PeaceFounder: e-voting by pseudonym braiding

Janis Erdmanis[0000−0002−8963−5963]

janiserdmanis@protonmail.ch

Abstract. The increasing pressure for adopting remote electronic voting systems has elevated concerns about privacy, transparency, and security. While current solutions aim to address these issues in large-scale elections, they often impose complex responsibilities on election authorities, such as threshold decryption ceremony coordination or managing a trusted setup phase due to trapdoors. In response, we introduce PeaceFounder, a system uniquely designed to simplify these challenges. Our approach guarantees voter anonymity before votes are cast through a transactional, single-mix process. It also provides verifiable proofs using a history tree based bulletin board, which unlike traditional blockchain or replication based buletin boards, ensures record immutability via consistency proofs across voter devices. This simplified architecture allows for single-person deployment and administration, making it particularly suitable for smaller communities. PeaceFounder also offers coercion and bribery resistance, detects device-level spyware and malware through a secondary channel, and enables the ongoing enrollment of new members without compromising anonymity.

Keywords: e-voting · privacy · transparency · verifiability · public auditability · reencryption mixnet · braiding · system design

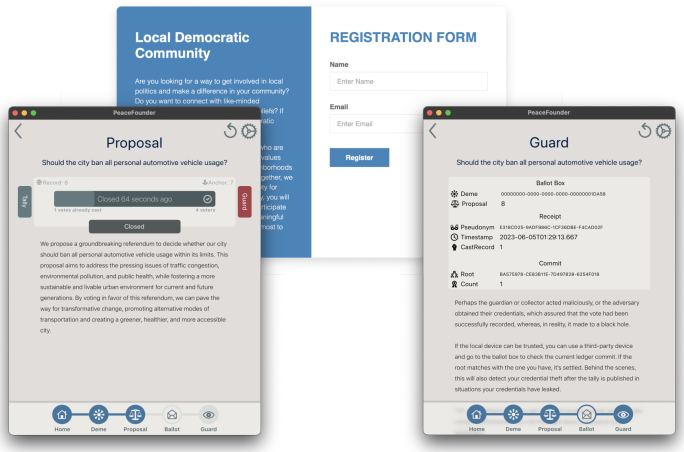
With the advancement of technology, remote electronic voting systems are becoming more prevalent in many contexts. E2E (end-to-end) verifiable voting systems have become the gold standard, providing universal and individual verifiability. Leading designs, like those in Estonia [14], Helios [1] and Belenios [3], utilise a re-encryption mix net, whereas others use homomorphic counting procedures coupled with a threshold decryption [9][10]. However, in these systems' privacy, transparency, and security remain in tension.[1]

A major hurdle in these systems is the multiparty protocol ceremony required for initiating a threshold decryption key and performing decryption at the end of the vote. The voter's anonymity demands independence of the involved parties,

[1] In this context, security is conceptualised as a behavior-enforcing system. On the other hand, transparency serves as a lens through which the actions and intents of all participating entities can be observed and verified, aligning closely with democratic principles of transparency as explored in [13]. The requirements of accountability and verifiability, as elaborated in [11], can be seen as the interplay between security and transparency—serving both as connecting and distinguishing factors between the two.

which introduces a risk of sabotage where election results could be left undecrypted and unannounced. Moreover, due to the need to encode ballot selection in a group element, only a limited number of ballot types can be supported and face challenges, for instance, for cardinal and budget planning ballots. This is even more restrictive when a homomorphic counting procedure is employed.

An alternative approach is to anonymise voter's credentials instead of the votes. The idea has been explored with blind signature schemes, but auditing the authority's issuance of signatures and detecting key leaks remains unresolved [12]. A subsequent method, proposed by Haenni & Spycher [8], leverages ElGamal re-encryption to verifiably exponentiate voters' public keys in tandem with a generator using zero-knowledge proofs. Together with a history tree bulletin board implementation [4], it forms the foundation for the design of the PeaceFounder voting system.



Are you looking for a way to get involved in local politics and make a difference in your community? Do you want to connect with like-minded

Should the city ban all personal automotive vehicle usage?

Record: 8 Anchor: 7

Closed 64 seconds ago

1 votes already cast 4 voters

We propose a groundbreaking referendum to decide whether our city should ban all personal automotive vehicle usage within its limits. This proposal aims to address the pressing issues of traffic congestion, environmental pollution, and public health, while fostering a more sustainable and livable urban environment for current and future generations. By voting in favor of this referendum, we can pave the way for transformative change, promoting alternative modes of transportation and creating a greener, healthier, and more accessible city.

Should the city ban all personal automotive vehicle usage?

Deme 00000000-0000-0000-0000-00000001DA58
Proposal 8

Pseudonym E318CD25-9ADF986C-1CF36DBE-F4CAD02F
Timestamp 2023-06-05T01:29:13.667
CastRecord 1

Root BA575978-CE83B11E-7D497828-6254F018
Count 1

Perhaps the guardian or collector acted maliciously, or the adversary obtained their credentials, which assured that the vote had been successfully recorded, whereas, in reality, it made to a black hole.

If the local device can be trusted, you can use a third-party device and go to the ballot box to check the current ledger commit. If the root matches with the one you have, it's settled. Behind the scenes, this will also detect your credential theft after the tally is published in situations your credentials have leaked.

Fig. 1. User experience overview of the Peacefounder system: the layout includes a voter's client interface with a custom registrar in the background. The left displays the proposal view, featuring key metadata such as the proposal's index in the braidchain and the relative generator, as determined by an anchor index, along with the number of votes cast. The right reveals the guard view providing the voter with essential details like a receipt to monitor local malware influence on their vote, as well as the current state of the ballot box ledger, indicating up to which point ledger immutability is assured with consistency proofs.

The PeaceFounder voting system builds upon the foundational work of Haenni & Spycher [8], serving as a practical implementation of their proposal. Nevertheless, PeaceFounder introduces several key features:

– A scalable bulletin board design with thin-member clients ensuring the immutability of all published records without replication;
– A registration protocol for new members that catches them up with the current relative generator;
– Mechanisms to handle uncooperative bulletin boards through auditors/proxies while preventing potential exploitation by coercers and bribers with time-restricted receipt freeness and revoting;
– A system allowing a member's device to detect private key leaks coming from spyware or bad cryptography via sequence numbers and bitmasks;
– A malware detection mechanism post-voting, where the device displayed receipt, is compared to a bulletin board while not being deceived into verifying another voter's vote.

Furthermore, PeaceFounder demonstrates that a single maintainer can feasibly deploy the system. That is possible due to the lack of any multi-party ceremony and member device accountability of the bulletin board. It also offers seamless integration opportunities with existing infrastructure and political environment for supporting different ways proposals are put to the ballot box, and member authenticity is verified and later audited. Additionally, the PeaceFounder showcases user experience for the voter, minimising their exposure to complex byte strings while maintaining cryptographic soundness along with other usability improvements.

Braiding

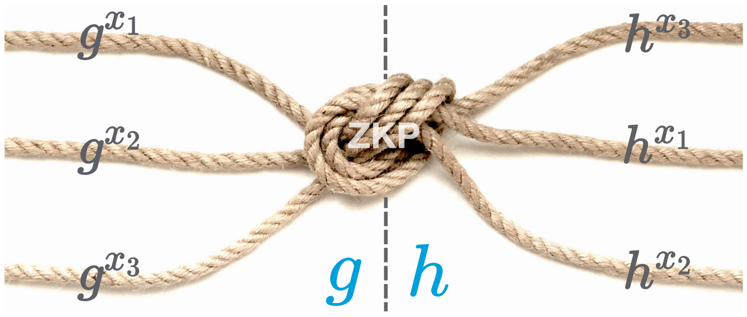


Fig. 2. A knot like structure formed in a braiding process. On the left inputs are shown a set of psueodnyms and a realtive genertor g. On the right the outputs are shuffled pseudonyms and relative generator exponentiated with a secret factor s. The knot itslef represents a zero knowledge proof assuring that output pseudonyms are computed correctly without allowing to link input to output pseudonyms. Since $h = g^s$ then $(g^{x_i})^s = (g^s)^{x_i} = h^{x_i}$ which can be computed by private key x_i owners.

The core primitive for PeaceFounder voting system revolves around the ability to generate digital signatures using a single private key for distinct generators, all while maintaining the security of the key. The signatures in such cases are supplemented with a corresponding public key for a relative generator[2] at which the signature has been issued. A relationship between these public keys can be established by showing an exponent connecting the relative generators or forming zero-knowledge proof demonstrating the equality of discrete logarithms [2].

The concept of unlinkability can be harnessed to create an interconnected structure using multiple private keys resembling a knot. In this structure, input **pseudonyms**—public keys derived by exponentiating input relative generator with private keys—are bound to output pseudonyms. To achieve this, a dealer exponentiates relative generator and pseudonyms with the same secret exponent and then shuffles the resulting output pseudonym list. We shall refer to this procedure as **braiding** to distinguish that from mixing objectives where input retains the original form after going through a mix cascade.

To ensure integrity in resulting braids, in particular, that braider had not replaced output pseudonyms with its own, zero-knowledge proofs can be used. This can be done by reformulating exponentiation as ElGamal re-encryption shuffle and consequent decryption as recently proposed in a novel e-voting system design [8]. The zero-knowledge proof of shuffle has been successfully made widely available for ElGamal re-encryption mixnets with Verificatum, which offers proof with relatively standard cryptographic assumptions on the difficulty of computing discrete logarithms and a decisional Diffie Hellman assumption [17][16][7]. Combined with zero knowledge proof of correct decryption, a braid proof can be formed, proving to everyone that computations have been performed honestly without revealing the secret exponentiation factor braider had used and can be safely forgotten afterwards. The resulting braid primitive is available in the *ShuffleProofs.jl* package, which also reimplements Verificatum-compatible proof of shuffle in Julia [6].

The braid primitive enables anonymisation to be transactional with one braider at a time, thus eliminating the need for complex coordination of parties as it is typical for many re-encryption mixnet or homomorphic-based e-voting systems [12]. In addition, it's also possible to publish this evidence on a bulletin board for everyone to verify without compromising participation privacy.

Bulletin Board

The PeaceFounder's bulletin board is a microservice that accepts transactional records and votes. The primary transactional records include member certificates issued in prospective **member** interaction with the **registrar**, braid records which recompute pseudonym member list and corresponding relative generator issued by a **braider** and proposal records announcing a vote issued by a **proposer**. The system is configured with a **guardian** issued deme[3] record, which

[2] The adjective *relative* is used here to signify that the generator is not an absolute constant in cryptographic operations.

[3] The term *deme* here denotes a PeaceFounder instance for a specific organization. It is inspired from its historical significance in Ancient Greece, where a *deme* rep-

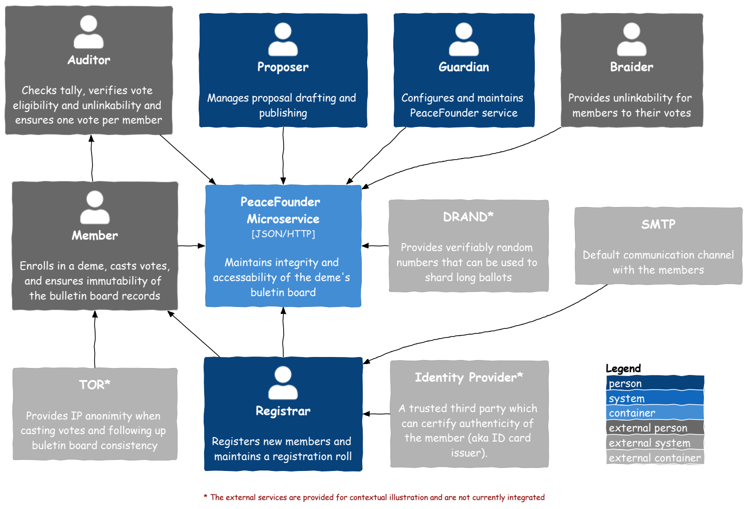

Auditor
Checks tally, verifies vote eligibility and unlinkability and ensures one vote per member
Proposer
Manages proposal drafting and publishing
Guardian
Configures and maintains PeaceFounder service
Braider
Provides unlinkability for members to their votes
Member
Enrolls in a deme, casts votes, and ensures immutability of the bulletin board records
PeaceFounder Microservice
[JSON/HTTP]
Maintains integrity and accessability of the deme's buletin board
DRAND*
Provides verifiably random numbers that can be used to shard long ballots
SMTP
Default communication channel with the members
TOR*
Provides IP anonimity when casting votes and following up buletin board consistency
Registrar
Registers new members and maintains a registration roll
Identity Provider*
A trusted third party which can certify authenticity of the member (aka ID card issuer).
Legend
person
system
container
external person
external system
external container
* The external services are provided for contextual illustration and are not currently integrated

Fig. 3. High-level diagram of the PeaceFounder system. With a blue colour, the internal components of the peacefounder system are indicated, which are necessary to bootstrap the system. The arrows represent dependencies for producing trust in the election process. Services like TOR [5] and DRAND [15] are shown for context and currently are not integrated.

sets cryptographic parameters, and a rooster, which includes the proposer, registrar, and bulletin board authority identities.

The peacefounder bulletin board, proposer, and registrar microservices are internal and can be managed or delegated by a **guardian**. This separation accommodates customisation for varied political consensus, criteria for proposal submissions to the ballot box, choice of identity provider, and methods for disclosing registrar information to auditors to verify the authenticity of members. To make testing and deployment easier for new organisations, a bundle that includes a registrar, proposer and bulletin board will be available and deployable on a preferred server of choice and will offer web access for a guardian with sane defaults and configuration options.

Member's client devices actively monitor the bulletin board, ensuring the immutability of records by tracking bulletin board commits. This method of oversight is scalable, as members only request history tree[4] consistency proofs [4], eliminating the need to replicate the actual bulletin board records. These proofs guarantee the protection of their votes and others, assuring that modifications to records are prevented when fresh entries are appended to the bulletin board. This streamlined approach enables prompt identification of bulletin board dishonesty,

resented a local administrative unit with its own diverse decision-making structure and governance rules.

[4] A *history tree* is a specialized Merkle tree designed to track chronological changes, enabling efficient and secure state verification at any historical point.

whether through removing or altering records or the malicious creation of a counterfeit ledger to exclude undesirable votes from the official tally.

Acountability via auditing

Integral to establishing trust a pivotal role is the presence of an **auditor**. The auditor is a judicial-like entity representing members and is vital for resolving conflicts. Member client devices create local proofs for altered or removed records or the presence of a counterfeit ledger, which are then sent to the auditor. Moreover, if votes aren't delivered, the auditor can act as a proxy, offering evidence that the bulletin board deliberately omitted specific votes. The auditor also ensures the integrity of the bulletin board's records, confirming each vote's eligibility and unlinkability and ensuring one vote per member.

Notably, the auditor can avoid a formal association with the guardian to verify the integrity of a resulting tally, as all relevant data is on the public bulletin board (except for the registration roll). This autonomy allows members the freedom to select their trusted auditors. If there are unresolved disputes with the bulletin board, members can even take on the auditor role and, if necessary, seek to replace the guardian. The system's transparency further allows auditors to cross-check each other's findings, promoting accuracy and preventing the spread of false claims.

The auditor plays a key role in assessing the registrar to confirm the authenticity of its members. The registrar maintains a registration roll, serving as evidence of every member's authenticity. Authenticity verification can be as simple as a trusted third party's digital signature on a document that includes the organisation's UUID and the index where a member's certificate is recorded. If a third-party identity provider isn't available, a photo or video of an individual displaying a page with the organisation's title and index might suffice. Regardless of the method, the recommended approach is to provide an auditor with a verifiably random subset of members where verifiable randomness, for instance, can be generated with DRAND service [15] to avoid data aggregation.

Coercion and bribery resistance

Another pillar that is necessary for ensuring democratic elections is to prevent coercion and vote buying. A significant risk to the PeaceFounder system is for a briber to ask members to forward their votes through a proxy channel they control. To counter this threat, the bulletin board hides the actual votes, showing only their hashes, and gives voters an option to revote, ensuring both receipt freeness and vote fairness. A sequence number along the vote ensures that only the latest cast vote on the device matters.

This method undermines the confidence of vote buyers and coercers, as it prevents them from ensuring that the votes they've acquired will be counted in the final tally. As a consequence, they can only return bribes after votes are

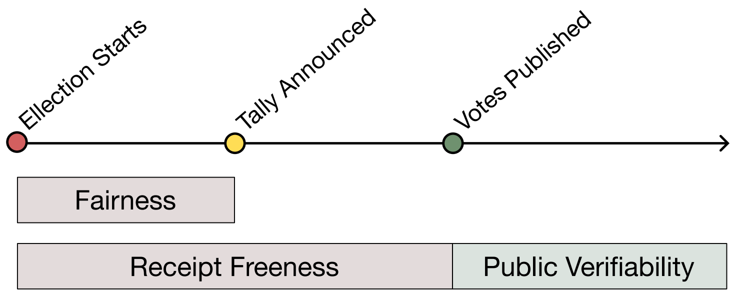


Fig. 4. Illustration of time restricted receipt freeness in the Peacefounder voting system. During the election period, the system maintains both receipt-freeness and fairness. However, after the tally is published, newly submitted votes lose their fairness. The votes themselves can be published on the buletin board latter to extend the time period for receipt-freeness, reducing the effectiveness of coercers and bribers. This comes at the expense of delaying the audit process for verifying that votes have been tallied as cast.

published on the bulletin board[5]. This arrangement erodes the credibility of bribers and coercers, making less likely for voters to engage with them in such transactions due to the lack of a guaranteed positive/negative outcome.

A secondary concern is the potential for a coercer to ask an individual to show how they had voted on their device. To address this, only a receipt is shown. However, this receipt can be linked to the specific vote on the bulletin board. If coercion becomes a significant threat, the receipt can be visible only briefly, such as 30 minutes after casting a vote. During this window, members can manually record their details in a logbook. While this approach may reduce user-friendliness, it still serves as a robust deterrent against malware attempting to cast votes on behalf of the voter.

Malware and spyware detection

The last piece of the puzzle is malware and spyware resistance. An adversary could issue votes without compromising the voter's device in case of key leakage. To counter this, every vote includes a sequence number, which records evidence on the bulletin board when a vote is cast from the voter's device. Moreover, if a vote with the same sequence number is already on the ledger, the first will override any subsequent vote. This mechanism prevents malware from silently replacing inactive voter's choices.

After voting concludes and the results are published, each voter receives a bitmask of the votes included in the final tally, along with consistency proofs. Given that this approach is scalable (e.g., 1kB can handle 8192 votes, and bit

[5] During the vote, only receipt hashes are published on the bulletin board. This serves to both commit the votes while maintaining fairness and receipt-freeness.

compression can further reduce size), voters device's compare this bitmask with the index from their most recent vote receipt. This allows them to detect any malware activity and display an alert to the voter.

To ensure the voting process's integrity, the voter's device must remain trustworthy. With the presence of malware, there's a risk that the device could falsely reassure the voter that their vote is cast as intended. To counter this threat, after the vote is submitted, the voter receives a receipt containing a *timestamp* of the vote's record, the *pseudonym* under which it was cast, and the *index* where the vote resides on the ledger. Once voting concludes and all votes are disclosed on the bulletin board, the voter can cross-reference their receipt with the bulletin board, verifying that the vote at the provided *index* aligns with their choice and matches the *timestamp* when the vote was cast, as well as checking that it was included in the final tally. By maintaining a written record, voters can ensure the accuracy of their vote, safeguarding against malware alterations, unauthorised revoting, or any attempts to redirect multiple voters to a singular vote.

However, it's essential to acknowledge that voters can only detect malware interference post-vote when comparing their receipt to the bulletin board. Additionally, a voter cannot provide evidence to others that their vote was compromised by malware, which means these instances aren't audited within the PeaceFounder system. As a result, members are encouraged and are responsible for utilising more secure devices less susceptible to malware attacks. For more advanced threats, like a briber mandating malware installation for monitoring or extracting the master key, the use of tamper-resistant hardware becomes essential – an extension larger organisations or states might consider.

Conclusion

In conclusion, in comparision with existing evoting systems PeaceFounder offers a new balance between privacy, transparency and security. To my knowledge it is the first system which allows fully centralized control while providing public evidence making it ideal for integration for small and medium sized organizations. The history tree bulletin board implementation ensures immediate finality, superior scalability, consolidates responsibility that simplifies deployment and audits, features externally imposed accountability and protects against comodified trust which hinders blockchain based solutions. On top of that fairness and time restricted receipt freeness can be supported by delaying publishing of the votes while still being kept accountable through published receipt hashes. The unlinkability of the votes are ensured by external braiders which can offer their services transactionally with a possible monetary incentives and can be brokered for discoverability. Multiple extensions to this system such as assymetric encryption of the ballot selection, coerced vote tagging, early private key leakage detection, ballot sharding and participation roll can be explored further with more development resources.

References

1. Ben Adida. Helios: Web-based open-audit voting. In *Proceedings of the 17th Conference on Security Symposium*, SS'08, page 335–348, USA, 2008. USENIX Association.
2. David Chaum and Torben Pryds Pedersen. Wallet databases with observers. In Ernest F. Brickell, editor, *Advances in Cryptology — CRYPTO' 92*, volume 740, pages 89–105. Springer Berlin Heidelberg, 1993. Series Title: Lecture Notes in Computer Science.
3. Véronique Cortier, Pierrick Gaudry, and Stéphane Glondu. Belenios: A simple private and verifiable electronic voting system. In Joshua D. Guttman, Carl E. Landwehr, José Meseguer, and Dusko Pavlovic, editors, *Foundations of Security, Protocols, and Equational Reasoning*, volume 11565, pages 214–238. Springer International Publishing, 2019. Series Title: Lecture Notes in Computer Science.
4. Scott A. Crosby and Dan S. Wallach. Efficient data structures for tamper-evident logging. In *Proceedings of the 18th Conference on USENIX Security Symposium*, SSYM'09, page 317–334, USA, 2009. USENIX Association.
5. Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The Second-Generation onion router. In *13th USENIX Security Symposium (USENIX Security 04)*, San Diego, CA, August 2004. USENIX Association.
6. Janis Erdmanis. ShuffleProofs.jl: Verificatum compatible verifier and prover for NIZK proofs of shuffle. `https://github.com/PeaceFounder/ShuffleProofs.jl`, 2022.
7. Rolf Haenni, Philipp Locher, Reto Koenig, and Eric Dubuis. Pseudo-code algorithms for verifiable re-encryption mix-nets. In *Financial Cryptography and Data Security*, volume 10323, pages 370–384. Springer International Publishing, 2017. Series Title: Lecture Notes in Computer Science.
8. Rolf Haenni and Oliver Spycher. Secure internet voting on limited devices with anonymized dsa public keys. In *Proceedings of the 2011 Conference on Electronic Voting Technology/Workshop on Trustworthy Elections*, EVT/WOTE'11, page 8, USA, 2011. USENIX Association.
9. Aggelos Kiayias, Michael Korman, and David Walluck. An internet voting system supporting user privacy. In *2006 22nd Annual Computer Security Applications Conference (ACSAC'06)*, pages 165–174, 2006.
10. Aggelos Kiayias, Thomas Zacharias, and Bingsheng Zhang. End-to-end verifiable elections in the standard model. *IACR Cryptol. ePrint Arch.*, 2015:346, 2015.
11. Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Accountability: definition and relationship to verifiability. In *Proceedings of the 17th ACM conference on Computer and communications security - CCS '10*, page 526. ACM Press, 2010.
12. Krishna Sampigethaya and Radha Poovendran. A framework and taxonomy for comparison of electronic voting schemes. *Computers & Security*, 25(2):137–153, 2006.
13. Carsten Schürmann, Randi Markussen, Olivier Bélanger, and Lorena Ronquillo. Framing electoral transparency: A comparative analysis of three e-votes counting ceremonies. In *Proceedings of 24th World Congress of Political Science: Panel RC10.05*, 2016.
14. Drew Springall, Travis Finkenauer, Zakir Durumeric, Jason Kitcat, Harri Hursti, Margaret MacAlpine, and J. Alex Halderman. Security analysis of the estonian internet voting system. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 703–715. ACM, 2014.

15. Ewa Syta, Philipp Jovanovic, Eleftherios Kokoris Kogias, Nicolas Gailly, Linus Gasser, Ismail Khoffi, Michael J. Fischer, and Bryan Ford. Scalable bias-resistant distributed randomness. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 444–460. IEEE, 2017.
16. Björn Terelius and Douglas Wikström. Proofs of restricted shuffles. In Daniel J. Bernstein and Tanja Lange, editors, *Progress in Cryptology – AFRICACRYPT 2010*, Lecture Notes in Computer Science, pages 100–113. Springer, 2010.
17. Douglas Wikstrom. How to implement a stand-alone verifier for the verificatum mix-net. *verificatum.org*, 2011.