A method for electronic voting with Coercion-free receipt

David J. Reynolds

(unaffiliated)

The central problem

1. How to get a DRE to properly encrypt a vote?

2. How to ensure encrypted votes are properly tallied?

Some Stricter Requirements

- End-to-end verifiable
  No ‘trust’ for integrity
  ‘Election authorities’ preserve privacy only

- ‘containment’ is distributed
  No one authority can expose a vote

- <u>no trusted computational devices</u>

  → Voter participates critically in verification

Expose fraud-in-collection using…

Chaum (optical)	---	Human optical skills
Neff	---	Temporal sequence
This system	---	Temporal sequence

How it works

Analogy

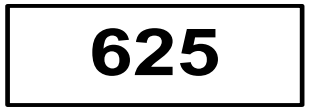
- Model DRE = ‘Collector’
- Collector has
  - invisible-ink pen = public key
  - invisible-ink writing = public-key encrypted
- Tallier has ‘magic-marker’
  - magic-marker = private key

- Meet with Collector
- Collector writes your vote using invisible-ink pen; you can't read invisible ink
- You can write in ordinary-ink, must not reveal vote
- Bring your vote to bulletin-board
- Tallier (privately) uses magic-marker to read invisible ink on your vote
- Can the Tallier detect fraud by collector?

YES!!!

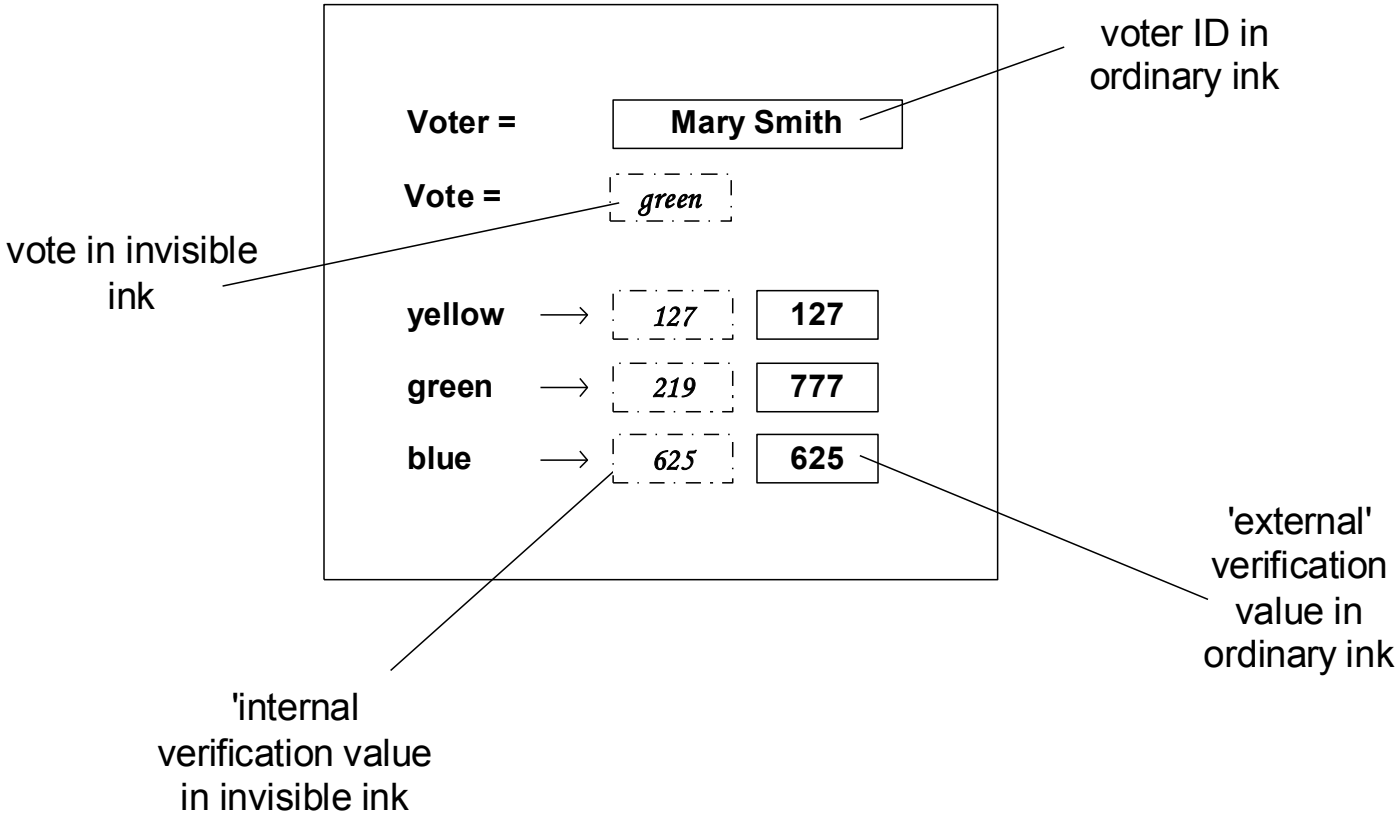
(convention)

Represents 625 in invisible ink
(= encrypted in public key)



Represents 625 in ordinary ink
(= plaintext)

Filled ballot (preview)



Terminology

- “On” = voted for
- “Off” = not voted for
- L options
- The ‘vote’ is the on-option
- The others are the off-options

- (K of L voting: K on-options, L-K off-options)

Polling process

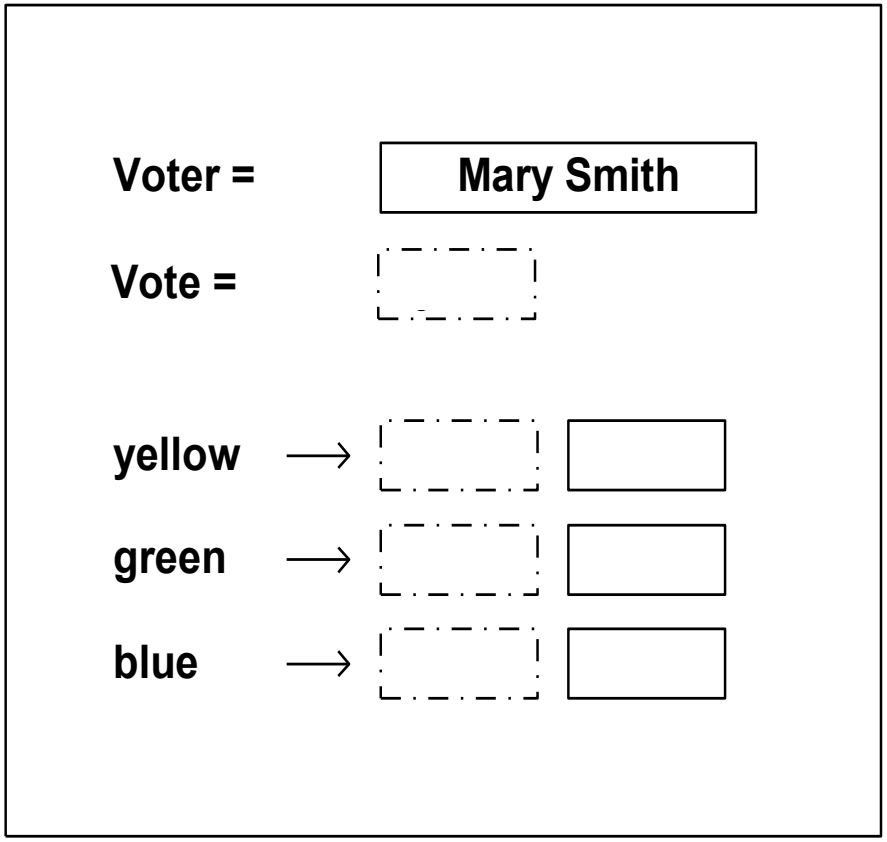
Voter announces vote=green

'Verification Phase 1': voter fills external verification values for off-options

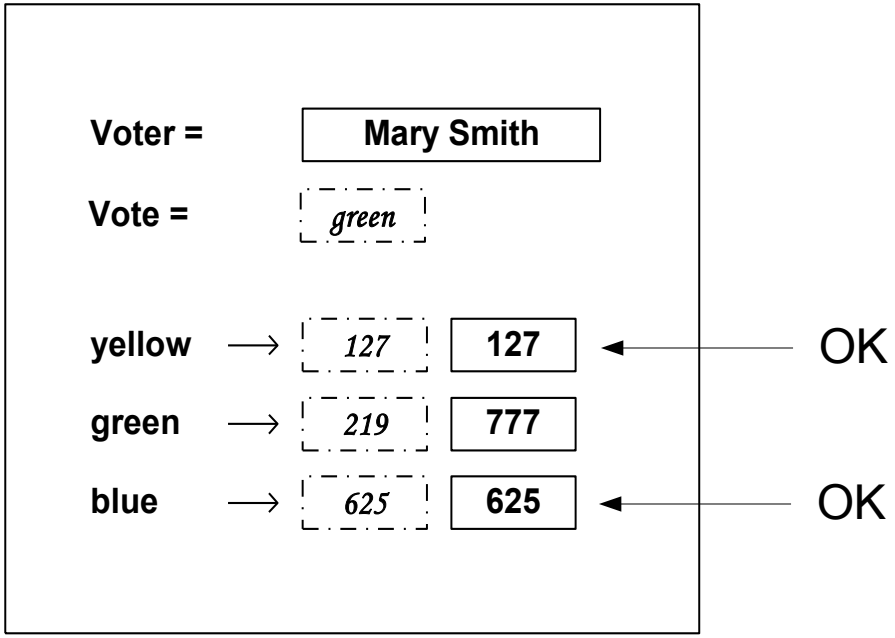
'Collector commit':

- **collector enters vote;**
- **copies external v.-values for off-options to internal**
- **Writes randomly-chosen internal v.-value for on-option**

'Verification Phase 2': voter fills external verification value for on-option



Verification process



Tallier checks that internal verification values equal external verification values for off-options

← 'Verification condition'

That's the method!!

The heart of the method

MUST MEET TWO CRITERIA

a) During verification/tallying, a condition is checked for each off-option (of the vote as encrypted)

b) The Collector can not* satisfy this condition for the on-option (of the true vote)

(*P_success = 1/1000)

That's all we need!!

- Fraud ➔ on-option of true vote = off-option of vote-as-encrypted

    a) … a condition is checked for each off-option….
    b) The Collector can not* satisfy this condition for the <u>on-option</u> (of the true vote)

a) is ensured by the tallying/verification arrangement

b) is ensured by the polling sequence and voter vigilance

Important feature

Voter just needs to

1) Ensure that the temporal sequence is OK (‘commit’ phase occurs before voter enters v.value for on-option)
2) That the v.value for on-option is as voter specified

Voter does not need to check verification-values for off-options

(Neff’s method has this feature too)

DRE & Coercion-properties

- Use identical UI and front-end receipting system to Neff's
- Requires printer with minimally-modified housing (commit must be seen to be made, but not readable)
- Fully coercion-free. Voter has full control over receipt outcome, regardless of vote.

Tallying methods

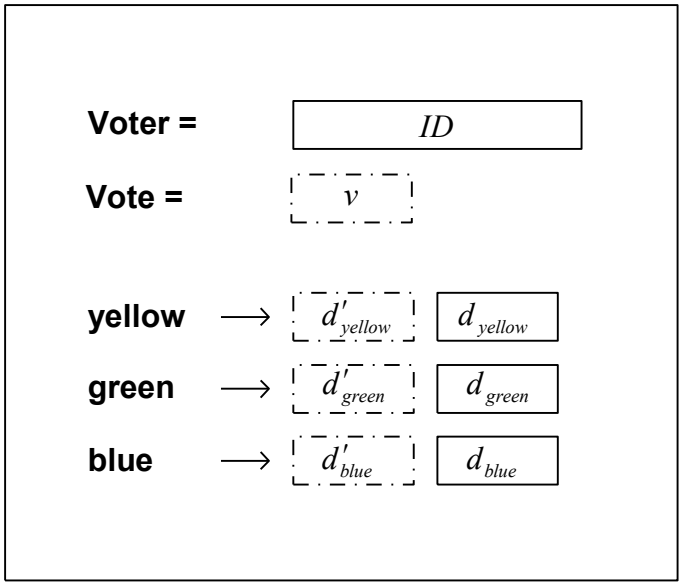
- Re-encryption mix-net
- Chaumian mix-net
- Without mix-net (with homomorphic encryption)

→ Complexity linear in L

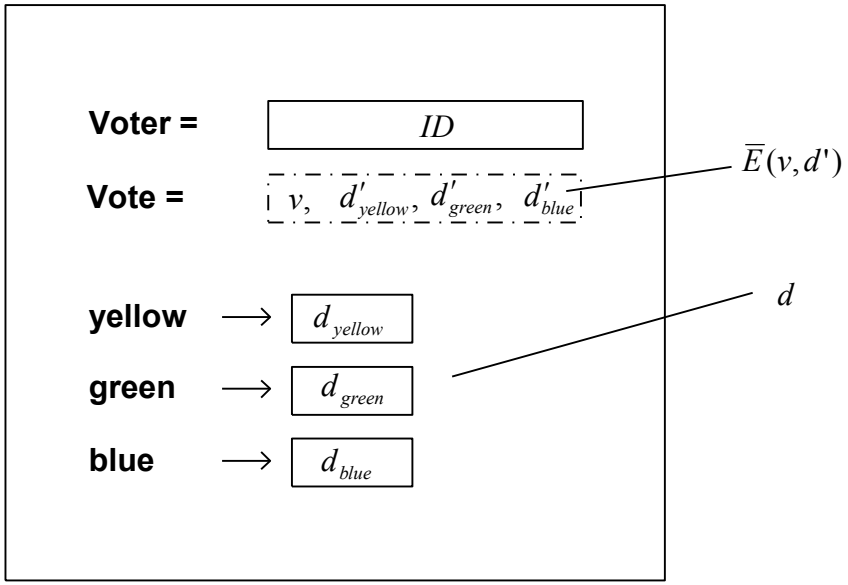
(Independent of K)

Notation

Layout in Analogy



True DRE receipt



Receipt is substantially:

ID, $\overline{E}(v,d')$, d

Homomorphic Tallying

Encrypting the vote

$$\overline{E}(v, d') \equiv \left\langle \bar{v}, \overline{d'} \right\rangle$$

$$\overline{d'_k} = E(d'_k)$$

$$\bar{v}_k = E(0), k \neq v \quad \bar{v}_v = E(1)$$

Encrypt vote as an L-tuple
(‘unitary’)

Homomorphic tallying

Proving the vote

a. Verification condition

DRE proves for each *k* in 1..L in Zero-knowledge

$$\overline{d'_k} = E(d_k) \text{ OR } \bar{v}_k = E(1)$$

b. Proving the vote 1-valued (long known method for 'unitary' approach)

DRE proves for each *k*

$$\bar{v}_k = E(0) \quad \text{OR} \quad \bar{v}_k = E(1)$$

To prove 1-of-L (not double-voted on issues)

Prove that the product of all $\bar{v}_k$ encrypts 1
→ simply reveal the randomizer of the product

This proving-1-valued is <u>linear in L</u>

Homomorphic tallying

Counting the vote

- **Trivially linear because of encrypting as L-tuple; all of the votes on options are encrypted separately**

  → Take the product of encrypted votes on each option (through votes of all voters) and Talliers decrypt result = total number of votes on that option

Adapting other methods to achieve homomorphic tallying, linear in L

- Assume DRE has already verifiably encrypted the vote

$$\bar{v}^* = E^*(v)$$

- Assume we can construct reasonable ZKP's of above form
- DRE encrypts vote again as L-tuple $\bar{v}$ (unitary) as specified
- Prove that the in the linear fashion shown above
- DRE proves that $\bar{v}$ encrypts same vote as $\bar{v}^*$

provides ZKP for each option *k* of the vote that

$$\bar{v}^* = E^*(k) \quad \textbf{OR} \quad \bar{v}_k = E(0)$$

Re-encryption Mix-net Tallying

Encrypting the vote

$$\overline{E}(v, d') \equiv \left\langle \overline{v}, \overline{d'} \right\rangle$$

$$\overline{d'_k} = E(d'_k)$$

$$\overline{v} = E(v)$$

Just need re-encrypt property

Re-encrypt. mix-net tallying

Proving the vote

a. Verification condition

DRE proves for each *k* in 1..L in Zero-knowledge

$$\overline{d}'_k = E(d_k) \quad \text{OR} \quad \overline{v} = E(k)$$

$\overline{v}$ Can now go into mix-net

Re-encryption mix-variant

- Leverage assumed homomorphic property to ‘subtract’ external from internal verifiers while they remain encrypted
- Results must travel with vote in mix-net
- Spares ZKPs from DRE, adds complexity to mix-net
- May be possible to reduce complexity by packing more than one number into 1 (familiar techniques)

(d_overall = d_1 + 1000 d_2 + 1000.1000 d_3)

Chaumian Mix-net Tallying

Encrypting the vote

$$\overline{E}(v,d') \equiv E^{onion}(v,-d')$$

Input-batch element:

$$\left\langle E^{onion}(v,-d'),d \right\rangle$$

Output-batch element:

$$\left\langle v,d-d' \right\rangle$$

Verification condition (on output element):

$$(d-d')_k = 0, k \neq v$$

DRE-Calculating ahead

- DRE can keep cache of calculations
- Assume voter often takes default verification-values for off-options
- ZKPs only need be calculated for on-option while voter waits
- Re-fill cache in separate thread

Conclusions

- Coercion-free verifiable system, very good security properties (p_detection=1/M)
- Tally with re-encryption/Chaumian mix-net or homomorphic encryption
- Homomorphic tallying linear in L

More material

- Search for ‘Reynolds’ on iacr’s eprint website
- www.iacr.org
- (Should be accepted soon!)