Ballot secrecy and ballot independence coincide

Ben Smyth[1] and David Bernhard[2]

[1] INRIA Paris-Rocquencourt, France
[2] University of Bristol, England

Abstract. We study ballot independence for election schemes:
- We formally define ballot independence as a cryptographic game and prove that ballot secrecy implies ballot independence.
- We introduce a notion of controlled malleability and show that it is sufficient for ballot independence. We also show that non-malleable ballots are sufficient, but not necessary, for ballot independence.
- We prove that ballot independence is sufficient for ballot secrecy under practical assumptions.

Our results show that ballot independence is necessary in election schemes satisfying ballot secrecy. Furthermore, our sufficient conditions enable simpler proofs of ballot secrecy.

1 Introduction

Voters should be able to express their free will in elections without fear of retribution; this property is known as privacy. Cryptographic formulations of privacy depend on the specific setting and *ballot secrecy*[1] [DKR06, BHM08, CS13] has emerged as a *de facto* standard privacy requirement of election schemes.

- *Ballot secrecy.* A voter's vote is not revealed to anyone.

Ballot secrecy provides privacy in an intimidation-free environment and stronger properties such as *receipt-freeness* and *coercion resistance* [DKR09] provide privacy in environments where intimidation may occur. Bernhard *et al.* [BCP+11, BPW12b,BPW12a] propose a cryptographic formalisation of ballot secrecy. However, we show that their definition allows election schemes that reveal voters' votes to be proven secure and we strengthen the definition to prevent this issue.

Ballot independence [Gen95, CS13] is seemingly related to ballot secrecy.

- *Ballot independence.* Observing another voter's interaction with the election system does not allow a voter to cast a meaningfully related vote.

Indeed, Cortier and Smyth [CS13, SC11, CS11] attribute a class of ballot secrecy attacks to the absence of ballot independence. However, ballot independence has not been formally defined and its relationship with ballot secrecy is unknown.

[1] The terms *privacy* and *ballot secrecy* occasionally appear as synonyms in the literature and we favour ballot secrecy because it avoids confusion with other privacy notions, such as receipt-freeness and coercion resistance, for example.

We provide a definition of ballot independence and show that ballot secrecy and ballot independence coincide in practical settings.

In traditional paper-based elections, physical mechanisms can be used to achieve privacy, for instance, ballots are completed in isolation inside polling booths, placed into locked ballot boxes, and mixed with other ballots before tallying. (See Schneier [Sch13] for a detailed, informal security analysis of Papal elections.) By comparison, the provision of ballot secrecy is more difficult in end-to-end verifiable election schemes, since ballots are posted on publicly readable bulletin boards. Nonetheless, ballot secrecy is a *de facto* standard property of election schemes and, hence, must be satisfied. The aforementioned physical mechanisms also provide an assurance of ballot independence in paper-based elections, however, the motivation for election schemes satisfying ballot independence is unclear, indeed, Bulens, Giry & Pereira [BGP11, §3.2] question whether ballot independence is a desirable property of election schemes and highlight the investigation of voting schemes which allow the submission of related votes whilst preserving ballot secrecy as an interesting research direction. Moreover, in the context of the Helios [Adi08, AMPQ09] election scheme, Desmedt & Chaidos [DC12] present a protocol which allows Bob to cast the same vote as Alice, with Alice's cooperation, and claim that Bob cannot learn Alice's vote. In this paper, we study the relationship between ballot secrecy and ballot independence and show that the two properties coincide in practical settings.

Contribution and Outline. In Section 3 we show that the definition of ballot secrecy by Bernhard *et al.* allows election schemes that reveal voters' votes to be proven secure and we present a stronger definition of ballot secrecy to prevent this issue. In Section 4 we propose a definition of ballot independence and give sufficient conditions to achieve this notion, including a definition of controlled-malleable encryption. In Section 5 we prove that ballot secrecy implies ballot independence, thereby providing an argument to end the ballot independence debate: ballot independence is a necessary property of election schemes (assuming ballot secrecy is required). In addition, we critique (Section 5.1) the results by Desmedt & Chaidos and argue that their security results do not support their claims. In Section 6 we present a practical class of election schemes (which includes Helios) for which ballot secrecy and ballot independence coincide.

Related work. The concept of independence was introduced by Chor *et al.* [CGMA85] and studied in the context of election schemes by Gennaro [Gen95]. Cortier and Smyth [CS11, SC11, CS13] have discovered attacks on ballot secrecy in several election schemes and considered the relationship to independence [CS13, Section 7]; their evidence suggests ballot secrecy implies ballot independence in homomorphic voting systems such as Helios. However, Cortier & Smyth did not make any formal claims, because ballot independence had not been formally defined. By comparison, in this paper, we present a formal definition of ballot independence and prove that ballot secrecy implies ballot independence. Bernhard, Pereira & Warinschi [BPW12b] show that a non-malleable

encryption scheme is sufficient to build an election scheme satisfying ballot secrecy and our work generalises their result.

2 Preliminaries

We adopt standard notation for the application of probabilistic algorithms: if A is a probabilistic algorithm, then $A(x_1, \ldots, x_n; r)$ is the result of running A on input $x_1, \ldots, x_n$ and coins r. We let $y \leftarrow A(x_1, \ldots, x_n)$ denote picking r at random and assigning the output of $A(x_1, \ldots, x_n; r)$ to the variable y. If S is a finite set, then $x \leftarrow S$ assigns a uniformly chosen element of S to x. If α is neither a probabilistic algorithm nor a set, then $x \leftarrow \alpha$ assigns α to x. Vectors are denoted using boldface, for example, $\mathbf{x}$. We extend set membership notation to vectors: we write $x \in \mathbf{x}$ (respectively, $x \notin \mathbf{x}$) if x is an element (respectively, x is not an element) of the vector $\mathbf{x}$.

2.1 Non-malleable encryption

Let us recall the standard syntax for *asymmetric encryption schemes*.

Definition 1 (Asymmetric encryption scheme). *An* asymmetric encryption scheme *is a triple of efficient algorithms* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *such that:*

- *The* key generation algorithm Gen *takes a security parameter* 1^n *as input and outputs a key pair* (pk, sk)*, where* pk *is a public key and* sk *is a private key.*
- *The* encryption algorithm Enc *takes a public key* pk *and message* m *as input, and outputs a ciphertext* c*.*
- *The* decryption algorithm Dec *takes a private key* sk *and ciphertext* c *as input, and outputs a message* m *or the special symbol* $\perp$ *denoting failure.*

Moreover, the scheme must be correct: for all $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$*, we have for all messages* m *and ciphertexts* $c \leftarrow \mathsf{Enc}_{pk}(m)$*, that* $\mathsf{Dec}_{sk}(c) = m$ *with overwhelming probability.*

Non-malleability [DDN91, BDPR98, DDN00] is a standard computational security model used to evaluate the suitability of encryption schemes. Intuitively, if an encryption scheme satisfies non-malleability, then an adversary is unable to construct a ciphertext *"meaningfully related"* to a challenge ciphertext, thereby capturing the idea that ciphertexts are tamper-proof. This notion can be captured by a pair of cryptographic games – namely, $\mathsf{Succ}^{\mathrm{CPA}}_{\mathcal{A},\Pi}$ and $\mathsf{Succ}^{\mathrm{CPA}}_{\mathcal{A},\Pi,\$}$ – between an adversary and a challenger. The first three steps of both games are identical. First, the challenger constructs a key pair (pk, sk). Secondly, the adversary $\mathcal{A}$ executes the algorithm A_1 on the public key pk and outputs the pair (M, s), where M is a sampling algorithm for some message space and s is some state information. Thirdly, the challenger randomly selects a plaintext x from the message space; at this point, the challenger in $\mathsf{Succ}^{\mathrm{CPA}}_{\mathcal{A},\Pi,\$}$ performs an additional step, namely, the challenger samples a second plaintext x'. Fourthly, the

challenger constructs a ciphertext $y \leftarrow \mathsf{Enc}_{pk}(x)$. Fifthly, the adversary executes algorithm A_2 which outputs a relation R and a vector of ciphertexts $\mathbf{y}$. Finally, the challenger decrypts $\mathbf{y}$ and outputs the corresponding plaintexts $\mathbf{x}$. The encryption scheme satisfies non-malleability if the adversary's relation R cannot meaningfully relate x and $\mathbf{x}$. Formally, Definition 2 recalls the non-malleability game proposed by Bellare *et al.* [BDPR98].

Definition 2 (Non-malleable encryption). *Let* $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *be an asymmetric encryption scheme,* $\mathcal{A} = (A_1, A_2)$ *be an adversary, and*

$$\mathsf{NM\text{-}CPA}_{\mathcal{A},\Pi}(n) := |\mathsf{Succ}^{CPA}_{\mathcal{A},\Pi}(n) - \mathsf{Succ}^{CPA}_{\mathcal{A},\Pi,\$}(n)|$$

where $\mathsf{Succ}^{CPA}_{\mathcal{A},\Pi}(n)$ *and* $\mathsf{Succ}^{CPA}_{\mathcal{A},\Pi,\$}(n)$ *are defined below, and* n *is a security parameter.*

$$\begin{aligned}\mathsf{Succ}^{CPA}_{\mathcal{A},\Pi}(n) = Pr[(pk, sk) \leftarrow \mathsf{Gen}(1^n);\ (M, s) \leftarrow A_1(pk);\\ x \leftarrow M;\ y \leftarrow \mathsf{Enc}_{pk}(x);\ (R, \mathbf{y}) \leftarrow A_2(M, s, y);\\ \mathbf{x} \leftarrow \mathsf{Dec}_{sk}(\mathbf{y}) : y \notin \mathbf{y} \wedge \perp \notin \mathbf{x} \wedge R(x, \mathbf{x})]\end{aligned}$$

$$\begin{aligned}\mathsf{Succ}^{CPA}_{\mathcal{A},\Pi,\$}(n) = Pr[(pk, sk) \leftarrow \mathsf{Gen}(1^n);\ (M, s) \leftarrow A_1(pk);\\ x, x' \leftarrow M;\ y \leftarrow \mathsf{Enc}_{pk}(x);\ (R, \mathbf{y}) \leftarrow A_2(M, s, y);\\ \mathbf{x} \leftarrow \mathsf{Dec}_{sk}(\mathbf{y}) : y \notin \mathbf{y} \wedge \perp \notin \mathbf{x} \wedge R(x', \mathbf{x})]\end{aligned}$$

In the above games we insist that the message space is valid (that is, $|x| = |x'|$ *for any* $x, x' \leftarrow M$ *given non-zero probability in the message space) and samplable in polynomial time, and the relation* R *is computable in polynomial time. We say* Π *satisfies* NM-CPA *if for all probabilistic polynomial-time adversaries* $\mathcal{A}$ *and security parameters* n*, there exists a negligible function* negl *such that* $\mathsf{NM\text{-}CPA}_{\mathcal{A},\Pi}(n) \leq \mathsf{negl}(n)$*.*

3 Election schemes and ballot secrecy

Based upon Bernhard *et al.* [BCP+11, BPW12b, BPW12a], we define a syntax for *election schemes* as follows.

Definition 3 (Election scheme). *An* election scheme *is a tuple of efficient algorithms* $(\mathsf{Setup}, \mathsf{Vote}, \mathsf{BB}, \mathsf{Tally})$ *such that:*

- *The* setup algorithm Setup *takes a security parameter* 1^n *as input and outputs a bulletin board* $\mathfrak{bb}$*, vote space* $\mathfrak{m}$*, public key* pk*, and private key* sk*, where* $\mathfrak{bb}$ *is a multiset and* $\mathfrak{m}$ *is a set.*
- *The* vote algorithm Vote *takes a public key* pk *and vote* $v \in \mathfrak{m}$ *as input, and outputs a ballot* b*.*

- *The* bulletin board algorithm BB *takes a bulletin board* $\mathfrak{bb}$ *and ballot* b *as input, where* $\mathfrak{bb}$ *is a multiset. It outputs* $\mathfrak{bb} \cup \{b\}$ *if successful (i.e.,* b *is added to* $\mathfrak{bb}$*) or* $\mathfrak{bb}$ *to denote failure (i.e.,* b *is not added).*
- *The* tally algorithm Tally *takes a private key* sk *and bulletin board* $\mathfrak{bb}$ *as input, where* $\mathfrak{bb}$ *is a multiset. It outputs a multiset* $\mathfrak{v}$ *representing the election result if successful or the empty set* $\emptyset$ *to denote failure, and auxiliary data* aux*.*

Moreover, the scheme must satisfy the following correctness property: for all parameters $(\mathfrak{bb}_0, \mathfrak{m}, pk, sk) \leftarrow \mathsf{Setup}(1^n)$*, votes* $v \in \mathfrak{m}$*, multisets* $\mathfrak{bb}$*, ballots* $b \leftarrow \mathsf{Vote}_{pk}(v)$*, bulletin boards* $\mathfrak{bb}' \leftarrow \mathsf{BB}(\mathfrak{bb}, b)$ *and tallying data* $(\mathfrak{v}, aux) \leftarrow \mathsf{Tally}_{sk}(\mathfrak{bb})$ *and* $(\mathfrak{v}', aux') \leftarrow \mathsf{Tally}_{sk}(\mathfrak{bb}')$*, we have with overwhelming probability that* $\mathfrak{bb}' = \mathfrak{bb} \cup \{b\}$ *and if* $\mathfrak{v} \neq \emptyset$*, then* $\mathfrak{v}' = \mathfrak{v} \cup \{v\}$ *and* $|\mathfrak{v}| = |\mathfrak{bb}|$*, otherwise,* $\mathfrak{v}' = \emptyset$*.*

In comparison with earlier presentations by Bernhard *et al.*, Definition 3 is stricter, since we explicitly define the bulletin board and election result as multisets. Moreover, the correctness condition, asserting that the election result corresponds to the multiset of votes cast, is new. Although the correctness condition restricts the applicability of our definition – for example, we cannot model schemes with weighted votes nor schemes which only reveal the winning candidate (as opposed to the number of votes for each candidate) – we believe it is useful for simplicity. In addition, there are some minor differences in error handling and we merge some functionality into a single function[2].

We demonstrate the applicability of our definition by recalling the construction (Definition 4) for election schemes proposed by Bernhard *et al.* [BCP+11, BPW12b]. We stress that more sophisticated schemes can also be captured – for example, Bernhard *et al.* [BCP+11, BPW12b, BPW12a] model Helios – but the following scheme is sufficient for our purposes.

Definition 4 ($\mathsf{Enc2Vote}$)**.** *Given an asymmetric encryption scheme* $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$*, we define the election scheme* $\mathsf{Enc2Vote}(\Pi)$ *as follows.*

- Setup *takes a security parameter* 1^n *as input and outputs* $(\emptyset, \mathfrak{m}, pk, sk)$*, where* $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$ *and* $\mathfrak{m}$ *is the encryption scheme's message space.*
- Vote *takes a public key* pk *and vote* $v \in \mathfrak{m}$ *as input, and outputs* $\mathsf{Enc}_{pk}(v)$*.*
- BB *takes a bulletin board* $\mathfrak{bb}$ *and ballot* b *as input, where* $\mathfrak{bb}$ *is a multiset. If* $b \in \mathfrak{bb}$*, then the algorithm outputs* $\mathfrak{bb}$ *(denoting failure), otherwise, the algorithm outputs* $\mathfrak{bb} \cup \{b\}$*.*
- Tally *takes as input a private key* sk *and a bulletin board* $\mathfrak{bb}$*, where* $\mathfrak{bb}$ *is a multiset. It outputs the multiset* $\{\mathsf{Dec}_{sk}(b) \mid b \in \mathfrak{bb}\}$ *and auxiliary data* $\perp$*.*

[2] In essence, the tally algorithm defined by Bernhard *et al.* outputs a tally τ and an additional algorithm is used to compute the election result $\mathfrak{v}$ from τ. We combine the functionality of these two algorithms into a single function but distinguish between the result $\mathfrak{v}$ and auxiliary data aux, which is typically used to store signatures of knowledge proving that the election result has been correctly computed from the bulletin board.

Intuitively, given an asymmetric encryption scheme Π satisfying NM-CPA, the construction Enc2Vote(Π) derives ballot secrecy from Π until tallying and the Tally algorithm maintains ballot secrecy by returning the number of votes for each candidate as an unordered multiset of votes[3].

Ballot Secrecy. Ballot secrecy is a *de facto* standard property of election schemes and, based upon Bernhard *et al.* [BCP+11, BPW12b, BPW12a], we formalise a cryptographic game for ballot secrecy (Definition 5). We will describe the differences between our formalisation and earlier presentations after our definition. Informally, our game proceeds as follows. First, the challenger executes the setup algorithm to construct a bulletin board $\mathfrak{bb}_0$, a vote space $\mathfrak{m}$, a public key pk, and a private key sk; the challenger also initialises a bulletin board $\mathfrak{bb}_1$ as a copy of $\mathfrak{bb}_0$ and selects a random bit β. Secondly, the adversary executes the algorithm A_1. The algorithm A_1 has access to an oracle $\mathcal{O}$ as follows: $\mathcal{O}(v_0, v_1)$ allows the adversary to honestly cast a vote $v_0 \in \mathfrak{m}$ on bulletin board $\mathfrak{bb}_0$ and honestly cast a vote $v_1 \in \mathfrak{m}$ on bulletin board $\mathfrak{bb}_1$, where the votes are cast using ballots constructed by the Vote algorithm; $\mathcal{O}(b)$ allows the adversary to cast a ballot b, where b is constructed by the adversary and might be rejected by the bulletin board; and $\mathcal{O}()$ returns the bulletin board $\mathfrak{bb}_\beta$. Thirdly, the challenger computes the election result $\mathfrak{v}$ as follows: if the honestly cast votes on the bulletin board $\mathfrak{bb}_0$ correspond to the honestly cast votes on the bulletin board $\mathfrak{bb}_1$, then the challenger reveals the election result for $\mathfrak{bb}_\beta$, otherwise, the challenger reveals the election result for $\mathfrak{bb}_0$, thereby preventing the adversary from trivially revealing β when the honestly cast votes differ. (The distinction between $\mathfrak{bb}_0$ and $\mathfrak{bb}_1$ is trivial when the honestly cast votes differ, because the adversary can test for the presence of honestly cast votes in the election result.) Formally, we introduce the multisets L_0 and L_1 to record the honestly cast votes on bulletin boards $\mathfrak{bb}_0$ and $\mathfrak{bb}_1$, and model the correspondence between bulletin boards as an equality test on L_0 and L_1, that is, we compute $(\mathfrak{v}, aux) \leftarrow \mathsf{Tally}_{sk}(\mathfrak{bb}_\alpha)$ such that $\alpha = \beta$, if $L_0 = L_1$, and $\alpha = 0$, otherwise. Finally, the adversary executes the algorithm A_2 on the election result $\mathfrak{v}$ and any state information s provided by A_1. The election scheme satisfies ballot secrecy if the adversary has less than a negligible advantage over guessing the bulletin board she interacted with.

Definition 5 (IND-SEC: Ballot secrecy). *Let $\Gamma = (\mathsf{Setup}, \mathsf{Vote}, \mathsf{BB}, \mathsf{Tally})$ be an election scheme, $\mathcal{A} = (A_1, A_2)$ be an adversary, and $\mathit{IND\text{-}SEC}_{\mathcal{A},\Gamma}(n)$ be the quantity defined below, where n is the security parameter.*

$$2 \cdot Pr[L_0 \leftarrow \emptyset; L_1 \leftarrow \emptyset; (\mathfrak{bb}_0, \mathfrak{m}, pk, sk) \leftarrow \mathsf{Setup}(1^n);\ \mathfrak{bb}_1 \leftarrow \mathfrak{bb}_0;\ \beta \leftarrow \{0,1\};$$
$$s \leftarrow A_1^{\mathcal{O}}(\mathfrak{m}, pk);\ (\mathfrak{v}, aux) \leftarrow \mathsf{Tally}_{sk}(\mathfrak{bb}_\alpha) : A_2(\mathfrak{bb}_\beta, \mathfrak{v}, aux, s) = \beta] - 1$$

[3] Definition 4 rectifies a mistake in the presentation by Bernhard, Pereira & Warinschi [BPW12b] which outputs a vector of votes (rather than a multiset) ordered by the time at which each vote was cast and therefore does not provide ballot secrecy, since there is a mapping between the order in which votes were cast and the votes. (Bernhard *et al.* [BCP+11] avoid this problem in a similar fashion.)

In the above game, L_0 and L_1 are multisets, the oracle $\mathcal{O}$ is defined below, and the bit α is defined as follows: if $L_0 = L_1$, then $\alpha = \beta$, otherwise, $\alpha = 0$.

- $\mathcal{O}(v_0, v_1)$ *executes* $L_0 \leftarrow L_0 \cup \{v_0\}; L_1 \leftarrow L_1 \cup \{v_1\}; b_0 \leftarrow \mathsf{Vote}_{pk}(v_0); b_1 \leftarrow \mathsf{Vote}_{pk}(v_1); \mathfrak{bb}_0 \leftarrow \mathsf{BB}(\mathfrak{bb}_0, b_0); \mathfrak{bb}_1 \leftarrow \mathsf{BB}(\mathfrak{bb}_1, b_1)$, *if* $v_0, v_1 \in \mathfrak{m}$.
- $\mathcal{O}(b)$ *assigns* $\mathfrak{bb}'_\beta \leftarrow \mathfrak{bb}_\beta$, *executes* $\mathfrak{bb}_\beta \leftarrow \mathsf{BB}(\mathfrak{bb}_\beta, b)$ *and if* $\mathfrak{bb}_\beta \neq \mathfrak{bb}'_\beta$, *then executes* $\mathfrak{bb}_{1-\beta} \leftarrow \mathsf{BB}(\mathfrak{bb}_{1-\beta}, b)$.
- $\mathcal{O}()$ *outputs* $\mathfrak{bb}_\beta$.

We say Γ satisfies ballot secrecy *if for all probabilistic polynomial-time adversaries $\mathcal{A}$ and security parameters n, there exists a negligible function* negl *such that* $\mathsf{IND\text{-}SEC}_{\mathcal{A},\Gamma}(n) \leq \mathsf{negl}(n)$.

Our game captures a setting where an adversary can cast ballots on behalf of a subset of voters, whom we call dishonest voters, and controls the distribution of votes cast by the remaining voters, whom we call honest voters, but honest voters always cast ballots constructed by the Vote algorithm. Furthermore, at the end of the election, the adversary obtains the election result. Intuitively, if the adversary loses the game, then the adversary is unable to distinguish between the bulletin boards $\mathfrak{bb}_0$ and $\mathfrak{bb}_1$, hence, the adversary cannot distinguish between an honest ballot $b_0 \in \mathfrak{bb}_0$ and an honest ballot $b_1 \in \mathfrak{bb}_1$, therefore, voters' votes cannot be revealed. On the other hand, if the adversary wins the game, then there exists a strategy to distinguish honestly cast ballots. For example, suppose an adversary in control of one dishonest voter can violate ballot secrecy in a referendum with two honest voters, when all voters participate, each voter casts a valid vote, and no auxiliary data is produced (as per the $\mathsf{Enc2Vote}$ construction, we can model the absence of auxiliary data using a constant symbol such as $\perp$). In this setting, we require a vote space $\{v_0, v_1\}$ and the adversary must make three oracle calls, namely, $\mathcal{O}(v_0, v_1)$, $\mathcal{O}(v_1, v_0)$, and $\mathcal{O}(b)$. It follows that the election result will be $\{v_0, v_1, v\}$, where v is the adversary's vote. Moreover, the adversary must have a strategy to generate b such that the adversary's vote v is related to either v_0 or v_1, otherwise, the election results from both bulletin boards will be equal and the adversary cannot win the game. We stress that a unanimous election result – for instance, the election result generated by tallying the bulletin board $\mathfrak{bb}_\beta$ produced by the oracle calls $\mathcal{O}(v_0, v_1)$, $\mathcal{O}(v_0, v_1)$, and $\mathcal{O}(b)$, where b contains the vote v_β – will always reveal all voters' votes and we tolerate this factor in our game by challenging the adversary to guess the bit β, rather than the distribution of votes.

Comparing IND-SEC and earlier definitions. In comparison with earlier definitions by Bernhard *et al.* [BCP+11, BPW12b, BPW12a], Definition 5 permits $\alpha \in \{0, 1\}$, whereas, earlier presentations implicitly[4] insist $\alpha = 0$. It follows that Definition 5 allows the adversary to access auxiliary data generated by tallying $\mathfrak{bb}_\beta$, whereas, earlier definitions only allow the adversary to access the auxiliary

[4] Earlier presentations do not explicitly define a bit α, however, they always tally $\mathfrak{bb}_0$ and this implicitly corresponds to $\alpha = 0$ in Definition 5.

data generated by tallying $\mathfrak{bb}_0$. Accordingly, earlier definitions implicitly assume that auxiliary data cannot be used to violate ballot secrecy, indeed, this corresponds to the description by Bernhard *et al.* [BCP+11, §2.2]: *"[ballot secrecy] is satisfied if an adversary [...] cannot learn anything about the votes of [...] honest voters beyond what can be inferred from the election result."* Unfortunately, however, it is possible that the auxiliary data can reveal voters' votes. For example, a variant of Enc2Vote (Definition 4) could define auxiliary data that maps ballots to decrypted ballots, thereby violating ballot secrecy; indeed, as highlighted in Footnote 3, Bernhard, Pereira & Warinschi [BPW12b] provided such a mapping in their variant of Enc2Vote. As discussed, we permit $\alpha \in \{0, 1\}$, rather than $\alpha = 0$, thereby strengthening Definition 5 in comparison with earlier definitions and, thus, overcoming the limitations of previous works.

4 Ballot independence

Intuitively, if an election scheme satisfies ballot independence, then an adversary is unable to construct a ballot that will be accepted by the election's bulletin board *and* be meaningfully related to a non-adversarial ballot from the bulletin board [CS13, Section 7.2], thereby capturing the notion that accepted ballots are tamper-proof. Building upon inspiration from non-malleable encryption, we formalise ballot independence as a non-malleability game.

4.1 Non-malleability game

The concept of non-malleability and first formalisation is due to Dolev, Dwork & Naor [DDN91, DDN00]. Bellare *et al.* [BDPR98] build upon these results to introduce NM-CPA (Definition 2) and based upon NM-CPA, we formalise ballot independence (Definition 6) as a pair of cryptographic games: $\mathsf{Succ}^{\mathrm{BB}}_{\mathcal{A},\Pi}$ and $\mathsf{Succ}^{\mathrm{BB}}_{\mathcal{A},\Pi,\$}$. The first three steps of both games are identical. First, the challenger sets up the keys, vote space, and bulletin board. Secondly, the adversary gets the vote space $\mathfrak{m}$, the public key pk and the board $\mathfrak{bb}$ as input and must return a distribution M on the vote space. The adversary may also read the board and submit ballots of his own. Thirdly, the challenger samples a vote v from M. At this point the two games diverge: in $\mathsf{Succ}^{\mathrm{BB}}_{\mathcal{A},\Pi}$, the challenger constructs a ballot $\mathsf{Vote}_{pk}(v)$ and adds it to the bulletin board; whereas, in $\mathsf{Succ}^{\mathrm{BB}}_{\mathcal{A},\Pi,\$}$, the challenger samples a second vote v' from M, constructs a ballot $\mathsf{Vote}_{pk}(v')$ and adds it to the bulletin board. Fourthly, the adversary must compute a relation R which is intended to distinguish the election results produced by the two games. Finally, the challenger tallies the election and evaluates the relation R on the vote v and, after removing the challenge vote, the election result. The adversary's advantage is the difference between the probabilities that his relation is satisfied in each game.

Definition 6 (NM-BB: Ballot independence). *Let* $\Gamma = (\mathsf{Setup}, \mathsf{Vote}, \mathsf{BB}, \mathsf{Tally})$ *be an election scheme,* $\mathcal{A} = (A_1, A_2)$ *be an adversary, and*

$$\mathit{NM\text{-}BB}_{\mathcal{A},\Gamma}(n) := |\mathit{Succ}^{BB}_{\mathcal{A},\Pi}(n) - \mathit{Succ}^{BB}_{\mathcal{A},\Pi,\$}(n)|$$

where $\mathsf{Succ}^{BB}_{\mathcal{A},\Pi}(n)$ *and* $\mathsf{Succ}^{BB}_{\mathcal{A},\Pi,\$}(n)$ *are defined below, and* n *is the security parameter.*

$$\mathsf{Succ}^{BB}_{\mathcal{A},\Pi}(n) = Pr[(\mathfrak{bb},\mathfrak{m},pk,sk) \leftarrow \mathsf{Setup}(1^n);\ (M,s) \leftarrow A_1^{\mathcal{O}}(\mathfrak{m},pk);\ v \leftarrow M;\ b \leftarrow \mathsf{Vote}_{pk}(v);\ \mathfrak{bb} \leftarrow \mathsf{BB}(\mathfrak{bb},b);\ R \leftarrow A_2^{\mathcal{O}}(s);\ (\mathfrak{v},aux) \leftarrow \mathsf{Tally}_{sk}(\mathfrak{bb}) : R(v,\mathfrak{v}\backslash\{v\})]$$

$$\mathsf{Succ}^{BB}_{\mathcal{A},\Pi,\$}(n) = Pr[(\mathfrak{bb},\mathfrak{m},pk,sk) \leftarrow \mathsf{Setup}(1^n);\ (M,s) \leftarrow A_1^{\mathcal{O}}(\mathfrak{m},pk);\ v,v' \leftarrow M;\ b \leftarrow \mathsf{Vote}_{pk}(v');\ \mathfrak{bb} \leftarrow \mathsf{BB}(\mathfrak{bb},b);\ R \leftarrow A_2^{\mathcal{O}}(s);\ (\mathfrak{v},aux) \leftarrow \mathsf{Tally}_{sk}(\mathfrak{bb}) : R(v,\mathfrak{v}\backslash\{v'\})]$$

In the above games we let $\mathcal{O}$ *be defined as follows:* $\mathcal{O}(b)$ *executes* $\mathfrak{bb} \leftarrow \mathsf{BB}(\mathfrak{bb},b)$ *and* $\mathcal{O}()$ *outputs* $\mathfrak{bb}$*. Moreover, we insist the vote space sampling algorithm* M *and the relation* R *are computable in polynomial time, and for all* $v \leftarrow M$ *we have* $v \in \mathfrak{m}$*. We say* Γ *satisfies* NM-BB *(or* ballot independence*) if for all probabilistic polynomial-time adversaries* $\mathcal{A}$ *and security parameters* n*, there exists a negligible function* negl *such that* $\mathsf{NM\text{-}BB}_{\mathcal{A},\Gamma}(n) \leq \mathsf{negl}(n)$*.*

Intuitively, if an adversary wins the game, then the adversary is able to construct a relation R which holds for a challenge ballot $b \leftarrow \mathsf{Vote}_{pk}(v)$ but fails for $b \leftarrow \mathsf{Vote}_{pk}(v')$. However, we must avoid crediting the adversary for trivial and unavoidable relations which hold iff the challenge vote appears in the election result, hence, we remove the challenge vote from the election result. By contrast, if the adversary can derive a ballot containing the challenge vote and the bulletin board accepts such a ballot, then the adversary can win the game. For example, suppose an election scheme allows the bulletin board to accept duplicate ballots and witness that an adversary can win the game as follows, namely, the adversary selects M as a uniform distribution on $\mathfrak{m}$, calls $\mathcal{O}(b)$ with the challenge ballot b, and defines a relation $R(v,\mathfrak{v})$ that holds iff $v \in \mathfrak{v}$. In this setting, $R(v,\{v\})$ always holds at the end of $\mathsf{Succ}^{\mathrm{BB}}_{\mathcal{A},\Pi}$, whereas, $R(v,\{v'\})$ holds with probability $1/\mathfrak{m}$ at the end of $\mathsf{Succ}^{\mathrm{BB}}_{\mathcal{A},\Pi,\$}$, since v' is sampled independently from v. Finally, if an adversary loses the game, then the adversary is unable to construct a suitable relation, hence, there is no ballot which the bulletin board will accept such that the ballot is related to $\mathsf{Vote}_{pk}(v)$ but not $\mathsf{Vote}_{pk}(v')$, therefore, the adversary cannot cast a ballot which is meaningfully related to an honest voter's ballot.

Comparing NM-BB *and* NM-CPA*.* The main distinction between the notion of non-malleability (Definition 2) and our definition of ballot independence is: NM-CPA universally quantifies over ciphertexts, whereas, NM-BB quantifies over ballots accepted by the bulletin board. It follows that non-malleability for encryption is intuitively stronger than ballot independence, since non-malleability for encryption insists that the adversary cannot construct ciphertexts meaningfully related to the challenge ciphertext, whereas, ballot independence tolerates meaningfully related ballots, assuming that they are rejected by the bulletin

board algorithm BB. For example, suppose an adversary $\mathcal{A}$ includes the challenge ciphertext in the vector $\mathbf{y}$ and observe that this adversary cannot win $\mathsf{NM\text{-}CPA}_{\mathcal{A},\Pi}(n)$, due to the constraint $y \notin \mathbf{y}$; by comparison, suppose an adversary $\mathcal{B}$ copies the challenge ballot b and observe that this adversary can win $\mathsf{NM\text{-}BB}_{\mathcal{B},\Gamma}(n)$. Nonetheless, for ballot independence, the bulletin board must not contain meaningfully related ballots and, hence, checking for meaningfully related ballots is a prerequisite of the bulletin board algorithm BB.

Non-malleable ballots are sufficient. Non-malleability for encryption prevents the adversary from constructing a ciphertext meaningfully related to the challenge ciphertext and, hence, it follows that non-malleable ballots are sufficient for ballot independence. Indeed, we can derive non-malleable ballots in our Enc2Vote construction using encryption schemes satisfying NM-CPA.

Proposition 7. *Given an encryption scheme Π satisfying NM-CPA, the election scheme* $\mathsf{Enc2Vote}(\Pi)$ *satisfies ballot independence.*

In Proposition 7, it is sufficient for the bulletin board algorithm, defined by $\mathsf{Enc2Vote}(\Pi)$, to reject ballots that already appear on the bulletin board since non-malleability prevents the adversary from creating ballots meaningfully related to honest voters' votes (except for exact copies). The proof is essentially the same as that of [BPW12b, Theorem 4.2].

More generally, we could adapt the non-malleability game for encryption (Definition 2) to a non-malleability game for ballots. In this setting, given an election scheme satisfying our non-malleability game for ballots and such that the bulletin board algorithm rejects duplicates, we believe that the election scheme satisfies ballot independence. Formalising this result is a possible direction for future research.

4.2 Indistinguishability game

Our non-malleability game (NM-BB) captures an intuitive notion of ballot independence, however, the definition is relatively complex and security proofs in this setting are relatively difficult. Bellare & Sahai [BS99] observed similar complexities with definitions of non-malleability for encryption and show that NM-CPA is equivalent to a simpler, indistinguishability-based notion. In a similar direction, we introduce an indisinguishability game IND-BB for ballot independence and, based upon Bellare & Sahai's proof, show that our games NM-BB and IND-BB are equivalent.

We model ballot independence as an indistinguishability game between an adversary and a challenger (Definition 8). Informally, the game proceeds as follows. First, the challenger initialises the bulletin board $\mathfrak{bb}$, defines the vote space $\mathfrak{m}$, and constructs a key pair (pk, sk). Secondly, the adversary executes the algorithm A_1 on the public key pk and vote space $\mathfrak{m}$, and outputs the triple (v_0, v_1, s), where $v_0, v_1 \in \mathfrak{m}$ and s is some state information. Thirdly, the challenger randomly selects a bit β, computes a challenge ballot b, and updates the bulletin

board with b. Fourthly, the adversary executes the algorithm A_2 which outputs some state t. Next, the challenger computes the election result $\mathfrak{v}$. Finally, the adversary executes the algorithm A_3 on the input t and $\mathfrak{v}\backslash\{v_\beta\}$. The election scheme satisfies ballot independence if the adversary has less than a negligible advantage over guessing the bit β.

Definition 8 (IND-BB: Ballot independence). *Let $\Gamma = (\mathsf{Setup}, \mathsf{Vote}, \mathsf{BB}, \mathsf{Tally})$ be an election scheme, $\mathcal{A} = (A_1, A_2, A_3)$ be an adversary, n be the security parameter and $\mathit{IND\text{-}BB}_{\mathcal{A},\Gamma}(n)$ the cryptographic game defined below.*

$$2 \cdot Pr[(\mathfrak{bb}, \mathfrak{m}, pk, sk) \leftarrow \mathsf{Setup}(1^n);\ (v_0, v_1, s) \leftarrow A_1^{\mathcal{O}}(\mathfrak{m}, pk);\ \beta \leftarrow \{0,1\};$$
$$b \leftarrow \mathsf{Vote}_{pk}(v_\beta);\ \mathfrak{bb} \leftarrow \mathsf{BB}(\mathfrak{bb}, b);\ t \leftarrow A_2^{\mathcal{O}}(s);\ (\mathfrak{v}, aux) \leftarrow \mathsf{Tally}_{sk}(\mathfrak{bb}) :$$
$$A_3(t, \mathfrak{v}\backslash\{v_\beta\}) = \beta] - 1$$

In the above game we let $\mathcal{O}$ be defined as follows:

- $\mathcal{O}(b)$ *executes* $\mathfrak{bb} \leftarrow \mathsf{BB}(\mathfrak{bb}, b)$
- $\mathcal{O}()$ *outputs* $\mathfrak{bb}$

Moreover, we insist that $v_0, v_1 \in \mathfrak{m}$. We say Γ satisfies IND-BB (or ballot independence*) if for all probabilistic polynomial-time adversaries $\mathcal{A}$ and security parameters n, there exists a negligible function* negl *such that $\mathit{IND\text{-}BB}_{\mathcal{A},\Gamma}(n) \leq \mathsf{negl}(n)$.*

Intuitively, if an adversary wins the game, then the adversary is able to distinguish between challenge ballots $b \leftarrow \mathsf{Vote}_{pk}(v_0)$ and $b \leftarrow \mathsf{Vote}_{pk}(v_1)$. As per our NM-BB game, we avoid trivial and unavoidable distinctions by removing the challenge vote from the election result.

Our ballot independence games are based on standard security models for encryption: NM-BB is based on non-malleability whereas IND-BB game is based on indistinguishability. Bellare and Sahai [BS99] have shown that non-malleability is equivalent to a notion of indistinguishability for encryption and we adapt their proof to show that NM-BB and IND-BB are equivalent.

Theorem 9 (NM-BB = IND-BB). *Given an election scheme Γ, we have Γ satisfies NM-BB if and only if Γ satisfies IND-BB.*

Theorem 9 relates the advantage of an adversary casting a vote meaningfully related to an honest voter's vote to an advantage in guessing the honest voter's vote, in a setting where the election result does not contain the honest voter's vote.

Proof. Let $\Gamma = (\mathsf{Setup}, \mathsf{Vote}, \mathsf{BB}, \mathsf{Tally})$. For the forward implication, suppose Γ does not satisfy IND-BB, hence, for any negligible function f, there exists an adversary $\mathcal{A} = (A_1, A_2, A_3)$ and a security parameter n such that $\mathsf{IND\text{-}BB}_{\mathcal{A},\Gamma}(n) > f(n)$, moreover, $\mathsf{IND\text{-}BB}_{\mathcal{A},\Gamma}(n) > 2 \cdot f(n)$, since since doubling a negligible function produces another negligible function. Let us show that Γ does not satisfy NM-BB, by constructing an adversary $\mathcal{B} = (B_1, B_2)$ as follows:

Algorithm B_1**.** Given input $\mathfrak{m}$ and pk, the algorithm computes $(v_0, v_1, s) \leftarrow A_1^{\mathcal{O}}(\mathfrak{m}, pk)$ and outputs $(\{v_0, v_1\}, (\{v_0, v_1\}, s))$.

Algorithm B_2**.** Given input $(\{v_0, v_1\}, s)$, the algorithm computes $t \leftarrow A_2^{\mathcal{O}}(s)$, selects some random coins r, and outputs the relation R such that $R(v, \mathfrak{v})$ holds if $v = v_g$ and fails otherwise, where $g \leftarrow A_3(t, \mathfrak{v}; r)$.

Let us consider executions of $\mathsf{Succ}^{\mathrm{BB}}_{\mathcal{B},\Pi}(n)$ and $\mathsf{Succ}^{\mathrm{BB}}_{\mathcal{B},\Pi,\$}(n)$.

- First, $\mathsf{Succ}^{\mathrm{BB}}_{\mathcal{B},\Pi}(n)$, where a single vote v is sampled from M. By inspecting the values provided to the embedded instance of $\mathcal{A}$, we see that the distribution of these values is identical to if $\mathcal{A}$ were interacting with IND-BB directly. The use of A_3 is in a non-black-box manner but this does not matter: it is still invoked exactly one time in the game. Hence, the probability that A_3's output matches the challenger's bit β is equal to the probability that $\mathcal{A}$ wins the IND-BB game, that is, strictly greater than $(2 \cdot f(n) + 1)/2$.
- Secondly, $\mathsf{Succ}^{\mathrm{BB}}_{\mathcal{B},\Pi,\$}(n)$, where two votes v and v' are sampled from M. The value v is independent of A's perspective, indeed, v could be sampled after A_3 has terminated and immediately before evaluating the relation R. It follows immediately that R holds iff $v = v_g$, where g is A_3's output and g is independent of v. Hence, the probability that R holds is $1/2$.

The advantage of our adversary $\mathcal{B}$ in NM-BB is therefore strictly greater than $(2 \cdot f(n)+1)/2-1/2 = f(n)$, concluding this direction of the proof by contraposition.

For the reverse implication, suppose Γ does not satisfy NM-BB, hence, for any negligible function f there exists an adversary $\mathcal{A} = (A_1, A_2)$ and a security parameter n such that $\mathsf{NM\text{-}BB}_{\mathcal{A},\Gamma}(n) > 2 \cdot f(n)$. Let us construct an adversary $\mathcal{B} = (B_1, B_2, B_3)$ against IND-BB as follows:

Algorithm B_1**.** Given input $\mathfrak{m}$ and pk, the algorithm computes $(M, s) \leftarrow A_1^{\mathcal{O}}(\mathfrak{m}, pk)$; $v_0, v_1 \leftarrow M$ and outputs $(v_0, v_1, (v_0, M, s))$.

Algorithm B_2**.** Given input (v_0, M, s), the algorithm computes $R \leftarrow A_2^{\mathcal{O}}(M, s)$ and outputs (v_0, R).

Algorithm B_3**.** Given input (v_0, R) and $\mathfrak{v}$, the algorithm evaluates $R(v_0, \mathfrak{v})$ and if the relation holds, then the algorithm outputs 0, otherwise, the algorithm outputs 1.

If the challenger selects $\beta = 0$ in IND-BB, then the embedded adversary $\mathcal{A}$ sees exactly the same distribution of values as in $\mathsf{Succ}^{\mathrm{BB}}_{\mathcal{B},\Pi}(n)$, otherwise ($\beta = 1$), $\mathcal{A}$ sees the same distribution as in the second $\mathsf{Succ}^{\mathrm{BB}}_{\mathcal{B},\Pi,\$}(n)$. Let g be $\mathcal{B}$'s guess in IND-BB. The success probability of B is:

$$\begin{aligned}
\Pr[\beta = g] &= \Pr[\beta = 0] \cdot \Pr[g = 0 \mid \beta = 0] + \Pr[\beta = 1] \cdot \Pr[g = 1 \mid \beta = 1] \\
&= 1/2 \cdot (\Pr[g = 0 \mid \beta = 0] + \Pr[g = 1 \mid \beta = 1]) \\
&= 1/2 \cdot (\Pr[R(v_0, \mathfrak{v})] + (1 - \Pr[R(v_1, \mathfrak{v})])) \\
&= 1/2 + 1/2 \cdot \mathsf{NM\text{-}CPA}_{\mathcal{A},\Pi}(n)
\end{aligned}$$

Since $1/2 + 1/2 \cdot \mathsf{NM\text{-}CPA}_{\mathcal{A},\Pi}(n) > 1/2 + f(n)$, the advantage of B is greater than $f(n)$, concluding the proof. □

4.3 Controlled malleability is sufficient

Recall that ballot independence tolerates meaningfully related ballots, assuming they are rejected by the bulletin board. It follows intuitively that we can weaken the requirement for an NM-CPA encryption scheme in Proposition 7, assuming we modify Enc2Vote's bulletin board algorithm to reject ballots meaningfully related to existing ballots on the bulletin board. We start with a simple example. Given an encryption scheme satisfying NM-CPA, we can derive a new encryption scheme by prepending a random bit to all ciphertexts and removing this bit before decryption. This new encryption scheme does not satisfy NM-CPA, however, we can derive an election scheme satisfying ballot independence using Enc2Vote if we modify Enc2Vote's bulletin board algorithm as follows: given a bulletin board $\mathfrak{bb}$ and ballot b, reject b if it is identical to any ballot already on $\mathfrak{bb}$ up to the first bit. This example shows that non-malleable ballots are not necessary for ballot independence. Let us now formalise a notion of *controlled malleability*[5], denoted NM-CPA/R (pronounced "NM-CPA modulo R"), which we will show is sufficient for ballot independence.

Definition 10 (Controlled malleability). *Let $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be an asymmetric encryption scheme and R be an efficiently computable equivalence relation on Π's ciphertext space. We say that Π satisfies NM-CPA/R (or* controlled malleability*) if for all efficient adversaries $\mathcal{A}$ the following probability is negligible*

$$Pr\left[(pk, sk) \leftarrow \mathsf{Gen}(1^n); \beta \leftarrow \{0,1\} \ : \ \mathcal{A}^{\mathsf{chal}_\beta, \mathsf{dec}}(pk) = \beta\right]$$

where the oracles chal *and* dec *are defined as follows and each oracle may be called once, in any order.*

- chal_β *takes two messages m_0 and m_1 of equal length as input, computes $c^* \leftarrow \mathsf{Enc}_{pk}(m_\beta)$, and outputs c^*.*
- dec *takes a vector $\mathbf{c}$ of ciphertexts as input. If chal_β has previously output a ciphertext c^* such that $R(c, c^*)$ holds for some $c \in \mathbf{c}$, then output $\perp$, otherwise, output $\mathsf{Dec}_{sk}(\mathbf{c})$.*

Our definition generalises non-malleability for encryption, in particular, NM-CPA = NM-CPA/R, when R is the identity. Moreover, we note that our definition could be adapted to a notion of CCA2/R by allowing arbitrarily many decryption queries. The construction Enc2Vote can be generalised to asymmetric encryption schemes satisfying controlled malleability as follows.

Definition 11 (Enc2Vote/R). *Suppose $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is an asymmetric encryption scheme and R is an efficiently computable equivalence relation on Π's ciphertext space, we define $\mathsf{Enc2Vote}/R(\Pi) = (\mathsf{Setup}, \mathsf{Vote}, \mathsf{BB}, \mathsf{Tally})$ as follows. Let the* Setup, Vote *and* Tally *algorithms be given by $\mathsf{Enc2Vote}(\Pi)$. The* BB *algorithm takes $\mathfrak{bb}$ and b as input, where $\mathfrak{bb}$ is a multiset. If there exists $b' \in \mathfrak{bb}$ such that $R(b, b')$, then* BB *outputs $\mathfrak{bb}$, otherwise,* BB *outputs $\mathfrak{bb} \cup \{b\}$.*

[5] The term is taken from Kohlweiss et al. [CKLM12] who introduce controlled malleability for zero-knowledge proofs.

Assuming that the relation R does not relate fresh, honestly generated ciphertexts in Π's ciphertext space to other values (Definition 12), we can ensure that $\mathsf{Enc2Vote}/R(\Pi)$ satisfies the correctness condition of election schemes and, hence, $\mathsf{Enc2Vote}/R(\Pi)$ is an election scheme satisfying ballot independence by (Proposition 13).

Definition 12 (Sparse relation). *Let $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be an asymmetric encryption scheme and R be an efficiently computable equivalence relation on Π's ciphertext space. We say R is a* sparse relation *if for all $(pk, sk) \leftarrow \mathsf{Gen}$, c and m, we have $c' \leftarrow \mathsf{Enc}(m, pk)$ yields $R(c, c') = 0$ with overwhelming probability.*

Proposition 13. *Suppose Π is an asymmetric encryption scheme and R is an efficiently computable and sparse equivalence relation on Π's ciphertext space such that Π satisfies NM-CPA/R. We have $\mathsf{Enc2Vote}/R(\Pi)$ satisfies ballot independence.*

The proof of Proposition 13 is similar to the proof of [BPW12b, Theorem 4.2].

Intuitively, we could adapt the controlled malleability game for encryption (Definition 10) to a controlled malleability game for ballots. In this setting, given an election scheme satisfying our controlled malleability game for ballots and such that the bulletin board algorithm rejects duplicates, we believe that the election scheme satisfies ballot independence. Moreover, the generalised definition would allow us to consider whether controlled malleability for ballots is necessary for ballot independence. (Clearly such results cannot be considered using controlled malleability for encryption, since this definition excludes election schemes based upon alternative cryptographic primitives, such as commitments, for example.) Formalising this result is a possible direction for future research.

Design paradigms and discussion. We derive the following design paradigms from our results: 1) use non-malleable ballots (Section 4.1), or 2) identify and reject related ballots using controlled malleability. The latter paradigm is particularly useful when ballots contain malleable data such as voter identities or pseudonyms, since we can tolerate malleability and provide provable security. Moreover, it facilitates more realistic models of election schemes in comparison with earlier work, for example, Bernhard *et al.* [BCP+11,BPW12b,BPW12a] abstractly model Helios ballots as non-malleable ciphertexts, whereas, in practice, Helios ballots embed non-malleable ciphertexts in malleable JavaScript Object Notation (JSON) data structures (this is particularly relevant, since Smyth & Cortier [SC10, §4.1] have shown that the JSON structures introduces vulnerabilities).

5 Ballot secrecy implies ballot independence

In this paper, all election schemes satisfy correctness: the bulletin board algorithm BB adds honestly constructed ballots to the bulletin board, the tally algorithm Tally includes honest votes in the election result, and the number of

votes in an election result corresponds to the number of ballots (that is, each ballot contains one vote). In this setting, an election scheme satisfying ballot secrecy also satisfies ballot independence.

Theorem 14 (Ballot secrecy implies ballot independence). *Given an election scheme* $\Gamma = (\mathsf{Setup}, \mathsf{Vote}, \mathsf{BB}, \mathsf{Tally})$ *satisfying ballot secrecy, we have* Γ *satisfies ballot independence.*

Theorem 14 relates an advantage in guessing an honest voter's vote in a setting where the election result *does not* contain the honest voter's vote to an advantage in the ballot secrecy game where the election result *does* include the honest voter's vote. It follows, by Theorem 9, that an advantage in casting a vote meaningfully related to an honest voter's vote translates into an advantage in guessing an honest voter's vote, hence, we have shown that ballot independence is necessary for ballot secrecy in election schemes defined by Definition 3.

The proof of Theorem 14 is standard: by contradiction, we construct an adversary $\mathcal{B} = (B_1, B_2)$ against $\mathsf{IND\text{-}SEC}$ from a successful adversary $\mathcal{A} = (A_1, A_2, A_3)$ against $\mathsf{IND\text{-}BB}$ such that $\mathcal{B}$ ensures $\mathcal{A}$'s perspective of the bulletin board and election result are consistent with $\mathsf{IND\text{-}BB}$. Before explaining how we ensure that $\mathcal{A}$'s perspective is consistent, let us briefly review the distinction between $\mathcal{A}$'s and $\mathcal{B}$'s perspectives of their respective bulletin board and election result. We first remark that the functionalities provided by $\mathcal{A}$'s oracle $\mathcal{O}_{\mathcal{A}}$ are a subset of those provided by $\mathcal{B}$'s oracle $\mathcal{O}_{\mathcal{B}}$. In $\mathsf{IND\text{-}BB}$, the adversary A_1 expects $\mathcal{O}_{\mathcal{A}}() = \mathfrak{bb}$ such that $b \in \mathfrak{bb}$ implies A_1 previously called $\mathcal{O}_{\mathcal{A}}(b)$. Moreover, adversaries A_2 and A_3 expect $\mathcal{O}_{\mathcal{A}}() = \mathfrak{bb} \cup \{b'\}$ such that b' is the challenge ballot and $b \in \mathfrak{bb}$ implies A_1 or A_2 previously called $\mathcal{O}_{\mathcal{A}}(b)$. Furthermore, A_3 observes the election result $\mathfrak{v} \backslash \{v_\beta\}$, where v_β is the challenge vote and $(\mathfrak{v}, aux) \leftarrow \mathsf{Tally}_{sk}(\mathfrak{bb})$. By comparison, the adversary $\mathcal{B}$ expects $\mathcal{O}_{\mathcal{B}}() = \mathfrak{bb}$ such that $b \in \mathfrak{bb}$ implies $\mathcal{B}$ previously called $\mathcal{O}_{\mathcal{B}}(b)$ or $\mathcal{O}_{\mathcal{B}}(v_0, v_1)$, and in the latter case the oracle constructed $b = \mathsf{Vote}_{pk}(v_\beta)$. Furthermore, $\mathcal{B}$ observes the election result $\mathfrak{v}$, where $(\mathfrak{v}, aux) \leftarrow \mathsf{Tally}_{sk}(\mathfrak{bb}_\alpha)$. It follows immediately that A_2 and A_3 will observe a challenge ballot on the bulletin board, whereas, $\mathcal{B}$ will not. In addition, the challenge vote will be removed from the election result observed by A_3, whereas no votes are removed from the election result observed by $\mathcal{B}$. Let us now informally explain how $\mathcal{B}$ ensures that $\mathcal{A}$'s perspective of the bulletin board and election result are consistent with $\mathsf{IND\text{-}BB}$. First, $\mathcal{B}$ ensures that a challenge ballot appears on the bulletin board observed by adversaries A_2 and A_3 by calling $\mathcal{O}_{\mathcal{B}}(v_0, v_1)$, where votes v_0 and v_1 are output by A_1. Secondly, the adversary $\mathcal{B}$ calls $\mathcal{O}_{\mathcal{B}}(v_1, v_0)$ and inputs the election result $\mathfrak{v}' \backslash \{v_1, v_0\}$ to A_3, where $(\mathfrak{v}', aux') \leftarrow \mathsf{Tally}_{sk}(\mathfrak{bb})$. We remark that tallying after the first step will produce an election result which includes the challenge vote v_β and does not correspond to the election result expected by $\mathcal{A}$; the second step overcomes this problem.

Proof of Theorem 14. Suppose $\Gamma = (\mathsf{Setup}, \mathsf{Vote}, \mathsf{BB}, \mathsf{Tally})$ is an election scheme with ballot secrecy that does not satisfy $\mathsf{IND\text{-}BB}$, hence for any negligible function f there exists an adversary $\mathcal{A} = (A_1, A_2, A_3)$ and security parameter n such

that $\mathsf{IND\text{-}BB}_{\mathcal{A},\Gamma}(n) > f(n)$. We construct an adversary $\mathcal{B} = (B_1, B_2)$ against IND-SEC as follows.

Algorithm B_1**.** Given input $\mathfrak{m}$ and pk, the algorithm proceeds as follows. First, B_1 computes $(v_0, v_1, s) \leftarrow A_1^{\mathcal{O}_{\mathcal{A}}}(\mathfrak{m}, pk)$, handling any oracle calls from A_1 as follows: if A_1 calls $\mathcal{O}_{\mathcal{A}}(b)$, then B_1 calls $\mathcal{O}_{\mathcal{B}}(b)$, similarly, if A_1 calls $\mathcal{O}_{\mathcal{A}}()$, then B_1 computes $\mathfrak{bb} \leftarrow \mathcal{O}_{\mathcal{B}}()$ and returns $\mathfrak{bb}$ to A_1. Secondly, B_1 creates and extracts the challenge ballot by computing $\mathfrak{bb} \leftarrow \mathcal{O}_{\mathcal{B}}(); \mathcal{O}_{\mathcal{B}}(v_0, v_1); \mathfrak{bb}' \leftarrow \mathcal{O}_{\mathcal{B}}(); b \leftarrow \mathfrak{bb}' \backslash \mathfrak{bb}$. Thirdly, B_1 computes $t \leftarrow A_2^{\mathcal{O}_{\mathcal{A}}}(s)$, handling any oracle calls from A_2 as before. Finally, B_1 computes $\mathfrak{bb}^{\dagger} \leftarrow \mathcal{O}_{\mathcal{B}}(); \mathcal{O}_{\mathcal{B}}(v_1, v_0); \mathfrak{bb}^{\ddagger} \leftarrow \mathcal{O}_{\mathcal{B}}(); b^{\dagger} \leftarrow \mathfrak{bb}^{\ddagger} \setminus \mathfrak{bb}^{\dagger}$ and outputs $(t, b^{\dagger})$.

Algorithm B_2**.** Given input $\mathfrak{v}$ and $(t, b^{\dagger})$, the algorithm computes $A_3(t, \mathfrak{v} \setminus \{v_0, v_1\})$ and outputs A_3's guess.

The embedded adversary $\mathcal{A}$ sees the same distribution of all elements as in the IND-BB game for the same value of β. Indeed, the challenge ballot is computed in the same manner, $\mathcal{O}_{\mathcal{A}}()$ produces the expected multiset of ballots (we stress that $b^{\dagger}$ – namely, the ballot introduced by B_1 to ensure that the election result is consistent with A_3's expectations – never appears in a multiset output by $\mathcal{O}_{\mathcal{A}}()$, since $b^{\dagger}$ is added to the bulletin board after all oracle calls by $\mathcal{A}$), and the election result observed by A_3 is as expected. It follows that $\mathcal{B}$ guesses β correctly with the same advantage as $\mathcal{A}$ and, therefore, $\mathsf{IND\text{-}SEC}_{B,\Gamma}(n) > f(n)$, concluding our proof. □

5.1 Critique of Desmedt & Chaidos's Helios variant

Intuitively, Theorem 14 contradicts the results by Desmedt & Chaidos [DC12], who claim to provide a variant of the Helios election scheme which allows Bob to cast the same vote as Alice, with Alice's cooperation, whilst preventing Bob from learning Alice's vote. In their protocol, Bob selects Alice's ballot from the bulletin board and communicates with Alice to generate a new ballot that is guaranteed to contain the same vote as Alice's. Desmedt & Chaidos's security claim is true *before the election result is announced*, since Bob gains no advantage in guessing Alice's vote. However, *after the election result is announced*, the claim is false. We can informally contradict this claim – using results by Cortier & Smyth [CS11, SC11, CS13] – in an election with voters Alice, Bob and Charlie: if Bob casts the same vote as Alice, then Bob can learn Alice's vote by observing the election result and checking which candidate obtained at least two votes (that is, Bob can learn Alice's vote when the election result is not unanimous). We believe the erroneous claim by Desmedt & Chaidos is due to an invalid inference from their computational security result. Indeed, although the result [DC12, Theorem 1] is correct, their model does not support their claims for real world security: Desmedt & Chaidos consider a passive adversary that cannot observe the election result, whereas, we believe a practical notion of security must consider an *active* adversary who can cast ballots and observe the election result, since this captures the capabilities of an attacker in the real world. Nonetheless, a weaker notion

of ballot secrecy may be satisfiable in Desmedt & Chaidos's variant of Helios, assuming Alice never cooperates with the adversary. Clearly, no claims can be made about Bob's knowledge of Alice's vote in this setting. We have shown Desmedt & Chaidos our results and Chaidos agrees with our findings [Cha13].

5.2 Discussion

We have shown that election schemes satisfying ballot secrecy must also satisfy ballot independence. However, we must concede that alternative formalisms of election schemes may permit different results. Indeed, Cortier & Smyth [CS13, Section 7.1] present a result to the contrary using anonymous channels, which are implicitly excluded from our model. Moreover, our model also excludes settings where the adversary cannot control a majority of voters and places some restrictions on the election result, namely, the election result is captured as a multiset which reveals the number of votes for each candidate. In this setting, an election result can be computed from a partial election result if the votes of the remaining voters are known. This property is implicitly used in our proof of Theorem 14, where we take the election result and challenge vote, and compute the partial election result which removes the challenge vote. On the other hand, some practical election schemes do not have this property. For example, consider an election scheme which announces the winning candidate, but does not provide a breakdown of the votes for each candidate [BY86, HK02, HK04, DK05]. It follows that knowledge of a partial election result can only be used to derive the election result if the adversary controls a majority of voters. Similarly, given an election result and knowledge of a minority of votes, a partial election result which excludes the known votes cannot be derived. In this setting, we believe election schemes can satisfy ballot secrecy but not ballot independence, since casting a minority of related ballots is not sufficient to reveal a voter's vote. Formal treatment of this case and consideration of whether such schemes are practical is a possible direction for future work.

6 Sufficient conditions for ballot secrecy

The main distinctions between our ballot secrecy (IND-SEC) and ballot independence (IND-BB) games are as follows.

1. The challenger in our ballot independence game explicitly defines a challenge ballot and adds the ballot to the bulletin board, whereas, the challenger in our ballot secrecy game provides the adversary with an oracle $\mathcal{O}_{\mathcal{B}}(\cdot,\cdot)$.

The two formulations are similar, indeed, the challenger's computation $b \leftarrow \mathsf{Vote}_{pk}(v_\beta)$; $\mathfrak{bb} \leftarrow \mathsf{BB}(\mathfrak{bb}, b)$ is similar to an oracle call $\mathcal{O}_{\mathcal{B}}(v_0, v_1)$. Moreover, a hybrid argument will show that it does not matter if we give the adversary only one challenge ballot or many oracle calls.

2. The adversary in our ballot secrecy game has access to the auxiliary data produced during tallying, but the adversary in our ballot independence game does not.

The second point distinguishes our two games; Theorem 14 shows that ballot secrecy is stronger than independence and Footnote 3 gives a case where it is strictly stronger: the presentation of the Enc2Vote construction by Bernhard, Pereira & Warinschi provides ballot independence, but the auxiliary data maps voters to votes, thereby violating ballot secrecy. Nonetheless, by restricting the adversary's access to auxiliary data we can show that the two games are equivalent (Theorem 15) and, hence, in the absence of auxiliary data, ballot independence is a sufficient condition for ballot secrecy, in particular, Enc2Vote and Enc2Vote/R are constructions for election schemes satisfying ballot secrecy.

Theorem 15 (NM-BB = IND-SEC, without auxiliary data). *Suppose* $\Gamma = (\mathsf{Setup}, \mathsf{Vote}, \mathsf{BB}, \mathsf{Tally})$ *is an election scheme such that there exists a constant symbol* $\perp$ *and for all parameters* $(\mathfrak{bb}_0, \mathfrak{m}, pk, sk) \leftarrow \mathsf{Setup}(1^n)$*, multiesets* $\mathfrak{bb}$ *and tallying data* $(\mathfrak{v}, aux) \leftarrow \mathsf{Tally}_{sk}(\mathfrak{bb})$*, we have* $aux = \perp$*. It follows that* Γ *satisfies ballot secrecy if and only if* Γ *satisfies ballot independence.*

Proof. Suppose Γ is an election scheme that does not satisfy IND-BB, hence for any negligible function f there exists an adversary $\mathcal{A}$ and security parameter n such that $\mathsf{IND\text{-}BB}_{\mathcal{A},\Gamma}(n) > f(n)$. The adversary $\mathcal{B}$ defined in the proof of Theorem 14 is such that $\mathsf{IND\text{-}SEC}_{B,\Gamma}(n) > f(n)$. It remains to show that ballot independence implies ballot secrecy.

Suppose Γ is an election scheme that does not satisfy IND-SEC, hence for any negligible function f there exists an adversary $\mathcal{B} = (B_1, B_2)$ and security parameter n such that $\mathsf{IND\text{-}SEC}_{\mathcal{B},\Gamma}(n) > f(n)$. By inspection of the game IND-SEC, the adversary B_1 makes an oracle call $\mathcal{O}_{\mathcal{B}}(v_0, v_1)$ for some votes v_0, v_1 in the voting scheme's vote space, hence, there exist adversaries $B_{1,1}$ and $B_{1,2}$ such that $B_1 = s' \leftarrow B_{1,1}; \mathcal{O}_{\mathcal{B}}(v_0, v_1); B_{1,2}(s')$ and $B_{1,2}$ outputs s. We wish to construct an adversary $\mathcal{A} = (A_1, A_2, A_3)$ against IND-BB out of $\mathcal{B}$ but the IND-BB game only gives us one opportunity to call the challenge oracle (as the output v_0, v_1 of A_1). We deal with the $\mathcal{O}_{\mathcal{B}}(\cdot, \cdot)$ calls using a hybrid argument:

Let q be an upper bound on the number of $\mathcal{O}_{\mathcal{B}}(\cdot, \cdot)$ queries that $\mathcal{B}$ makes. Since $\mathcal{B}$ is efficient, assume that q is polynomial in the security parameter. Let G_0 be the IND-SEC game modified such that the bit β is always chosen as 0 and let G_q be the game in which β is always 1; the adversary $\mathcal{B}$'s advantage against IND-SEC is clearly his distinguishing advantage between G_0 and G_q.

Let G_i for $0 < i < q$ be the IND-SEC game in which the first i queries to $\mathcal{O}_{\mathcal{B}}(\cdot, \cdot)$ are answered as if $\beta = 1$ and the rest as if $\beta = 0$. If $\mathcal{B}$ has non-negligible advantage in distinguishing G_0 from G_q then there must be an i such that $0 \leq i < q$ and $\mathcal{B}$ distinguishes G_i from G_{i+1} with non-negligible advantage too; more precisely if $\mathsf{IND\text{-}SEC}_{\mathcal{B},\Gamma}(n) = \alpha$ then there must be an i for which the distinguishing advantage is at least i/q.

Suppose we know such an i. We construct $\mathcal{A}$ as the following adversary.

A_1 Receive $\mathfrak{m}, pk$ from the challenger and run B_1 on these inputs; forward $\mathcal{O}()$ and $\mathcal{O}(b)$ calls to the challenger and return its replies. Initialise two lists L_0 and L_1 to be empty. For the first i $\mathcal{O}_{\mathcal{B}}(v_0, v_1)$ calls that B_1 makes, create a ballot b for v_1 and issue an $\mathcal{O}(b)$ call to the challenger. Also store the votes

(v_0, v_1) by appending them to L_0 and L_1 respectively. When A_1 makes the $i+1$-st two-element call with parameters v_0, v_1, suspend the execution of B_1, save its state as s, add the parameters to the current call to L_0 and L_1 and return $(v_0, v_1, (s, L_0, L_1))$.

A_2 Parse the received state as (s, L_0, L_1). Resume B_1 using state s and handle oracle calls as above except that for oracle calls $\mathcal{O}_{\mathcal{B}}(v_0, v_1)$, create a ballot b for v_0 and issue a $\mathcal{O}(b)$ call for it. Keep adding the inputs to two-element oracle calls to L_0 and L_1.
When B_1 outputs some state s, make a $\mathfrak{bb} \leftarrow \mathcal{O}()$ call to obtain the current board and return $(\mathfrak{bb}, s, L_0, L_1)$. (If B_1 outputs state s before it has made $i+1$ two-element oracle queries, we abort $\mathcal{B}$ and let A_3 guess a random bit $g \xleftarrow{R} \{0,1\}$.)

A_3 We get a state $t = (\mathfrak{bb}, s, L_0, L_1)$ and a tally $\mathfrak{v}' = \mathfrak{v} \backslash \{v_\beta\}$ as input. We remove all votes from $\mathfrak{v}'$ that came from $\mathcal{O}_{\mathcal{B}}(\cdot, \cdot)$ calls (except the "challenge" one which has already been removed by our challenger). We can do this using the lists L_0 and L_1. Next, we add all the votes in L_0 back on to the result; call the multiset thus created $\mathfrak{v}_{\mathcal{B}}$.
We run $\mathcal{B}_2$ on $\mathfrak{bb}$, $\mathfrak{v}_{\mathcal{B}}$, auxiliary data[6] $\perp$ and state s until it terminates with a guess g, which we return to our challenger and halt.

We claim that this construction provides a view of either G_i or G_{i+1} towards $\mathcal{B}$ and retains the distinguishing advantage. In the case that $\mathcal{B}$ makes fewer than $i+1$ two-element oracle queries, its advantage must be $1/2$ since the two games are identical until the $i+1$-st of these queries. Therefore we can let A_3 just guess randomly in this case. From now on, assume that $\mathcal{B}$ makes at least $i+1$ two-element oracle queries.

Suppose that our challenger chose $\beta = 0$. In this case, the first i two-element queries result in a ballot for the "right-hand" vote (i.e. v_1 if the submitted pair of votes is (v_0, v_1), the $i+1$-st query is forwarded to our challenger who chooses v_0 due to $\beta = 0$ and all further calls are answered using v_0: this is exactly the procedure of G_i. If $\beta = 1$ then the only difference is that the $i+1$-st oracle call is answered using v_1, resulting in exactly the procedure of G_{i+1}.

Therefore, we preserve the distinguishing advantage $f(n)$ of $\mathcal{B}$ in our adversary $\mathcal{A}$ against IND-BB. □

Intuitively, we can generalise Theorem 15 to election schemes in which the auxiliary data can be simulated. Since the auxiliary data output by election schemes typically consists of signatures of knowledge proving that the election result has been correctly computed from the bulletin board, we expect many practical election schemes will satisfy zero-knowledge auxiliary data, indeed, Helios outputs partial ElGamal decryptions [Ped91, CP93] and proofs demonstrating knowledge of discrete logarithms [CEGP87, CEG88, Sch90] which can be simulated. In this context, we believe ballot secrecy and ballot independence

[6] In a proof of Theorem 16, at this point we would use the zero-knowledge simulator to generate auxiliary data, in the appropriate manner for the model of zero knowledge under consideration.

coincide (Remark 16). Unfortunately, formalising zero-knowledge is a complex issue – in particular, the simulator needs some extra capabilities compared to the election officials (otherwise the officials could publish simulated proofs!) – to which there is no general solution and, hence, there is no general proof of Remark 16. Nonetheless, we believe Remark 16 can be shown to hold for particular formalisations of zero-knowledge, for instance, a proof could be constructed in the programmable random oracle model (the proof would essentially be that of Theorem 15 with the simulator being run at the appropriate point; we briefly comment on this in the proof of Theorem 15) and, hence, a proof of ballot secrecy can be reduced to a proof of ballot independence.

Remark 16 (NM-BB = IND-SEC for zero-knowledge auxiliary data). Given an election scheme Γ satisfying zero-knowledge auxiliary data (informally, zero-knowledge auxiliary data means that the auxiliary data can be simulated given the result), we have Γ satisfies ballot secrecy if and only if Γ satisfies ballot independence.

Remark 16 suggests that ballot independence is a sufficient condition for ballot secrecy in election schemes where auxiliary data can be simulated. Coupled with earlier results [BPW12a], this should facilitate a proof of ballot secrecy in Helios. (Bernhard *et al.* [BCP+11] provide a proof of ballot secrecy in a variant of Helios which uses the Naor & Yung transformation [NY90] to derive non-malleable ballots and Bernhard, Pereira & Warinschi [BPW12a] prove that Helios satisfies ballot secrecy in the special case of referendums, however, a full proof of ballot secrecy in Helios is not currently known.)

7 Conclusion

We have formalised *ballot independence* in a variant of the model for election schemes proposed by Bernhard *et al.* Our main results are as follows. Ballot secrecy implies ballot independence; the converse holds too if there is no auxiliary data. Moreover, we have argued that ballot independence and ballot secrecy coincide if auxiliary data is "zero knowledge;" since auxiliary data typically consists of zero knowledge proofs, this assumption is realistic and holds for election schemes such as Helios, for instance. Furthermore, we provide some sufficient conditions for ballot independence and, hence, ballot secrecy: we show that non-malleable ballots are sufficient but not necessary for independence and secrecy, and introduce a weaker notion of controlled-malleable encryption which we show is sufficient, moreover, this notion is better suited to modelling the way ballots are handled in practice (for example, by Helios). In addition, we show that the notion of ballot secrecy proposed by Bernhard *et al.* does not capture attacks which rely on auxiliary data and we adopt a stronger definition. Furthermore, we show that the variant of Helios proposed by Desmedt & Chaidos does not satisfy ballot secrecy.

Acknowledgements. We are particularly grateful to Bogdan Warinschi and the anonymous reviewers who read earlier versions of this paper and provided useful guidance. This work has been partly supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC project *CRYSP* (259639) and by EPSRC via grant EP/H043454/1.

References

[Adi08] Ben Adida. Helios: Web-based Open-Audit Voting. In *USENIX Security'08: 17th USENIX Security Symposium*, pages 335–348. USENIX Association, 2008.

[AMPQ09] Ben Adida, Olivier de Marneffe, Olivier Pereira, and Jean-Jacques Quisquater. Electing a University President Using Open-Audit Voting: Analysis of Real-World Use of Helios. In *EVT/WOTE'09: Electronic Voting Technology Workshop/Workshop on Trustworthy Elections*. USENIX Association, 2009.

[BCP+11] David Bernhard, Véronique Cortier, Olivier Pereira, Ben Smyth, and Bogdan Warinschi. Adapting Helios for provable ballot privacy. In *ESORICS'11: 16th European Symposium on Research in Computer Security*, volume 6879 of *LNCS*, pages 335–354. Springer, 2011.

[BDPR98] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations Among Notions of Security for Public-Key Encryption Schemes. In *CRYPTO'98: 18th International Cryptology Conference*, volume 1462 of *LNCS*, pages 26–45. Springer, 1998.

[BGP11] Philippe Bulens, Damien Giry, and Olivier Pereira. Running Mixnet-Based Elections with Helios. In *EVT/WOTE'11: Electronic Voting Technology Workshop/Workshop on Trustworthy Elections*. USENIX Association, 2011.

[BHM08] Michael Backes, Cătălin Hriţcu, and Matteo Maffei. Automated Verification of Remote Electronic Voting Protocols in the Applied Pi-calculus. In *CSF'08: 21st Computer Security Foundations Symposium*, pages 195–209. IEEE Computer Society, 2008.

[BPW12a] David Bernhard, Olivier Pereira, and Bogdan Warinschi. How Not to Prove Yourself: Pitfalls of the Fiat-Shamir Heuristic and Applications to Helios. In *ASIACRYPT'12: 18th International Conference on the Theory and Application of Cryptology and Information Security*, volume 7658 of *LNCS*, pages 626–643. Springer, 2012.

[BPW12b] David Bernhard, Olivier Pereira, and Bogdan Warinschi. On Necessary and Sufficient Conditions for Private Ballot Submission. Cryptology ePrint Archive, Report 2012/236 (version 20120430:154117b), 2012.

[BS99] Mihir Bellare and Amit Sahai. Non-malleable Encryption: Equivalence between Two Notions, and an Indistinguishability-Based Characterization. In *CRYPTO'99: 19th International Cryptology Conference*, volume 1666 of *LNCS*, pages 519–536. Springer, 1999.

[BY86] Josh Benaloh and Moti Yung. Distributing the Power of a Government to Enhance the Privacy of Voters. In *PODC'86: 5th Principles of Distributed Computing Symposium*, pages 52–62. ACM Press, 1986.

[CEG88] David Chaum, Jan-Hendrik Evertse, and Jeroen van de Graaf. An Improved Protocol for Demonstrating Possession of Discrete Logarithms and Some

Generalizations. In *EUROCRYPT'87: 4th International Conference on the Theory and Applications of Cryptographic Techniques*, volume 304 of *LNCS*, pages 127–141. Springer, 1988.

[CEGP87] David Chaum, Jan-Hendrik Evertse, Jeroen van de Graaf, and René Peralta. Demonstrating Possession of a Discrete Logarithm Without Revealing It. In *CRYPTO'86: 6th International Cryptology Conference*, volume 263 of *LNCS*, pages 200–212. Springer, 1987.

[CGMA85] Benny Chor, Shafi Goldwasser, Silvio Micali, and Baruch Awerbuch. Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults. In *FOCS'85: 26th Foundations of Computer Science Symposium*, pages 383–395. IEEE Computer Society, 1985.

[Cha13] Pyrros Chaidos. Private email communication, March/April 2013.

[CKLM12] Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Sarah Meiklejohn. Malleable Proof Systems and Applications. In *Advances in Cryptology — Eurocrypt 2012*, volume 7237 of *LNCS*, pages 281–300, 2012.

[CP93] David Chaum and Torben P. Pedersen. Wallet Databases with Observers. In *CRYPTO'92: 12th International Cryptology Conference*, volume 740 of *LNCS*, pages 89–105. Springer, 1993.

[CS11] Véronique Cortier and Ben Smyth. Attacking and fixing Helios: An analysis of ballot secrecy. In *CSF'11: 24th Computer Security Foundations Symposium*, pages 297–311. IEEE Computer Society, 2011.

[CS13] Véronique Cortier and Ben Smyth. Attacking and fixing Helios: An analysis of ballot secrecy. *Journal of Computer Security*, 21(1):89–148, 2013.

[DC12] Yvo Desmedt and Pyrros Chaidos. Applying Divertibility to Blind Ballot Copying in the Helios Internet Voting System. In *ESORICS'12: 17th European Symposium on Research in Computer Security*, volume 7459 of *LNCS*, pages 433–450. Springer, 2012.

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-Malleable Cryptography. In *STOC'91: 23rd Theory of computing Symposium*, pages 542–552. ACM Press, 1991.

[DDN00] Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable Cryptography. *Journal on Computing*, 30(2):391–437, 2000.

[DK05] Yvo Desmedt and Kaoru Kurosawa. Electronic Voting: Starting Over? In *ISC5: International Conference on Information Security*, volume 3650 of *LNCS*, pages 329–343. Springer, 2005.

[DKR06] Stéphanie Delaune, Steve Kremer, and Mark Ryan. Coercion-Resistance and Receipt-Freeness in Electronic Voting. In *CSFW'06: 19th Computer Security Foundations Workshop*, pages 28–42. IEEE Computer Society, 2006.

[DKR09] Stéphanie Delaune, Steve Kremer, and Mark D. Ryan. Verifying privacy-type properties of electronic voting protocols. *Journal of Computer Security*, 17(4):435–487, July 2009.

[Gen95] Rosario Gennaro. Achieving independence efficiently and securely. In *PODC'95: 14th Principles of Distributed Computing Symposium*, pages 130–136. ACM Press, 1995.

[HK02] Alejandro Hevia and Marcos A. Kiwi. Electronic Jury Voting Protocols. In *LATIN'02: Theoretical Informatics*, volume 2286 of *LNCS*, pages 415–429. Springer, 2002.

[HK04] Alejandro Hevia and Marcos A. Kiwi. Electronic jury voting protocols. *Theoretical Computer Science*, 321(1):73–94, 2004.

[NY90] Moni Naor and Moti Yung. Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attacks. In *STOC'90: 22nd Theory of computing Symposium*, pages 427–437. ACM Press, 1990.

[Ped91] Torben P. Pedersen. A Threshold Cryptosystem without a Trusted Party. In *EUROCRYPT'91: 10th International Conference on the Theory and Applications of Cryptographic Techniques*, number 547 in LNCS, pages 522–526. Springer, 1991.

[SC10] Ben Smyth and Véronique Cortier. Does Helios ensure ballot secrecy? Cryptology ePrint Archive, Report 2010/625 (version 20101217:132825), 2010.

[SC11] Ben Smyth and Véronique Cortier. A note on replay attacks that violate privacy in electronic voting schemes. Technical Report RR-7643, INRIA, June 2011. `http://hal.inria.fr/inria-00599182/`.

[Sch90] Claus-Peter Schnorr. Efficient Identification and Signatures for Smart Cards. In *CRYPTO'89: 9th International Cryptology Conference*, volume 435 of *LNCS*, pages 239–252. Springer, 1990.

[Sch13] Bruce Schneier. Hacking the Papal Election. `https://www.schneier.com/blog/archives/2013/02/hacking_the_pap.html`, 2013.