NotaryCoin — open ethereum network with proof of authority consensus by U.S. notaries

Igor Barinov, Viktor Baranov, Roman Storm et al.
www.notarycoin.com

Abstract

In this paper we propose a new public ethereum network based on Proof of Authority (PoA) consensus protocol where private actors serve on public duty. The network's first applications will be an official U.S. electronic notary journal and support of admissable in court timestamping. The paper adopts the new Q&A style of a document flow.

General questions

What is NotaryCoin?

NotaryCoin (NTC) is a public ethereum network based on Proof of Authority (PoA) consensus by U.S. public notaries.

What is the difference between NotaryCoin and Ethereum?

NotaryCoin is the network based on the ethereum protocol and software with additional management contracts and distributed apps (DAPPs). It is fully compatible with the Parity client version 1.6 or higher. The main technical difference between NotaryCoin and Ethereum (ETH) public network is the consensus protocol. ETH uses Proof of Work (PoW), and NotaryCoin uses Proof of Authority (PoA). NotaryCoin will support the broad ecosystem of existing ethereum tools and solutions.

Who are authorities in NotaryCoin?

Certified U.S. public notaries who get mining licenses from NotaryCoin Foundation during initial distribution of those licenses. After the initial distribution of licenses, an additional participant can be added through voting on the blockchain and election by a majority of voters.

Why notaries required as authorities in the network?

Notaries run mining nodes which validate transactions and create blocks. The approach was to find an independent third party who can be easily verified by anyone outside of the network. It could be any licensed professional, for example tax preparer, architect, lawyer, doctor, pilot. The advantage of notaries is that the state entrusts them with the work of authenticating records and third parties trust them.

Which modifications will you add to a Parity client?

Block attribution and alternative reward split will be main features.

Block attribution: In this open network, miners will be well identified and known parties. Any 3rd party can independently check notary license and residence of each miner with Secretary of State where a notary has his license. Notaries will sign a legal agreement to not block transactions in the network. For some use cases, attribution of a mined block by miners metadata can bring positive feedback from regulators. Therefore, miner's identity will be added to each mined block and to a blockchain explorer.

Alternative reward split: The network will keep a list of miners in a smart contract and update it using a voting dapp. When one miners proposes a block and other miners accept it, all miners get proportional part of the reward. That reward will be distributed only to miners who have an active node in the network. For security reasons, a miner will propose a block from one address (mining address) and get the reward to another address(payout address)

What are the benefits of using NotaryCoin for smart contracts and distributed apps?

In NotaryCoin each block is created by an authorized U.S. public notary. Notary license requirements vary from state to state, although a public notary should be either a citizen or permanent resident (“green card”)

and a resident of the state where he or she received the license. The U.S. government does a background check on each applicant for the public notary license. Thus, each block in the NotaryCoin blockchain will be attributed by a citizen or permanent resident, with known identity and authority given by the government.

Miners of the network will sign the non-affiliation agreement. [TODO]
Miners of the network will sign the mining agreement. [TODO]

What is NotaryCoin Foundation?

NotaryCoin Foundation is a U.S. non-profit created to promote the network, to support the development of open source software of smart contracts and distributed apps within the network, and to distribute initial mining licenses.

Use cases

Do you have any use cases for NotaryCoin?

It’s possible to use any code created for the Ethereum network on NotaryCoin. NotaryCoin Foundation and BlockNotary, Inc. will provide several products that work on the NotaryCoin chain. The first product will be an electronic notary journal and trusted timestamping (OpenTimestamps) admissible in court and arbitration in the US.

What is Electronic Notary Journal?

Electronic Notary Journal replaces the paper notary journal for recording notary acts with open source and a decentralized application. It works on mobile and web platforms, is easy to use, and is built on decentralized trust. When notaries put their acts in Journal, they can trust they’re secure with end-to-end encryption, and their data is their own. The blockchain protects authenticity and timestamp for each notary act. Notarized by blockchain records are recognized in arbitration in the U.S. and in courts in the state of Vermont.

What is OpenTimestamps?

OpenTimestamps is a timestamping proof standard for blockchain timestamping. It’s vendor and blockchain independent and NotaryCoin Foundation will provide open calendar servers for timestamping in NotaryCoin.

Licenses distribution and initial liquidity

How many mining licenses will be distributed?

Altogether, 52 mining licenses will be distributed during initial distribution.

The first 12 notaries will get mining licenses after signing the non-compete memorandum and mining agreement. The first 12 notaries can be from any U.S. state. For the remaining 40 licenses, a notary from a state which is not in the NotaryCoin network can get a license after signing the non-compete memorandum and the mining agreement. If there is already a notary from the same state in the NotaryCoin, the applicant will be placed in a queue to participate in the smart contract voting. After the initial distribution, miners will vote to add new miners.

One license will be reserved for a notary from NotaryCoin Foundation and one license will be reserved for a notary from the ethereum ecosystem. Those licenses will incentivize development and support of the network. Each organization should have a licensed U.S. public notary and sign the non-compete memorandum and mining agreement.

What is the non-affiliation agreement and mining agreement?

The non-affiliation agreement is a legal document which describes relations between notaries who mine blocks in the NotaryCoin network. For example, a notary can have only one license, and it’s prohibited to get a mining license if a notary has a close relative who is a miner in the NotaryCoin network. In addition, notaries should vote for adding and removing new miners, and the voting process should be independent. The document describes the process for selling and buying mining licenses.

The mining agreement is a legal document which describes the rules of operation of a node in the network. For example, a participant should keep his or her node updated with software patches and ensure it has enough resources such as CPU, RAM, and disk space. A mining node should also run in a trusted environment. A miner should keep his or her node running with 99% uptime and participate in voting.

Who will do the background checks on applicants for the mining licenses?

Initial background checks of new applicants will be performed by the NotaryCoin Foundation and later on will be decided by the community of miners.

How will the network provide initial liquidity?

NotaryCoin will provide initial liquidity by an Initial Coin Offering (ICO) and it will follow industry best practices. Information about the ICO will be provided on the NotaryCoin website.

Threats and attack vectors

Will it be possible for miners to change their keys?

Yes. Once you receive your initial key, we highly recommend you generate keys for mining, voting, and payout accounts. After successful execution, your initial key will be invalid.

If you decide to change any of your generated keys in the future, you can always do so by executing contract’s methods or in the DAPP interface.

What if a miner’s key is compromised?

If your miner’s key is compromised, you'll be able to vote for replacing your miner's key using your voting key in the voting DAPP. If your voting and miner keys are compromised, you will need to record a statement with another notary in the network and provide that statement to some other notary who has a voting key.

To protect a network, any signer may only mint 1 block out of every K. This ensures that damage is limited, and the rest of the miners can vote out the malicious user.

Attack vector: Censoring signer

(Ref: https://github.com/ethereum/EIPs/issues/225)

One attack vector is if a signer (or group of signers) attempts to censor out blocks that vote on removing them from the authorization list. To work around this, we restrict the allowed minting frequency of signers to 1 out of N/2. This ensures that malicious signers need to control at least 51% of signing accounts, at which case it's game over anyway.

Attack vector: compromising miner's key

According to parity wiki's page regarding exposing to external connections, your miner's key will be always at risk of compromisation. Hence why we have payout key which you should always be running in your local network.

[TODO]

Questions from notaries

Is there any more documentation besides the white paper?

NotaryCoin Foundation will provide User's guide for your mining node, payout client, and voting application before launch of the system.

When will I be performing notary (I couldn't figure it out)?

You will not act as a notary to validate blocks. Your notary status and signing of a non-affiliation agreement with fellow notaries will provide a source of trust to the underlying consensus, built on proof of authority algorithm, where private identified actors serve on public duty. Meanwhile, the system will support legaltech innovations and applications on NotaryCoin and electronic notary journal will be the first application. As a miner you will not need to do anything besides signing blocks

Does mining just mean running a node or will I need to manually validate things, and how frequently?

To mine you'll need to initiate your keys and run a node in fully automatic mode. You'll need to vote to add and remove miners from the network,e.g. On license expiration, offline more than 72 hours etc.

How many keys do I get during initial distribution?

You will get an initial key from NotaryCoin Foundation and will be able to set mining, voting, and payout accounts in the management DAPP inside the Parity client.

What will I receive for participating in the network?

A reward of 5 coins (equivalent to ether in Ethereum network) will be deposited to your payout account after each block you mine.The miner will also receive all the gas in fees that it generates from the transactions in the block that it verifies. The price of coins will be determined by the market.

How can I spend my reward?

You will be able to sell it for bitcoins on a cryptocurrency exchange which will support NTC ticker. If the state jurisdiction allows operations with cryptocurrency, then you can exchange bitcoins for fiat money on sites like Coinbase. Otherwise, wait for legalization in your state.

How can I vote?

After you receive your initial key, you will replace it for voting, mining, and payment keys in the interface. You will receive an invitation to vote and using your voting key will be able to express your position on the issue that requires voting in a NotaryCoin Voting DAPP.

Where do I get the software?

NotaryCoin Foundation will provide docker images for manual installation and cloud images on Azure, IBM, and Google cloud for automatic installation. The management UI will work on local Windows and Mac computers.

Which PoA protocol does NotaryCoin use?

NotaryCoin will use PoA consensus protocol implemented in the Parity client.

Appendix A

Class of a smart contract to get list of miners and their identity

Description

The user of the interface with the information about the miners will be able to receive information about the active miners and their data, for example, full name, notary license number, validity of the notarial license, address and state. The contract will provide information for the interface

Code

```
pragma solidity ^0.4.11;


contract ValidatorsManager {
    address owner;

    //initial validators (for demo)
    address[] public validators = [0x0083d7E016DEb94836443aAF997f63DcA13cd66f,
0x004705e3796589ba255EFD3B4C13f85dAEBd66cE];
    address[] public disabledValidators = [0x00bA7Af2c7a8d724BF526e79B965D1Ae7748AA33];


    function ValidatorsManager() {
        owner = msg.sender;
    }


    function getValidators() constant returns (address[]) {
        return validators;
    }


     function getDisabledValidators() constant returns (address[]) {
         return disabledValidators;
    }
     function kill() {
         if (msg.sender != owner) return;
         selfdestruct(owner);
     }
}
```

```
contract ValidatorClass {
    address owner;
    struct Validator {
        uint id;
        string fullName;
        string streetName;
        string state;
        uint zip;
        uint licenseId;
        uint licenseExpiration;
        uint disablingDate;
        string disablingTX;
    }

    mapping(address => Validator) public validator;

    function ValidatorClass() {
        owner = msg.sender;

        //initial validators (for demo)
        validator[0x0083d7E016DEb94836443aAF997f63DcA13cd66f] = Validator({              id: 34882346728, fullName: "John A Stearn",
streetName: "4604 Ridenour Street", state: "IN", zip: 46785, licenseId: 4029833344, licenseExpiration: 1553126400, disablingDate: 0,
disablingTX: ""});
        validator[0x004705e3796589ba255EFD3B4C13f85dAEBd66cE] = Validator({id: 838472384, fullName: "Joan W Brink", streetName:
"748 Summit Park Avenue", state: "MI", zip: 48342, licenseId: 652162544616, licenseExpiration: 1764892800, disablingDate: 0,
disablingTX: ""});
        validator[0x00bA7Af2c7a8d724BF526e79B965D1Ae7748AA33] = Validator({id: 3849328423, fullName: "Daniel C Liss",
streetName: "1943 College Street", state: "GA", zip: 30030, licenseId: 253532376, licenseExpiration: 1604275200, disablingDate:
1490400000, disablingTX: "0xfbc9899cc374b95bdee3f042ba2b12b69a9cfeca008d701c11896bf38f167118"});
    }

    function getValidatorId(address addr) constant returns (uint) {
        return validator[addr].id;
    }

    function getValidatorFullName(address addr) constant returns (string) {
        return validator[addr].fullName;
    }

    function getValidatorStreetName(address addr) constant returns (string) {
        return validator[addr].streetName;
    }
```

```
    function getValidatorState(address addr) constant returns (string) {
        return validator[addr].state;
    }

    function getValidatorZip(address addr) constant returns (uint) {
        return validator[addr].zip;
    }

    function getValidatorLicenseId(address addr) constant returns (uint) {
        return validator[addr].licenseId;
    }

    function getValidatorLicenseExpiration(address addr) constant returns (uint) {
        return validator[addr].licenseExpiration;
    }

    function getValidatorDisablingDate(address addr) constant returns (uint) {
        return validator[addr].disablingDate;
    }

    function getValidatorDisablingTX(address addr) constant returns (string) {
        return validator[addr].disablingTX;
    }

    function kill() {
        if (msg.sender != owner) return;
        selfdestruct(owner);
    }
}
```


Smart contract to change initial key, mining key, voting key, and payout address

Description

With the help of a decentralized application, the miner will replace the initialization key with three keys: a mining key, a voting key, and a key for paying a reward for mining. The key for mining will be located in a

container launched in the cloud. The container will be provided as standard software without the need for additional modification.When the container is launched, the miner will provide a key for mining. The container will update itself with the new version of the software and security updates

Code

```
contract KeysClass {
    address owner;

    struct InitialKey {
        bool isNew;
    }
    struct PayoutKey {
        bool isActive;
    }
    struct MiningKey {
        bool isActive;
    }
    struct VotingKey {
        bool isActive;
    }

    mapping(address => InitialKey) initialKeys;
    mapping(address => MiningKey) miningKeys;
    mapping(address => PayoutKey) payoutKeys;
    mapping(address => VotingKey) votingKeys;
    uint8 private initialKeysIssued;
    uint8 private initialKeysLimit;

    function KeysClass() {
        owner = msg.sender;
        initialKeysIssued = 0;
        initialKeysLimit = 12;
        //initial key (for demo)
        initialKeys[0x70de02424c1b3b1ada0fab8dd1d70e04727bd082] = InitialKey({isNew: true});
    }

    function addInitialKey(address key) {
        if (msg.sender != owner) return;
        if (initialKeysIssued >= initialKeysLimit) return;
        if (checkInitialKey(key)) return;
        initialKeysIssued++;
        initialKeys[key] = InitialKey({isNew: true});
    }
```

```
    function checkInitialKey(address key) constant returns (bool) {
        if (msg.sender != key) return;
        return initialKeys[key].isNew;
    }

    function createKeys(address miningAddr, address payoutAddr, address votingAddr) {
        if (!checkInitialKey(msg.sender)) return;
        miningKeys[miningAddr] = MiningKey({isActive: true});
        payoutKeys[payoutAddr] = PayoutKey({isActive: true});
        votingKeys[votingAddr] = VotingKey({isActive: true});
        //invalidateInitialKey
        delete initialKeys[msg.sender];
    }

    function checkMiningKeyValidity(address addr) constant returns (bool) {
        if (msg.sender != addr) return;
        return miningKeys[addr].isActive;
    }

    function checkPayoutKeyValidity(address addr) constant returns (bool) {
        if (msg.sender != addr) return;
        return payoutKeys[addr].isActive;
    }

    function checkVotingKeyValidity(address addr) constant returns (bool) {
        if (msg.sender != addr) return;
        return votingKeys[addr].isActive;
    }
}
```


Peer review

[TODO]

Appendix B

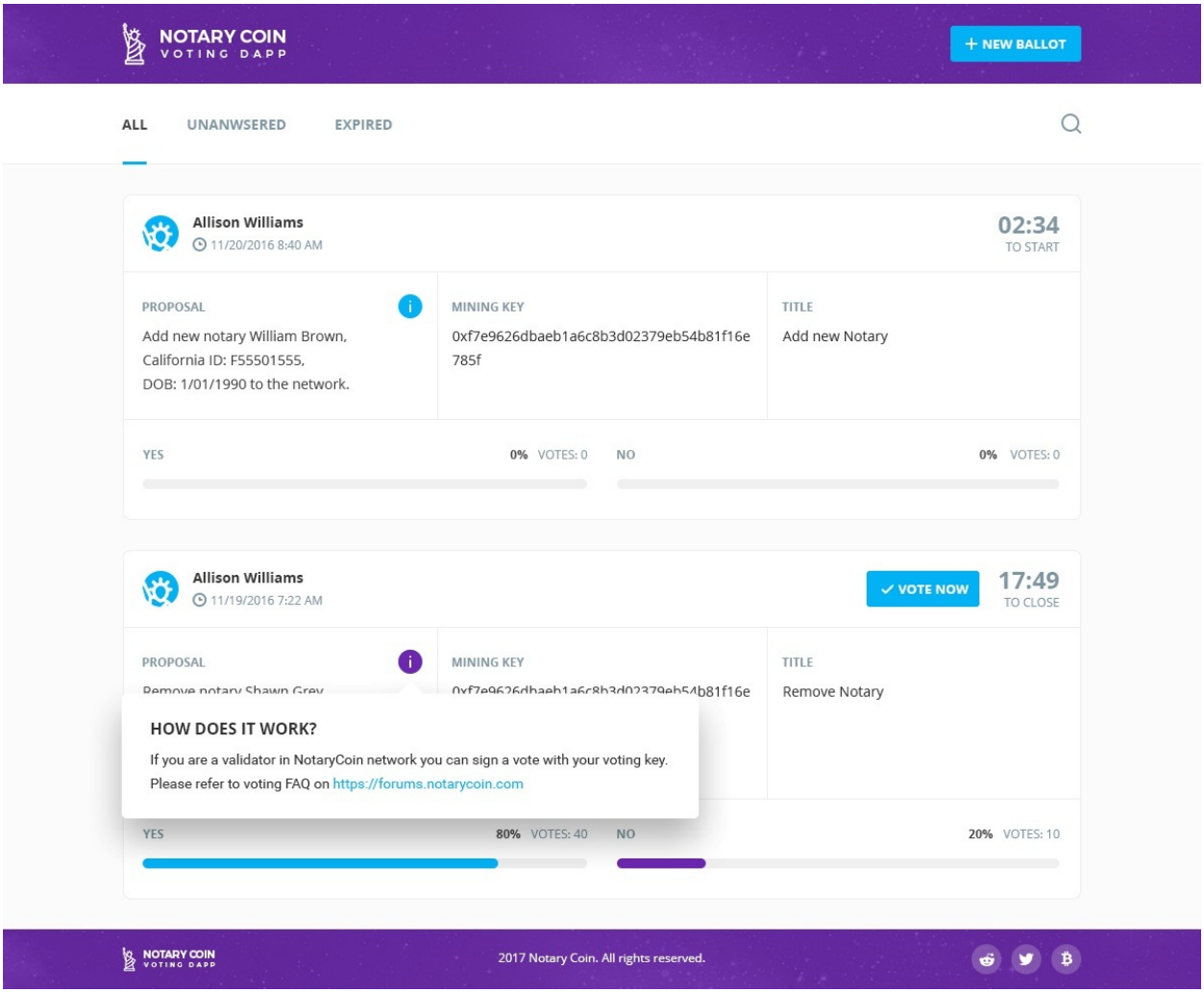
UI for DAPPS

While one can interact with the contracts using the Ethereum clients, having a web-based user interface (UI)

for a contact makes it more convenient for the end-users. Decentralized applications (DAPPS) provide this functionality. A DAPP is backed by one or more smart contracts and provides a web-based user interface for the contracts. Blockchain client can connect their blockchain accounts to the DAPP thru trusted signer function of Parity client and send transactions to the underlying smart contracts.

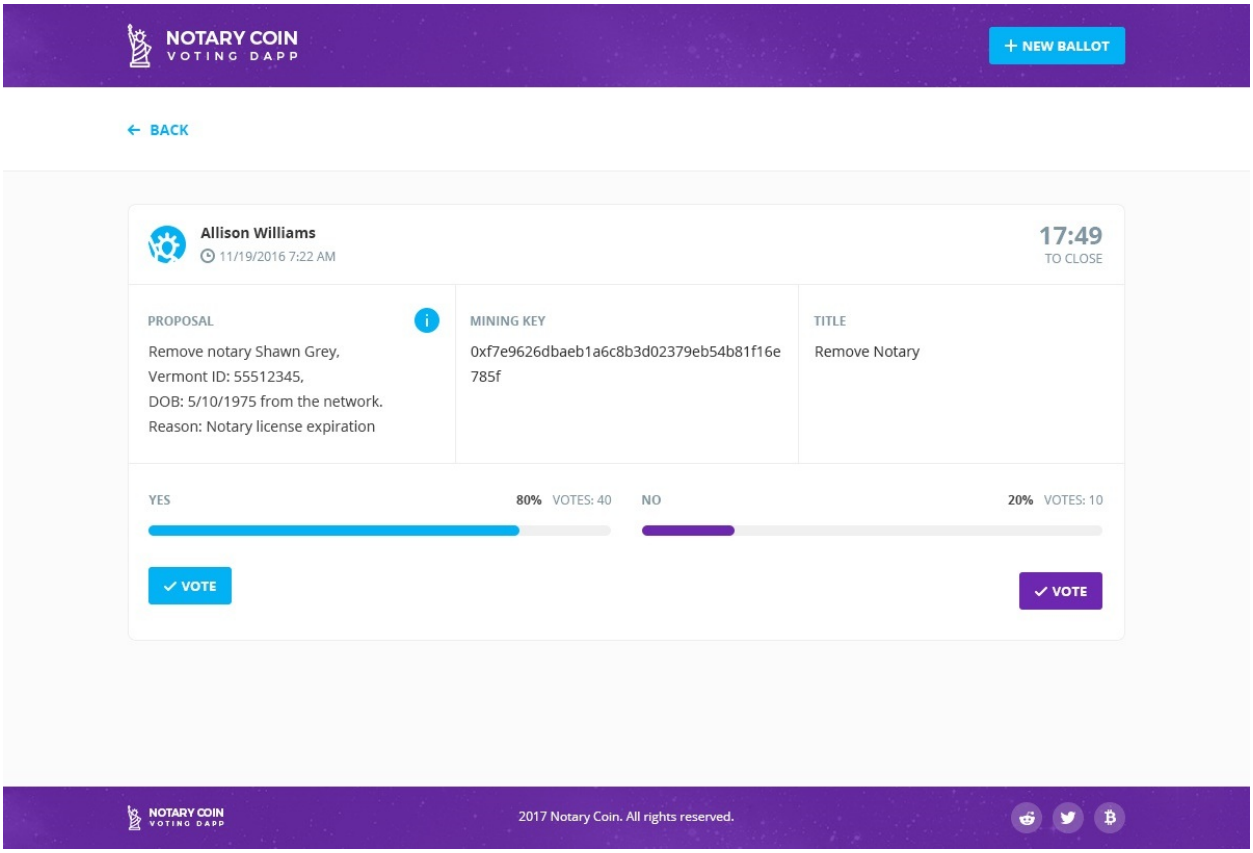
Voting DAPP

On the main screen of the application, the user sees a list of ballots. For its ease of use it is possible to switch between sections of popular, recent, unanswered, expired. In popular ballots are all open ballots in recent are registered recently, ballots in unanswered are those ballots for which the user has not yet voted, ballots which are expired deadline for a vote.
The user can create a new one by clicking New ballot. After filling in the required fields, the user will sign the ballot using the key.
Each ballot on this screen contains information about the originator, date of beginning and end of voting, the number of votes, the proposal, the affected key, the action and the number of voters who voted in the ballot.

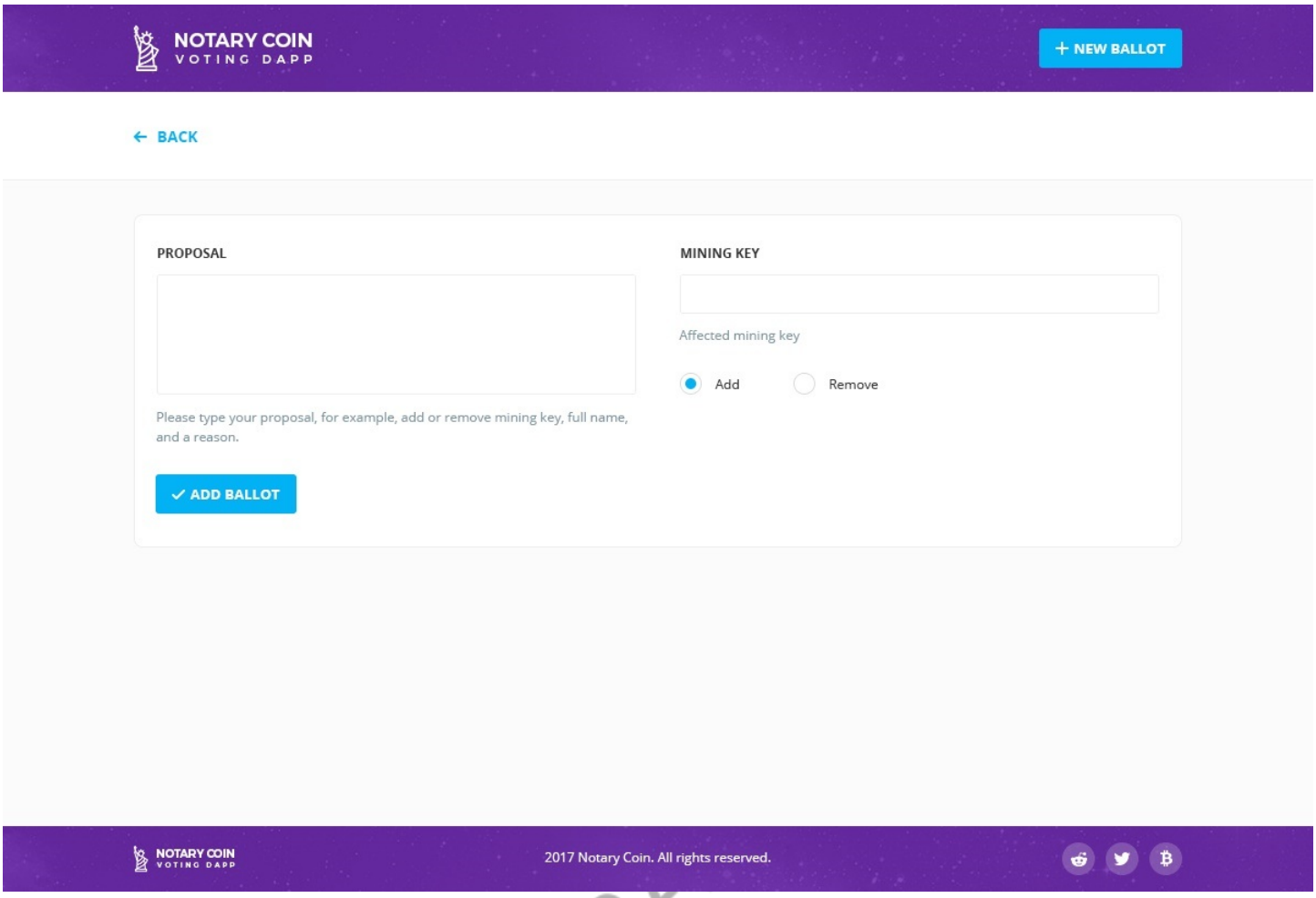


Voting screen

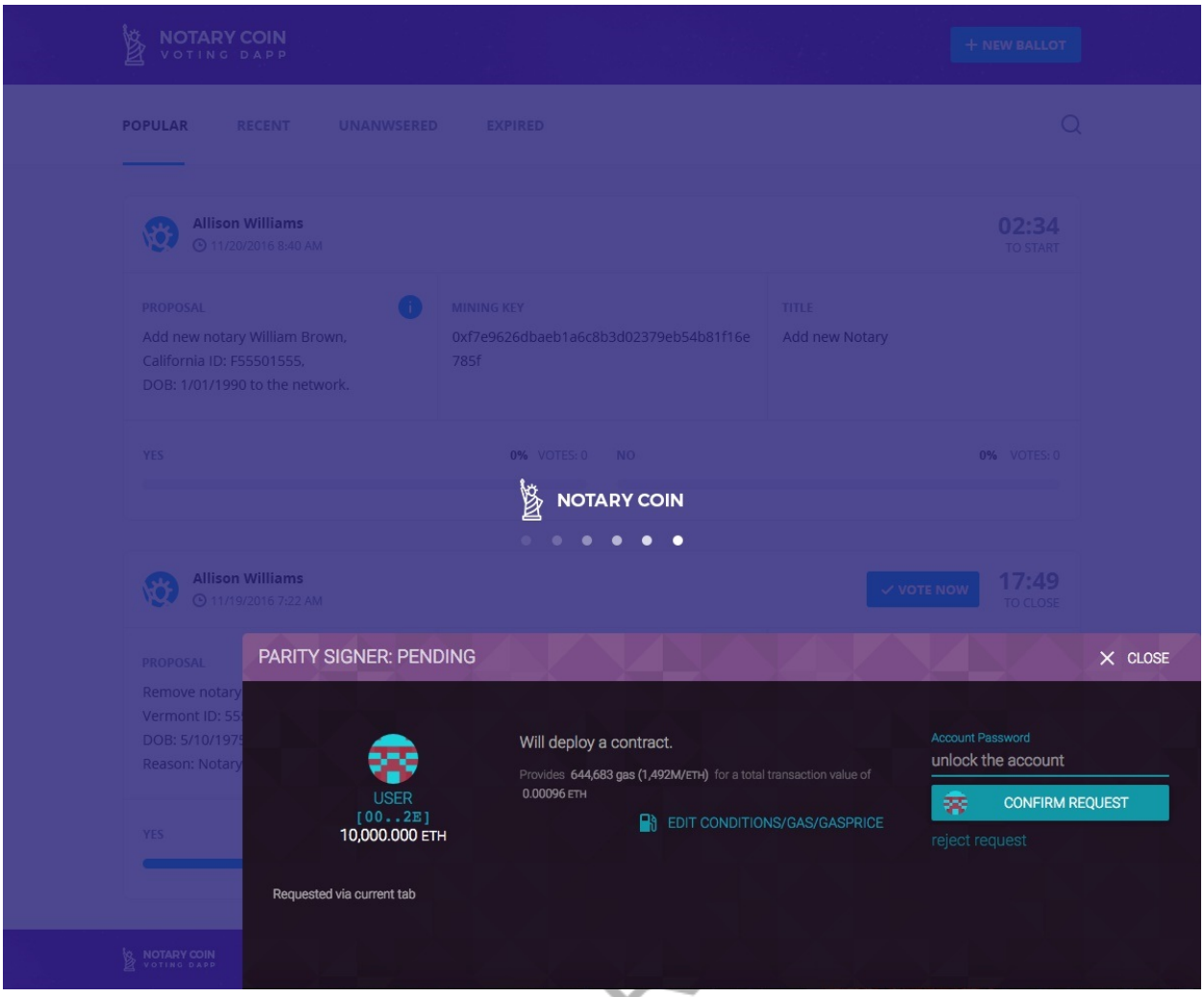
On this screen the voter may re-examine voting options and sign a transaction using his private key with agreement or disagreement with the proposal .



Add Ballot

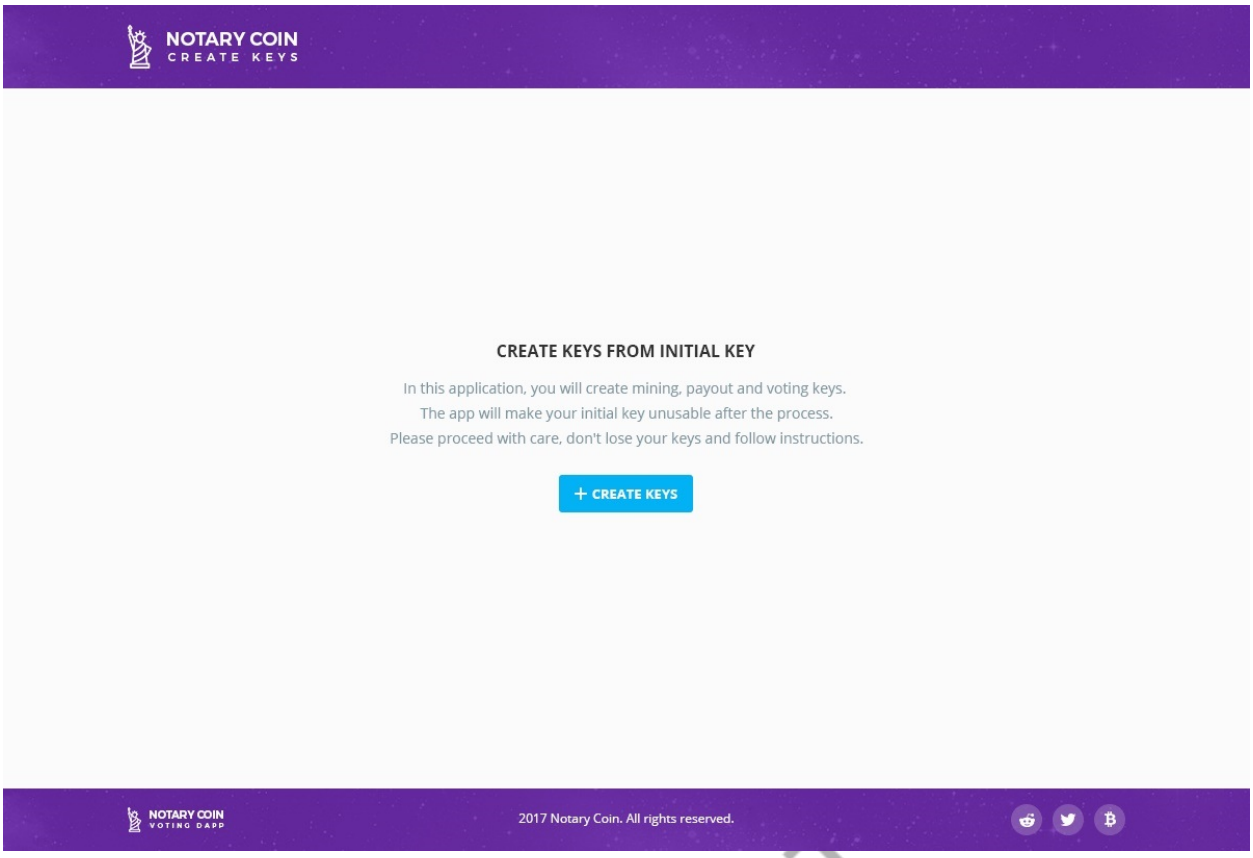


Sign a new ballot with Parity

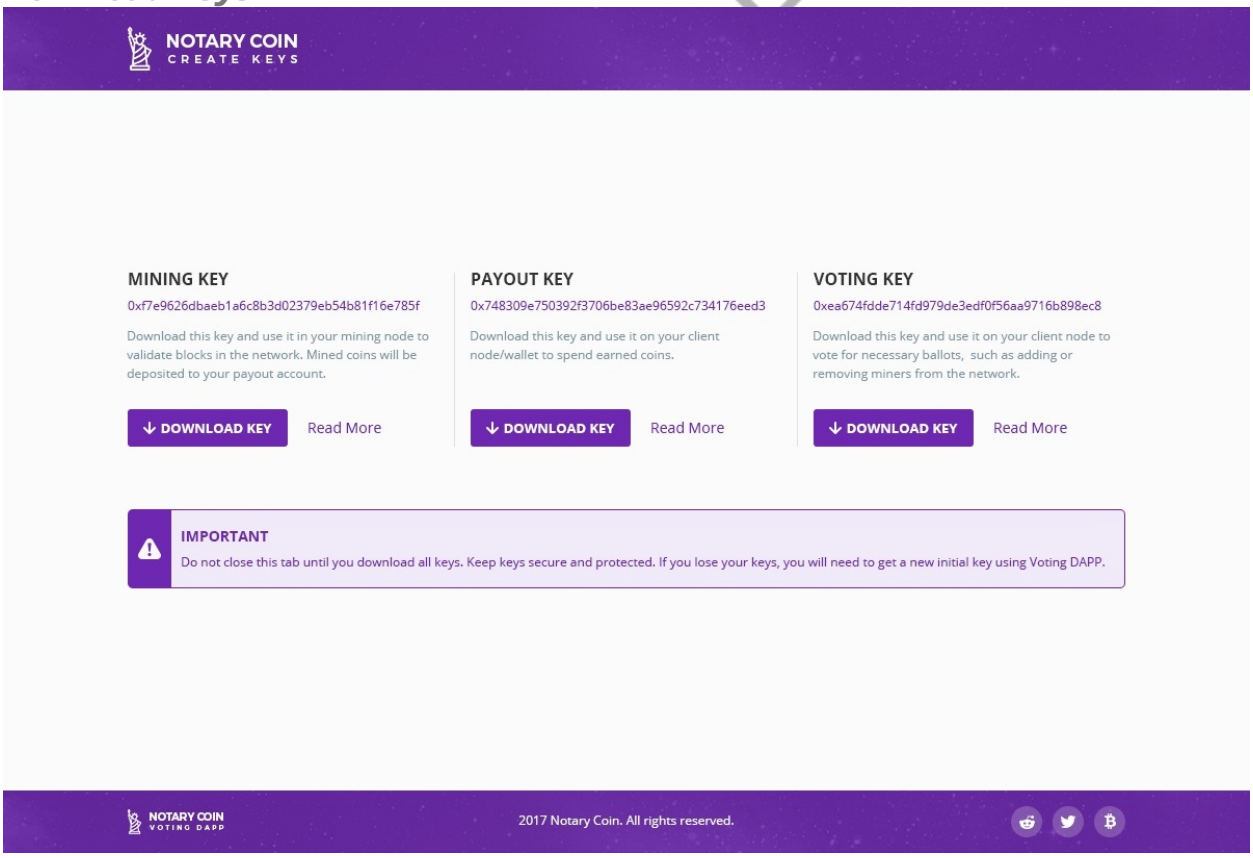


Create Keys DAPP

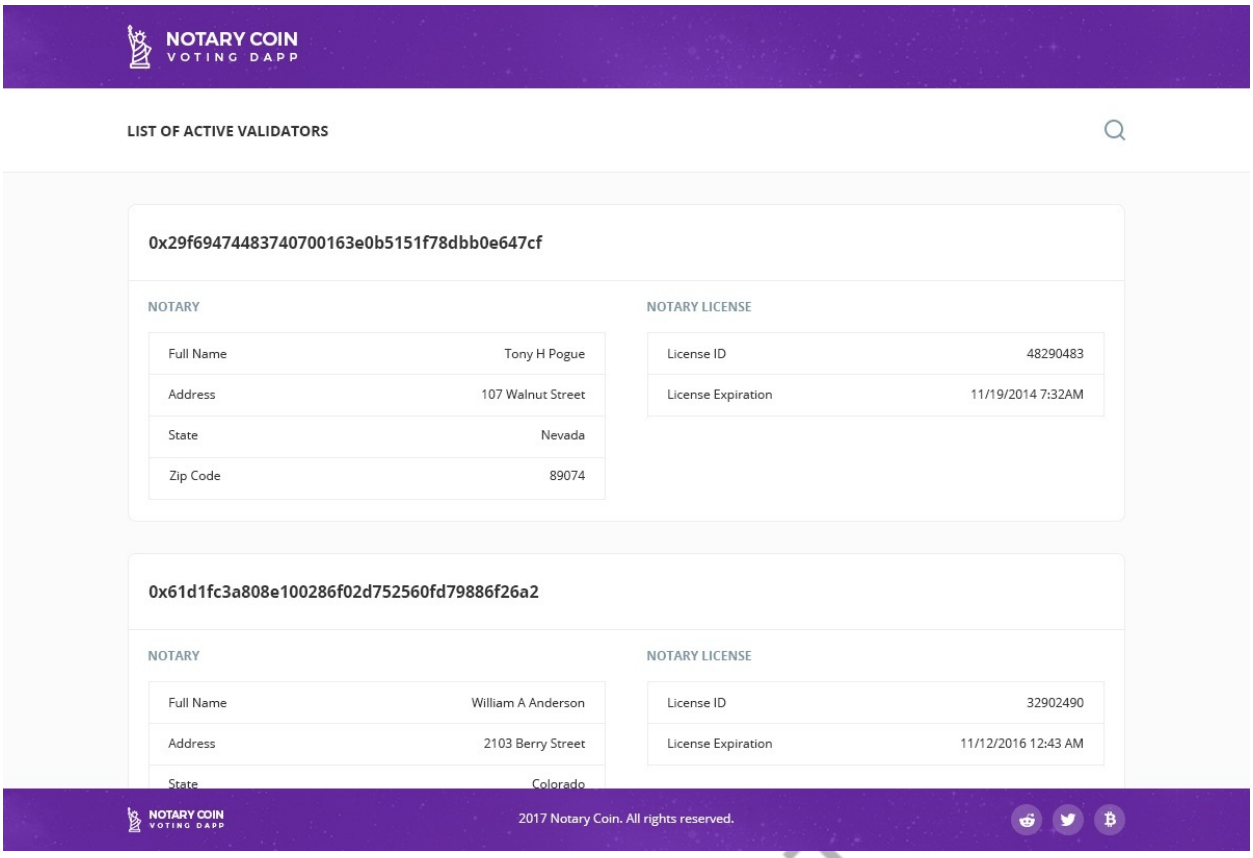
Preview of DAPPs logic: https://youtu.be/FCRDgSIe7GA



Download keys



List of Miners DAPP


NOTARY COIN
VOTING DAPP
LIST OF ACTIVE VALIDATORS
0x29f69474483740700163e0b5151f78dbb0e647cf
NOTARY
Full Name
Tony H Pogue
Address
107 Walnut Street
State
Nevada
Zip Code
89074
NOTARY LICENSE
License ID
48290483
License Expiration
11/19/2014 7:32AM
0x61d1fc3a808e100286f02d752560fd79886f26a2
NOTARY
Full Name
William A Anderson
Address
2103 Berry Street
State
Colorado
NOTARY LICENSE
License ID
32902490
License Expiration
11/12/2016 12:43 AM
NOTARY COIN
VOTING DAPP
2017 Notary Coin. All rights reserved.