7th Estate Technical Diagram Explanation

The complete detailed step-by-step recipe, based on the ingredients in the introductory documents, is explained and detailed here with reference to the technical diagram and the eight steps enumerated in the frieze across its top.

Step 1.

The “voter roster” (shown light blue) is a complete list of all potential voters who can cast a ballot in the election. Each potential voter should appear as an entry in the roster, and of course no voter should appear more than once. The roster is posted on a blockchain so as to be immutable. Each voter entry consists of the voter name and mailing address. The first voter entry is given a sequential position number of zero, the second is given position number one, and so on all the way up to the number of voters minus one, 9999 in the example.

The names of voters can be used to order the voter entries alphabetically, which makes it easy to verify the absence of duplicates when names are unique. Lists of registered voters are typically available to political parties from governments, through brokers, and increasingly simply posted online; lists of households or residents are more widely available for commercial purposes.

The “committed summands” (shown green) are one cryptographically-generated random number for each of the 1000 ballots. The summands are in the example each a whole number between 0 to 9999, but no two summands should be equal. There is one summand entry for each voter that will be selected. They serve to randomize which voters are selected, much like a flipped coin on a wrist that is still hidden under a hand before a coin flip outcome randomizes the eventual coin flip. The encryption algorithm ideally is such that there is exactly one key that allows each entry to be decrypted and thereby to reveal the value that was “committed” to by that encryption; thus, each entry can be decrypted in one way only and locks in the value committed. The committed summands should be stored immutably on the blockchain before step 2; they need only be decrypted to allow audit of who ballots were sent to in step 8.

The columns 1 and 3 (shown yellow in the diagram) are each in effect 50 lists of 2000 individually encrypted entries, for a combined total of 200,000 encryptions. The encryption algorithm used for these column entries, like that just described for the committed summands, should ideally lock-in or commit each value. Unlike the committed summands, here each entry in each column and plane is encrypted with a different key. The first “plane” posted on the blockchain in this step consists of columns [1,1] and [3,1]. It contains 4000 encryptions, each of a value like that shown: ballot number and a single vote code for column 1 and “no,” “yes, or “decoy” for column 3. There are 50 planes, each containing exactly the same information, but arranged in a different order called the “plane permutation.”

Each plane permutation is the same for all data in that plane (columns 1 and 3 described here, as well as column 2 to be described with reference to step 6). These plane permutations are created in effect unpredictably at random. For instance, the values of a cryptographic pseudorandom sequence, determined by the election key, could be used to select a ballot number and one of its two “ballot positions” (one for “yes” and the other for “no”) to fill each successive row, but those positions already selected are skipped to give a uniform probability of each permutation being selected. (More efficient though more complex algorithms are well known.) The plane permutations themselves need not be committed to, but rather are used consistently within each plane, and this

consistency allows all the opening of column commitments in step 6 to result in mutually consistent data.

There are 1000 ballots in the example and each ballot has its two positions, “yes” and “no” printed adjacent its respective “vote code.” The vote codes can be generated cryptographically using the election key in a manner that makes them each in effect independent of each other and the other generated values, including the committed summands and plane permutations. Furthermore, each vote code can be unique; however, separating ballot serial numbers from the vote codes may also be more familiar to voters.

Vote codes ideally would have a redundancy structure encoded in them, not shown in the diagram examples to keep them compact. This redundancy can make data entry errors by voters very likely recognizable by the electronic polling place servers, allowing voters to be asked to correct their entry errors. Apart from the redundancy, however, there should ideally be enough valid codes satisfying that part of the structure known to polling places for an electronic polling place server to have more than a negligible chance of guessing what will turn out in the tallying of step 8 to be a valid vote code. The underlying vote codes, apart from the data-entry-protection redundancy, should be generated cryptographically so as to be difficult to predict by those who may wish to flood the system with random fabricated votes. Vote codes in practice could be about 20 digits long; about the same difficulty for voters to enter as a payment card number, but large enough that getting the system to accept guessed codes would in effect be similar to a so-called “denial of service attack” (something addressed further with reference to Step 5).

Each plane of columns contains a row for each position. The column 1 entries for a row contain a ballot serial number and a vote code; the column 3 entries for that row contain the respective one of the words “yes,” “no,” or “decoy.” Each real ballot has a single “yes” and a single “no” row, whereas each decoy ballot has two “decoy” rows. (The two positions for a particular ballot are in effect independently and randomly determined by the plane permutation, and so will likely not appear adjacent and apparently be completely randomly located.)

Step 2.

The “drawn summands” are in the same quantity and over the same range as the committed summands: in the example, 1000 rows each containing a different number between 0 and 9999. Step 2 in effect makes a verifiably random selection of voters, those who will receive ballots, from the voter roster. To accomplish this, the drawn summands are chosen in a publicly verifiable and independent manner, but only well after the voter roster and committed summands have been immutably fixed on the blockchain. Within the blockchain paradigm, the draw can be as simple as the hash of the root of a pre-designated subsequent block. This hash can readily be turned into a list of drawn summands by a fixed public method. For instance, a pre-defined cryptographic pseudorandom sequence generator, such as one that uses successive hashing, produces a list from which exactly the first suitable values are extracted. The result of this step is that the drawn summands are added modulo 10000 to the committed summands, with the resulting 1000 distinct values, each between 0 and 9999, being used to select an enry from the roster of voters.

Step 3.

This step separately prints both the addresses and the ballots, so that they can be randomly paired once the order of each is lost through physical tossing and optional shuffling in step 4. The address printing prints the voter names and addresses that were selected in step 2 from the roster, in the example 1000 names and addresses. The addresses are printed in the example on labels that are then affixed to the envelopes and hidden under opaque tape before the ceremony.

The ballots are printed on index cards in the example. (Ballots printed on paper may be more difficult to physically randomize in step 4.) The serial numbers of ballots are simply chosen sequentially, in the example here, from 1 to 1000. (Serial numbers are used in the example for clarity, as mentioned already with reference to step 2.) The other information to be printed on each ballot, apart from the fixed voter instructions, is the pairing of “yes” and “no” vote indications with the vote codes. Printing the vote indications in a random order per ballot is known, at least in conventional elections, to provide a more unbiased choice by voters. The codes on ballots are hidden by scratch off, which has the dual advantage of hiding them before mailing and providing evidence of which if any codes were revealed after mailing.

A website collecting votes should be able to provide feedback to voters in case the voter makes an error while copying the code from the printed ballot to the online form (as mentioned earlier, however, such what might be called redundancy in the codes should not be large enough to make it too easy to guess valid codes, and of course should not reveal the corresponding voter choice.)

Step 4.

The “ceremony” conducted in this step by a number of “attendees.” It results in placing ballots unpredictably in pre-addressed envelopes and then in postal collection boxes. It also provides some auditing. Evnelopes are verifiably randomly-associated and stuffed with ballots. The envelopes have addresses hidden by tape, and after stuffing but while hidden from view, the tape is removed from the envelopes and they are inserted into the postal mailing system.

The first type of audit, the “public audit,” is attendee-verifiable random selection of some completed stuffed envelopes and ballots and the revealing of their printing. Address as well as ballot serial number and vote codes of these mailers are displayed to the attendees. All the encrypted information on the blockchain related to these particular ballots, but only that related to these ballots, is then decrypted by those with access to the system key. The decryption allows verification that at least all of this printing was correct. This type of audit has elsewhere been referred to as a “print audit” as it serves to check that, at least for those audited ballots, vote codes were printed properly associated with vote indications according to the commits of columns 1 and 3 on each of the fifty planes. The physical ballots audited can be destroyed, in the example by shredding. The publication of serial numbers prevents them from being accepted as valid in step 5.

The “private audit” allows the auditors to take half the vote codes and half the serial numbers of some ballots; the other halves of these ballots are shredded publicly. A halve taken is enough to allow any one private auditor to show that a particular ballot was not mailed, but the halves are not enough to allow the private auditors even working together to vote any of them.

(One optional type of audit, the “private audit,” lets auditors selected from the attendees learn the address on some ballots that were otherwise destroyed without anyone being able to learn more about those ballots; this then lets these auditors later check with voters to make sure that the voters were not contacted by those with access to secret keys or ballots.

(Another optional audit variant allows the ballot to be sent voters, with only those auditors checking with those voters to ensure that the ballots were received—reducing the need to make the set of voters pubic. If roster entries were encrypted, auditors should also be able to check at least the respective encrypted roster entries.)

A third type of audit, “audit of decoy,” is used in the case of decoy ballots. The decoy ballots, owing to their potential scarcity, are audited before they are placed in the remaining envelopes. They may

later be selected in the public audit, but if selected for private audit they should be easily recognized as decoys by the extra thickness of the label and handled as part of the pubic audit.

Sep 5.

Voters vote their ballots online by supplying the serial number and vote code, both revealed by scratching off adjacent their choice. Votes can be provided to any one of the potentially more than one polling location online. These sites can inform voters when a vote code is invalid, as already mentioned with reference to step 3, but this redundancy will in practice not let them create valid codes or learn to which vote a vote code corresponds.

Step 6.

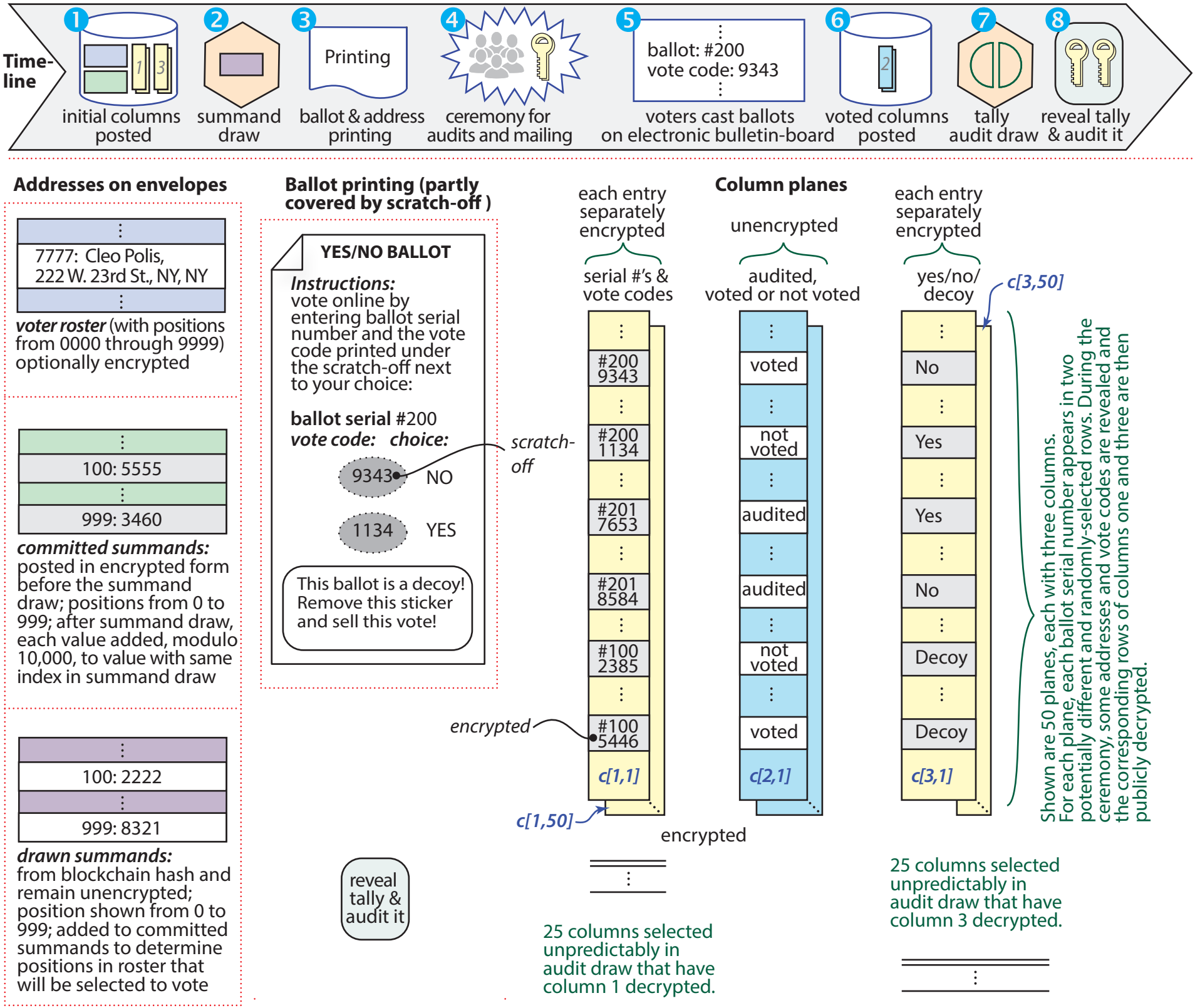
The vote codes from the online polling locations are posted to the blockchain. Possession of the election key allows the software of those conducting the election to compute from these the list of which positions have been “voted,” “not voted”; the positions marked “audited” were determined by the public audit during the ceremony of step 4. Next, for each of the 50 planes, the corresponding one of the 50 plane permutations, already described with reference to Step 1, is used to re-arrange the respective unencrypted list. Each such list is then separately posted in the clear to the blockchain. Thus, the software can use the system key to re-determine the plane permutations and ensure that the permutation of each center (blue) column 2 matches that of the corresponding two surrounding encrypted columns 1 and 3 (yellow).

Step 7.

Fifty random bits, in the example, are determined and made public. These bits should result from a process that is easy for anyone to verify but extremely difficult for anyone to manipulate, such as that used for the drawn summands. (Another example such process, one that has been used in public-sector elections, is an algorithm that combines a fixed set of published prices of publicly traded things at a pre-determined time after completion of step 6.)

Step 8.

In this final step, the fifty-bit sequence determined in step 7 is used by the custodian(s) of the election key to determine which of the two columns of each plane of first and second columns to provide decryption keys for: “0” bits in the sequence correspond to the left yellow column 1 having its decryption keys revealed for that plane; “1” bits in the sequence correspond to the right yellow column 3 having its decryption keys revealed for the plane. (Note that all entries for ballots audited have ideally already had their decryption keys revealed in public audit, allowing their decryption already in step 4, and some of these entries may be selected to have their keys revealed again here.)


Time-line
1
initial columns posted
2
summand draw
3
Printing
ballot & address printing
4
ceremony for audits and mailing
5
ballot: #200
vote code: 9343
voters cast ballots on electronic bulletin-board
6
voted columns posted
7
tally audit draw
8
reveal tally & audit it
Addresses on envelopes
7777: Cleo Polis, 222 W. 23rd St., NY, NY
voter roster (with positions from 0000 through 9999) optionally encrypted
100: 5555
999: 3460
committed summands: posted in encrypted form before the summand draw; positions from 0 to 999; after summand draw, each value added, modulo 10,000, to value with same index in summand draw
100: 2222
999: 8321
drawn summands: from blockchain hash and remain unencrypted; position shown from 0 to 999; added to committed summands to determine positions in roster that will be selected to vote
Ballot printing (partly covered by scratch-off)
YES/NO BALLOT
Instructions: vote online by entering ballot serial number and the vote code printed under the scratch-off next to your choice:
ballot serial #200
vote code: choice:
9343 NO
1134 YES
scratch-off
This ballot is a decoy! Remove this sticker and sell this vote!
reveal tally & audit it
each entry separately encrypted
serial #'s & vote codes
#200 9343
#200 1134
#201 7653
#201 8584
#100 2385
#100 5446
c[1,1]
c[1,50]
encrypted
unencrypted
audited, voted or not voted
voted
not voted
audited
audited
not voted
voted
c[2,1]
each entry separately encrypted
yes/no/ decoy
c[3,50]
No
Yes
Yes
No
Decoy
Decoy
c[3,1]
Shown are 50 planes, each with three columns. For each plane, each ballot serial number appears in two potentially different and randomly-selected rows. During the ceremony, some addresses and vote codes are revealed and the corresponding rows of columns one and three are then publicly decrypted.
Column planes
encrypted
25 columns selected unpredictably in audit draw that have column 1 decrypted.
25 columns selected unpredictably in audit draw that have column 3 decrypted.