ClearVote: An End-to-End Voting System that Distributes Privacy Between Printers

Stefan Popoveniuc
KT Consulting
Gaithersburg, MD
stefan@popoveniuc.com

Richard Carback
UMBC
Baltimore, MD
carback1@umbc.edu

ABSTRACT

In many end-to-end voting systems there is a single entity that produces each ballot. This entity can be the printer in the case of paper ballots, or the voting machine in the case of an electronic interface. While not able to change election results, this powerful entity has access to confidential information and can reveal selections made by the voters which, along with the voter's identities, can compromise the secrecy of the ballot.

We propose ClearVote, a new end-to-end voting system that has no single entity that can reveal ballot selections. The ClearVote ballot has three sheets of transparent plastic, each sheet coming from a different printer. Assuming no two printers collude, there is no single entity with enough knowledge to reveal ballot selections.

Categories and Subject Descriptors

J.1 [**Computer Applications**]: ADMINISTRATIVE DATA PROCESSING—*Government*; H.4.0 [**Information Systems**]: INFORMATION SYSTEMS APPLICATIONS—*General*

General Terms

Design, Security, Verification

Keywords

cryptographic voting, end-to-end verifiable election systems

1. INTRODUCTION

Paper ballot end-to-end (E2E) verifiable voting systems often make the assumption that the printer used to create the ballots is trusted with voter privacy. This assumption is a weakness with regard to voter privacy. An attacker that compromises the printer could use information printed on the voter's receipt to determine how she voted. ClearVote spreads the trust out among multiple independent printers, requiring two printers to collude in order to violate voter privacy.

WPES'10, October 4, 2010, Chicago, Illinois, USA.

Privacy problems of this nature are pervasive in voting systems and similar attacks exist in traditional systems as well. If a voter interacts with a machine to vote, the machine is trusted with voter privacy. In a traditional system, the machine knows each voter's choices, and, in an E2E system, the machine could identify the selections from a privacy preserving receipt. In traditional paper systems, information that can identify voters can be added to the ballot. One example is England's ballot act of 1872, which introduced the secret ballot. It requires a serial number on the back of the ballot and counterfoil stubs that can be used to identify voters. This counterfoil process is still used.

Our contribution is a system that resists pervasive privacy attacks of the nature we describe above. A printer cannot determine how a voter has voted based on her receipt without at least the collusion of one other printer. A scanner or election official cannot determine how a voter has voted based on the portion of the ballot scanned. In this way, we reduce the trust required to protect voter privacy to the voting booth (*i.e.,* the attacker will have to look over the voter's shoulder or have the voter take recording equipment into the voting booth).

2. RELATED WORK

Our work is part of a family of approaches that descend from an earlier paper-based system by Chaum [3]. This initial system prints voter selections on two transparent sheets using visual cryptography. The voter "encrypts" the ballot by destroying one sheet. The surviving half is publicly posted and processed through a mixnet [6] that verifiably and anonymously computes election results. Prêt à Voter [5], PunchScan [10, 7], and Scantegrity [4] are descendants of this system. They uses pre-printed ballots which are marked directly by the voter.

While these systems 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 concern for privacy.

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

Moran and Naor [9] 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. There are known techniques to distribute the back-end to multiple independent trustees such that no small coalition can compromise privacy and associate coded votes with clear text votes, but this task is much more difficult for the front-end part. We discuss both in this section.

3.1 The Ballot & Voter Experience

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 3.2.3.

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 ballot 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.

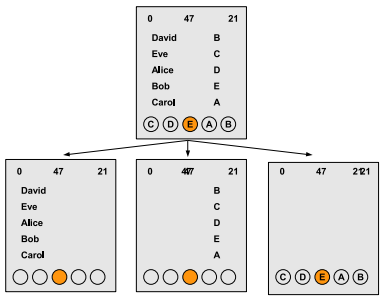

Figure 1: A ClearVote ballot is composed of 3 transparent sheets stacked on top of each other. The letters next to candidates indicate which position to select in the row of characters below the candidate list. The voter selected Bob in this example.

5. After voting, the voter shreds the bottom two (non-receipt) sheets. 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 two things on the bulletin board: The ballot serial numbers, and a reconstruction of her receipt.

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 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. We expect readers to be familiar with exponential Elgamal.

3.2.1 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 au-

thority 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.

3.2.2 Ballot Printing

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 $\overline{m}_i^j = -m_i^j \ mod\ c = 0$, guaranteeing that, with high probability, the commitments that were not opened are consistent with those in the encrypted tuples (P^j, Q^j).

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.

Print Audit. For each of the sheets that the voters kept as their receipts, the corresponding election authority opens the commitment to that sheet. 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.

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 $(2/3)^k$. For example, for if 20 sheets were misprinted then the probability of not detecting any of them is approximately 0.03%.

3.2.3 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 $[v_0, (P_{i_0}^0, Q_{i_0}^0), (P_{i_1}^1, Q_{i_1}^1), (P_{i_2}^2, Q_{i_2}^2)]$,

Authority Zero initially reads the data from the bulletin board: $[v_0, (P_{i_0}^0, Q_{i_0}^0), (P_{i_1}^1, Q_{i_1}^1), (P_{i_2}^2, Q_{i_2}^2)]$. Using that, she computes two random shifts $m_i^{'}, m_i^{''} \in \mathbb{Z}_c$, decrypts the message that is encrypted with her public key $Q_{i_0}^0/((P_{i_0}^0)^{x_0})$ and finds $\overline{m}_i^0$. Then she adds the computed shift to the coded vote and subtracts the two random shifts $v_1 = v_0 + \overline{m}_i^0 - m_i^{'} - m_i^{''}$, re-encrypts the encrypted message for authority One and adds one of the random shifts to the encrypted shift $(P_{i_1}^1, Q_{i_1}^1) = (P_{i_1}^1 * g^{r_{i_1}^{'}}, Q_{i_1}^1 * h^{r_{i_1}^{'}} * g^{m_i^{'}})$, and re-encrypts the encrypted message for authority Two and adds the other random shift to the encrypted shift $(P_{i_2}^2, Q_{i_2}^2) = (P_{i_2}^2 * g^{r_{i_1}^{''}}, Q_{i_2}^2 * h^{r_{i_1}^{''}} * g^{m_i^{''}})$. 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 $[v_1, (P_{i_1}^1, Q_{i_1}^1), (P_{i_2}^2, Q_{i_2}^2)]$. She computes a random shift $m_i^{'} \in \mathbb{Z}_c$, decrypts the message that is encrypted with her public key $Q_{i_1}^1/((P_{i_1}^1)^{x_1})$, and finds $\overline{m}_i^{11}$. Then adds the computed shift to the coded vote, subtracts the random shift $v_2 = v_1 + \overline{m}_i^{11} - m_i^{'}$, re-encrypts the encrypted message for authority Two and adds the random shift $(P_{i_2^{''}}^2, Q_{i_2^{''}}^2) = (P_{i_2^{''}}^2 * g^{r_{i_2}^{''}}, Q_{i_2^{''}}^2 * h^{r_{i_2}^{''}} * g^{m_i^{'}})$. 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 $[v_2, (P_{i_2^{''}}^2, Q_{i_2^{''}}^2)]$. She decrypts the final message which is encrypted with her public key $Q_{i_2^{''}}^2/((P_{i_2^{''}}^2)^{x_2})$, finds $\overline{m}_i^{22}$, and 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.

To audit the tally, 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 $1/2^k$ for each authority.

4. SYSTEM PROPERTIES

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

4.1 Privacy

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.

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

During the mixing phase, because authority Zero also 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 does not vote for any candidate this will be visible on the receipt. Thus an attacker can force a voter to abstain. The same attack 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. 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, we point to indirection and specifying a partial printing processes for multiple authorities as possible solutions.

6. REFERENCES

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

[2] R. Carback, S. Popoveniuc, A. T. Sherman, and D. 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] D. Chaum. Secret-ballot receipts: True voter-verifiable elections. *IEEE Security and Privacy*, pages 38–47, January/February 2004.

[4] D. Chaum, A. Essex, R. Carback, J. Clark, S. Popoveniuc, A. T. Sherman, and P. Vora. Scantegrity: End-to-end voter verifiable optical-scan voting. *IEEE Security and Privacy*, May/June 2008.

[5] D. Chaum, P. Y. A. Ryan, and S. 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] D. L. Chaum. Untraceable electronic mail, return address, and digital pseudonym. *Communication of ACM*, February 1981.

[7] K. Fisher, R. Carback, and A. 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] M. Jakobsson, A. Juels, and R. 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] T. Moran and M. Naor. Split-ballot voting: everlasting privacy with distributed trust. In *ACM Conference on Computer and Communications Security*, pages 246–255, 2007.

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