On the Security of Ballot Receipts in E2E Voting Systems

Jeremy Clark, Aleks Essex, and Carlisle Adams

School of Information Technology and Engineering (SITE),
University of Ottawa
{jclar037,aesse083,cadams}@site.uottawa.ca

Abstract. This paper examines and compares the security of ballot receipts in three end-to-end auditable (E2E) voting systems: Prêt à Voter, Punchscan, and ThreeBallot. Ballot receipts should have two properties: from a privacy perspective, they should provide no information as to how the ballot was cast, and from an integrity perspective, they should provide no information that would assist an adversary in tampering with the tallying process. We find that Prêt à Voter and Punchscan have similar security properties with respect to ballot receipts, and provide no non-negligible information on the receipt itself that could compromise privacy or security (assuming the underlying cryptography is secure). However we show that ThreeBallot receipts leak partial information that is useful for compromising voter privacy and the integrity of the tally.

1 Introduction

In recent years, there has been an increased interest in cryptographic voting systems designed to protect voter privacy and provide voter-verifiable integrity. A set of voting guidelines published by the US Election Assistance Commission in 2005 includes a section on "end-to-end cryptographic independent verification" (E2E) voting systems [1]. The criteria for an E2E system includes, but is not limited to, the use of cryptography to secure voted ballots prior to tallying, the issuing of ballot receipts to voters, and an end-to-end audit mechanism for independent verification of the election results. The scope of this paper is limited to a security analysis of the ballot receipts in three E2E systems. In keeping with [1], we propose that any E2E ballot receipt should satisfy the following two properties:

Property 1. *The ballot receipt should provide no information that would increase an adversary's ability to determine how the ballot was cast.*

Property 2. *The ballot receipt should provide no information that would increase an adversary's ability to corrupt ballots without detection.*

The former is a privacy property designed to ensure a secret ballot. In section 2, we will define what is meant by 'no information' and derive an attack game to test for this property. In section 3, we will apply it to the three E2E systems. The second property is critical to preserving the integrity of the election. Section 4 and 5 will examine the three systems with respect to this property, both in terms of the risk to the adversary of being caught and the potential benefits to be gained.

We compare the following three systems: Prêt à Voter [4, 11], Punchscan [8, 2], and ThreeBallot [10, 9]. A Prêt à Voter ballot contains a candidate list in a randomized order. To vote, the voter simply places a mark beside their chosen candidate. The candidate list is then detached and shredded while the position of the mark is recorded and kept by the voter as a receipt. The receipt also contains a mix network onion which can be used to reconstruct the destroyed information during the tally.

Punchscan uses two sheets of paper. The top sheet contains a fixed-order candidate list, a set of symbols beside the names in a randomized order, and a series of holes. The bottom sheet contains the same set of symbols, also in random order, printed such that they show through the holes. To vote, the voter notes the symbol located beside their chosen candidate, then finds the hole containing the same symbol and daubs it with ink—filling in the hole and leaving a rim of ink around it. The voter shreds one of the sheets, and the marked positions on the other sheet are recorded by the poll worker. This sheet is then returned to the voter to be kept as a receipt. The receipt contains a serial number which can be used to reconstruct the vote from the half-ballot during the tally.

ThreeBallot employs a "multiballot," which is simply a set of three conventional ballots. To vote, the voter makes a mark for each candidate once on any of the three ballots and then completes her vote by voting a second time for the chosen candidate. The voter retains a copy of one of the three ballots as a receipt, and then the three ballots are separated and deposited into a ballot box. For the purpose of this paper, we consider ThreeBallot to be an E2E system because it is designed to provide the same properties as E2E systems [9], however it should be noted that it accomplishes this without the use of cryptography and therefore would not strictly meet the EAC's specification [1].

2 A Ballot Receipt Attack Game

The first property of an E2E ballot receipt is that it protects voter privacy by not revealing which candidate(s) the voter chose. However it is important to qualify the amount of information about the voter's choice that is not revealed by a ballot receipt.

Definition 2.1. *A ballot receipt contains* ***Insufficient Information*** *if it cannot be used in any manner to prove the cast vote of its respective ballot with certainty.*

Definition 2.2. *A ballot receipt contains* ***No Non-negligible Information*** *if it cannot be used in any manner to guess the cast vote of its respective ballot with non-negligible advantage over a random guess by a PPT-bounded (probabilistic polynomial-time) adversary.*

The former definition is less strict than the latter. Prêt à Voter, Punchscan, and ThreeBallot all generally meet the first definition. Note that in the case of ThreeBallot, with both the ballot receipt and the bulletin board of all marked ballots, it has been shown that ballot receipts may not meet the first criterion when the number of contests and candidates is sufficiently high [12]. Countermeasures have been suggested [10]. The analysis in this paper is concerned with security properties that exist even at the base case of 1 contest and 2 candidates. In order to test the systems against the second definition, we formulate an attack game. Attack games are common in establishing the formal provable security of cryptographic systems (i.e., [6]). Our attack game is outlined in Algorithm 1.

Algorithm 1: Attack Game with a Random Voting Oracle

```
1  Oracle:
2      Choose random candidate $c^* \in C$.
3  end
4  Oracle:
5      Run voting function $r^* = \mathbf{V}(c^*)$ as follows:
6          If applicable, choose random permutation $p \in P$.
7          If applicable, choose random marking style $s \in S$.
8          Generate ballot mark $m$ for $c^*$.
9          If applicable, choose random ballot fraction $f \in F$.
10         Return receipt $r \subseteq \langle p, s, m, f \rangle$.
11 end
12 Adversary:
13     Run some guessing algorithm $c_g = A(r^*)$.
14     Return guessed candidate $c_g$.
15 end
16 Oracle:
17     if $c_g = c^*$ then
18         Return TRUE.
19     else if $c_g \neq c^*$ then
20         Return FALSE.
21 end
```

The attack game involves two participants: an adversary who is PPT bounded and a random voting oracle. When queried, the random voting oracle chooses a random candidate to vote for and then performs the voting function. The exact nature of the voting function changes from system to system. In Prêt à Voter, the order of the candidates, p, is randomly generated and the receipt contains the position of the mark: $r = \langle m \rangle$. For Punchscan, the order of the symbols for both sheets, p_t and p_b, is chosen randomly, either the top or bottom sheet, f, is randomly chosen to be kept, and the receipt contains the position of the mark, the permutation of the kept sheet, and which sheet was kept $r = \langle m, p_f, f \rangle$. For ThreeBallot, there is more than one way to mark the multiballot in order to vote for a given candidate. A marking style, s, is randomly chosen, and either the first, second, or third ballot is kept (f). A ThreeBallot receipt contains the position(s) marked on the kept ballot: $r = \langle m \rangle$.

Given a receipt from the oracle, the adversary's goal is to guess which candidate was voted for. The adversary may employ some algorithm to assist in this guess. The oracle will evaluate the guess and declare it correct or incorrect. For an election with N candidates, we define the informational advantage of having a ballot receipt as,

$$Adv = \left| \frac{1}{N} - \Pr[c^* \leftarrow \mathrm{A}(r^*) | r^* = \mathbf{V}(c^*) \subseteq \langle m, p_f, f \rangle] \right| . \tag{1}$$

If a ballot receipt contains no non-negligible information, the adversary's algorithm will be equivalent to random guessing—that is, a probability of correctness of $1/N$ and thus no more than a negligible ($\epsilon < 1/2^N$) advantage.

This attack game is a base case, as it only considers the amount of information leaked by the marks on the receipt. This minimal level of security should be considered necessary *but not sufficient* for a voting system. We provide the formal framework of an attack game to illuminate the analysis in this paper, but we also hope it will be utilized in future security analysis involving stronger attacks. For a voting system to be provably secure, other potential information sources including the receipt's serial number or cryptographic onion, the set of other published post-election receipts/ballots, and any other information revealed in the auditing process must also be shown to reveal only negligible information [7]. The random voting oracle is also idealized as the random operations will not be perfectly uniform selections in practice. Rather, they will be either psuedorandom selections based on a secret key or product of the voter's choice. If the underlying pseudorandomness or cryptographic primitives are insecure, the adversary may be able to mount an attack, and similarly if voters are psychologically predisposed to certain selections or primed to make them.[1]

3 Mark Information

3.1 Prêt à Voter and Punchscan

In this section, we consider the adversary's advantage in the attack game of Algorithm 1 with a Prêt à Voter and a Punchscan random voting oracle. Beginning with a Prêt à Voter ballot, recall that the candidate names are randomized. For N candidates, there are $N!$ equally probable permutations of the names. If a receipt is marked in a given position, the number of permutations that would result in this marked position counting as a vote for the ith candidate is $(N-1)!$. Therefore the probability that any possible mark corresponds to the ith candidate is,

$$\Pr[r = \mathbf{V}(c_i), \forall i] = \frac{(N-1)!}{N!} = \frac{1}{N}. \tag{2}$$

A Punchscan ballot has two random permutations of indirection pointers—one permutation for the top sheet, and one for the bottom. For N candidates, there are $(N!)^2$ equally probable permutations for the full ballot (i.e., both sheets). The receipt—one sheet without the other—contains $N!$ permutations itself. If it is marked in position i, there are $(N-1)$ permutations on the other sheet that would map the vote to a specific candidate. Taken together, the probability that any possible mark corresponds to the ith candidate is,

[1] This issue does not emerge in Prêt à Voter as voters are not required to make any random selections. A recent Punchscan case study [5] found that 85% of voters choose the bottom sheet instead of the top, however this particular information is not useful for attacking voter privacy. In ThreeBallot, voters are required to make random choices about how to mark the ballot along with which fraction to keep. Taken together, this could facilitate privacy attacks along with integrity attacks.

$$\Pr[r = \mathbf{V}(c_i), \forall i] = \frac{(N-1)!^2}{N!(N-1)!} = \frac{1}{N}. \tag{3}$$

The results of these equations are perhaps intuitive; the marks on both a Prêt à Voter and a Punchscan ballot receipt provide no information as it is equally probable that the mark corresponds to any of the candidates on the ballot. The security of the marks is premised on the use of secure underlying cryptographic primitives to generate the random permutations.

3.2 ThreeBallot

We now consider ThreeBallot by first recalling that a ThreeBallot election need not use three ballots. In a ThreeBallot election with B ballots (where every candidate receives $B-2$ votes, and the chosen candidate receives $B-1$) and N candidates, the number of ways of marking a multiballot for a particular candidate is:

$$M = \binom{B}{B-1}\binom{B}{B-2}^{(N-1)}. \tag{4}$$

M multiballots contain R potential receipts:

$$R = N \cdot B \cdot M. \tag{5}$$

Consider the base case of a 3 ballot multiballot with 1 contest of 2 candidates. There are 9 ways of marking the multiballot for each candidate, creating 54 equally probable receipts. Figure 1 shows the complete set. In the notation we are using, a receipt of $\{0,1\}$ indicates no vote recorded for candidate A and a vote recorded for candidate B. As denoted in Figure 1 as shaded receipts, there are 12 equally probable ways of receiving a $\{0,1\}$ receipt if candidate B was voted for, while there are only 3 equally probable ways if candidate A was voted for. Thus this particular receipt leaks information. Similarly $\{1,0\}$ is 12 to 3 in favor of candidate A. Both $\{0,0\}$ and $\{1,1\}$ do not leak information as they could have come from candidate A or B with equal (6 to 6) probability.

The adversary can exploit the information leaked by the ballot receipts with asymmetrical probabilities by using Algorithm 2. In the base case of three ballots and two candidates, an attacker using this algorithm gains an advantage of 16.67% over that of a random guess. Figure 2-a shows the expected advantage for three ballots and an increasing number of candidates. Figure 2-b shows the expected advantage for two candidates and an increasing number of ballots.

Algorithm 2: Adversary's Algorithm

```
1 if $r = \{0, 0, \ldots, 0\}$ then
2     Guess random candidate: $c_i \in C$.
3 else
4     Guess random candidate with marked vote: $c_i \in C_v | C_v \subset C$ s.t. $r_i = 1 \ \forall i$
```

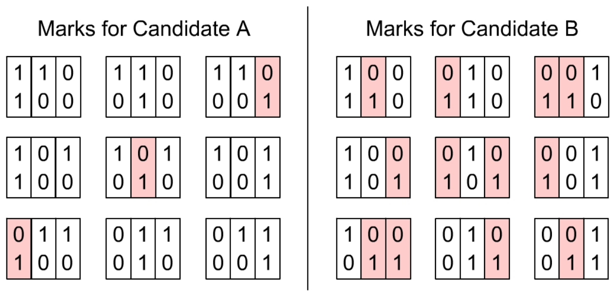

Fig. 1. The 18 possible ways to mark a ThreeBallot multiballot in a two-candidate election (9 for each candidate). The shaded ballots demonstrate that the probability of getting a $\{0, 1\}$ ballot receipt is not equal for both candidates.

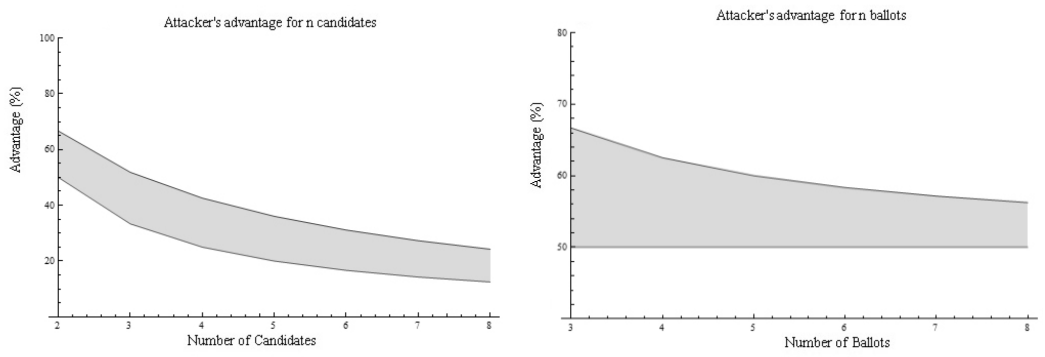

Fig. 2. Adversary's advantage over a random guess at determining the candidate voted for from a ThreeBallot receipt with increasing (a) number of candidates and (b) number of ballots in the multiballot.

ThreeBallot could be made secure against this attack by only permitting a suitable subset of the possible marking patterns, such that all receipts occur with equal probability for all candidates. Unfortunately by definition of how votes are marked, no suitable subset can exist. Every marking pattern must by necessity contain at least one receipt that is probable for the chosen candidate. Furthermore, every marking pattern that contains at least one receipt that is probable for another candidate must be complemented by at least two receipts that is probable for the chosen candidate. This should be apparent from Figure 1 for the base case; the only $\{0, 1\}$ receipts possible for candidate A occur in multiballots with two $\{1, 0\}$ receipts.

4 Preventing Ballot Tampering with Receipts

The second property of a ballot receipt is that it should not provide information useful to an adversary wanting to tamper with recorded ballots before they are tallied. In E2E systems, the receipts may be compared to the inputs of the tallying function to help ensure the integrity of the tallying process. In ThreeBallot, the tallying function is simply the sum of the votes on the set of all ballots minus the number of votes. Since a random third of the ballots on the bulletin board have copies circulating as receipts, an adversary who modifies one ballot faces a 1/3 chance of being caught if every voter checks her receipt.

It was orginally thought that a ThreeBallot receipt contains no useful information, and that an adversary will not gain any advantage by merely seeing the receipt [9]. Voters with no intention of checking their ballot themselves might be encouraged to give a copy to an organization who will check on their behalf. Unfortunately this is useful information for attacking the integrity of a ThreeBallot election. Each receipt has a serial number and if the adversary sees a receipt, she will not modify the corresponding ballot on the bulletin board when choosing a ballot to tamper with. If she knows a receipts, the probability of being detected when modifying one ballot (out of the set of B ballots) is $\left(\frac{\frac{B}{3}-a}{B-a}\right)$.

In the extreme case, the adversary has seen all the receipts ($a = B/3$) and her probability of being caught is zero. Essentially the adversary is partitioning the set of posted ballots into two subsets: one of known receipts and one of unknown status. She will always draw a ballot to tamper with from the second subset. Every receipt she sees moves a ballot from the second subset to the first, and thus every receipt leaks useful information. At a philosophical level, ThreeBallot's mechanism for receipt checking fails because cut-and-choose protocols require the choice to be unpredictable [3] and every ThreeBallot receipt reveals useful information about the choice. This problem does not arise in Prêt à Voter or Punchscan because all the inputs to the tallying function are receipts.

5 Cost-Benefit Analysis of Tampering

In the previous section, we examined the scenario where an adversary modifies the inputs to the tallying function. The probability of getting caught tampering with election results can be thought of as a cost to the adversary. We have shown that revealing Prêt à Voter and Punchscan receipts to an adversary does not decrease the cost to the adversary, while every ThreeBallot receipt does (to the extreme case where the adversary has seen all the receipts and the cost is zero). However our analysis is not complete unless if we also consider how to minimize the benefit of the attack to the adversary. It is acceptable that a voting system has a lower cost if (and only if) the associated payoff is lower as well. The best system design maximizes the cost and minimizes the benefit of tampering.

In Prêt à Voter and Punchscan, the best an adversary can hope to achieve is apply a random mapping between which candidate was voted for and which candidate gets the vote. If there are two candidates, Alice and Bob, the adversary can make a vote for Alice count for Bob only by also making a vote for Bob count for Alice. If the adversary is attempting to rig the election for Bob, she may inadvertently take votes from Bob and give them to Alice. Unless if she knows the vote associated with the receipt, the benefit of tampering with Prêt à Voter and Punchscan tally-inputs is uncertain.

In the case of ThreeBallot, an adversary can explicitly take a vote away from one candidate and give it to another candidate. In order to ensure the total number of votes does not exceed the number of cast ballots, the adversary may have to modify two ballots: take a vote from a candidate on one ballot and give it to another candidate on a second ballot. Either way, a tampering attack on ThreeBallot is more beneficial than on Prêt à Voter or Punchscan; and it is at a lower cost as well.

6 Concluding Remarks

We have formulated a basic attack game that demonstrates that ThreeBallot receipts can leak partial information about which candidate was voted for. For this reason, it cannot be asserted that ThreeBallot receipts contain no non-negligible information about how a voter voted. This can, however, be asserted of both Prêt à Voter and Punchscan, if we assume the underlying cryptography is secure.[2] In addition, we have demonstrated that merely showing a ThreeBallot receipt to an attacker compromises the integrity of the tallying by a non-negligible quantity. This property does not exist in a Prêt à Voter or Punchscan election. Finally, we show that if an attacker attempts to rig an election for her preferred candidate, tampering with a ThreeBallot election yields greater benefit than Prêt à Voter or Punchscan. For these reasons combined, the use of either Prêt à Voter or Punchscan in an E2E election appears to be a Pareto improvement[3] over ThreeBallot with respect to the security of ballot receipts.

References

1. Voting system performance guidelines. *2005 Voluntary Voting System Guidelines*, Volume 1, United States Election Assistance Commission, Version 1.0, 2005.
2. K. Fisher, R. Carback and A.T. Sherman. Punchscan: introduction and system definition of a high-integrity election system. Proceedings of *Workshop on Trustworthy Elections 2006*.
3. D. Chaum, C. Crepeau, and I. Damgard. Multiparty unconditionally secure protocols. Proceedings of the twentieth annual *ACM Symposium on Theory of Computing*, 1988.
4. D. Chaum, P.Y.A. Ryan, and S. Schneider. A practical, voter-verifiable election scheme. Technical Report CS-TR-880, University of Newcastle upon Tyne, 2004.
5. A. Essex, J. Clark, R. Carback, and S. Popoveniuc. Punchscan in practice: an E2E election case study. Proceedings of *Workshop on Trustworthy Elections 2007*.
6. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270299, 1984.
7. C. Karlof, N. Sastry, and D. Wagner. Cryptographic voting protocols: a systems perspective. Proceedings of the 14th *USENIX Security Symposium*, 2005.
8. S. Popoveniuc and B. Hosp. An introduction to Punchscan. Proceedings of *Workshop on Trustworthy Elections 2006*.
9. R. Rivest. The ThreeBallot voting system. October 1, 2006. Online: http://theory.csail.mit.edu/∼rivest/Rivest-TheThreeBallotVotingSystem.pdf
10. R. Rivest and W.D. Smith. Three Voting Protocols: ThreeBallot, VAV, and Twin. Proceedings of *USENIX/ACCURATE Electronic Voting Technology (EVT) 2007*.
11. P.Y.A. Ryan and T. Peacock. Prêt à Voter: a systems perspective. Technical Report CS-TR-929, University of Newcastle upon Tyne, 2005.
12. C.E.M. Strauss. A critical review of the triple ballot voting system. Part 2: cracking the triple ballot encryption. Draft Version 1.5, October 8 2006. Online: http://cems.browndogs.org/pub/voting/tripletouble.pdf

[2] Even if it were not, it is unclear whether the system would reach unicity. So long as the keyspace is larger than the permutation space, there will be spurious keys.

[3] Improving one or more aspects without worsening any others.