



HAL
open science

Belenios with cast-as-intended: towards a usable interface

Véronique Cortier, Pierrick Gaudry, Anselme Goetschmann, Sophie Lemonnier

► To cite this version:

Véronique Cortier, Pierrick Gaudry, Anselme Goetschmann, Sophie Lemonnier. Belenios with cast-as-intended: towards a usable interface. EVote-ID 2024 - 9th International Joint Conference on Electronic Voting, Oct 2024, Terragona, Spain. hal-04646244

HAL Id: hal-04646244

<https://inria.hal.science/hal-04646244v1>

Submitted on 12 Jul 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

Belenios with cast-as-intended: towards a usable interface

Véronique Cortier¹, Pierrick Gaudry¹, Anselme Goetschmann¹, and Sophie Lemonnier^{1,2}

¹ Loria – Université de Lorraine, CNRS, Inria – France

² Université de Lorraine, PErSEUs, F-57000 Metz, France

Abstract. In this work we consider Belenios-CaI, a protocol offering a cast-as-intended mechanism and building upon Belenios, a voting system used in about 7000 elections to date. We modify the design of Belenios-Cai from the user perspective without changing its core cryptographic mechanism. The goal is to increase its usability by letting the voter simply check whether two symbols are equal or different.

We conducted a user-study among 165 participants in a research center to evaluate the usability of our implementation of Belenios-CaI. Since the cast-as-intended mechanism assumes that voters make some random choices, we also evaluate whether the choices made by voters are sufficiently “random” to provide verifiability and whether it could affect their privacy. The study shows that, for our population, Belenios-CaI is considered as usable with the random choices of the voters seeming sufficient for verifiability and privacy.

1 Introduction

E-voting aims at two main security properties, namely vote privacy and verifiability; the latter is often split in several sub-properties:

- cast-as-intended: the ballot cast by the voter contains their intended vote;
- recorded-as-cast: the ballot recorded in the ballot box corresponds to the one cast by the voter;
- tallied-as-recorded: the result corresponds to the ballots recorded in the ballot box;
- eligibility: ballots recorded in the ballot box only come from legitimate voters.

In this paper, we focus on cast-as-intended, which aims at protecting against a malicious voting device that could try to modify the vote of a voter, e.g. when encrypting it. Specifically, we consider the recently proposed Belenios-CaI system [5], which builds upon Belenios [6], a system now in production for 8

This work was partially supported by the French National Research Agency under the France 2030 programme with the reference ANR-22-PECY-0006 and ANR Chair IA ASAP (ANR-20-CHIA-0024), with support from the region Grand Est, France.

years and with more than 700,000 voters in total. In Belenios-CaI, voters audit their ballot by checking a control value. Instead of the sole encryption $\text{enc}(v)$ of the voter’s vote v , the ballot consists of three encryptions $\text{enc}(v)$, $\text{enc}(a)$, $\text{enc}(b)$ and a zero-knowledge proof that $v + a = b$ modulo c , where c is some constant greater than the number of voting options. The voting device displays the ballot and the values v, a, b, c . The voter should check that $v + a = b$ modulo c and randomly challenge their voting device to open either $\text{enc}(a)$ or $\text{enc}(b)$. The voting device has to provide the corresponding randomness and the correctness of the encryption is then discharged to auditors: the voter simply checks that either a or b appears next to their ballot on the public bulletin board and the auditors check that the randomness corresponds to the audited encryption. An interesting feature of Belenios-CaI is that the voter does not need any second device nor additional secure channel to receive extra material. However, Belenios-CaI presents usability issues. First of all, voters need to compute arithmetic operations (modular addition). Moreover the audit of the ballot involves several verifications that must be performed by the voter, which seems cumbersome.

Our contributions. We propose an encoding of addition to ease the voter’s journey. Namely, we consider addition modulo 2 only, and encode addition by asking voters to tell whether two symbols are identical or different, which is a much simpler task. We design and implement a voter interface that guides the voter to perform the checks for each voting choice. We also extend the current implementation of Belenios to support the new ballot format and verification. This yields the first implementation of Belenios-CaI. The code is open-source and covers all parts of the elections (server and voting client).

In order to check the usability of Belenios-CaI with our approach, we conducted an experiment with 165 participants, which were randomly assigned either Belenios-CaI or the original Belenios. The goal was to study whether Belenios-CaI introduces a reasonable overhead of complexity w.r.t. Belenios, a system that is used on an everyday basis. Moreover, in Belenios-CaI, voters must choose to open either $\text{enc}(a)$ or $\text{enc}(b)$ “at random”, a difficult task for voters, who are not perfect random generators. Hence we study whether the bias in the randomness could affect verifiability but also privacy. More precisely, we investigate three research questions.

Q1 - Usability *Is Belenios-CaI usable?*

We aim at understanding (i) whether voters still manage to vote with Belenios-CaI and (ii) how the additional step affects perceived usability w.r.t. Belenios.

Q2 - Secrecy *Is the secrecy of the vote degraded by the control pattern?*

Belenios-CaI exposes additional information on the public bulletin board under the form of a control pattern selected by the voter (the selection of a or b for each voting choice). Although there is by design no correlation between the voting choice and the control pattern of a ballot, it is possible that the voter’s control pattern selection is influenced by their choice of voting option. This would represent an important leakage since it can compromise the secrecy of the vote.

Q3 - Verifiability *Is the randomness provided by the voters sufficient to provide cast-as-intended verifiability?*

We aim at determining whether the random selection performed by voters is sufficient to prevent an adversary controlling voting clients from manipulating an election with a very low probability of being detected.

Our experiment shows that Belenios-CaI remains reasonably usable compared to Belenios and provide sufficient guarantees w.r.t. cast-as-intended. Moreover, our statistical analysis did not detect any meaningful correlation between votes and control patterns, indicating that Belenios-CaI does not threaten vote privacy. It is important to note, however, that our experiment was conducted in a research center in computer science, where most participants are researchers, PhD students or engineers. As future work, it would be necessary to conduct a wider experiment on the general population in order to determine whether the bias of our population could affect the conclusion of the study.

Related work. Several cast-as-intended mechanisms have been proposed, as summarized in Table 1. A natural approach is to use a second device checking that the voting device has correctly encrypted the voter’s intended vote. This is the approach followed by Estonia [8] where the voting device exports in a QR-code the randomness used for encrypting the vote. The Polyas system [14] refines this approach to provide a better receipt-freeness resistance. The Benaloh’s challenge [3] follows the same idea except that the audited ballot is never the one that is cast, again to mitigate vote-buying attacks. While the Benaloh’s challenge does not require explicitly a second device, it needs an independent party that can check the correctness of the encryption. Another approach is to use *return codes*, as proposed in Switzerland [7, 17]. Voters receive a voting sheet where each voting choice is associated to a return code (specific to each voter). Once a voter casts a vote, their voting device displays a code that must match the one written on their voting sheet (for their voting choice). This approach relies on a secure postal channel to distribute voting sheets and on a heavy infrastructure (several independent online servers). In order to avoid a second device and keep the infrastructure simple, some systems (e.g. Select [10], Selene [16], Hyperion [15]) introduce a *tracker* that appears on the public bulletin board, next to the voter’s vote, allowing them to check that their vote has been counted. One important advantage of this approach is its simplicity w.r.t. voters: they *see* their vote. However, in case of a vote manipulation, voters can only detect it once the election is over and tallied, which renders dispute-resolution even more complex.

User studies have already been conducted on other cast-as-intended mechanisms. In particular, Marky *et al.* [13] show that the Benaloh’s challenge is very difficult to conduct for voters. In a recent study, Hilt *et al.* [9] investigate how voters react to vote manipulations for systems that provide cast-as-intended through a second device while Volkamer *et al.* [18] study whether code-voting and QR-codes may respectively increase security and usability. Marky *et al.* [12] study voter’s perception when physical printed audit trails are used in parallel with online voting. Of course, no user study was applied to Belenios-CaI yet, due to its recent design.

	Audit using randomness of ballot			Return codes	Trackers	Belenios-CaI
	<i>Estonia</i> [8]	<i>Polyas</i> [14]	<i>Benaloh</i> [3]	[7, 17]	[10, 15, 16]	
Single device				✓	✓	✓
Verify and go	✓	✓	✓	✓		✓
Cast ballot is audited	✓	✓		✓	✓	✓
Single online server		✓	✓		✓	✓

Table 1. Comparison of cast-as-intended mechanisms.

2 Context

2.1 Overview of Belenios-CaI

We provide a brief overview of Belenios-CaI, that was introduced in [5] as a variant of Belenios. We refer to this article for the precise description and security analysis. We assume here that the reader is familiar with Helios-like e-voting systems. As in Belenios, the actors are the voters, a voting server, some decryption trustees, a credential authority (a.k.a. registrar), and a public bulletin board, with external auditors. In Belenios-CaI, the roles of the decryption trustees and the credential authority are unchanged, so we will not talk about them further.

For a given question, there is a set of possible answers (the candidates), and each of them can be selected, possibly with a limit on selections. The ballot is a set of micro-ballots, one for each answer, that encodes the selected / not-selected choice of the voter. Each micro-ballot takes the following form:

$$\text{bal} = (\text{enc}(v), \text{enc}(a), \text{enc}(b), \pi),$$

where v has a value of 0 or 1, which encodes the choice of the voter, a and b are random integers chosen by the voting device such that $b \equiv v + a \pmod{10}$, and π is a zero-knowledge proof that the three plaintexts hidden in the three ciphertexts are indeed integers that verify these arithmetic properties.

Once the ballot is sent to the server by the voting device, the voter receives from the server (on a channel not controlled by the voting device) a tracking number that also serves as a commitment on the ballot. This number can no longer be changed, because the voter will look for it on the public bulletin board.

Then, the audit phase starts. The voting device shows the a and b values to the voter, who must check that the modular equality $b \equiv v + a \pmod{10}$ indeed holds. A key point, here, is that this operation must be done by the voter themselves, and not by their voting device. Then, the voter picks one of the two values a and b at random, and the voting device sends to the server the randomness that was used to encrypt this value, so that the server can open the ciphertext. It then publishes on the board the tracking number, the ballot, the revealed randomness, and the corresponding a or b decrypted value. The voter visits the board (possibly

	Selected?	Audit codes
Alice	1	$1 + 5 = 6$
Bob	0	$0 + 3 = 3$
Charlie	1	$1 + 9 = 10$

Check that the additions are correct

Fig. 1. Original Belenios-CaI audit phase (picture from [5]). The voter checks an addition modulo 10, and randomly selects one of the a or b value for each line.

with another device), and checks that everything is as expected. In addition to their usual tasks, the auditors must check that the revealed randomness indeed opens the ciphertext to the value a or b published by the server on the board.

In a typical setting, there are several micro-ballots, and therefore there is one (a, b) pair for each possible answer. In [5], it is suggested to present them as in Figure 1, with all the data corresponding to one micro-ballot put on a single line, thus forming a table. Using additions modulo any number at least 2 instead of 10 is possible, and yields the same theoretical security.

2.2 Terminology used in the present study

In our paper, we will use slightly different names for the various elements in the audit phase of the voter’s journey:

- The **control values** are the values a and b , such that the **vote** v verifies $b \equiv v + a \pmod{10}$. These were called audit codes in [5].
- The **mask** is the random choice made by the voter of which control value will be revealed. This is just one bit per line (whether the blue box is on the left or on the right, in Figure 1).
- The **control pattern** is the combination of the mask and of the control values that must be revealed. On Figure 1, this is the set of blue boxes, with their positions in the table, and the values inside them. The control pattern is the data that is visible on the public bulletin board at the end of the voter’s journey and that they should compare to what their device showed to them.

3 User Interface Design

As first contribution, we present the design of a user interface for Belenios-CaI.

3.1 Challenges

Non-trivial process. Belenios-CaI requires the voter to perform a verification and a random selection for each voting item. The risk is that most of the voters

Table 2. Possible values for v , a and b modulo 2: on the left, the two cases where the voter did not select the answer, and on the right, the two cases where they did. In the first two columns, 0 is represented by *crossed* and 1 by *checked*. In the third, 0 corresponds to *thumb-up* and 1 to *thumb-down*. (Symbols taken from Google Fonts [1].)

v	+	a	=	$b \bmod 2$		v	+	a	=	$b \bmod 2$
<input type="checkbox"/>		<input type="checkbox"/>		0	<input type="thumbs-up"/>	<input checked="" type="checkbox"/>		<input type="checkbox"/>		1
0		0		0		1		0		1
<input type="checkbox"/>		<input checked="" type="checkbox"/>		1	<input type="thumbs-down"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>		0
0		1		1		1		1		0

would not actually verify what is asked but just validate when possible, to complete the process faster. This risk is exacerbated by the fact that voters perform this task online and anonymously [11]. Since this would compromise the verifiability of the cast-as-intended property, the system should enforce the verification.

More generally, we aim at designing an interface that guides voters linearly through simple tasks.

Arithmetical operations. Further, the system should be usable by anyone and we should not assume technical knowledge of the voter. In particular, the voter cannot be relied on to perform arithmetical operations. Since Belenios-CaI depends on the computation of a modular addition for each voting option, a workaround had to be found.

Voter randomness. Another critical aspect of the protocol is the randomness provided by the voter during the selection of a mask. Although a slightly biased randomness is sufficient for verifiability as explained in Section 4.3, it should not be influenced by the intention of the voter to preserve vote secrecy.

3.2 Design choices

Compute instead of verify. In order to bring the voter to do the verification required by Belenios-CaI, we ask them to perform the computation instead of just verifying it. This forces the voter to examine each line and select one of the answers. If they make a mistake, a warning indicates the issue.

Modulo 2 with symbols. The computation modulo 10 can be performed more intuitively using a simple trick: by choosing $m = 2$, computing $v + a \bmod m$ is equivalent to answering the question “Is v identical to a ?” and mapping “yes” to $b = 0$ and “no” to $b = 1$.

In order to make the task clearer and the resulting pattern easier to visualize, we replaced the numbers with symbols. A first pair of symbols was needed to represent whether a voting option is chosen or not. We used checkboxes as

illustrated in Table 2, which we will denote as *checked* and *crossed* in the following. Since the value of a is compared with v , their domain have to be the same. The symbols for b were selected to represent “yes” and “no”, denoted as *thumb-up* and *thumb-down*.

Hide vote during selection. Since voters have to select random control values, independently of their voting option, the voting option and choice are hidden during this step of the audit phase, as illustrated in Figure 2.

3.3 Audit flow

Steps. In order to make the flow as simple and linear as possible, the interface lets the voter perform the necessary operations in four steps as depicted in Figure 2. More details about this process can be found in Figure 1 of the initial paper [5].

1. (*check*) The relationship between v , a and b is verified by answering the question “Are the two symbols identical?” for each line, i.e., for each voting item.
2. (*select*) The voter randomly picks one of the control values, for each line.
3. (*save*) The control pattern resulting from the selection is saved by the voter in the form of a PDF file.
4. (*ballot box*) After casting the ballot, the voter verifies that the control pattern corresponding to their ballot is identical to the one in the *save* step.

Note that the first two steps include an action for each voting item, therefore the voting time increases significantly for an election with a larger number of candidates.

Layout. Since the goal of the audit phase is to prevent a malicious voting client from manipulating the cast ballot by making any tampering visible, the interface should have a stable layout with components moving as little as possible. Thus, the grid structure with one row per voting item and three columns (for v , a and b) remains unchanged across the three steps.

During the *select* step, the voter should pick one control value or the other independently of whether the line in question corresponds to an option voted for, therefore the first column, containing v , is grayed out (see Figure 2b). In the *save* step, the masking of the chosen voting option is kept to prevent a voter from taking a screenshot of their vote along with the control pattern, thus producing a receipt with their vote in clear. Nevertheless, the name of the voting item has to be displayed to avoid a malicious voting client from swapping two items in the interface without being detected.

3.4 Prototype

We implemented a fully functional and publicly available prototype³ of our design based on the existing Belenios project. Next to implementing the user interface of the audit phase as outlined in this section, other adaptations were made to Belenios while developing the prototype.

³ <https://gitlab.inria.fr/agoetsch/belenios-cai>


Your vote	Control value
Cheese cake <input checked="" type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>
Tiramisù <input type="checkbox"/>	<input type="checkbox"/> Are the two symbols identical? <input checked="" type="radio"/> yes <input type="radio"/> no
Blank vote <input type="checkbox"/>	<input checked="" type="checkbox"/>

(a) *check*: the voter checks the control values.

Your vote	Control pattern
<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> Randomly pick one of the two symbols
<input type="checkbox"/>	<input type="checkbox"/> <input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>

(b) *select*: the voter selects one of the control values.

Your vote	Control pattern
Cheese cake <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>
Tiramisù <input type="checkbox"/>	<input type="checkbox"/> <input checked="" type="checkbox"/>
Blank vote <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>



(c) *save*: the voter saves the resulting control pattern.

Tracking number 0mPx27/qMwguqEqvY098Mn4K9NT0PdkKU/trqxr+aL4

Cheese cake <input type="checkbox"/>	<input type="checkbox"/>
Tiramisù <input type="checkbox"/>	<input checked="" type="checkbox"/>
Blank vote <input type="checkbox"/>	<input type="checkbox"/>

(d) *ballot box*: the voter verifies their control pattern in the ballot box.

Fig. 2. Four steps of the audit phase.

- The zero-knowledge proofs needed by Belenios-CaI were added to the ballot data structure. They ensure, e.g., that the relationship $v + a \equiv b \pmod 2$ holds for the encryption of each voting item.
- To improve the usability of the system, the instruction to copy and save the ballot tracking number manually was replaced by the download of a PDF document containing the tracking number. The control pattern can be downloaded in a similar way. The voter should check that the downloaded documents match what is displayed by the voting client.
- The static ballot box page was improved to include a search function, only displaying the ballots with a tracking number including characters entered by the user. The control pattern was also added along its ballot.

4 Experimental setting

4.1 Participants

We organized an experiment inviting the employees working in our French computer science research center (Loria and Inria), to vote using Belenios. We discuss the potential biases and their effect in Section 6.

The median age of the 129 participants who completed the experiment and fully filled the survey is 40 years old, 32% of them identified as a woman and 67% as a man. More details can be found in Figure 6 in appendix. 11%, denoted as “admin staff”, are not working in the field of computer science, while the other participants are researchers in computer science (permanent, postdoc or PhD students).

Participants had to vote using either our prototype Belenios-CaI (denoted by `cai`) or the original software Belenios-base (denoted by `base`). The assignment to a group was made by alternating `cai` and `base` when participants connected to the voting system (i.e. in a round-robin manner).

4.2 Methods

The experiment was conducted online and lasted for a week.

Participants task. Participants were contacted by email. They were proposed to first vote online for their favorite dessert among 5 choices from the local canteen. This voting question, asked with the consent of the pastry chef, satisfied both requirements of low impact (no focus on the integrity in this context), and of high motivation (to incentivize more participants). The invitation message included a short description of the data collected for the experiment, together with a link to a web page with a more detailed data management policy. Participants were told that the experiment was aiming at evaluating usability but they were not told that two versions were running.

When connecting to the voting system, participants are asked for their personal credential (included in the invitation email). Then they select their voting choice. After reviewing their choice, the participants authenticate themselves using a short code received by email. Lastly, if they are assigned Belenios-CaI, they have to audit their encrypted ballot as described in Section 3.3. If assigned to Belenios-base, they are directed immediately to the “success” page.

Then they were asked to answer a survey. This survey was mentioned in the invitation email and voters were reminded about the survey once they had voted.

Tools. In order to understand the behavior of the participants and identify obstacles in their journey, the voting system was modified to record the time spent on each step of the process as well as the number of mistakes made in the process and the number of clicks on specific elements of the interface. It was also modified in order to assign either Belenios-CaI and Belenios-base in a round-robin manner.

Since the election organized for the experiment had no impact and the results were not used, this version sacrifices privacy to understand the behavior of voters.

In particular, the private decryption key was owned by our voting server in order to later decrypt the individual ballots. This was needed to measure the correlation between audit patterns and the voting choice. To keep the experiment anonymous, participants were identified with their voting credential only, letting us link data from the voting system with their vote and the survey results without keeping track of their vote identifiers (email address).

Survey. Available in French or English and hosted on a local instance of LimeSurvey, the survey was split in three sections. The first one contained personal questions about the participant, the second the questions to obtain a System Usability Score (SUS) [4] and the third some further questions about the voting process. SUS takes the form of a 10-question survey on a 5-level Likert scale and results in a usability score on a scale of 0-100. We selected SUS for its simplicity and robustness [2].

The invitation email and the survey are provided in Appendix B.

4.3 Data collection and analysis

Since the research questions address different aspects of the practicality of Belenios-CaI, we describe separately the data collected to address them.

Q1 - Usability. Our goal is to measure if the addition of an audit phase in Belenios-CaI negatively impacts its usability compared to Belenios-base. We consider three main aspects to compare the usability of Belenios-CaI w.r.t. Belenios-base:

- effectiveness: we first record the success rate of the `cai` and `base` groups, i.e. whether a voter manages to cast her ballot;
- satisfaction: we compute the SUS scores of both groups resulting from the survey;
- efficiency: we record the time spent on each voting step and the number of mis-clicks to understand where the voters struggled during the process. To focus on the voting interface we denote *normalized voting time* the time spent in the voting booth without the authentication step, where the voter has to enter a validation code sent by email.

In each case, we compare the measure (success rate, SUS score, or voting time) between `cai` and `base`, to see if they differ significantly. For this, we compute the mean value and the standard error, and check if the intervals overlap.

Q2 - Secrecy. In order to investigate whether the control pattern reveals some information about the vote (Q2), ballots were individually decrypted at the end of the experiment and paired with their corresponding control pattern.

We investigate three possible correlations; for each of them, we performed a Chi-squared independence test:

- Do voters prefer to select the right control value (*b*) on items they voted for?
- Do voters select *a* more easily when the symbol is *checked* on non-voted items?

- Do voters select more easily a positive symbol (*thumb-up* or *checked*) on items they voted for?

Note that in our interface, the symbols of control values for a voted item are always either both positive or both negative (see the right part of Table 2), thus we do not expect a revealing behavior on voted items.

Q3 - Verifiability. The main desired property of the system is the verifiability of cast-as-intended: a ballot should contain the vote intended by the voter, and any manipulation attempt by an adversary controlling voting clients should be detected with high probability. Since Belenios-CaI relies on the randomness provided by the voters to ensure this property, we evaluate whether the data collected during our experiment supports this hypothesis. The set of observed control masks was the only data needed to perform this analysis.

If one mask was significantly more frequent, e.g. if voters would select control values on the right in 99% of cases, an adversary could modify a ballot without being detected with high probability by adapting the control values on the left.

Strategy of an adversary. To modify a voting item without being detected, an adversary has to adapt the control value which will not be selected by the voter during the *select* step. In the case of an election where voters choose one among k options, modifying a ballot only requires changing 2 voting items, the one chosen by the voter and the one chosen by the adversary. Therefore, the adversary needs to predict the left-or-right selection on these 2 lines.

Adversary success rate. Assuming that the adversary knows the distribution of control masks, they know in particular the distribution of the sub-mask composed of the 2 lines they need to modify D . We call *peak* the most frequent sub-mask and $p = D(\textit{peak})$ its frequency. Since the adversary succeeds when the voter selects the predicted mask, their success rate when attempting to modify the same two lines of m ballots is $p_{\text{adv}} = p^m$.

For example, let us consider an attacker attempting to manipulate the ballots of $m = 10$ voters who are voting for the first candidate by creating ballots for the second candidate instead. Further, we assume that the sub-mask distribution for the first two voting items is $D(0,0) = 0.1$, $D(0,1) = 0.1$, $D(1,0) = 0.7$, $D(1,1) = 0.1$. In this case we have $\textit{peak} = 1,0$ and $p = 0.7$, which yields $p_{\text{adv}} = p^m = 0.7^{10} = 0.028$.

We observe that the success rate of the adversary decreases exponentially with the number of ballots they attempt to manipulate, which indicates that Belenios-CaI is effective mostly in large elections, since in that case an adversary needs to modify proportionally more ballots to impact the result.

Assessing observed distribution. Given the actual distribution of masks observed during our experiment, we compute the pair of voting items with the highest maximal frequency, which indicates the most important weakness in terms of security. We call this observed peak in the mask distribution p_{obs} . To evaluate whether the experiment provides enough evidence that the frequency of *peak* is at

most a chosen acceptable value p_{acc} , we perform a statistical test where the null hypothesis H_0 is defined as “ $p \geq p_{acc}$ ”, in order to determine whether H_0 can be rejected with a significance level $\alpha = 0.01$. We consider the number of occurrences of the *peak* mask and model its distribution as a binomially distributed random variable $X \sim B(n, p_{acc})$, where n is the size of the group of our experiment.

5 Results

5.1 Participants’ group

An invitation email was sent to 880 employees. It contained a link to the voting system, a personal voting credential, a link to the survey and information about data privacy in the experiment. Among them, 178 started voting and 165 could cast a ballot, as illustrated in Figure 3. 138 participants completed the survey, but 6 of them did not enter their voting credentials when asked, which prevented us from linking their answer to their ballot. This leaves 129 participants who successfully voted and completed the survey.

Groups. A group of participants (*cai*) was assigned to the Belenios-CaI system while the other part (*base*) used the original system without cast-as-intended. The *base* group contains 71 participants while *cai* only 58. The difference is due to a weakness of our round-robin assignment: when a person connected to the voting system and was assigned to *cai* but did not complete their vote, their

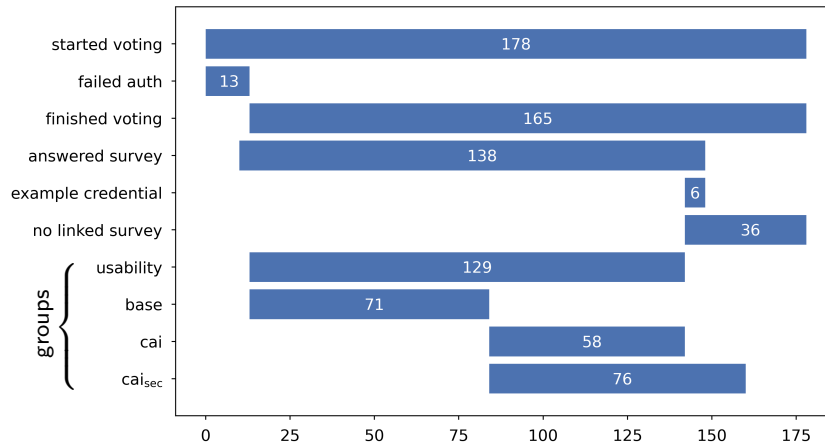


Fig. 3. Number of participants in the different groups. Some participants could not authenticate during the voting process. Among the ones who filled the survey, some used the credential given as example, which prevented us from linking their survey answer to their vote. The *usability* group was used to address Q1, with *cai* denoting the participants assigned to Belenios-CaI and *base* the ones assigned to Belenios-base. The group *cai_{sec}* denotes the participants assigned to Belenios-CaI even if they did not answer the survey. It is used to address the security-oriented questions Q2 and Q3.

participation was not recorded as exploitable but resulted in two consecutive participants assigned to *base*.

To address the first research question Q1, we used the data of the participants in *base* and *cai*, forming the usability group. In the case of Q2 and Q3, since these research questions address security properties of Belenios-CaI and are not related to the survey, we considered all participants who used the Belenios-CaI system, also including those who did not fill the survey. We designate this group as *cai_{sec}*, consisting of 76 participants (see Figure 3).

5.2 Q1 - Usability

Success rate. First, we observe that all of the 76 participants which were assigned to group *cai* and managed to authenticate could successfully conclude the audit phase and cast their ballot. This indicates that the usability of the audit step was sufficient to let every participant complete it. More details about the number of voters and their progress can be found in Table 4 of Appendix C.

SUS score. The SUS score seems to differ between the *cai* and the *base* groups, with a higher score for *base* (mean value of 78.45, std. error of 1.61) than for *cai* (mean of 72.07, std. error of 1.82), hence the intervals do not overlap. Figure 4a illustrates in a more qualitative way this difference, showing the median and the quartiles in both cases. We observe that the median score of both groups can be classified as “good” according to [2]. Figure 4b displays the average score to each of the questions in the SUS survey for both groups, showing that the difference is distributed over the different aspects measured by SUS.

Voting time. Similarly, the *normalized voting time* (see Section 4.3) of *cai* is higher than *base*. The mean value for *cai* is 113.2 sec. (std. error 8.16) and for *base* it is 29.9 sec. (std. error of 3.86). Figure 5 complements this data with the median values and the shape of the distributions in both cases.

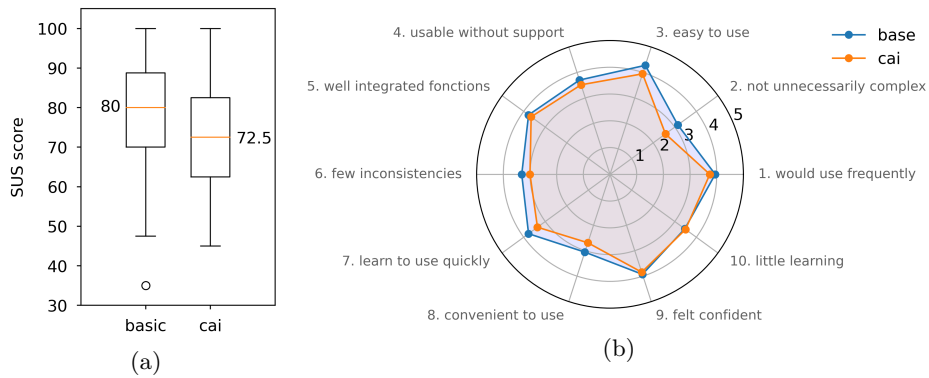


Fig. 4. On the left, comparison between SUS scores of *base* and *cai* groups. On the right the score of each question displayed separately; for the “negative” questions 2, 4, 6, 8, 10, we displayed the reverted score, so that on this picture, higher is always better.

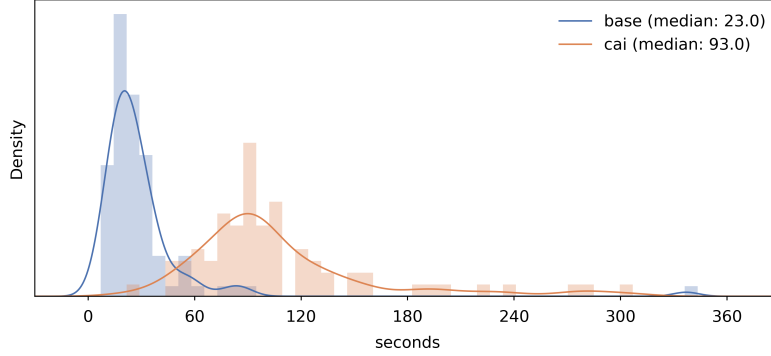


Fig. 5. Normalized voting time distribution of **base** and **cai** groups.

The difference between the voting times of both groups is explained integrally by the time on the audit phase (median: 71 s). Although this additional time is important, the overall voting time remains under 2 minutes, a time still acceptable to complete a voting process.

5.3 Q2 - Secrecy

The control patterns selected by the voters are published in the ballot box next to their ballot tracker. Thus, any dependence between a vote and its corresponding control pattern is a threat to vote secrecy. We investigate here whether the data collected during our experiment contains evidence of such a dependence.

Data. Each vote is composed of one item per possible voting option; each item is a triple (v, a, b) , where only one of a and b is displayed on the board. This results in a total of 380 items ($= 5 \cdot |\text{cai}_{\text{sec}}|$, for the 5 voting options of each ballot and a population $|\text{cai}_{\text{sec}}| = 76$). As detailed in Section 3, v and a are represented in the voter interface as *checked* or *crossed* (for 1 and 0, respectively), and b takes the shape of *thumb-up* or *thumb-down*. Note that we consider each item of the ballots separately, distinguishing between 76 *voted items* (corresponding to the chosen option) and 304 *non-voted items*.

Independence tests. We performed the three hypothesis tests listed in Section 4.3. The results of the Pearson's χ^2 tests are displayed in Table 3, for three pairs of variables that should be independent, otherwise an attacker could deduce some information on the vote and break privacy. In all cases, the test indicates independence.

5.4 Q3 - Verifiability

We start by aggregating the observed control masks, i.e., the left-right selection in the control patterns. The mask with the highest frequency is 1, 1, 1, 1, 1, present in 10 out of $|\text{cai}_{\text{sec}}| = 76$ ballots, i.e., in 13.2% of cases. This observation suggests

Table 3. Results of Pearson’s χ^2 independence tests, for three pairs of variables, where correlation could lead to a privacy leak. For each line, the number of choices for each variable is 2, therefore the degree of freedom is 1, and the reference value for χ^2 at a significance level of 0.05 is 3.84. Any computed value less than 3.84 means that the variables are independent.

In the third line, we define $positive = a \cdot \overline{mask} + \bar{b} \cdot mask$, which has value 1 when the selected control value is *checked* or *thumb-up* and 0 otherwise.

Question	Variables	χ^2	indep.?
Do voters prefer to select the right control value (b) on items she voted for?	$v, mask$	0.024	yes
Do voters select a more easily when the symbol is <i>checked</i> on non-voted items?	$mask, a$	0.049	yes
Do voters select more easily a positive symbol (thumb-up or check mark) on items she voted for?	$v, positive$	0.52	yes

that the distribution is not too skewed and that cast-as-intended can be ensured. For completeness, the distribution of control masks observed in our experiment is plotted in Figure 8 of Appendix C.

We follow the method proposed in Section 4.3 to assess the observed distribution and decide whether this leads to an acceptable advantage for an attacker. Among the $\binom{5}{2} = 10$ combinations, it turns out to be the first two voting items with $peak = 0, 0$ give the highest frequency and $p_{obs} = \frac{31}{76} = 0.41$. This value is the result of our experiment with a sample size of $|cai_{sec}| = 76$.

This 0.41 value is well below what we decided to be an acceptable value $p_{acc} = 0.7$. For the sake of completeness, we computed the probability that we observed a peak at 0.41, if the real distribution includes a peak with 0.7. This probability is $1.25 \cdot 10^{-7}$ (details are given in Appendix C) and confirms that, based on our observation, an adversary will gain only an acceptable advantage regarding the verifiability property.

5.5 Feedback given in the free text questions

In the third part of the survey, participants were given the opportunity to provide comments on various aspects and we thank them for their suggestions.

Regarding security, many participants say they trust the system, but they also acknowledge that this is mostly due to the fact that they know the authors of this experiment and trust their expertise. They do not understand how the various steps in the voter’s journey have an impact on the security. Some participants are more skeptical and seem to be reluctant against Internet voting in general.

Feedback about usability is mostly positive, in the sense that many participants consider that the number of steps is high but that this is acceptable if justified for security. However, many also mention that not understanding the precise reasons for these steps generates some frustration. Among the difficulties mentioned by the participants, going back and forth between the mailbox and the browser is

mentioned several times as a problem. Some also say that they managed to use the system but that they believe it might be difficult for others (elders, in particular). Some remarks were specifically related to the cast-as-intended functionality, and one participant explicitly mentioned the hesitation when having to randomly choose the mask.

More generally, feedback from participants confirms that Belenios-CaI does not seem to be hindering the voting system, but it does offer some ideas for improving the user experience and acceptance of the online voting system.

6 Discussion

Considering the cost of ensuring cast-as-intended in perceived usability and in voting time, we observe that it might not be worth the additional security in the case of small scale elections since a few modifications may still happen with realistic probability. However, for large elections, the protection against malicious voting clients can make the traded usability acceptable. Indeed, it is unlikely that an attacker can modify sufficiently many votes without being detected.

Limitations. The recruitment of participants was performed in the research center where the authors work. Therefore, most participants had at least a Master in Computer Science. They may have a different understanding of how to “randomly” select the control values, as compared to the general population. Furthermore, although participation was anonymous, the fact that many participants know the authors of the study might have introduced a bias in the responses to the survey. But since we are interested in the comparison between Belenios and Belenios-CaI and not in a system alone, the possible bias should be equivalent for both systems and not affect the difference in perceived usability.

The main goal of the study was to determine whether Belenios-CaI is an acceptable evolution of Belenios. It does not evaluate whether the underlying cast-as-intended mechanism is more usable than others such as Benaloh’s challenge [3] or return codes [7, 17].

Future work. The analysis regarding secrecy and verifiability performed in Section 5.3 only provides insight about settings similar to our experiment. To make stronger statements concerning the impact of voter-generated randomness on vote privacy, a study on a larger population sample, more representative of the general population, would be necessary.

Another interesting aspect would be the evaluation of verification efficiency, i.e., whether Belenios-CaI allows voters to detect a manipulation, and a comparison with other systems providing cast-as-intended verifiability.

Acknowledgments. We are very grateful to Stéphane Glondu for his help with Belenios implementation.

References

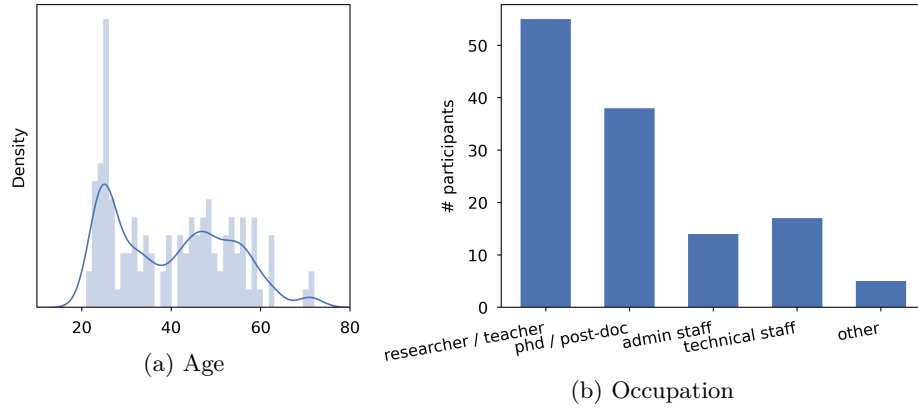
1. Google Fonts: Material Icons. <https://fonts.google.com/icons> (2024), accessed: 2024-03-15

2. Bangor, A., Kortum, P.T., Miller, J.T.: An empirical evaluation of the system usability scale. *Intl. Journal of Human-Computer Interaction* **24**(6), 574–594 (2008), <https://doi.org/10.1080/10447310802205776>
3. Benaloh, J.: Simple verifiable elections. In: *Electronic Voting Technology Workshop, Usenix* (2006)
4. Brook, J.: SUS: a “quick and dirty” usability scale. *Usability evaluation in industry* (1996)
5. Cortier, V., Debant, A., Gaudry, P., Glondu, S.: Belenios with cast as intended. In: *VOTING (FC workshop)*, Springer (2023), https://doi.org/10.1007/978-3-031-48806-1_1
6. Cortier, V., Gaudry, P., Glondu, S.: Belenios: a simple private and verifiable electronic voting system. *Foundations of Security, Protocols, and Equational Reasoning: Essays Dedicated to Catherine A. Meadows* pp. 214–238 (2019), https://doi.org/10.1007/978-3-030-19052-1_14
7. Haenni, R., Koenig, R.E., Locher, P., Dubuis, E.: CHVote Protocol Specification. *Cryptology ePrint Archive* (2017), <https://eprint.iacr.org/2017/325>
8. Heiberg, S., Martens, T., Vinkel, P., Willemson, J.: Improving the verifiability of the Estonian Internet voting scheme. In: *EvotID*, Springer (2017), https://doi.org/10.1007/978-3-319-52240-1_6
9. Hilt, T., Berens, B., Truderung, T., Udovychenko, M., Neumann, S., Volkamer, M.: Systematic user evaluation of a second device based cast-as-intended verifiability approach. In: *VOTING (FC workshop)*, Springer (2024)
10. Küsters, R., Müller, J., Scapin, E., Truderung, T.: sElect: A lightweight verifiable remote voting system. In: *CSF, IEEE* (2016), <https://doi.org/10.1109/CSF.2016.31>
11. Lelkes, Y., Krosnick, J.A., Marx, D., Judd, C.M., Park, B.: Complete anonymity compromises the accuracy of self-reports. *Journal of Experimental Social Psychology* **48**(6), 1291–1299 (2012), <https://doi.org/https://doi.org/10.1016/j.jesp.2012.07.002>
12. Marky, K., Gerber, N., Krumb, H., Khamis, M., Mühlhäuser, M.: Investigating voter perceptions of printed physical audit trails for online voting. In: *S&P, IEEE* (2024), <https://doi.org/10.1109/SP54263.2024.00136>
13. Marky, K., Kulyk, O., Renaud, K., Volkamer, M.: What Did I Really Vote For? On the Usability of Verifiable E-Voting Schemes. In: *CHI, ACM* (2018), <https://doi.org/10.1145/3173574.3173750>
14. Müller, J., Truderung, T.: Caused: A protocol for cast-as-intended verifiability with a second device. In: *EvotID*, Springer (2023), https://doi.org/10.1007/978-3-031-43756-4_8
15. Ryan, P., Roenne, P., Rastikian, S.: Hyperion: An enhanced version of the Selene end-to-end verifiable voting scheme. In: *EvotID* (2021)
16. Ryan, P., Rønne, P., Iovino, V.: Selene: Voting with transparent verifiability and coercion-mitigation. In: *VOTING (FC workshop)*, Springer (2016), https://doi.org/10.1007/978-3-662-53357-4_12
17. Swiss Post: Swiss Post Voting System: System Specification. Version 1.1.1. Tech. rep., Swiss Post (October 2022)
18. Volkamer, M., Kulyk, O., Ludwig, J., Fuhrberg, N.: Increasing security without decreasing usability: A comparison of various verifiable voting systems. In: *SOUPS, Usenix* (2022)

Appendix: Supplementary Material

A Characteristics of the participants population

We note that the participants were asked to first vote and then fill the survey, and some of them did not complete this second step. Thus, the characteristics of the population, provided in Figure 6, are only known for the part which answered the survey.



	usability (129)	base (71)	cai (58)
Age	Q1: 26.0	Q1: 27.0	Q1: 26.0
	Q2: 40.0	Q2: 38.0	Q2: 41.0
	Q3: 51.0	Q3: 50.5	Q3: 50.75
Gender	F: 42	F: 21	F: 21
	M: 86	M: 49	M: 37
	N/A: 1	N/A: 1	N/A: 0
Occupation	researcher/teacher: 55	researcher/teacher: 26	researcher/teacher: 29
	phd/post-doc: 38	phd/post-doc: 23	phd/post-doc: 15
	admin staff: 14	admin staff: 7	admin staff: 7
	tech staff: 17	tech staff: 14	tech staff: 3
	other: 5	other: 1	other: 4

(c) Characteristics per group. Q1, Q2 and Q3 stand for quartiles.

Fig. 6. Age, gender and occupation distribution of the participants.

B Material provided to voters

We provide here the invitation email sent to voters and the three page survey, which voters had to complete once they voted.

B.1 Invitation email

(slightly modified for anonymity)

Hello everyone!

We are asking for your help to test a possible evolution of the online voting system Belenios. The experiment is open until November 12th and done entirely online, from your computer and whenever you want. The estimated total time is 10-15 minutes and we need around 100 volunteers to get relevant results.

=== Participate now! ===

Your voting credential: <CREDENTIAL>

Link to the voting system: <VOTING_LINK>

The questionnaire to fill after voting: <QUESTIONNAIRE>

=== More information ===

In order to make the experiment more interesting, we propose to choose your favorite dessert among a selection of Isabelle's specialities:

- Mousse au chocolat
- Éclair meringué au citron
- Cheese cake
- Tiramisù

After voting, we kindly ask you to answer to some questions in order to evaluate the usability of our system. This should not take more than 5 minutes.

If you would like to go further, you can then head to room B231, where a michoko will be waiting for you. This will also be an opportunity to chat and find out more about the experiment or Belenios in general. It's also okay to just pass by to get your michoko as a reward for participating!

Belenios (<https://www.belenios.org>) is an online voting system developed over the last ten years by the LORIA laboratory, in connection with research about security properties such as verifiability and secrecy of the vote.

The version set up for this experiment is instrumented (behavior and response times) to allow us to evaluate a new functionality, so some security properties are not guaranteed to the same level as in the

system deployed for real elections. Details about personal data protection can be found here: <DATA_POLICY>.

Thanks in advance for your participation!

<Anonymized>

B.2 Survey

Personal information

1. What is your voting credential?
This credential can be found in the invitation email. Example: 3ac-xUs-eCn-xzy-6cF
2. Age
3. Gender
 - Male
 - Female
 - Other
 - No answer
4. Are you color-blind?
 - Yes
 - No
 - No answer
5. What is your occupation?
 - Researcher / teacher / professor
 - Doctoral student / post-doc / student
 - Administrative staff
 - Technical staff
 - Other
6. Usage of IT systems
Please select the option that suits you best: never, once a week, multiple times per week, once a day, more than once a day.
 - I search for information on internet.
 - I use a computer.
 - I use a smartphone or a tablet.
7. Usage of voting systems
Please select the option that suits you best: never, once, more than once, once a year, more than once a year.
 - I have already taken part in elections at a polling station.
 - I have already voted by post.
 - I have already voted online.

Usability Please select the option that suits you best: strongly disagree, disagree, neither agree nor disagree, agree, strongly agree.

Belenios is the voting system used during the experiment

1. I think that I would like to use Belenios frequently.
2. I found Belenios unnecessarily complex.
3. I thought Belenios was easy to use.
4. I think that I would need the support of a technical person to be able to use Belenios.
5. I found the various functions in Belenios were well integrated.
6. I thought there was too much inconsistency in Belenios.
7. I would imagine that most people would learn to use Belenios very quickly.
8. I found Belenios very cumbersome to use.
9. I felt very confident using Belenios.
10. I needed to learn a lot of things before I could get going with Belenios.

Voting process

1. I What type of device did you use to vote?
 - Computer
 - Smartphone
 - Tablet
 - Other
2. Did you manage to vote?
 - Yes
 - No

Please enter your comment here
3. How did you save your tracking number?

Example of tracking number: jXPQiFcKCwIrMdgL04S3IjhhPhzr4P9+MUktNCV031k

 - Download the PDF
 - Picture or screenshot
 - On a paper
 - Not saved
 - Other
4. Did you verify that your ballot is in the ballot box?
 - Yes
 - No
 - No but I will do it
 - Other
5. Did you see a control pattern while voting?
 - Yes
 - No
6. How did you save your control pattern?
 - Download the PDF
 - Picture or screenshot
 - On a paper
 - Not saved

- Other
- 7. Is the control pattern in the ballot box identical to the one you selected?
 - Yes
 - No
 - I don't know
 - Other
- 8. Do you think that Belenios protects the secrecy of your vote?
 - Yes
 - No
 Please enter your comment here
- 9. Do you think that Belenios prevents the manipulation of the voting result?
 - Yes
 - No
 Please enter your comment here
- 10. Do you think that using Belenios is too complex? In which part and to what extent?
- 11. Do you have other remarks or comments about the experiment?

C Raw data and additional statistics

C.1 Usability data

The participants' success rate for each phase is given in Table 4. Note that it raises a potential problem in the authentication phase: 4 voters in the `base` group and 9 in `cai` could not authenticate by entering a code received by email. But since this is independent of the audit phase (it occurs before, in the voter's journey), we do not investigate the issue here.

Table 4. Number of participants who completed voting phases. The participants had to enter their credentials, chose a voting option and authenticate with a code sent by email. The ones in the `cai` group additionally completed the audit phase to finalize their vote.

voting step	base	cai
entered credentials	93	86
entered voting choice	93	85
authenticated	89	76
completed audit		76
voted	89	76
filled survey	71	58

The raw data for the SUS score and of the global voting time that yielded the statistics of Section 5.2 are the following.

```
SUS_base = [ 67.5, 100, 72.5, 67.5, 95, 95, 97.5, 70, 80, 60, 85,
92.5, 97.5, 80, 97.5, 47.5, 62.5, 80, 85, 87.5, 80, 87.5, 95, 77.5,
77.5, 87.5, 90, 87.5, 60, 82.5, 90, 90, 67.5, 77.5, 50, 85, 67.5, 85,
80, 90, 90, 35, 75, 85, 80, 72.5, 95, 65, 72.5, 82.5, 75, 70, 60, 95,
80, 57.5, 70, 90, 77.5, 85, 95, 70, 67.5, 57.5, 65, 65, 70, 100,
77.5, 87.5, 75 ]
```

```
SUS_cai = [ 45, 60, 100, 85, 80, 90, 62.5, 77.5, 45, 70, 62.5, 80,
47.5, 45, 72.5, 77.5, 85, 90, 85, 72.5, 62.5, 90, 72.5, 87.5, 65, 70,
60, 72.5, 82.5, 75, 82.5, 62.5, 57.5, 97.5, 92.5, 62.5, 75, 75, 67.5,
57.5, 70, 82.5, 67.5, 52.5, 95, 62.5, 70, 85, 82.5, 85, 52.5, 55,
82.5, 72.5, 67.5, 62.5, 52.5, 82.5 ]
```

```
Vote_time_base = [ 8.0, 19.0, 11.0, 10.0, 30.0, 8.0, 31.0, 27.0,
20.0, 36.0, 32.0, 53.0, 24.0, 28.0, 20.0, 20.0, 78.0, 14.0, 34.0,
33.0, 17.0, 18.0, 25.0, 22.0, 21.0, 51.0, 62.0, 84.0, 28.0, 38.0,
19.0, 7.0, 16.0, 16.0, 24.0, 9.0, 29.0, 18.0, 18.0, 15.0, 20.0, 43.0,
12.0, 337.0, 21.0, 45.0, 36.0, 21.0, 17.0, 24.0, 25.0, 25.0, 26.0,
57.0, 22.0, 39.0, 24.0, 14.0, 23.0, 12.0, 25.0, 34.0, 24.0, 57.0,
19.0, 15.0, 15.0, 15.0, 30.0, 17.0, 37.0, 16.0, 25.0, 16.0, 10.0,
23.0, 10.0, 36.0, 21.0, 24.0, 34.0, 28.0, 16.0, 17.0, 88.0, 16.0,
31.0, 11.0, 32.0 ]
```

```
Vote_time_cai = [ 96.0, 104.0, 106.0, 79.0, 97.0, 65.0, 90.0, 108.0,
60.0, 155.0, 107.0, 90.0, 199.0, 70.0, 108.0, 92.0, 519.0, 130.0,
62.0, 88.0, 99.0, 73.0, 53.0, 49.0, 92.0, 99.0, 304.0, 62.0, 157.0,
56.0, 122.0, 91.0, 75.0, 136.0, 92.0, 83.0, 79.0, 124.0, 92.0, 97.0,
66.0, 282.0, 222.0, 236.0, 106.0, 45.0, 92.0, 146.0, 91.0, 82.0,
73.0, 45.0, 80.0, 82.0, 128.0, 88.0, 81.0, 84.0, 122.0, 128.0, 94.0,
94.0, 132.0, 105.0, 109.0, 86.0, 58.0, 191.0, 119.0, 186.0, 23.0,
98.0, 68.0, 148.0, 274.0, 80.0 ]
```

In Figure 7, we give the breakdown of time spent in the audit phase.

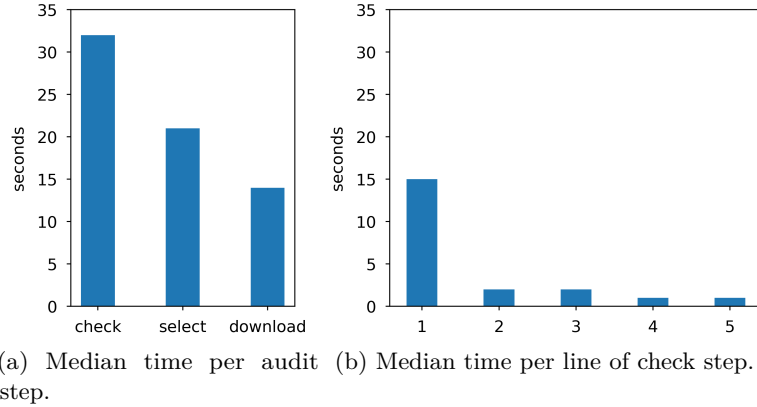
C.2 Secrecy data

The contingency tables for the three χ^2 computations of Table 3 are the following.

- Does the voter prefer to select the right control value (*b*) on items she voted for?

v\mask	left	right	total
yes	34	42	76
no	139	165	307
total	173	207	380

- Does the voter select *a* more easily when the symbol is *checked* on non-voted items?

**Fig. 7.** Time of audit steps.

mask\ a	checked	cross	total
left	70	69	139
right	81	84	165
total	151	153	304

- Does the voter select more easily a positive symbol (*thumb-up* or *checked*) on items she voted for?

v\ positive	yes	no	total
yes	35	41	76
no	154	150	304
total	189	191	380

C.3 Verifiability data

The observed distribution of the control masks is given in Figure 8.

We perform a statistical test to evaluate whether the experiment provides enough evidence that the frequency of *peak* is at most a chosen acceptable value $p_{\text{acc}} = 0.7$.

Recall from Section 4.3 that the null hypothesis H_0 is defined as “ $p \geq p_{\text{acc}}$ ”. We consider the number of occurrences of the *peak* mask and model its distribution as a binomially distributed random variable $X \sim B(n, p_{\text{acc}})$, where $n = 76$ is the size of the group of our experiment.

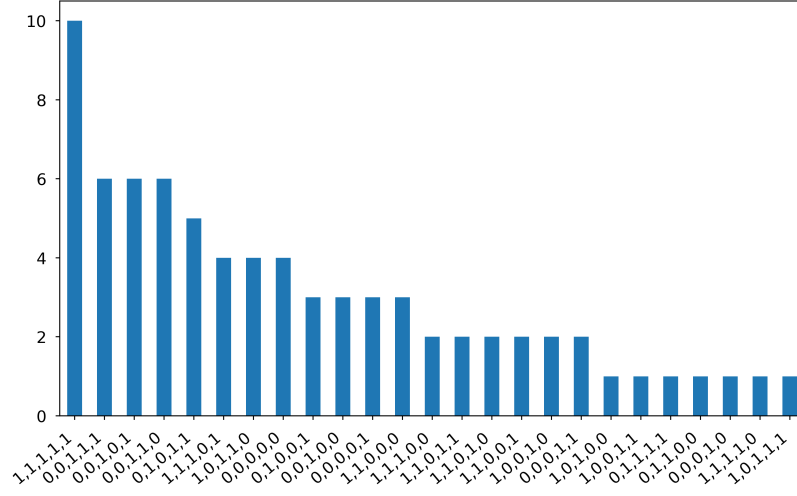


Fig. 8. Distribution of the sample of 76 control masks. Each digit corresponds to one of the five voting items, 0 is a selection of the left control value and 1 of the right one.

We then compute the probability of X being at most $occ_{peak} = 31$.

$$\begin{aligned}
 P(X \leq occ_{peak}) &= \sum_{0 \leq k \leq occ_{peak}} P(X = k) \\
 &= \sum_{0 \leq k \leq occ_{peak}} \binom{n}{k} p^k (1-p)^{n-k} \\
 &= 1.25 \cdot 10^{-7}
 \end{aligned}$$

Since $1.25 \cdot 10^{-7}$ is below the significance level α we reject H_0 , meaning we have enough evidence suggesting that $p < 0.7$. Thus, the success rate of an adversary modifying m ballots can be considered to be at most 0.7^m .