TrueVote: A Transparent Electronic Voting System validated by the Bitcoin Blockchain

Brett Morrison, June 1, 2021

brett@truevote.org, https://truevote.org

Abstract

True democracy starts with voting. True democracy requires that every eligible voter can vote and that every vote is counted properly. Analog and semi-automated systems have proven to be unreliable and inaccurate. This reality has contributed to disenfranchisement, fraud, voter suppression, and lack of confidence in the election process.

We shop, we bank, we share, and we even find love on the internet through our devices. Our analog voting systems are obsolete – It's time we vote on our phones!

To ensure a true democracy, what's needed is an open, fully digital, tamper-proof, verifiable system. TrueVote is designed to fill that need.[1]

TL;DR

When voting using today's conventional analog systems, there is no way of knowing whether election authorities counted your vote properly. You are trusting people in power to handle your ballot correctly.

TrueVote solves two fundamental problems:

1. Voting on a mobile device.
2. Verifying your ballot's accuracy and its inclusion in an aggregate count.

This white paper outlines a method of voting in elections built on encryption, a public ledger (blockchain), and checksums of all stored data. Smart contracts unlock the election results at designated times during and/or after the election. The voter can confirm their ballot is included in the aggregate count via a public "Ballot Explorer" and confirm that the public version of their ballot matches their ballot submission. When the results are unlocked, the election authorities, the general public, and the media can transparently see the election data and can validate their accuracy via hashed checksums. Ballot data is de-coupled from voter identifying data.

Introduction

Immutable - not capable of or susceptible to change[2]

TrueVote is an online system that enables voters to cast their ballot using their smartphone, validate their vote was counted, and view the aggregate results of an election.

TrueVote stores all non-user identifying voting data in a publicly accessible database, viewable using a "Ballot Explorer" (similar to a "Blockchain Explorer"[3]).

Designated Election Commissioners (DEC) are assigned by election officials as authorities of an election within TrueVote. Election officials typically assign only one or two DECs to each election district. TrueVote publicly posts their names. DECs have permission to create an election. For example, for statewide elections, this would be a state's Secretary of State office.

Administrators create elections on truevote.org (similar to creating online surveys) using TrueVote's Election Configuration Administration Portal (ECAP). Once created and scheduled, valid participants of the election (voters) receive a notification reminding them of the upcoming election.

Ballot entry submissions are made on the device as one would expect: by choosing between lists of candidates and ballot measures. Before the ballot is submitted, the choices are hashed on the device with a checksum using the same hashing method as the server uses. Once the server receives the data and the checksum, the server re-hashes and confirms that both checksums match. TrueVote then stores the data, using the checksum as a branch of a Merkle tree[3].

Peter Todd, an early Bitcoin core developer, built OpenTimestamps[4] as a hashing protocol for creating Merkle trees and connecting them into a single Merkle root[5]. DCRTime[6] is an enhanced derivative of Todd's research. TrueVote uses both hashing protocols to maximize throughput for election use cases.

Periodically, the Merkle root is stamped on the Bitcoin blockchain in the OP_RETURN field. The timestamp pattern ensures that the data is immutable as of a *known point in time*. This approach of storing the "chain of proofs" on the established, resilient, tamper-proof, world-distributed Bitcoin blockchain guarantees that all data forming the Merkle root has not been altered by anyone in any way.

During the election, TrueVote encrypts and stores ballot data at rest. At the "time of settlement" (reveal time), ballot data is decrypted and pushed from TrueVote's private data store to its public data store. For example, if the polls close at 9pm and the election rules permit the posting of results beginning at 3pm, TrueVote unlocks the results at 3pm, updating the results in frequent batches (near real-time) as votes counted between 3pm - 9pm are posted.

This method employs the hybrid design of a moderately complex database with normalized data models, forming a traditional internet-scale application augmented with layered on-chain checksums ensuring each voter's data is provably immutable.

"I think timestamps is the most interesting and the most important non-monetary use of Bitcoin"
Aaron van Wirdrum - Technical Editor, Bitcoin Magazine [7]

Transparency

Open Source

TrueVote's Core Platform code is open source under the MIT License[8] and maintained by the TrueVote community with peer review, valuable feedback, pull-requests, and test suites.

Note: TrueVote, Inc. may choose to add some value-added services that are closed source on top of the TrueVote Core Platform.

Designated Election Commissioners (DEC)

All DECs are publicly identified per appropriate jurisdiction, with information publicly posted as part of the Election Masthead (e.g., Name, Email, Election Configurator Issuance Dates, Scope of Election Jurisdiction).

The DECs give the voter an accountable human responsible for election support issues.

Repeatable Builds

Repeatable builds ensure the code running on a device is the exact same as the publicly available open source code.

All apps that land on Apple and Google stores will have "repeatable build" checksums as part of the startup lifecycle. Telegram[9], the messaging app, has executed this successfully and TrueVote apps will be modeled after their approach.

Election Configuration Admin Portal (ECAP)

The ECAP is where the ballots and the election are created. It's the "back-end" interface that DECs will log in to in order to set up an election. Here, the DEC configures the title, text, images, date ranges, eligible voter lists, and ballot items for the election.

Voter Authentication

Today's systems typically use a combination of email, password, multi-factor authentication (MFA), and 3rd party authentication frameworks such as oAuth, Facebook login, and Google login, to name a few. TrueVote will also employ these ubiquitous login methods, augmented with a zero-knowledge proof[10] security model. Additionally, TrueVote requires MFA upon registration. All data on the device is encrypted in secure storage while at rest and during transmission.

The private key needed to access elections will always remain on the voter's device, making it impossible for anyone to access the raw TrueVote data and impersonate a voter account. Should a device be lost or damaged, the account holder will be required to perform a reset procedure typical of any secure online service.

Voter Identity

Voter registration and identity verification is a separate process and is outside the scope of this document. Registration processes will continue to be handled by registration authorities, whose records link to TrueVote via a foreign key[11] connecting the voter's TrueVote account to the registrar's record.

This external link provides defense against Sybil[12] attacks. Voters cannot just create an account on TrueVote and begin voting without linking back to the registration provider. This *binding* is only accessible to the voter when signed in to their account. The link between their TrueVote account and registrar (e.g., John Smith, registered as Independent, in Los Angeles County, DOB 04/04/2001) is displayed on the app.

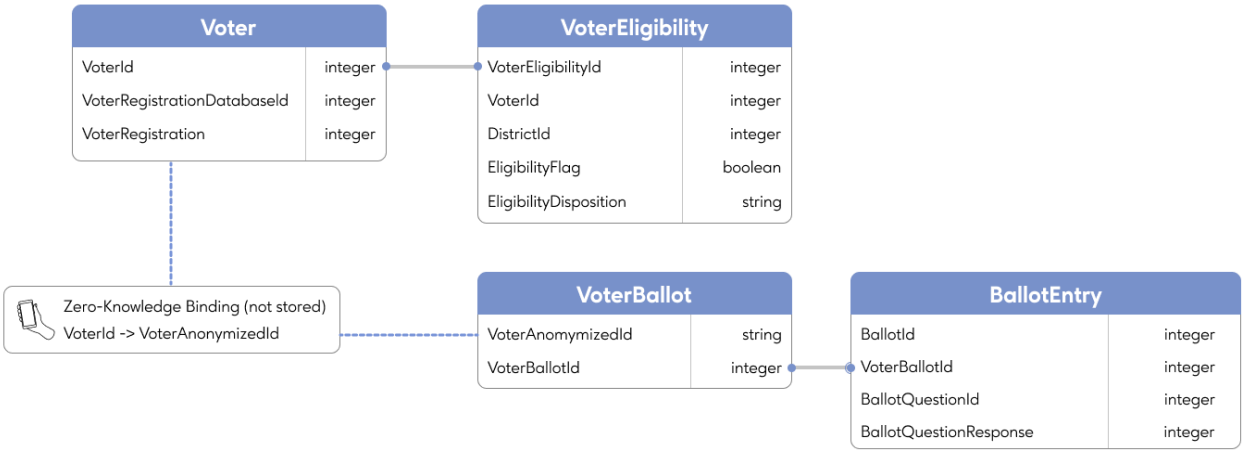
Voter Identity v2

There are many innovations developing in digital identification services. The Decentralized Identity Foundation[13], Microsoft ION[14], Persona[15], Cognito Flo[16], and Civic[17] are projects that TrueVote may incorporate as we move from the current fragmented registration systems to a more systematic, automated process. Microsoft ION, for example, has an *attestation* mechanism to build trust with an identity as it becomes verified by more and more services (DMV, Passports, Schools, Employers, etc.) and builds on the reputation of a person's Decentralized Identifiers (DID).

De-Coupling of VoterId and BallotId

VoterId and BallotId are only bound within the voter's account and are stored encrypted. The signed-in voter account context is the only way these two pieces of data are mapped together. The connection is not viewable in any other context in the TrueVote platform. The encryption key for the binding is part of the account's client-side storage, which TrueVote never stores. The encryption occurs on the client, and then the encrypted binding data is stored encrypted on truevote.org, enabling voter sign-in on multiple devices and account recovery. TrueVote cannot decrypt this data and is comparable to LastPass Password Manager[18] in the way it stores data.

De-coupled Voter/Ballot Data Model



Account Recovery

TrueVote account recovery is very similar to any secure service (e.g., financial institutions) in using a "forgot password" workflow and MFA recovery codes. However, the encrypted bindings mentioned earlier will not be recoverable unless the voter has at least one device where they are signed in. In this case, the following happens:

1. New sign-in generates encryption key and peers with existing device for key exchange.
2. Existing device encrypts existing key with a new key and sends it to new device.
3. Both devices refresh their keys with new ones and re-encrypt the binding data.

Both LastPass[19] and the Edge SDK[20] have employed similar mechanics for ensuring that encrypted user data is zero-knowledge when stored; TrueVote will build off of these industry examples.

Election Result Tabulation

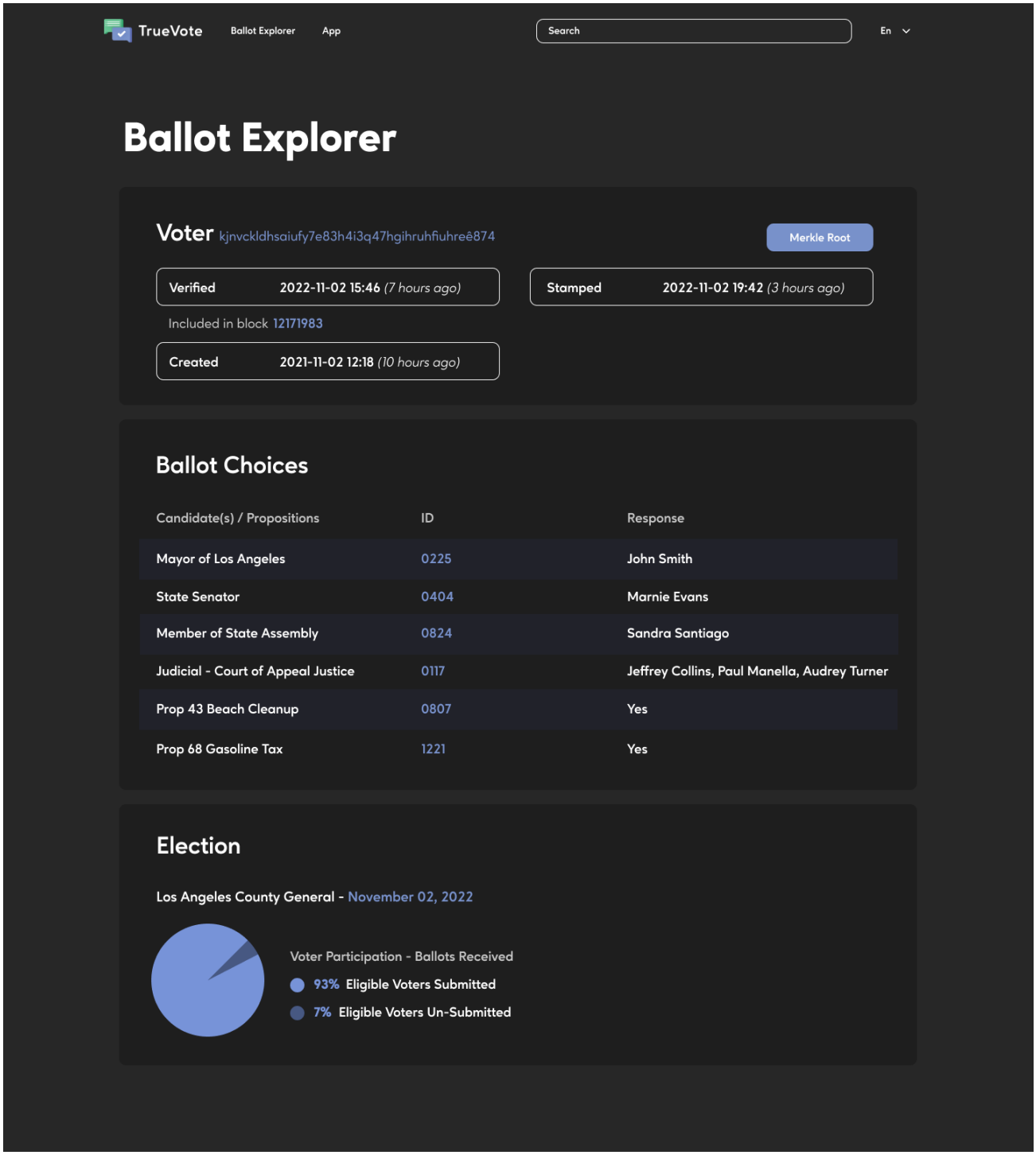
TrueVote reveals posted votes after executing the timelocked smart contract, which may vary for each district. For example, if a smart contract for a North Carolina district unlocks at 5pm EST, TrueVote will automatically tabulate the unlocked votes; all derived data is then viewable on the Ballot Explorer.

The election authorities, media, and public can now query the Ballot Explorer through the Election Report Portal (ERP) to view the election results. If permitted by the election configuration, ballots can be submitted after the timelock contract, which are then viewed as additions in real-time or held until the next timelocked smart contract executes.

Publicly Viewing Voter Data - Decentralized

TrueVote will have both a public REST and GraphQL API which the Ballot Explorer will access. TrueVote will build a bulk download tool, allowing for the auditing, analysis, and decentralized storage of all non-voter identifying data, similar to hosting a Bitcoin full node where the entire blockchain lives on each full node device. TrueVote will encourage the community to build tools and analysis reports for the data trove, which will help analyze election trends historically unavailable to the general public, opening up a world of discovery never before seen with legacy voting systems.

Ballot Explorer



Election Fraud

The transparency of the data eliminates election fraud. Because all the data is transparent and the source code is open source, it's not possible for ballot counts to be altered. It is possible for an entity to generate fake accounts and fake registrations. However, as digital IDs and decentralized ID systems mature, creating fictitious records will become much more difficult.

Voter Fraud

It is still possible for an individual to commit voter fraud by falsifying information with a registration authority, voting for a deceased person and other nefarious attempts to violate election rules. TrueVote will build detection signals for these types of actions, flagging suspicious accounts and ballots. These alarm methods may be more optimally implemented as closed source to create some obfuscation between the attacker and TrueVote.

Hostile Actor Attack Surfaces

Distributed Denial of Service (DDoS)[21]

DDoS attacks are the hardest to defend against, and require assistance from network and edge service providers. A hostile actor would likely attempt to DDoS any publicly accessible internet resource.

TrueVote will be a "juicy" target for DDoS attacks. TrueVote will work with vendors such as Cloudflare to help shield such attacks. Having a larger voting window helps mitigate this target. When everyone votes at a certain time, an attacker can plan. Early voting helps to spread the attack surface across the span of an election and alleviate the consequences of a DDoS on a single day during waking hours.

App Vulnerabilities & Rooted Devices

All sites and apps are susceptible to attackers looking to exploit the product and potentially gain control of an account. TrueVote will use best practices, take recommendations, and engage with industry auditors to scrutinize the development and deployment lifecycles for possible vulnerabilities.

A malicious actor with physical access to a device may "root" the device and potentially extract private data from the device. TrueVote will employ best practices published by Apple and Google to defend against this type of attack.

Third Party Vendor Frameworks and Packages

All elements of a software stack are potential exploitation targets, ranging from the hardware, the operating system, the application framework, plugins, libraries, and other application-dependent services. TrueVote will consider each of these elements as part of the entire ecosystem and will constantly monitor for any discovered vulnerabilities and remediation recommendations as encountered.

Man In the Middle (MITM) & DNS Spoofing[22]

An attacker can create a version of the TrueVote server infrastructure and poison DNS in a network, forcing the application to connect to a spoofed set of servers impersonating the TrueVote protocol. TrueVote will use best practices and certificate transparency[23] for issued certificates from Let's Encrypt[24] and other industry-ubiquitous certificate providers to best detect this type of attack.

Voter Intimidation / Coercion

Like traditional voting, online voting systems can be vulnerable to "off-line" attacks and are subject to voter coercion and voter intimidation, where an adversary with nefarious motives can influence an individual's vote. Someone can potentially "force" a voter to vote for a particular choice with both in-person voting and mail-in voting. This forcing can also translate to online systems where intimidation can happen similarly to mail-in voting, with no physical voting official nearby at the time of ballot entry. TrueVote will develop a "decoy ballot" subsystem to help mitigate this type of voter fraud.

Unknowns

TrueVote is open source, which means that attackers may have more opportunities to look for possible weaknesses. This is also a strength, as the community at large can work on finding and fixing any potential problems before a hostile actor does. Security audits will be encouraged and embraced.

Software Principles and Guidelines

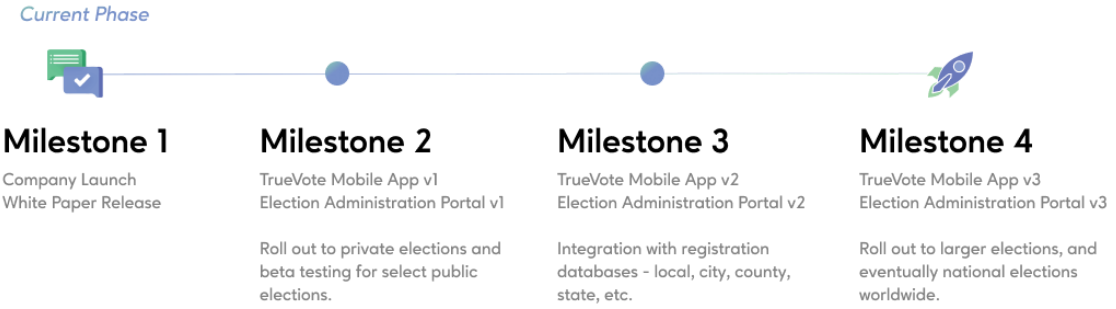
There will be numerous stringent procedures in place to ensure a healthy and maintainable codebase for years to come. Here are a few:

- All open source code published on GitHub, MIT license
- Governance / standards of repositories modeled after the Bitcoin project (Merges, Test Coverage Thresholds, etc.)
- Authors must sign all commits
- All UI presented strings localized with dictionary pattern (no embedded English, TrueVote is an international project)
- Strict linting enabled across all languages
- Repository specific contributor guidelines specified in each repository's root
- Performance and scalability will be considered and measurable as core to the engineering ethos
- All architectural decisions logged using the Architectural Decision Records (ADR) pattern[25]

Timeline

The initial version will be built from the principles outlined in this document. The product will then first be rolled out to private elections. As the product becomes proven, edge cases will be tuned and larger and larger elections added until eventually all elections worldwide are using TrueVote.

Timeline



Conclusion

The invention of blockchain has changed our world and allowed this opportunity to happen. Bitcoin and blockchain "cannot be uninvented."[26] By de-coupling voter identifying data from ballot submission and storing the Merkle root of all data on the Bitcoin blockchain, TrueVote solves the challenges of electronic voting. As an open source project from inception, TrueVote builds on the spirit of worldwide community, demystifying the software behind the most cherished of institutional processes - elections.

TrueVote is now born to ensure immutable accuracy, honesty, freedom, and truth in voting. This is true democracy, reimagined for the digital age.

References

[1] Satoshi Nakamoto, Inspiration, Bitcoin White Paper, https://bitcoin.org/bitcoin.pdf
[2] Immutable, Websters, https://www.merriam-webster.com/dictionary/immutable
[3] Blockchain Explorer, https://en.bitcoin.it/wiki/Block_chain_browser
[4] Merkle tree, https://en.wikipedia.org/wiki/Merkle_tree
[5] Peter Todd, OpenTimestamps, https://opentimestamps.org
[6] Marco Peereboom, https://blog.decred.org/2017/06/14/dcrtime-Blockchain-based-Timestamps
[7] Aaron van Wirdrum - Technical Editor, Bitcoin Magazine, https://youtu.be/hUFNqD3DWWQ?t=1577
[8] MIT License, https://opensource.org/licenses/MIT
[9] Telegram Reproducible Builds, https://core.telegram.org/reproducible-builds
[10] Zero Knowledge Proof Security Model, https://en.wikipedia.org/wiki/Zero-knowledge_proof
[11] Foreign Key, https://en.wikipedia.org/wiki/Foreign_key
[12] Sybil Attack, https://www.microsoft.com/en-us/research/wp-content/uploads/2002/01/IPTPS2002.pdf
[13] Decentralized Identify Foundation, https://identity.foundation
[14] Microsoft ION, https://techcommunity.microsoft.com/t5/identity-standards-blog/ion-booting-up-the-network/ba-p/1441552
[15] Persona Identity, https://withpersona.com
[16] Cognito Flo, https://cognitohq.com/products/identity-verification-service
[17] Civic, https://www.civic.com
[18] LastPass Security Model, https://logmeincdn.azureedge.net/legal/LastPass-SPOC.pdf
[19] LastPass Technical White Paper, https://logmeincdn.blob.core.windows.net/lporcamedia/document-library/lastpass/pdf/en/Sales-Tool-LastPass-Technical-Whitepaper.pdf
[20] Edge SDK, https://edge.app/wp-content/uploads/2018/09/2017-10-Edge-White-Paper.pdf
[21] DDoS, https://en.wikipedia.org/wiki/Denial-of-service_attack
[22] DNS Spoofing, https://en.wikipedia.org/wiki/DNS_spoofing
[23] Certificate Transparency, https://certificate.transparency.dev
[24] Let's Encrypt Certificates, https://letsencrypt.org
[25] Architectural Decision Records, https://adr.github.io
[26] Andreas Antonopoulos, "Bitcoin cannot be uninvented", https://twitter.com/aantonop/status/410470820167688192

Acknowledgements

[*] Pedram Hasid, TrueVote Co-Founder, for his friendship, support, contributions, and teamwork
[*] John Matthews, TrueVote strategic partner, for his mastery of the elections process
[*] Hannah Morrison, Daughter, for her linguistic acumen and grammar policing
[*] Sima Morrison, Wife, my muse, my guiding light, for her immutable love and support