to confirm that the ballot is read properly. The codes are posted and processed through the backend. This approach sacrifices some privacy properties—the original ballots are uniquely identifiable with codes next to candidates—for better usability—the voter experience is nearly identical to traditional optical scan systems.

While for the most part these systems all use techniques which distribute the election authority into parts, the printer which is used to print the blank ballots is still a single monolithic entity which has legitimate access to all the information that is printed on the ballots. This single point of contention is of great concern for privacy.

Carback et al. [2] reduce this trust in PunchScan using independent ballot sheets, giving either printer a 50% chance to break voter privacy by printing the sheets separately. We build on this work and propose a three sheet ballot, each coming from an independent authority.

Moran and Naor [10] propose a PunchScan-like front-end with four sheets composed of 2 different sets of letters and 2 layers of indirection. This construction achieves the same property ClearVote achieves with three sheets and only 1 level of indirection: the ballot printers cannot know how the voter voted unless they collude. Also, ClearVote only requires voters to mark a ballot once instead of twice to indicate a choice.

Our front-end requires a back-end that supports re-encryption. In particular, we use a variant of the back-end found in the Helios voting system [1].

3. CLEARVOTE SYSTEM DEFINITION

All End-to-End voting systems have two stages. The first is the front-end that associates a coded vote to a candidate on each ballot and publishes the coded vote on a public bulletin board. The second is a back-end which computes the tally based on all coded votes. Mixnet-based back-ends transform the coded votes back to clear text candidate names, while homomorphic back-ends aggregate all coded votes into a coded tally and then decrypt the coded tally. While there are known techniques to distribute the back-end to multiple independent trustees such that no single one or no small coalition can compromise privacy and associate coded votes with clear text votes, this task is much more difficult for the front-end part.

A Prêt à Voter ballot directly associates candidates to coded votes. Thus the printer which produces this ballot knows the association. The PunchScan ballot makes the association indirectly by having two sheets of paper. Since one of the sheets becomes the voter's receipt and is publicly posted, the printer that prints the other page can still associate the coded vote with the voter's selections. ClearVote splits the ballot into three pages, each page being generated and printed by a different entity. Each page contains an equal amount of information. In this sense, the pages are symmetric. If we assume that no two entities collude, no entity is able to break ballot privacy. From a back-end perspective, in ClearVote, each trustee is responsible for decrypting its own part independently of the other two entities, thus preserving the truly distributed nature of the back-end.

For the front-end, each ClearVote trustee contributes a shift amount to the clear text vote. The shift amounts are additive and cyclical, thus the order of composition does not matter. For example, assuming there are 5 candidates (0,1,2,3 and 4), a clear text vote of 1 (a vote for the second candidate) can be transformed into coded vote 2 by the following three shift amounts: the first trustee adds a shift of 2, the second one a shift of 1 and the third one a shift of 3. If we add the clear text vote 1 to the three shift amounts we get 2 mod 5. To recover the clear text vote, each trustee subtracts from the coded vote the shift amount that he initially contributed. Thus, 2-2-1-3=1 (mod 5).

The voter chooses at random one of the shift amounts to be made public as a receipt. This public shift amount can be used to probabilistically check the correct printing of the entire ballot. Each trustee only knows his own shift amount, and the public shift amount from the receipt (provided by another trustee). Without the third shift amount from the last trustee, it is equally likely that the coded vote corresponds to any clear text vote. For example, if the coded vote is 2, the public shift amount is 3, and the trustees shift amount is 1, this would result in a partially decoded value of 2-3-1=3. Since the shift amount of the third trustee can be 0,1,2,3 or 4, equally likely, the clear text vote may be 3,2,1,0 or 4, equally likely. Thus a single trustee does not learn any information about the clear text vote from the coded vote, the public shift and its own shift.

3.1 Voter Experience

The voter experience is similar to traditional optical scan voting with a few extra steps when the ballot is issued and before the receipt is scanned. When a voter arrives at the polling place she is directed through the following procedures:

1. The voter is asked to authenticate herself. After proper authentication, the voter is handed a ballot card. The ballot card acts as an authentication token for election judges at the next step.

2. The voter is directed to three ballot issuing tables. At each table the voter selects a sheet in an unpredictable fashion. After the voter selects the sheet she presents the ballot card to the election judge. The judge writes the serial number of the selected sheets on the card. At the end of this process, the voter has three ballot sheets and a ballot card with three serial numbers on it. The voter can check that the serial numbers on her ballot sheets are consistent with the serial numbers from the ballot card.

3. With her three sheets, the voter is directed to the voting booth, where the voter stacks the three sheets. The sheets are printed on plastic transparencies (similar to those used for overhead projectors), so the stacking order of the non-receipt sheets does not matter. Figure 1 shows how the ballot is formed.

4. To vote, the voter finds her desired candidate in the list, makes a mental note of the symbol next to that candidate, and marks that symbol in a selection area under the candidate list. For the example in Figure 1,

Figure 1: A ClearVote ballot is composed of 3 transparent sheets stacked on top of each other. The letters next to candidates indicate which position to select in the row of characters below the candidate list.

if the voter wants to vote for Bob, she sees that Bob has "E" next to his name, and marks the third circle. It is sufficient if only the top sheet contains the mark

5. After voting, the voter shreds the bottom two (non-receipt) sheets. This ensures the information on those sheets is securely destroyed. The voter hands the surviving top sheet and ballot card to a judge at a scanning station. Both the card and sheet are scanned and recorded. The information on each is digitally signed and the signature is printed back onto the sheets. The voter gets back the signed sheets, and keeps them as a receipt.

After the polls close the voter can inspect a public bulletin board (*e.g.,* a web site). The voter can locate her receipt on the bulletin board by the serial number of her receipt (or any of the three serial numbers on the ballot card). She sees three things on the bulletin board:

1. all three serial numbers as written down on the yellow piece of paper.
2. the marked circle consistent with the circle marked on her receipt.
3. one of the three following lists (only one of them), depending on which sheet the voter kept:
   (a) the order of the candidates.
   (b) the order of the symbols next to the candidates.
   (c) the order of the symbols in the circles.

If the information on the public bulletin board is inconsistent with the information that the voter has on her receipt, she can bring the receipt as proof of malfeasance. If not, the voter has verified that her ballot was correctly printed and recorded.

3.2 The ClearVote Ballot

As shown in Section 3.1, the voter interacts with a three sheet ballot. There are three separate authorities that create this ballot. Each authority produces a type of sheet that contains a different area of the ballot: (1)The candidates in a random order. (2)The candidate symbols in a random order. (3)The marking area symbols in a random order.

The random orders are random shifts rather than random permutations. Composing shifts is commutative whereas composing permutations is not commutative, and commutativity is needed for securely decrypting the votes, as presented in Section 3.3.8.

Recall that the voter receives one sheet as a receipt and destroys the other sheets. Calculating the plaintext choice requires taking the positional data from the receipt (the number of the marked circle) and "subtracting" the three random shifts which contributed to the transformation of the clear text vote into this coded vote.

3.3 Back-end

In this section, we describe the cryptographic protocol carried out by the printing authorities, including auditing steps. Readers accustomed with the Helios system [1] should find this section familiar. After briefly describing background on exponential Elgamal, we describe each step of the election from initializing the system to the final tally audit.

3.3.1 Elgamal Background

Let g be a generator of a cyclic group $\mathbf{G}$ in a typical Elgamal setting. Let x be the private key of George, and $h = g^x$ the public key. To encrypt message m, Alice chooses a random r and computes $(P, Q) = (g^r, g^m h^r)$. To decrypt (P, Q), George computes $Z = \frac{Q}{P^x} = g^m$ and tries all possible m to see which one is consistent with Z. If the message m can only take few values, recovering m from g^m can be done using a look-up table. In ClearVote, m takes values from $\mathbb{Z}_c$ for a small value of c, where c represents the number of candidates in a race.

To re-encrypt (P, Q), Bob chooses an appropriate random r' and computes $(P', Q') = (Pg^{r'}, Qh^{r'}) = (g^{r+r'}, g^m h^{r+r'})$. Bob provides a proof that (P', Q') is a re-encryption of (P, Q). Bob can add a message to the one already encrypted by computing $(P', Q') = (P, Qg^{m'}) = (g^r, g^{m+m'} h^r)$. Bob can do both operations, re-encryption and message addition, without the private key and without the need to know what message is encrypted.

3.3.2 Election Initialization

Each of the three authorities—Zero, One, and Two—compute a shift amount for each sheet, publishes a commitment to the shift amount to be used for printing his sheets, and publishes an encryption of the opposite of the shift amount.

Let n be the number of ballots and let c be the number of candidates on the ballot. Let m_i^j be the shift amount for ballot i computed by authority j, $\forall i \in \mathbb{Z}_n$, $\forall j \in \mathbb{Z}_3$ and $\forall m_i^j \in \mathbb{Z}_c$.

Each ballot sheet has a unique serial number which is the numeral j followed by numeral i. All sheets produced by authority Zero start with prefix 0, all sheets produced by authority One start with prefix 1 and all sheets produced by authority Two start with prefix 2. A ballot consists of sheets $0s_1$, $1s_2$, $2s_2$, i.e. it is not necessary to have the same suffix, but the prefix (sheet type) must be different.

Let g be an Elgamal generator for a fixed group $\mathbf{G}$ in a typical Elgamal setting. Let x_j be the private key of authority j and let $h_j = g^{x_j}$ be the public key of authority j.

For each $i \in \mathbb{Z}_n$, each authority $j \in \mathbb{Z}_3$ computes the shift amount $m_i^j \in \mathbb{Z}_c$ to be printed on the ballot sheet ji, and the additive inverse of the shift amount, which is

going to be used for decryption $\overline{m}_i^j$, $m_i^j + \overline{m}_i^j \ mod \ c = 0$. For each ballot, each authority publishes on a public bulletin board a commitment to each shift amount (i, j, m_i^j). The authority does not post the tuple itself, but only a commitment to it. The authority also publishes an encryption of the amount of shift for decryption, the tuple $(i, j, P_i^j, Q_i^j) = (i, j, g^{r_i^j}, g^{\overline{m}_i^j} h_j^{r_i^j})$, where r_i^j is a Elgamal random number.

3.3.3 Pre-Printing Audit

For each authority j, an independent auditor[1] selects a statistically significant random set of indexes $A_j \subset \mathbb{Z}_n$. For each $i \in A_j$, authority j opens the commitment to the tuple (j, i, m_i^j) and reveals the randomness r_i^j used in computing the Elgamal encryption for that i. The auditor can check the commitments and can check that the Elgamal encryption contains $\overline{m}_i^j = -m_i^j \ mod \ c = 0$.

This check ensures that the shift amounts promised to be used when printing the ballot sheets are going to be correctly compensated in the decryption process. In other words, if the the opened commitment reveals a shift amount of 3, then the Elgamal encryption contains a shift amount of -3.

Since the set A_j is random and statistically significant, it follows that the commitments that were not opened for checking, contain, with high probability, information that is consistent with the ones in the encrypted tuples (P^j, Q^j).

3.3.4 Printing

Each authority j prints all the ballot sheets that were not opened in the first audit, i.e. $i \in \mathbb{Z}_n - A_j$. Each of the authorities prints only one of 3 ballot sheet types.

3.3.5 Voting

Recall from Section 3.1 that the voter goes to the polling place and randomly chooses three sheets from the three different authorities to form a ballot. A record is made of what serial numbers the voter chose. Both the voter and the poll workers must have irrefutable proof about the particular triple of serial numbers, such that the voting system cannot later claim that the voter chose a different set of sheets (the voter would have a proof in this case), nor can the voter make the same claim (because the poll workers would have a proof).

After making her selections, the voter shreds the sheets that do not have marks on them (the two bottom ones), scans the sheet with the marks (the top one) and keeps a signed copy of the sheet with the marks. The signed copy prevents the voter from adding extra marks after the sheet was scanned. The signed copy, along with the other two serial numbers of the shredded sheets, becomes the voter's receipt. All the receipts are posted on the public bulletin board.

3.3.6 Auditing the Printing

For each of the sheets that the voters kept as their receipts, the corresponding election authority opens the commitment to that sheet. If sheet ji was kept as a receipt, then authority j opens the commitment to (j, i, m_i^j) and posts this information on the public bulletin board. The consistency between the opened commitments and the posted receipts can be checked by anyone.

The voter can access the public bulletin board and type in the serial number of her receipt ji to get a copy of the sheet she kept. The voter can check that the printing on the sheet she has is consistent with the shift amount m_i^j. If not, her receipt serves as irrefutable proof of malfeasance.

Given a fixed sheet ji the probability that the voter is going to keep this ji sheets as her receipt is $\frac{1}{3}$. It follows that if authority j misprinted the sheet ji, and if the voter checks her receipt, the probability that she does not detect the incorrect printing is $\frac{2}{3}$ (because any of the other two sheets may have been kept by the voter). Since we assume that the voters choose independently at random which sheet to keep as a receipt, it follows that if authority j misprinted k sheets, then the probability that no voter detects this misprint is $(\frac{2}{3})^k$. For example, for if 20 sheets were misprinted then the probability of not detecting any of them is approximately 0.03%.

3.3.7 Checking Receipt Correctness

The position that the voter marked represents the coded vote and it is posted on the public board, along with the other two serial numbers of the sheets that the voter shredded. The voter can check that the bulletin board contains the same information as her receipt. If not, the voter's receipt is irrefutable proof of malfeasance. Anybody can check that the posted receipts do not contain over-votes.

3.3.8 Tallying Election Results

Each receipt contains the three serial numbers of the sheets that the voter chose. Given a serial ji, the message (i, j, P_i^j, Q_i^j) is identified. Each receipt is translated into the tuple $[v_0, (i_0, 0, P_{i_0}^0, Q_{i_0}^0), (i_1, 1, P_{i_1}^1, Q_{i_1}^1), (i_2, 2, P_{i_2}^2, Q_{i_2}^2)]$, where v_0 represents the coded vote, and $(P_{i_j}^j, Q_{i_j}^j)$ represents the encryption of the shift amount $\overline{m}_i^j$ for ballot i, authority j. The first two arguments in each encrypted message are stripped off, to result in $[v_0, (P_{i_0}^0, Q_{i_0}^0), (P_{i_1}^1, Q_{i_1}^1), (P_{i_2}^2, Q_{i_2}^2)]$,

Each authority is running a mixnet. The input to each mixnet consists of a number of tuples, each tuple containing a coded vote and the encrypted shifts for all the other authorities. To simplify the presentation, we describe what each mixnet does to each input message, but we ignore the mixing part. The reader should keep in mind that the mixnet also reorders the output messages before publishing them such that no output message can be traced to an input message.

Authority Zero receives $[v_0, (P_{i_0}^0, Q_{i_0}^0), (P_{i_1}^1, Q_{i_1}^1), (P_{i_2}^2, Q_{i_2}^2)]$, and performs the following computations:

1. computes two random shifts $m_i', m_i'' \in \mathbb{Z}_c$
2. decrypts the message that is encrypted with her public key $\frac{Q_{i_0}^0}{(P_{i_0}^0)^{x_0}}$ and finds $\overline{m}_i^0$.

[1] We could select multiple auditors, *e.g.,* representatives from each candidate.

3. adds the computed shift to the coded vote and subtracts the two random shifts $v_1 = v_0 + \overline{m}_i^0 - m_i^{'} - m_i^{''}$

4. re-encrypts the encrypted message for authority One and adds one of the random shift to the encrypted shift $(P_{i_1^{'}}^1, Q_{i_1^{'}}^1) = (P_{i_1}^1 * g^{r_{i1}^{'}}, Q_{i_1}^1 * h^{r_{i1}^{'}} * g^{m_i^{'}})$

5. re-encrypts the encrypted message for authority Two and adds the other random shift to the encrypted shift $(P_{i_2^{''}}^2, Q_{i_2^{''}}^2) = (P_{i_2}^2 * g^{r_{i1}^{''}}, Q_{i_2}^2 * h^{r_{i1}^{''}} * g^{m_i^{''}})$

The resulting outputs of the form $[v_1, (P_{i_1^{'}}^1, Q_{i_1^{'}}^1), (P_{i_2^{''}}^2, Q_{i_2^{''}}^2)]$ are randomly shuffled and posted on the public bulletin board.

Authority One receives $[v_1, (P_{i_1^{'}}^1, Q_{i_1^{'}}^1), (P_{i_2^{''}}^2, Q_{i_2^{''}}^2)]$ and performs the following computations:

1. computes a random shift $m_i^{'} \in \mathbb{Z}_c$

2. decrypts the message that is encrypted with her public key $\frac{Q_{i_1^{'}}^1}{(P_{i_1^{'}}^1)^{x_1}}$ and finds $\overline{m}_i^{11}$

3. adds the computed shift to the coded vote and subtracts the random shift $v_2 = v_1 + \overline{m}_i^{11} - m_i^{'}$

4. re-encrypts the encrypted message for authority Two and adds the random shift
$(P_{i_2^{''}}^2, Q_{i_2^{''}}^2) = (P_{i_2^{''}}^2 * g^{r_{i2}^{''}}, Q_{i_2^{''}}^2 * h^{r_{i2}^{''}} * g^{m_i^{'}})$

The resulting outputs of the form $[v_2, (P_{i_2^{''}}^2, Q_{i_2^{''}}^2)]$ are randomly shuffled and posted on the public bulletin board.

Authority Two receives $[v_2, (P_{i_2^{''}}^2, Q_{i_2^{''}}^2)]$ and performs the following computations:

1. decrypts the message that is encrypted with her public key $\frac{Q_{i_2^{''}}^2}{(P_{i_2^{''}}^2)^{x_2}}$ and finds $\overline{m}_i^{22}$

2. adds the computed shift to the coded vote $v_3 = v_2 + \overline{m}_i^{22}$

The resulting outputs of the form $[v_3]$ are randomly shuffled and posted on the public bulletin board. They represent the clear text votes and can be tallied by anyone. For example, 0 represents a vote for Alice, 1 a vote for Bob, 2 a vote for Carol, etc.

In summary, the voter transforms an initial clear text vote v into coded vote v_0 by marking the three sheet ballot. $v_0 = v + m_{i_0}^0 + m_{i_1}^1 + m_{i_2}^2$. The back-end recovers $v_3 = v$ from v_0 by performing $v_3 = v_0 + (\overline{m}_{i_0}^0 - m_{i_0}^{'} - m_{i_0}^{''}) + (\overline{m}_{i_1}^1 + m_{i_0}^{'} - m_{i_1}^{'}) + (\overline{m}_{i_2}^2 + m_{i_0}^{''} + m_{i_1}^{'}) = v_0 + \overline{m}_{i_0}^0 + \overline{m}_{i_1}^1 + \overline{m}_{i_2}^2$. Since $m_{i_j}^j = -\overline{m}_{i_j}^j$, then $v_3 = v$.

3.3.9 Post-Election Mixnet Audit

Each authority publishes a proof that the operations it performed were correctly done for each input message, and that it did not incorrectly transform, delete or inject any messages.

We present a simple Randomized Partial Checking [8] proof. Each authority controls a mixnet that consists of two mixes. For example $\overline{m}_i^j$ is split into two numbers that sum up to $\overline{m}_i^j$; also, the re-encryption of the other messages in the tuple is performed by each mix. The output of the first mix of each mixnet publishes its intermediary results. Then, the mixnet is audited by flipping a coin on each of the outputs of the first mix. If the coin is heads, the pre-image of this output is revealed and the mix publicly shows how it performed all the operations, including the mixing (permutation), for that output. If the coin is tails, the post-image of the intermediary output is revealed, and the second mix publicly shows how it performed all the operations, including the mixing (permutation) for that input. This way, no link from the output of the entire mixnet to the input of the mixnet is fully revealed. The partial links that are revealed prove that, if k transformations were incorrectly done, then the probability that none of the incorrect transformations are detected is $\frac{1}{2^k}$ for each authority.

4. SYSTEM PROPERTIES

The mixnet verifiability is not new so we focus on privacy and usability.

4.1 Privacy

ClearVote is based on the assumption that no two authorities collude. If the assumption holds, no single authority can find out how any of the voters voted.

The voter cannot use the receipt that she kept to prove how she voted. All three sheets are needed to reconstruct the clear text vote. The receipt only contains a third of the information, insufficient to make an educated guess about what candidate received the vote.

Since there is no single printer that prints all three ballot sheets, the printer is no longer a single point of failure for ballot confidentiality. For example, if a voter chose the sheet printed by authority Zero to keep as a receipt, and thus the shift amount from that sheet is fully revealed, authority One cannot find out how the voter voted, because, although it knows the shift amount that authorities Zero and One contributed, it does not know the shift amount that is added by authority Two. The same argument is valid for any authority.

During the mixing phase, because authority Zero re-encrypts the shift amounts of authority One, authority One cannot trace the message through the mixnet of authority Zero. Authority Zero can modify the shift amount inside the encryption without needing to know the shift amount already inside the encryption (the homomorphic property of exponential Elgamal). As a result, authority One does not break the input and the output of the mixnet of authority Zero into smaller privacy sets by decrypting the corresponding parts of the input and output and classifying the decryption

according to the shift amount. This is because authority Zero modifies the shift amounts inside the encryption corresponding to authority One. A similar discussion is valid for authorities One and Two.

In conclusion, no authority can trace any of the messages through the mixnet of another authority.

The ClearVote ballot suffers from some of the same privacy attacks as the PunchScan ballot. If the voter doesn't want to vote for any candidate, this will be visible on the receipt, since there will not be a selection. Thus an attacker can force a voter to abstain. The same attacks can be conducted by forcing the voter not to go to the polling place.

Forced randomization attacks are also possible. The attacker can coerce the voter into bringing a receipt that always has the first position marked, or to mark a position based on the order observed on the receipt sheet. This would essentially force the voter to cast her ballot for a random candidate, since the first position corresponds to an unknown candidate.

To avoid attacks based on which sheet the voter chooses to keep as a receipt after she sees the shift amount on all sheets, the system can force the voter to commit to the chosen receipt sheet before she sees any of the three sheets.

4.2 Usability

Indirection on the PunchScan ballot is often raised as a usability concern, and ClearVote may have slightly worse usability properties. These problems stem from the transparencies, the multiple sheets, and the requirment to record serial numbers on the ballot card.

Instructing voters to put the receipt sheet on top is likely to be a major problem. Voters may not comply with this instruction, and by the time a mistake can be caught the unmarked sheets will have already been destroyed. A physical mechanism that forces the receipt sheet to be the top sheet (via a poll worker locking it into place) would be better. The additional level of indirection provided by the randomized candidate order may also confuse voters.

The transparencies may be harder to read for some voters. They may also be harder to mark and to scan. It may present a challenge to find a marking device that voters can use which does not smear from voter's hands or in the scanner.

5. CONCLUSIONS

We have shown that indirection is a useful approach to improve privacy in E2E voting systems. ClearVote takes PunchScan's indirection and adds an additional layer of indirection found in the randomized candidate order of Prêt à Voter. The result is no dependence on a single printer and thus no single entity which can break the confidentiality of cast ballots.

Ideally, trust would be spread out among an arbitrary number of authorities. It is possible to further improve on this design, but not likely by adding more sheets. Instead, indirection and specifying a partial printing processes for multiple authorities may be a promising direction.

6. REFERENCES

[1] Ben Adida. Helios: Web-based open audit voting. In *Proceedings of the Fourteenth USENIX Security Symposium*. Usenix, July 2008.

[2] Richard Carback, Stefan Popoveniuc, Alan T. Sherman, and David Chaum. Punchscan with independent ballot sheets: Simplifying ballot printing and distribution with independently selected ballot halves. In *IAVoSS Workshop On Trustworthy Elections (WOTE 2007)*, University of Ottawa, Canada, June 2007.

[3] David Chaum. Secret-ballot receipts: True voter-verifiable elections. *IEEE Security and Privacy*, pages 38–47, January/February 2004.

[4] David Chaum, Aleks Essex, Richard Carback, Jeremy Clark, Stefan Popoveniuc, Alan T. Sherman, and Poorvi Vora. Scantegrity: End-to-end voter verifiable optical-scan voting. *IEEE Security and Privacy*, May/June 2008.

[5] David Chaum, Peter Y. A. Ryan, and Steve Schneider. A practical voter-verifiable election scheme. In *In Sabrina De Capitani di Vimercati, Paul F. Syverson, and Dieter Gollmann, editors, ESORICS, volume 3679 of Lecture Notes in Computer Science*, pages 118–139. Springer, 2005.

[6] David L. Chaum. Untraceable electronic mail, return address, and digital pseudonym. *Communication of ACM*, February 1981.

[7] Kevin Fisher, Richard Carback, and Alan T. Sherman. Punchscan: Introduction and system definition of a high-integrity election system. In *IAVoSS Workshop On Trustworthy Elections (WOTE 2006)*, Robinson College, Cambridge UK, June 2006.

[8] Markus Jakobsson, Ari Juels, and Ronald L. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In *Proceedings of the 11th USENIX Security Symposium*, pages 339–353, Berkeley, CA, USA, 2002. USENIX Association.

[9] David Lundin. Component based electronic voting systems. In *IAVoSS Workshop On Trustworthy Elections (WOTE 2007)*, University of Ottawa, Canada, June 2007.

[10] Tal Moran and Moni Naor. Split-ballot voting: everlasting privacy with distributed trust. In *ACM Conference on Computer and Communications Security*, pages 246–255, 2007.

[11] Stefan Popoveniuc and Ben Hosp. An introduction to PunchScan. In *IAVoSS Workshop On Trustworthy Elections (WOTE 2006)*, Robinson College, Cambridge UK, June 2006.

[12] Stefan Popoveniuc and Poorvi Vora. A framework for secure electronic voting. In *IAVoSS Workshop On Trustworthy Elections (WOTE 2008)*, Katholieke Universiteit, Leuven, Belgium, July 2008.