# Fair Voting Systems

Ben Smyth

*University of Birmingham*

February 10, 2025

**Abstract**

Generalising verifiability for a plurality of key holders uncovers an oversight. A single key holder can tally and announce the result, along with evidence, themselves; a plurality of key holders cannot. Their tallying procedure is three phase: An encrypted tally is computed, each key holder partially decrypts, and the (decrypted) tally is recovered. For a plurality of key holders, the tally is computed, rather than verified. I formalise soundness (tallies correspond to votes expressed in cast ballots) of that computation, along with completeness (such tallies must always be computable) and unforgeability (only credential holders can authorise ballots). We state each property as a game against an adversary, and show how intuitive constructions such as signed ballots can fail them.

*Keywords:* Voting; verifiability; fairness; cast-as-intended.

## Introduction

Public forums create legitimacy. Everyone can count raised hands. Secrecy creates a quandary: Decisions must reflect stakeholder will. In public, stakeholders vote openly. Legitimacy is irrefutable. Anyone present can determine the outcome. But in public, we aren't free. Conversely, private decisions protect freedom, but lack legitimacy, there's no visible proof.

Everybody lies. Don't trust. Verify. Decisions must demonstrably reflect stakeholder will. There must be an absolute path from decision to stakeholder votes. No one can be fooled into accepting the wrong result. Correct results are always accepted. Formalisations have proven elusive, these principles aren't captured by existing definitions [1, 2]. Voting systems vulnerable to attack can be proven to satisfy current formalisations. With Michael and Steven, I present a new one [3].

We consider a key holder computing a tally along with evidence of correct tallying. Those outputs are verified to determine whether the tally corresponds to votes expressed in cast ballots. We formulate reachability games capturing the aforementioned principles with respect to tallying and verification procedures. Generalising for a plurality of key holders leads to a surprising conclusion: Philosophically, we want fairness, not verifiability.

A fair voting system must ensure tallies correspond to votes expressed in cast ballots (soundness) and such tallies must always be computable (completeness). Whilst a single key holder can tally and announce results themselves, a plurality of key holders cannot. The idea that a tally is announced and verified doesn't transpose. For a plurality of key holders, tallying comprises three phases: An encrypted tally is computed, each keyholder partially decrypts, and the (decrypted) tally is recovered. Formalising fairness in terms of a verification procedure accepting a tally is natural with a single key holder, whereas such a procedure is irrelevant when a plurality of key holders compute partial decryptions. Instead, we want the tallying algorithm guaranteeing soundness in the third-phase, and that algorithm always computing a tally when public key shares are system generated (completeness). Herein I formalise these properties.

## Syntax

Key and credential generation each input a security parameter, the former outputs a public key share and a private key share, the latter a public voter credential and a private voter credential. Encrypted, authenticated ballot construction inputs public key shares, a voter's private credential, and a boolean matrix. (One-by-one for referenda, boolean array for approval and first-past-the-post, two dimensional boolean matrix for scoring and ranking.) Three-phrase tallying: Computing an encrypted tally on input of purported ballots, public key shares, and public voter credentials. (Tallying discards duplicate ballots, non-ballots, ill-formed ballots, unauthorised ballots, early votes, etc.) Each keyholder partially decrypting the encrypted tally with their private key share. (Keyholders re-tally the en-

crypted tally themselves, to avoid being duped into becoming a decryption oracle.) And, recovering a tally from an encrypted tally, partial decryptions, and public keys, or throwing an error if partial decryptions are insubstantial. Algorithms may take additional input including metadata, e.g., vote matrix dimensions, candidate names, and security parameters.

### Towards formalising fairness

Consider an adversary orchestrating a system execution with a tally differing from votes expressed in collected ballots; in the absence of such an adversary, we infer the voting system delivers voters' will.

*Soundness.* A voting system is sound when tallies match votes expressed in ballots: Any system computed tally from adversary computed inputs (public key shares, public voter credentials, ballots, and partial decryptions) is the sum of matrices expressed in authorised ballots, excluding ballots authorised by the same private credential, which are all discarded, except for the last such ballot.

Soundness precludes invalid tallies.

Voting systems must also preclude executions in which an adversary forces an error state—from which no tally can be derived—to avoid suppression of "unfavourable" outcomes: Consider an adversary orchestrating a system execution from which no tally can be derived, we infer tallies are always computable in the absence of such an adversary.

*Completeness.* A voting system is complete when tallies are always derivable: For system computed key shares, adversary computed public voter credentials and ballots, a tally is system computable.

Completeness guarantees systems are capable of deriving tallies. Taken together, soundness and completeness ensure tallies match votes expressed in ballots (soundness), and such tallies are always computable (completeness).

Voting systems must also guarantee that only those possessing private voter credentials can authorise ballots.

*Unforgeability.* A voting system ensures unforgeability when ballots are authenticated by private voter credentials: For adversary computed key shares and system computed voter credentials, authorised ballots are only computable with private voter credentials.

Sound and complete voting systems deliver tallies matching votes expressed in authorised ballots. Unforgeability limits authorisation to private credential holders.

### Twilight Zone

Secrecy follows an established narrative: An adversary is challenged to unravel some cryptographic object computed in accordance with some primitive(s) to discover a secret. Verifiability twists the narrative, objects are computed by an adversary. We must discern truth. For example, extract a tally from a vote amongst utterly corrupt parties. Surprising consequences emerge. For instance, using a key pair (belonging to the codomain of an asymmetric encryption scheme's key generation algorithm) to produce and decrypt a ciphertext may result in recovering plaintext distinct from what was encrypted—(probabilistic) correctness makes no guarantees when key pairs aren't computed by key generation algorithms, there may exist non-functional keys arising with negligible probability under normal operation, an adversary may know of their existence. (Fortunately, perfect correctness works as expected.)

Formulations of verifiability are fiendish (cf., [1, 2]); we discern voters' will from authorised ballots in a voting system's ballot construction codomain, necessitating identification of elements in that codomain (using zero-knowledge proofs of construction, for example) and inversion of authorised ballots to extract voters' votes (typically by thought experiment).

Many intuitively verifiable voting systems will fall foul of our formalisation. For instance, consider a (non-secret) ballot as a timestamped digital signature over a vote matrix, and a tally as the sum of matrices authenticated by public credentials (disregarding matrices with earlier timestamps authenticated by the same credential). Soundness seems satisfied, tallies match votes expressed in ballots, *non*? Maybe; it depends on properties of the signature scheme: If there exists private and public keys from which some cryptographic object (other than a signature) can be constructed, such that the object is authenticated by the public key, the scheme will tally a ballot outside the system's ballot construction codomain. Unforgeability (of signatures) doesn't preclude such constructions because a private key holder is permitted to construct objects authenticated by their public key—verifiability is an unsettling beast.[1]

### Cryptographically fair

Whether a voting system is sound depends on whether tallies match votes expressed in ballots authorised by public voter credentials,[2] such determinations can be established by considering voting system executions in which an adversary attempts to subvert the system:

**Definition** (Soundness)**.** *A voting system is* sound *if on input of adversary computed public key shares, public voter credentials, ballots, partial decryptions, and metadata (inc. vote matrix dimensions), any system computed tally—derived by tallying ballots (with respect to public key shares, public voter credentials,*

[1] I've given up seeking intuitive examples of sound & complete voting systems ensuring unforgeability. Whilst signed ballots seemingly lend themselves to the task, we've just uncovered the necessity for nuanced assumptions. I favour studying deployed systems over contrived examples (such as [4, §3.3]).

[2] Probabilistic, computational guarantees suffice, absolute/perfect, information-theoretic guarantees are (typically) unnecessary; we needn't be concerned with attacks that cannot be executed (i.e., attacks in non-polynomial time or with negligible probability of success), herein all algorithms (including those used by adversaries) are probabilistic polynomial time and all security properties need only hold with overwhelming probability.

*and vote matrix dimensions) to obtain an encrypted tally which decrypts (using partial decryptions) to recover a tally (aborting if partial decryptions are insubstantial)—is a unique sum of matrices, each matrix containing one at most once per row, being otherwise zero-filled, and after discarding duplicate ballots, ballots non-conformant with vote matrix dimensions (non-ballots), ballots outside the system's ballot construction codomain (ill-formed ballots), ballots inverting to inputs including a private voter credential that doesn't pair with one of the public voter credentials to appear in the system's credential generation codomain (unauthorised ballots & ill-formed credentials), and, for each distinct public voter credential, discarding ballots inverting to inputs including a timestamp and a private voter credential that pairs with that public voter credential to appear in the system's credential generation codomain (early votes), except for any ballot with the highest timestamp (which is retained), the remaining ballots invert onto those matrices (and other inputs).*

That's heavy! Yet, far more accessible than my earlier iterations. Let's unpack: We challenge an adversary to orchestrate a system execution (comprising public key shares, public voter credentials, ballots, partial decryptions, and metadata) producing a tally that doesn't express voters' will. Duplicate ballots are ignored, only a voter's last has influence. Ill-formed ballots (outside the system's ballot construction codomain) and votes cast with ill-formed credentials (outside the system's credential generation codomain) or unauthorised credentials are ignored too, cast-as-intended assures well-formedness. That leaves ballots authorised by public voter credentials; those authenticated by the same public voter credential are also ignored, except for the last. Soundness assures irrelevant ballots are disregarded and tallies are the sum of matrices expressed in remaining ballots—the last ballots cast by voters—those expressing voters' will.

You've earned a rest, completeness is super lightweight; we challenge an adversary to force an error state.

**Definition** (Completeness)**.** *A voting system is* complete *if given some system computed public key shares, no adversary can compute public voter credentials, ballots, and metadata (inc. vote matrix dimensions) causing a system error during tallying.*

Completeness assures voting systems cannot be derailed, tallies must always be derivable.

We challenge an adversary to forge an authorised ballot:

**Definition** (Unforgeability)**.** *A voting system ensures* unforgeability *if an adversary computes public key shares and makes requests for (1) a public voter credential (the corresponding private voter credential being kept secret); (2) a ballot authorised by a private voter credential, the adversary providing a vote (matrix), a timestamp, and the corresponding public voter credential; and (3) a private voter credential, the adversary providing the corresponding public voter credential; yet the adversary cannot derive a ballot (beyond copies of requested ballots) inverting to inputs including a private voter credential (beyond those requested) that pairs with one of the public voter credentials.*

Unforgeability assures only private voter credential holders can authenticate ballots.

Each of these definitions can be stated algorithmically, such as those I presented in a technical report [5, Appendix D]. Whilst there's merit to such presentation, they're less accessible, whereas the above can be more broadly understood.

## Outlook

"Stop the steal," rallied President Donald Trump, moments before the Capitol was stormed. "This is not just an attack on me and my campaign. This is an attack against our country. This is about the integrity of our democracy and the security of our nation." The House is in agreement. "Serious questions have arisen about the accuracy and reliability of new electronic voting machines," raged Hillary Clinton, "including concerns that they can be susceptible to fraud and computer hacking." Science has long taken this position. Yet 'merica, the so called leader of the free world, is awash with voting technology that was never fit for purpose.

"The whole election is...rigged," cried Donald Trump, four years earlier. "The whole thing is one big fix. One big ugly lie." Promptly rebranding the election "beautiful," as he clamoured for victory. Rival Hillary blaming her defeat on Vladimir Putin and his Russian hackers. The truth: Our election systems are vulnerable to mass fraud. We must demand free and fair elections to preclude demonstrations of evil.

After nearly two decades of contemplating inner workings of voting systems—pondering what exactly they're supposed to do; postulating formalisations of secrecy and verifiability; dismissing much of voting system theory as unfit, uncovering vulnerabilities as a consequence—I'm finally reaching (after many iterations and failures) an understanding of what free & fair voting actually means.

Doubt over election results erode democracy, descend civilised society into chaos: On 6 January 2021, a mob stormed the Capitol, attempting to overthrow those that had 'stolen' the election from Donald Trump. Their mission: To restore democracy. "If you don't fight like hell, you're not going to have a country anymore;" I'm writing to equip democracy builders with tools for certifiably free and fair elections—rotten voting technology must be eradicated before a U.S. election is thrown by hackers, a plausible extinction catalyst. Trust, but verify.

## References


[1] B. Smyth, Surveying global verifiability, Information Processing Letters 163 (2020).

[2] B. Smyth, M. R. Clarkson, Surveying definitions of election verifiability, Information Processing Letters 177 (2022).

[3] B. Smyth, S. Frink, M. R. Clarkson, Election Verifiability: Cryptographic Definitions and an Analysis of Helios and JCJ, Cryptology ePrint Archive, Report 2015/233 (2021).

[4] B. Smyth, A foundation for secret, verifiable elections, Cryptology ePrint Archive, Report 2018/225 (2018).

[5] B. Smyth, Athena: A verifiable, coercion-resistant voting system with linear complexity, Cryptology ePrint Archive, Report 2019/761 (2019).