Implementation Phase of Decentralized Voting Audit Solution for Latin America

Contest

March 15, 2021 – Jul 14, 2021 at 23:59 UTC. There will be a 24-hour countdown clock on the last day of possible entry.

Voting cycle:

20 days

Background:

The auditing process of Guatemala's voting results has off-chain flaws that require a comprehensive on-chain solution. The authors of the current Guatemalan volunteer-based voting audit process have partners in other countries such as El Salvador, Ecuador, Honduras, and several other Latin American countries where voting audits face similar problems. Those problems are based on an old system where the paper is trusted Acta#4. Pictures of this Acta#4 are given to volunteers on a flash drive, which means potential manipulation before this information is provided to volunteers. To combat this issue, Carlos Toriello Herrerias, a Guatemalan activist and blockchain enthusiast and an avid proponent of blockchain technology, created an app to help mitigate this problem; however, it is still an imperfect system since the incoming information is potentially compromised. All of the details are described here.

In November-December 2020, the first phase of a "Decentralized Solution for Voting Audit for Latin America" contest took place, aiming at crowdsourcing solution ideas.

Base on the contest's first phase winner's submission 3, a more formal specification was developed to be implemented by contestants (see Addendum).

Requirements:

To develop a DeAudit smart contract system based on the specification in Addendum.

DeAudit web-based explorer should be developed following the specification.

Should include DeBots for all system user interfaces.

Should include auto-tests designed as a smart contract or a script to test scenarios.

A solution should have a Free Software license. (Various Licenses and Comments about Them - GNU Project - Free Software Foundation 1).

A system should be deployed and tested on the DevNet, and Jury should be able to access it for testing.

Evaluation criteria and winning conditions:

All actions inside a solution should be easily accessible via DeBots interfaces.

A solution should pass the attached tests.

If a test does not cover some scenarios from requirements, then jurors can develop their own tests, but it should reduce such a submission score.

The solution should be scalable to millions of participants.

Voting:

Jurors whose team(s) intend to participate in this contest by providing submissions lose their right to vote in this contest.

A jury from other sub-governance groups could be added to this contest to provide additional technical expertise.

Each juror will vote by rating each submission on a scale of 1 to 10.

Jurors should provide feedback on each submission.

The jury will reject duplicate, subpar, incomplete, or inappropriate submissions.

Reward:

Only submissions with an average score equal to or more than 4.0 can get a reward.

1st prize……200,000 TONs

2nd prize…… 150,000 TONs

3rd prize…… 100,000 TONs

4th place …… 75,000 TONs

5th place …… 50,000 TONs

6th-10th place ……25,000 TONs

Total prizes: 700,000

Note: If the number of winning submissions is less than the number of rewards available, any remaining rewards are not subject to distribution and are considered void.

Jury rewards:

An amount equal to 7% of all total tokens actually awarded will be distributed equally between all jurors who vote and provide feedback. Both voting and feedback are mandatory in order to collect the reward.

Governance rewards:

An amount equal to 2 % of the prize fund will be allocated to members who participated in organizing the contest, to be distributed equally among them:

@prigolovko

@chuck_bogorad

@Ronmillow

@Futurizt

@blazingangels

@emmorozov

@gdache

@carlosguate

@noBapnymuHa

Procedural remarks:

Participants must upload their work correctly so it can be viewed and accessible in the formats described. If work is inaccessible or does not fit the criteria described, the submission may be rejected by jurors.

Participants must submit their work before the closing of the filing of applications. If not submitted on time, the submission will not count.

Addendum: Specification of a decentralized solution for voting audit for Latin America

NOTE: A participant can make implementation changes against the specification below, but they must support all described user functionality.

Glossary

DeAudit – decentralized elections audit, the process described in this specification.

DeAudit library (DAL) – a unique smart contract containing all code for all DeAudit system smart contracts, used purely for the client-side application's convenience to get the required code to deploy.

AuditDapp – a web-based and/or mobile application (at contestant's choice) designed to perform specific actions in the DeAudit process, which is inconvenient or impossible to perform using DeBots interfaces. AuditDapp should include DeAudit explorer functionality described in section 12 below.

Action Team (AT) – a reputable group of people from a country that initiates an DeAudit.

SMV – soft majority voting, see Declaration of Decentralization 1.

AT smart contract (ATSC) – an SMV-based smart contract system (see link 1to the reference implementation), which has additional specific functionality described in this specification. ATSC represents AT and could be a root for several DeAudit smart contracts. Details of its deployment are presented below.

ATSC should have a DeBot interface in parallel with AuditDapp.

Participant's smart contract (partSC) – main "user interface" smart contract for participation in DeAudit. Should have a DeBot interface with maximum possible functionality in parallel with AuditDapp.

Act4 – a paper form called Acta#4 filled by hand by election officials and volunteers.

Act4 collation smart contract (A4SC) – a smart contract deployed from partSC representing some reference to Act4 and data from it, including List of candidates(LoC), the number of votes, and additional Act4 information. A4SC sends results to DASC after the end of DeAudit, only if validators approved data. Any AT member should be able to trigger that transaction.

DeAudit smart contract (**DASC**) – DASCs are deployed from ATSC after AT voting in multiple instances.

DeAudit data smart contract (dataSC) – a smart contract which contains all parameters for DeAudit and deployed by ATSC. Also, this smart contract receives messages from A4SCs and sums them up into the election results. It calculates % of Act4 received and % where consensus was reached, which affects rewards calculation based on data collected from A4SCs and centSCs.

Democracy Token (DT) – the main reward token for participants in DeAudit.

Voting center (**VC**) – any location where citizens can come to vote, with the issuance of Acta#4 at the end of voting.

Voting center smart contract (centSC) – a contract which acts as a root for A4SC deployment. Anyone can initiate centSC deployment at an address calculated as a hash a centSC code and VC number from a DASC VC ledger.

Collator – anyone who initiated new Act4 verification and locked a stake for it.

List of candidates (**LoC**) – the list of election candidates approved by AT prior to deployment of DASC.

Validator – any randomly selected person deployed a validator's smart contract and registered it in DASC following the process described in this specification.

Collator-validator cycle (C-V cycle) – one cycle of Act4 collation and subsequent validation.

SMV base – several validators allocated to a VC. For s series of C-V cycles, SMV base is calculated as a number of all active validators (those who voted) during previous cycles and several validators allocated during the current cycle.

Pre-audit process

First of all, an Action Team (AT) should appear and deploy an AT smart contract (ATSC). In addition to standard SMV smart contract system functionality, i.e. allowing adding/removal of members, ATSC should allow:

Vote on DeAudit initiation following SMV principles. Any AT member should be able to initiate such voting.

Send trigger transactions to A4SC by any AT member.

Technically there could be several ATs, and they will be able to decide which group to support depending on its public reputation.

DeAudit initialization

DeAudit process could be launched by ATs voting. As a result there will be DASCs (see below) and dataSC deployed with the following fields:

Sequential number of DeAudit initiated by ATSC;

List of candidates (LoC);

List of voting centers (LoVC);

Time of audit start;

Validation period, collation period;

Collation base reward in Democracy Tokens (CBRwd);

Validator’s base reward in Democracy Tokens (VBRwd);

Collator’s min stake in TONs (ColStake);

Validator’s min stake in TONs (ValStake);

Additional data fields in Act4. It should be extremely strict and clear format to prevent misunderstanding of collators and validators.

Max allowed number of C-V cycles;

Number of bits used to address DASCs from partSC.

DASCs are deployed in multiple instances: number of times equal 2^number of bits in (l)=DAN. This is required for the sharding partSC registration process.

Each DASC will contain at deploy only the following parameters (Stateint):

Address of dataSC;

Address of ATSC.

A sequential number from 1 to DAN

Address of dataSC is also should be added into ATSC ledger (DeAudit ledger), thus for each DeAudit it is possible to calculate all DASC addresses knowing only ATSC address.

Additional decisions about new C-V cycles for a specific DeAudit should be written to DeAudit ledger of ATSC and into dataSC.

Democracy Token

TIP-3 ‘Democracy Token’ (DT) should be the main reward issued based on the principles described below.

AT can be the same for many election cycles. Issuance of DTs is not limited, but it is tied to some real work to be done by people, i.e. its supply is limited and it could have monetary value.

Validators and Collators should get 2 types of tokens a reward: transferable (DT1) and nontransferable (DT2). Names could be offered by a contestant.

If society supports DeAudit then merchants can accept DT1 as payment to show off their civil position.

DT2 could be used later in future applications where reputation based on participation in DeAudits can have some value.

Participants

Anyone can become a participant of DeAudit. For this purpose, a smart contract (partSC) should be deployed.

PartSC can collate Act4 or register as a validator by sending messages to one of DASC with address calculated based on first XX bits of partSC address (as stated in dataSC).

PartSC should have a DeBot interface to be accessible inside any front-end supporting DeBots. At the same time if a contestant believes that there are no required DeBot interfaces available for some described functionality, then this part can be developed only in AuditDapp. A contestant will be required to push request for missed interfaces to DeBot consortium (link 1)

The collation phase

Any participant can collate Act4. It could be people on the ground in Guatemala, at voting centers who will be able to take a photo of an original Acta#4; volunteers, temporary workers, and witnesses.

To do a collation a participant should send a message from partSC to centSC with min ColStake. The address of centSC could be calculated based on a voting center number and centSC code. If centSC is not deployed partSC should deploy it first.

Prior to the collation of Act4 a collator should upload a picture of Act4 and all other photo evidence that Act4 is original (voting site photo, on premise witnesses photo) in any standard format using an AuditDapp/DeBot into any decentralized storage (at a choice of a contestant).

CentSC deploys A4SC and writes inside its ledger partSC's address as a collator.

Deploy message should contain:

Min ColStake;

Hash and a reference of Act4;

Candidates numbers of votes from Act4;

Additional mandatory information from Act4 as required by DataSC.

Prior to deployment AuditApp/DeBot should check previously collated Acta4 for this centSC and notify a user that her collation will be the number X.

If a collator will upload a new “Act4 evidence”, then she must supply a comment detailing the incorrect information in the previously uploaded form.

CentSC should not accept a new collation if input data (votes per candidates and other required supplementary numbers) is identical to one of existing collated Act4. The idea is that a collator should provide alternative evidence only if previously collated information is incorrect.

A contestant can propose and develop in her solution a mechanics for adding more evidence to previously collated Act4 (co-collation).

The validation phase

Registration

Registration is required for several purposes:

To prevent an attack where someone could try to calculate and deploy partSCs to be linked to a particular centSC (address mining).

To have an exact number of validators to calculate SMV base for each A4SC.

Anyone can participate in DeAudit as a validator. To be registered as a validator, a user should send from partSC a message with ValStake to a corresponding DASC (based of first bits of partSC address), which triggers the following:

DASC deploys ValSC with the following statInit = ValSC code + seqno.

DASC send to new ValSC address a message setting up a parSC address as an owner.

DASC returns back to partSC its seqno, thus partSC will be able to calculate its ValSC address.

DASC increases seqno.

This functionality should be accessible from AuditApp and/or DeBot.

Validator registration is available any time during the collation phase but stops when the validation phase starts.

Validation

When the validation phase begins centSCs should stop accepting new collations.

A contestant should provide justification of a statistically reliable number of validations per 1 centSC, which should depend on the number of registered validators.

Anyway, any registered validator should have a possibility to participate in not less than 3 validations to promote involvement and trust in DeAudit.

When a validation phase of DeAudit starts ATSC should publish a cycle random number (CRN), which should be used for the calculation of attribution of registered validators to centSCs.

ATSC gets the number of all registered validators for a particular C-V cycle by receiving all seqno from all DASCs = cycle validators number (CVN). Using CRN and CVN everyone should be able to calculate this

voting center – validator link. The exact algorithm should be proposed by a contestant, but all validators should be divided between centSCs in an equal, deterministic, but random way.

AuditDapp (and DeBot if it will be supported) should be able to calculate it and notify the user that she has the ability to start validation. AuditDapp/ DeBot should provide the functionality to view Act4 any registered data from A4SC to users. During the validation cycle a user can select one valid collated Act4 and confirm that data in A4SC is corresponds to a photo or select to reject all collated Act4.

When centSC receives a message from partSC it should check first that this partSC is a validator linked to this voting center by calculation based on CRN and CVN. Then centSC sends received votes for and against into A4SCs.

Interim slashing phase

Collators interim slashing

If for any cumulative number of C-V cycles number of votes against some collated Act4 more than a SMV supermajority threshold then A4SC should be destroyed, its address should be excluded from the ledger in centSC and collator’s stake will be slashed, i.e. should be sent to ATSC balance.

Validators interim slashing

When the validation phase begins, all validators’ stakes should be transferred from DASCs to corresponding centSCs in amount roundup for total number of registered validators/number of VCs. It could be a bit more than stakes provided by validators, thus DASCs should have a reserve.

At the end of the validation phase stakes should be sent back to all validators who voted. Other stakes should be slashed, i.e. should be sent to ATSC balance.

Collation-validation cycles

The purpose of DeAudit is to select in a decentralized way one Act4 for each VC and confirm the correctness of input of election data. The ultimate success of DeAudit is achieved if for all voting centers (VCs) Act4s were collated and for all VCs validators come to a consensus and selected 1 Act4 in terms of SMV.

In reality, there could be gaps in Act4 collation as well as there could be insufficient validators activity. In such a case AT can vote to run a new collation-validation cycle.

Amendment of SMV base

If AT decided to have another C-V cycle, then a SMV base for all A4SCs should be reduced by a number of non-active validators during the previous round. So for the next round the total SMV base will be calculated and the sum of all previously active validators and the current number of allocated validators.

Reward phase

If AT does not vote for a next C-V cycle or the maximum number of C-V cycles have been reached, then DeAudit is over and the reward phase is started.

Collators who were not slashed get rewards in DTs in the amount equal to CBRew ^ (1+rounddown((% of Act4 received – 50%)/10%).

Validators who were not slashed get rewards in DTs in the amount equal to VBRew ^ (1+rounddown((% of Act4, where consensus was reached – 50%)/10%).

The idea is to provide extra rewards for a joint community effort to get maximum Act4 uploaded and verified.

Audit slashing

This section describes a process called Audit slashing. This is not a normal process for a DeAudit, but it should exist as a protection measure

from “buying of a DeAudit”. The existence of such an outcome should make “buying of a DeAudit” useless and a costly venture, which by itself reduces the likelihood that someone will try to spoil DeAudit.

But still, there will be a chance that some wealthy group will decide to collate and validate the wrong set of Act4s. To do so this group will require, on average, 50% of all validation power.

If it becomes obvious for and AT that this what happening, AT can vote to initiate a DeAudit slashing.

This decision should freeze all stakes and processes in centSCs. After that, a trusted verification should happen.

ATs will vote for a closed list of trusted validators. Such validators will have to vote for and against A4SCs. Results of such vote should be used to return back of stakes of those collators/ordinary validators who provided/voted for correct Act4s. The rest stakes will be slashed, but not transferred to IDSC as in other sections slashing but should be distributed among honest collators and honest ordinary validators based on their staked amounts.

Audit explorer

The following minimal visualization capabilities should be a part of AuditDapp and/or DeBot:

List of ATSCs (searched by the hash of a code)

For an ATSC:

All started and passed DeAudits

DTs issued

Locked in stakes in TONs

For a DeAudit:

Start and finish time

Stats of a current DeAudit:

Number of centerVSs where Act4 was collated even once and % of all VCs;

Total number of collated Act4;

Number of validated Act4 and % of total VCs;

Current voting calculation results.

Number of registered and slashed validators and

Number of collators and number slashed collators

Audit slashing process indicator

For Audit slashing:

Metrics related to Audit slashing

There should be a search string for an address. If that address belongs to any DeAudit smart contract system, it should be displayed with related information.