voters choice is immutably entered into the blockchain and the tally is verifiable by all.

5. Election result:

   - Once the election is over, due to the nature of the smart contract design, no more votes can be added for any candidate.

   - The tally for each candidate is publicly verifiable by anyone along with all of the funded transactions casting votes.

1.2 Docker

Over the past few years, container technology has become increasingly promising as a means to seamlessly make software available across a wider range of platforms & allow developers to worry less about the eventual runtime environment (as this can be standardized). Docker containers provide a way to “wrap up a piece of software in a complete filesystem that contains everything it needs to run” [51].

There are several benefits to the use of Docker containers; This could substantially reduce the effort required to create and validate a new software releases, since docker containers create their own dedicated environment, testing on one OS means that the application will run the same on any OS capable of running Docker. In addition, docker containers provide a quick and easy way to install and use a software release, for our application this could mean faster patches if needed as you would simply need to swap out the docker image being used. Other benefits include faster bootup of containers (compared to just virtualization), closer development to production parity, immutable infrastructure and improved scaling (on a per-container basis) [52]. There is also a security argument to be made for using containers, as they offer a degree of isolation for each enclosed application and only expose those services which you decide. The previous point about patching also add to security, as legacy applications often forgo patches in sensitive environments due to the possibility of breakages. When using containers, changes can be fully tested in the container which can then be swapped into the production environment [53].

All of my development for this project was conducted inside of Docker containers. I decided on this because there are very distinct separations between the applications which make up my system (this is described in section LINK TO SECTION), so running each one inside a docker container seemed the logical thing to do. It also meant that I could control the startup of the system as a whole & expose (between containers) only those services necessary to communicate.

1.3 Ethereum

1.3.1 What is Ethereum

In summary, Ethereum is an open software platform based on blockchain technology that enables developers to build and deploy decentralized applications. The block chain is a decentralized network of computers who, at the most basic level, all maintain a ledger in consensus with each other. One block is added at a time, each block contains a mathematical proof that verifies it's addition to the chain and the transactions within are protected by a strong cryptography.

Ethereum is poised to become the next greatest innovation based on block chain technology so is Ethereum similar to Bitcoin? The answer is sort of, but not really. Like Bitcoin, Ethereum is a distributed public blockchain network, however there are some significant technical differences between the two. The most important distinction to note is that Bitcoin and Ethereum differ substantially in purpose and capability. Bitcoin offers one particular application of blockchain technology, a peer to peer electronic cash system that enables online payments. While the bitcoin blockchain is used to track ownership of digital currency (bitcoins), the Ethereum blockchain focuses on running the programming code of decentralized application [55].

decentralized

Ethereum enables developers to build and deploy decentralized applications. A decentralized application or Dapp serves some particular purpose to its users. Bitcoin, for example, is a Dapp that provides its users with a peer to peer electronic cash system that enables online Bitcoin payments. Because decentralized applications are made up of code that runs on a blockchain network, they are not controlled by any individual or central entity.

Any services that are centralized can be decentralized using Ethereum. Think about all the intermediary services that exist across hundreds of different industries. From obvious services like loans provided by banks to intermediary services rarely thought about by most people like title registries, voting systems, regulatory compliance and much more.

Ethereum can also be used to build Decentralized Autonomous Organizations (DAO). A DAO is fully autonomous, decentralized organization with no single leader. DAOs are run by programming code, on a collection of smart contracts written on the Ethereum blockchain. The code is designed to replace the rules and structure of a traditional organization, eliminating the need for people and centralized control. A DAO is owned by everyone who purchases tokens, but instead of each token equating to equity shares & ownership, tokens act as contributions that give people voting rights.

What are the benefits of Ethereum decentralized Platform? Because decentralized applications run on the blockchain, they benefit from all of its properties.

Immutability A third party cannot make any changes to data.

Corruption & tamper proof Apps are based on a network formed around the principle of consensus, making censorship impossible.

Secure With no central point of failure and secured using cryptography, applications are well protected against hacking attacks and fraudulent activities.

Zero downtime Apps never go down and can never be switched off.

Whats the downside of decentralized applications?

Despite bringing a number of benefits, decentralized applications arent faultless. Because smart contract code is written by humans, smart contracts are only as good as the people who write them. Code bugs or oversights can lead to unintended adverse actions being taken. If a mistake in the code gets exploited, there is no efficient way in which an attack or exploitation can be stopped other than obtaining a network consensus and re-writing the underlying code. This goes against the essence of the blockchain which is meant to be immutable. Also, any action taken by a central party raises serious questions about the decentralized nature of an application.

1.3.2 Blockchain

1.3.3 Ether

mining

In the Ethereum blockchain, instead of mining for bitcoin, miners work to earn Ether, a type of crypto token that fuels the network. Beyond a tradeable cryptocurrency, Ether is also used by application developers to pay for transaction fees and services on the Ethereum network [55].

1.3.4 Transaction Costs & Gas

1.3.5 Smart Contracts

solidity Smart contract is just a phrase used to describe computer code that can facilitate the exchange of money, content, property, shares, or anything of value. When run on the blockchain a smart contract becomes like a self-operating computer program that automatically executes when specific conditions are met. Because smart contracts run on the blockchain, they run exactly as programmed without any possibility of censorship, downtime, fraud or third party interference. While all blockchains have the ability to process code, most are severely limited. Ethereum is different. Rather than giving a set of limited operations, Ethereum allows developers to create whatever operations they want. This means developers can build thousands of different applications that go way beyond anything we have seen before.

1.3.6 Ethereum Clients

geth

1.3.7 Something...

1.4 System Design