Polling with Physical Envelopes: A Rigorous Analysis of a Human-Centric Protocol*

Tal Moran[1] and Moni Naor[1]**

Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel

Abstract. We propose simple, realistic protocols for polling that allow the responder to plausibly repudiate his response, while at the same time allow accurate statistical analysis of poll results. The protocols use simple physical objects (envelopes or scratch-off cards) and can be performed without the aid of computers. One of the main innovations of this work is the use of techniques from theoretical cryptography to rigorously prove the security of a realistic, physical protocol. We show that, given a few properties of physical envelopes, the protocols are unconditionally secure in the universal composability framework.

1 Introduction

In the past few years, a lot of attention has been given to the design and analysis of electronic voting schemes. Constructing a protocol that meets all (or even most) of the criteria expected from a voting scheme is generally considered to be a tough problem. The complexity of current protocols (in terms of how difficult it is to describe the protocol to a layperson) reflects this fact. A slightly easier problem, which has not been investigated as extensively, is that of polling schemes.

Polling schemes are closely related to voting, but usually have slightly less exacting requirements. In a polling scheme the purpose of the pollster is to get a good statistical profile of the responses, however some degree of error is admissible. Unlike voting, absolute secrecy is generally not a requirement for polling, but some degree of response privacy is often necessary to ensure respondents' cooperation.

The issue of privacy arises because polls often contain questions whose answers may be incriminating or stigmatizing (e.g., questions on immigration status, drug use, religion or political beliefs). Even if promised that the results of the poll will be used anonymously, the accuracy of the poll is strongly linked to the trust responders place in the pollster. A useful rule of thumb for polling sensitive questions is "better privacy implies better data": the more respondents trust that their responses cannot be used against them, the likelier they are to

* This work was partially supported by the Minerva Foundation

** Incumbent of the Judith Kleeman Professorial Chair

answer truthfully. Using polling techniques that clearly give privacy guarantees can significantly increase the accuracy of a poll.

A well-known method for use in these situations is the "randomized response technique" (RRT), introduced by Warner [25]. Roughly, Warner's idea was to tell responders to lie with some fixed, predetermined, probability (e.g., roll a die and lie whenever the die shows one or two). As the probability of a truthful result is known exactly, statistical analysis of the results is still possible[1], but an individual answer is always plausibly deniable (the respondent can always claim the die came up one).

Unfortunately, in some cases this method causes its own problems. In pre-election polls, for example, responders have a strong incentive to always tell the truth, ignoring the die (since the results of the polls are believed to affect the outcome of the elections). In this case, the statistical analysis will give the cheating responders more weight than the honest responders. Ambainis, Jakobsson and Lipmaa [1] proposed the "Cryptographic Randomized Response Technique" to deal with this problem. Their paper contains a number of different protocols that prevent malicious responders from biasing the results of the poll while preserving the deniability of the randomized response protocol. Unlike Warner's original RRT, however, the CRRT protocols are too complex to be implemented in practice without the aid of computers. Since the main problem with polling is the responders' lack of trust in the pollsters, this limitation makes the protocols of [1] unsuitable in most instances.

The problem of trust in complex protocols is not a new one, and actually exists on two levels. The first is that the protocol itself may be hard to understand, and its security may not be evident to the layman (even though it may be formally proved). The second is that the computers and operating system actually implementing the protocol may not be trusted (even though the protocol itself is). This problem is more acute than the first. Even for an expert, it is very difficult to verify that a computer implementation of a complex protocol is correct.

Ideally, we would like to design protocols that are simple enough to grasp intuitively and can also be implemented transparently (so that the user can follow the steps and verify that they are correct).

1.1 Our Results

In this paper we propose two very simple protocols for cryptographic randomized response polls, based on *tamper-evident seals* (introduced in a previous paper by the authors [18]). A tamper-evident seal is a cryptographic primitive that

[1] For instance, suppose $p > \frac{1}{2}$ is the probability of a truthful response, n is the total number of responses, x is the number of responders who actually belong in the "yes" category and R is the random variable counting the number of "yes" responses. R is the sum of n independent indicator random varables, so R is a good estimation for $E(R) = px + (1-p)(n-x) = x(2p-1) + n(1-p)$. Therefore, given R, we can accurately estimate the actual number of "yes" responders: $x = \frac{E(R)-n(1-p)}{2p-1}$.

captures the properties of a sealed envelope: while the envelope is sealed, it is impossible to tell what's inside, but if the seal is broken the envelope cannot be resealed (so any tampering is evident). In fact, our CRRT protocols are meant to be implemented using physical envelopes (or scratch-off cards) rather than computers. Since the properties of physical envelopes are intuitively understood, even by a layman, it is easy to verify that the implementation is correct.

The second important contribution of this paper, differentiating it from previous works concerning human-implementable protocols, is that we give a formal definition and a rigorous proof of security for the protocols. The security is unconditional: it relies only on the physical tamper-evidence properties of the envelopes, not on any computational assumption. Furthermore, we show that the protocols are "universally composable" (as defined by Canetti [3]). This is a very strong notion of security that implies, via Canetti's Composition Theorem, that the security guarantees hold even under general concurrent composition

Our protocols implement a relaxed version of CRRT (called *weakly secure* in [1]). We also give an inefficient strong CRRT protocol (that requires a large number of rounds), and give impossibility results and lower bounds for strong CRRT protocols with a certain range of parameters (based on Cleve's lower bound for coin flipping [8]). These suggest that constructing a strong CRRT protocol using scratch-off cards may be difficult (or even impossible if we require a constant number of rounds).

1.2 Related Work

Randomized Response Technique. The randomized response technique for polling was first introduced in 1965 [25]. Since then many variations have been proposed (a survey can be found in [6]). Most of these are attempts to improve or change the statistical properties of the poll results (e.g., decreasing the variance), or changing the presentation of the protocol to emphasize the privacy guarantee (e.g., instead of lying, tell the responders to answer a completely unrelated question). A fairly recent example is the "Three Card Method" [14], developed for the United States Government Accountability Office (GAO) in order to estimate the size of the illegal resident population. None of these methods address the case where the responders maliciously attempt to bias the results.

To the best of our knowledge, the first polling protocol dealing explicitly with malicious bias was given by Kikuchi, Akiyama, Nakamura and Gobioff. [17], who proposed to use the protocol for voting (the protocol described is a randomized response technique, although the authors do not appear to have been aware of the previous research on the subject). Their protocol is still subject to malicious bias using a "premature halting" attack (this is equivalent to the attack on the RRT protocol in which the responder rolls a die but refuses to answer if result of the die is not to his liking). A more comprehensive treatment, as well as a formal definition of cryptographic randomized response, was given by Ambainis et al. [1]. In their paper, Ambainis et al. also give a protocol for *Strong* CRRT, in which the premature halting attack is impossible. In both the papers [17, 1], the protocols are based on cryptographic assumptions and require computers to implement.

Independently of this work, Stamm and Jakobsson show how to implement the protocol of [1] using playing cards [24]. They consider this implementation only as a visualization tool. However, if we substitute envelopes for playing cards (and add a verification step), this protocol gives a Responder-Immune protocol (having some similarities to the one described in Section 3.2).

Deniable and Receipt-Free Protocols. The issues of deniability and coercion have been extensively studied in the literature (some of the early papers in this area are [2, 22, 4, 5, 15]). There are a number of different definitions of what it means for a protocol to be *deniable*. Common to all of them is that they protect against an adversary that attacks actively only *after* the protocol execution: in particular, this allows the parties to lie about their random coins. Receipt-Free protocols provide a stronger notion of security: they guarantee that even if a party is actively colluding with the adversary, the adversary should have no verifiable information about which input they used. Our notion of "plausible deniability" is weaker than both "traditional" deniability and receipt-freeness, in that we allow the adversary to gain some information about the input. However, as in receipt-freeness, we consider an adversary that is active before and during the protocol, not just afterwards.

Secure Protocols Using "Real" Objects. The idea of using real objects to provide security predates cryptography: people have been using seals, locks and envelopes for much of history. Using real objects to implement protocols that use them in non-obvious ways is a newer notion. Fagin, Naor and Winkler [16] propose protocols for comparing secret information that use various objects, from paper cups to the telephone system. In a more jocular tone, Naor, Naor and Reingold [19] propose a protocol that provides a "zero knowledge proof of knowledge" of the correct answer to the children's puzzle "Where's Waldo" using "low-tech devices" (e.g., a large newspaper and scissors). In all these works the security assumptions and definitions are informal or unstated. Crépeau and Kilian [10] show how to use a deck of cards to play "discreet" solitary games (these involve hiding information from yourself). Their model is formally defined, however it is not malicious; the solitary player is assumed to be honest but curious.

A related way of using real objects is as aids in performing a "standard" calculation. Examples in this category include Schneier's "Solitaire" cipher [23] (implemented using a pack of cards), and the "Visual Cryptography" of Naor and Shamir [21] (which uses the human visual system to perform some basic operations on images). The principles of Visual Cryptography form the basis for some more complex protocols, such as the "Visual Authentication" protocol of Naor and Pinkas [20], and Chaum's human verifiable voting system [7].

Tamper-Evident Seals. This work can be viewed as a continuation of a previous work by the authors on tamper-evident seals [18]. In [18], we studied the possibility of implementing basic cryptographic primitives using different variants of physical, tamper-evident seals. In the current work we focus on their use in realistic cryptographic applications, rather than theoretical constructs (for instance, there is a very sharp limit on the number of rounds and the number of envelopes that can be used in a protocol that we expect to be practical for humans). We limit ourselves to the "distinguishable envelope" (DE) model, as this model has a number of intuitive physical embodiments, while at the same time is powerful enough, in theory, to implement many useful protocols[2] (an informal description of this model is given in Section 2.3; for a formal definition see [18]).

Overview of Paper. In Section 2, we give formal definitions of the functionalities we would like to realize and the assumptions we make about the humans implementing the protocols. Section 3 gives an informal description of the CRRT protocols. In Section 4, we show how to amplify a weak CRRT protocol in order to construct a strong CRRT protocol, and give some impossibility results and lower bounds for strong CRRT protocols. Finally, a discussion and some open problems appear in Section 5.

The formal protocol specification and proof of security for our Pollster-Immune CRRT protocol appears in Appendix A. Due to space constraints, the complete specifications and formal proofs for the other protocols will appear only in the full version of this paper.

2 The Model

Ideal Functionalities. Many two-party functionalities are easy to implement using a trusted third party that follows pre-agreed rules. In proving that a two-party protocol is secure, we often want to say that it behaves "as if it were performed using the trusted third party". The "Universal Composability" framework, defined by Canetti [3], is a formalization of this idea. In the UC model, the trusted third party is called the *ideal functionality*. If every attack against the protocol can also be carried out against the ideal functionality, we say the protocol realizes the functionality. Canetti's Composition Theorem says that *any* protocol that is secure using the ideal functionality, will remain secure if we replace calls to the ideal functionality with executions of the protocol.

Defining the security guarantees of our protocols as ideal functionalities has an additional advantage as well: it is usually easier to understand what it means for a protocol to satisfy a definition in this form than a definition given as a list of properties. Below, we describe the properties we wish to have in a CRRT protocol, and give formal definitions in the form of ideal functionalities.

2.1 Cryptographic Randomized Response

A randomized response protocol involves two parties, a pollster and a responder. The responder has a secret input bit b (this is the true response to the poll

[2] Although the "indistinguishable envelope model" (also defined in [18]) is stronger (e.g., oblivious transfer *is* possible in this model), it seems to be very hard to devise a secure, physical realization of this functionality.

question). In the ideal case, the pollster learns a bit c, which is equal to b with probability p (p is known to the pollster) and to $1 - b$ with probability $1 - p$. Since p is known to the pollster, the distribution of responders' secret inputs can be easily estimated from the distribution of the pollster's outputs.

The essential property we require of a Randomized Response protocol is *plausible deniability*: A responder should be able to claim that, with reasonable probability, the bit learned by the pollster is not the secret bit b. This should be the case even if the pollster maliciously deviates from the protocol.

A *Cryptographic* Randomized Response protocol is a Randomized Response protocol that satisfies an additional requirement, *bounded bias*: The probability that $c = b$ must be at most p, even if the responder maliciously deviates from the protocol. The bounded bias requirement ensures that malicious responders cannot bias the results of the poll (other than by changing their own vote). Note that even in the ideal case, a responder can always choose any bias p' between p and $1-p$, by randomly choosing whether to vote b or $1-b$ (with the appropriate probability).

Strong p-CRRT In a *strong* CRRT protocol, both the deniability and bounded bias requirements are satisfied. Formally, this functionality has a single command:

Vote b The issuer of this command is the responder. On receiving this command the functionality tosses a weighted coin c, such that $c = 0$ with probability p. It then outputs $b \oplus c$ to the pollster and the adversary.

Unfortunately, we do not know how to construct a practical strong CRRT protocol that can be implemented by humans. In Section 4, we present evidence to suggest that finding such a protocol may be hard (although we do show an impractical strong CRRT protocol, that requires a large number of rounds). The protocols we propose satisfy relaxed conditions: The first protocol is immune to malicious pollsters (it is equivalent to strong CRRT if the pollster is honest), while the second is immune to malicious responders (it is equivalent to strong CRRT if the responder is honest).

Pollster-Immune p-CRRT (adapted from Weak CRRT in [1]) This is a weakened version of CRRT, where a malicious pollster cannot learn more than an honest pollster about the responder's secret bit. A malicious responder can bias the result by deviating from the protocol (halting early). A cheating responder will be caught with fixed probability, however, so the pollster can accurately estimate the number of responders who are cheating (and thus bound the resulting bias). When the pollster catches the responder cheating, it outputs $\boxtimes$ instead of its usual output. Formally, the ideal functionality accepts the following commands:

Query The issuer of this command is the pollster, the other party is the responder. The functionality ignores all commands until it receives this one.

On receiving this command the functionality chooses a uniformly random bit r and a bit v, such that $v = 1$ with probability $2p - 1$. If the responder is corrupted, the functionality then sends both bits to the adversary.

Vote b On receiving this command from the responder, the functionality checks whether $v = 1$. If so, it outputs b to the pollster, otherwise it outputs r to the pollster.

Halt This command captures the responder's ability to cheat. On receiving this command from a corrupt responder, the functionality outputs $\boxtimes$ to the pollster and halts.

The functionality is slightly more complex (and a little weaker) than would appear to be necessary, and this requires explanation. Ideally, the functionality should function as follows: the responder casts her vote, and is notified of the actual bit the pollster would receive. The responder then has the option to halt (and prevent the pollster from learning the bit). Our protocol gives the corrupt responder a little more power: the responder first learns whether the pollster will receive the bit sent by the responder, or whether the pollster will receive a bit fixed in advance (regardless of what the responder sends). The responder can then plan her actions based on this information. The functionality we describe is the one that is actually realized by our protocol (for $p = \frac{3}{4}$).

Responder-Immune *p*-CRRT In this weakened version of CRRT, malicious responders cannot bias the results more than honest responders, but a malicious pollster can learn the responder's secret bit. In this case, however, the responder will discover that the pollster is cheating. When the responder catches the pollster cheating, it outputs $\boxtimes$ to signify this. The functionality accepts the following commands:

Vote b The issuer of this command is the responder. On receiving this command the functionality tosses a weighted coin c, such that $c = 0$ with probability p. It then outputs $b \oplus c$ to the pollster and adversary.

Reveal The command may only be sent by a corrupt pollster *after* the Vote command was issued by the responder. On receiving this command, the functionality outputs b to the adversary and $\boxtimes$ to the responder.

Test x : The command may only be sent by a corrupt pollster, after the Vote command was issued by the responder. On receiving this command:
- if $x = b$, then with prob. $\frac{1}{2}$ it outputs b to the adversary and $\boxtimes$ to the responder, and with prob. $\frac{1}{2}$ it outputs $\perp$ to the adversary (and nothing to the responder).
- if $x = 1 - b$ the functionality outputs $\perp$ to the adversary (and nothing to the responder).

Ideally, we would like to realize responder-immune CRRT without the **Test** command. Our protocol realizes this slightly weaker functionality (for $p = \frac{2}{3}$). It may appear that a corrupt pollster can cheat without being detected using the **Test** command. However, for any corrupt pollster strategy, if we condition on the pollster's cheating remaining undetected, the pollster gains no additional information about the responder's choice (since in that case the response to the **Test** command is always $\perp$).

2.2 Modelling Humans

The protocols introduced in this paper are meant to be implemented by humans. To formally prove security properties of the protocols, it is important to make explicit the abilities and limitations we expect from humans.

Following Instructions. The most basic assumption we make about the parties participating in the protocol is that an honest party will be able to follow the instructions of the protocol correctly. While this requirement is clearly reasonable for computers, it may not be so easy to achieve with humans (e.g., one of the problems encountered with the original randomized response technique is that the responders sometimes had difficulty understanding what they were supposed to do). The ability to follow instructions depends on the complexity of the protocol (although this is a subjective measure, and hard to quantify). Our protocols are secure and correct only assuming the honest parties are actually following the protocol. Unfortunately, we do not know how to predict whether this assumption actually holds for a specific protocol without "real" experimental data.

Random Choice. Our protocols require the honest parties to make random choices. Choosing a truly random bit may be very difficult for a human (in fact, even physically tossing a coin has about 0.51 probability of landing on the side it started on [13]). For the purposes of our analysis, we assume that whenever we require a party to make a random choice it is uniformly random. In practice, a random choice may be implemented using simple physical means (e.g., flipping a coin or rolling a die). In practice, the slight bias introduced by physical coin flipping will not have a large effect on the correctness or privacy of our protocols.

Non-Requirements. Unlike many protocols involving humans, we do not assume any additional capabilities beyond those described above. We don't require parties to forget information they have learned, or to perform actions obliviously (e.g., shuffle a deck without knowing what the permutation was). Of particular note, we don't require the parties to watch each other during the protocol: this means the protocols can be conducted by mail.

2.3 Distinguishable Envelopes

Our CRRT protocols require a physicial assumption: tamper-evident envelopes or scratch-off cards. Formally, we model these by an ideal functionality we call "Distinguishable Envelopes" (defined in [18]). Loosely speaking, a distinguishable envelope is an envelope in which a message can be sealed. Anyone can open the envelope (and read the message), but the broken seal will be evident to anyone looking at the envelope.

3 An Informal Presentation of the Protocols

It is tempting to try to base a CRRT protocol on oblivious transfer (OT), since if the responder does not learn what the pollster's result is, it may be hard to influence it (in fact, one of the protocols in [1] is based on OT). However, OT is impossible in the DE model [18]. As we show in Section 4.1, this proof implies that in any CRRT protocol using distinguishable envelopes, the responder *must* learn a lot about the pollster's result. In both our protocols, the responder gets complete information about the final result.

To make the presentation more concrete, suppose the poll question is "do you eat your veggies?". Clearly, no one would like to admit that they do not have a balanced diet. On the other hand, pressure groups such as the "People for the Ethical Treatment of Salad" have a political interest in biasing the results of the poll, making it a good candidate for CRRT.

3.1 Pollster-Immune CRRT

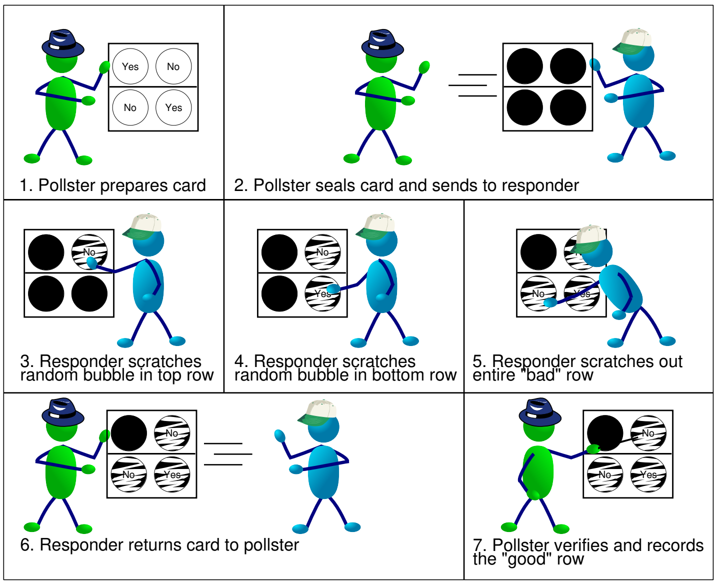

Fig. 3.1. Sample execution of pollster-immune protocol

This protocol can be implemented with pre-printed scratch-off cards: The responder is given a scratch-off card with four scratchable "bubbles", arranged in two rows of two bubbles each. In each row, the word "Yes" is hidden under one bubble and the word "No" under the other (the responder doesn't know which is which). The responder scratches a random bubble in each row. Suppose the responder doesn't eat her veggies. If one of the rows (or both) show the word "No", she "wins" (and the pollster will count the response as expressing dislike of vegetables). If both bubbles show "Yes", she "loses" (and the pollster will count the response as expressing a taste for salad). In any case, before returning the card to the pollster, the responder "eliminates" the row that shows the unfavored answer by scratching the entire row (she picks one of the rows at random if both rows show the same answer) Thus, as long as the responder follows the protocol, the pollster receives a card that has one "eliminated" (entirely scratched) row and one row showing the result he will count. An example of protocol execution appears in Figure 3.1.

Security Intuition. Note that in exactly $\frac{3}{4}$ of the cases the counted result will match the responder's intended result. Moreover, without invalidating the entire card, the responder cannot succeed with higher probability. On the other hand, this provides the responder with plausible deniability: she can always claim both rows were "bad", and so the result didn't reflect her wishes. Because the pollster doesn't know which were the two bubbles that were scratched first, he cannot refute this claim. An important point is that plausible deniability is preserved even if the pollster attempts to cheat (this is what allows the responder to answer the poll accurately even when the pollster isn't trusted). Essentially, the only way the pollster can cheat without being unavoidably caught is to put the *same* answer under both bubbles in one of the rows. To get a feeling for why this doesn't help, write out the distribution of responses in all four cases (cheating/honest, Yes/No). It will be evident that the pollster does not get any additional information about the vote from cheating in this way.

On the other hand, the responder learns the result before the pollster, and can decide to quit if it's not to her liking (without revealing the result to the pollster). Since the pollster does not know the responder's outcome, this has the effect of biasing the result of the poll. However, by counting the number of prematurely halted protocol executions, the pollster can accurately estimate the *number* of cheating responders.

The formal protocol specification and proof appear in Appendix A.

Generalizing to any rational p. The protocol above realizes Pollster-Immune $\frac{3}{4}$-CRRT. In some cases we require a p-CRRT protocol for different values of p. In particular, if we need to repeat the poll, we need the basic protocol to have p closer to $\frac{1}{2}$ (in order to maintain the plausible deniability).

The following protocol will work for any rational $p = \frac{k}{n}$ (assume $k > \frac{1}{2}n$): As in the former protocol, the pollster generates two rows of bubbles. One row contains k "Yes" bubbles and $n - k$ "No" bubbles in random order (this row is the "Yes" row), and the other contains k "No" bubbles and $n - k$ "Yes" bubbles (this row is the "No" row). The rows are also in a random order. The responder's purpose is to find the row matching her choice. She begins by scratching a single bubble in each row. If both bubbles contain the same value, she "eliminates" a random row (by scratching it out completely). Otherwise, she "eliminates" the row that does not correspond to her choice. The pollster's output is the majority value in the row that was not eliminated. The probability that the pollster's output matches the responder's choice is exactly p.

Unfortunately, this protocol is completely secure only for a semi-honest pollster (one that correctly generates the scratch-off cards). A malicious pollster can cheat in two possible ways: he can replace one of the rows with an invalid row (one that does not contain exactly k "Yes" bubbles or exactly k "No" bubbles), or he can use two valid rows that have the same majority value (rather than opposite majority values). In both cases the pollster will gain additional information about the responder's choice. This means the protocol does not realize the ideal Pollster-Immune CRRT functionality.

If the pollster chooses to use an invalid row, he will be caught with probability at least $\frac{1}{2}(1-p)$ (since with this probability the responder will scratch identical bubbles in both rows, and choose to eliminate the invalid row). We can add "cheating detection" to the protocol to increase the probability of detecting this attack. In a protocol with cheating detection, the pollster gives the responder ℓ scratch-off cards rather than just one (each generated according to the basic protocol). The responder chooses one card to use as in the basic protocol. On each of the other cards, she scratches off a single row (chosen randomly), and verifies that it contains either exactly k "Yes" bubbles or exactly k "No" bubbles. She then returns all the cards to the pollster (this step is necessary to prevent the responder from increasing her chances by trying multiple cards until one gives the answer she wants). A pollster that cheats by using an invalid row will be caught with probability $1 - \frac{1}{\ell}$.

A malicious pollster can still cheat undetectably by using two valid rows with identical majorities. This gives only a small advantage, however, and in practice the protocol may still be useful when p is close to $\frac{1}{2}$.

3.2 Responder-Immune CRRT

The responder takes three envelopes (e.g., labelled "1", "2" and "3"), and places one card containing either "Yes" or "No" in each of the envelopes. If she would like to answer "No", she places a single "Yes" card in a random envelope, and one "No" card in each of the two remaining envelopes. She then seals the envelopes and gives them to the pollster (remembering which of the envelopes contained the "Yes" card).

The pollster chooses a random envelope and opens it, revealing the card to the responder. He then asks the responder to tell him which of the two remaining envelopes contains a card with the *opposite* answer. He opens that envelope as well. If the envelope does contain a card with the opposite answer, he records the answer on the first card as the response to the poll, and returns the third (unopened) envelope to the responder.

If both opened envelopes contain the same answer, it can only be because the responder cheated. In this case, the pollster opens the third envelope as well. If the third envelope contains the opposite answer, the pollster records the answer on the first card as the response to the poll. If, on the other hand, all three

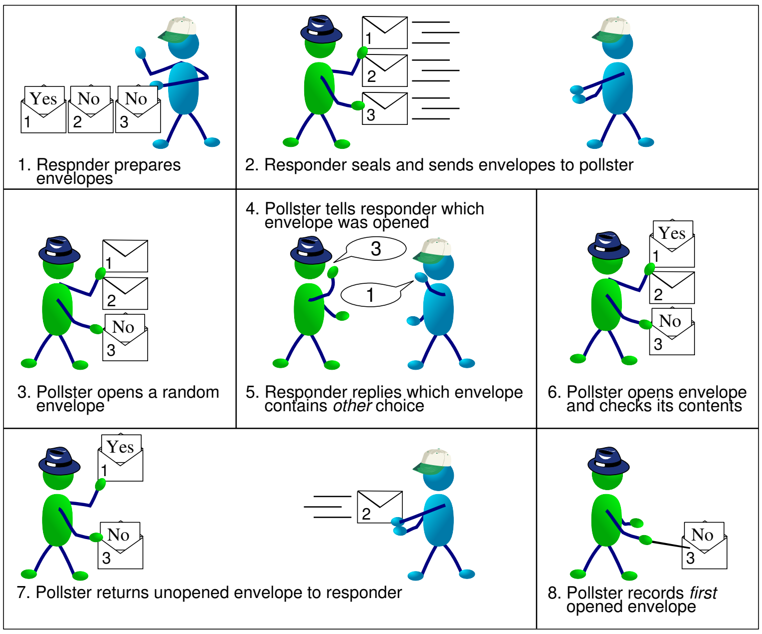

Fig. 3.2. Sample execution of responder-immune protocol

envelopes contain the same answer, the pollster rolls a die: A result of 1 to 4 (probability $\frac{2}{3}$) means he records the answer that appears in the envelopes, and a result of 5 or 6 means she records the opposite answer. An example of protocol execution (where both parties follow the protocol) appears in Figure 3.2.

Security Intuition. In this protocol, the responder gets her wish with probability at most $\frac{2}{3}$ no matter what she does. If she follows the protocol when putting the answers in the envelopes, the pollster will choose the envelope containing the other answer with probability $\frac{1}{3}$. If she tries to cheat by putting the same answer in all three envelopes, the pollster will roll a die and choose the opposite answer with probability $\frac{1}{3}$. The pollster, on the other hand, can decide to open all three envelopes and thus discover the real answer favored by the responder. If he does this, however, the responder will see that the seal on the returned envelope was broken and know the pollster was cheating.

The pollster may also be able to cheat in an additional way: he can open two envelopes before telling the responder which envelope he opened, and hope that the responder will not require him to return an envelope that was already opened. This attack is what requires us to add the **Test** command to the functionality.

Implementation Notes. This protocol requires real envelopes (rather than scratch-off cards) to implement, since the responder must choose what to place in the envelopes (and we cannot assume the responder can create a scratch-off card).

In general, tamper-evidence for envelopes may be hard to achieve (especially as the envelopes will most likely be provided by the pollster). In this protocol, however, the pollster's actions can be performed in full view of the responder, so any opening of the envelopes will be immediately evident. When this is the case, the responder can tell which envelope the pollster opened first, so the protocol actually realizes the stronger version of the Responder-Immune CRRT functionality (without the **Test** command).

If the penalty for a pollster caught cheating is large enough, the privacy guaranteed by this protocol, may be enough to convince responders to answer accurately in a real-world situation even with the weaker version of the functionality. This is because any pollster cheating that can possibly reveal additional information about the responder's choice carries with it a corresponding risk of detection.

Generalizing to any rational p. When the pollster's actions are performed in view of the responder (in particular, when the responder can see exactly which envelopes are opened by the pollster), this protocol has a straightforward generalization to any rational $p = \frac{k}{n}$, where $k > \frac{1}{2}n$: the responder uses n (rather than 3) envelopes, of which k contain her choice and $n - k$ contain its opposite. After the pollster chooses an envelope to open, the responder shows him $n - k$ envelopes that contain the opposite value.

Note that when this generalized protocol is performed by mail, it does *not* realize the ideal functionality defined in Section 2.1.

4 Strong CRRT Protocols

Ideally, we would like to have CRRT protocols that cannot be biased at all by malicious responders, while perfectly preserving the responder's deniability, even against malicious pollsters. Unfortunately, the protocols described in Section 3 do not quite achieve this. At the expense of increasing the number of rounds, we can get arbitrarily close to the Strong-CRRT functionality defined in Section 2.1.

Consider a protocol in which the pollster and responder perform the pollster-immune p-CRRT protocol r times, one after the other (with the responder using the same input each time). The pollster outputs the majority of the subprotocols' outputs. If the responder halts at any stage, the pollster uses uniformly random bits in place of the remaining outputs.

This protocol gives a corrupt responder at most $O(\frac{1}{\sqrt{r}})$ advantage over an honest responder. We give here only the intuition for why this is so: Clearly, if a corrupt responder wants to bias the result to some bit b, it is in her best interest to use b for all the inputs. Since the subprotocol securely realizes p-CRRT, the only additional advantage she can gain is by halting at some round i. However, halting affects the result only if the other $r - 1$ rounds were balanced (this is the only case in which the outcome of the i^{th} round affects the majority). In the case where $p = \frac{1}{2}$, it is easy to see that the probability for this occurring is

$O(\frac{1}{\sqrt{r}})$. However, the probability that $r-1$ independent weighted coin flips are balanced is maximized when $p=\frac{1}{2}$. Thus, the additional advantage that can be gained by the adversary is at most $O(\frac{1}{\sqrt{r}})$.

The problem with the amplification protocol described above is that the probability that an honest responder will get the result she wants tends to 1 as the number of rounds grows, for any constant $p>\frac{1}{2}$. Therefore, to preserve plausible deniability we must use a p-CRRT protocol where p is very close to $\frac{1}{2}$, such as the protocol described in Section 3.1 that works for any rational p. This adds further complexity to the protocol (e.g., our generalized Pollster-Immune protocol requires $\Omega(\frac{1}{\epsilon})$ bubbles on the scratch-off card for $p=\frac{1}{2}+\epsilon$). This, this multi-round protocol is probably not feasible in practice.

4.1 Lower Bounds and Impossibility Results

In this section we attempt to show that constructing practical strong CRRT protocols is a difficult task. We do this by giving impossibility results and lower bounds for implementing subclasses of the strong CRRT functionality. We consider a generalization of the strong p-CRRT functionality defined in Section 2.1, which we call (p,q)-CRRT. The (p,q)-CRRT functionality can be described as follows:

Vote b The issuer of this command is the responder. On receiving this command the functionality tosses a weighted coin c, such that $c=0$ with probability p. It then outputs $b\oplus c$ to the pollster. The functionality supplies the responder with exactly enough additional information so that she can guess c with probability $q\geq p$.

In the definition of strong CRRT given in Section 2.1, we specify exactly how much information the pollster learns about the responder's choice, but leave completely undefined what a cheating responder can learn about the pollster's result. The (p,q)-CRRT functionality quantifies this information: in a (p,p)-CRRT, the responder does not gain any additional information (beyond her preexisting knowledge that the pollster's result will equal her choice with probability p). In a $(p,1)$-CRRT, the responder learns the pollster's result completely. We show that (p,p)-CRRT implies oblivious transfer (and is thus impossible in the DE model), while $(p,1)$-CRRT implies strong coin-flipping (and thus we can lower-bound the number of rounds required for the protocol). For values of q close to p or close to 1, the same methods can still be used to show lower bounds.

(p,q)-CRRT when q is close to p: First, note that when $p=q$ we can view the (p,q)-CRRT functionality as a binary symmetric channel (BSC) with error probability $1-p$. Crépeau and Kilian have shown that a protocol for Oblivious Transfer (OT) can be constructed based on any BSC [9]. However, it is impossible to implement OT in the Distinguishable Envelope (DE) model [18]. Therefore (p,p)-CRRT cannot be implemented in the DE model. It turns out that this is also true for any q close enough to p. This is because, essentially, the (p,q)-CRRT functionality is a $(1-q, 1-p)$-Passive Unfair Noisy Channel (PassiveUNC), as defined by Damgård, Kilian and Salvail [12]. A (γ, δ)-PassiveUNC is a BSC with error δ which provides the corrupt sender (or receiver) with additional information that brings his perceived error down to γ; (i.e., a corrupt sender can guess the bit received by the receiver with probability $1-\gamma$, while an honest sender can guess this bit only with probability $1-\delta$). For γ and δ that are close enough (the exact relation is rather complex), Damgård, Fehr, Morozov and Salvail [11] show that a (γ, δ)-PassiveUNC is sufficient to construct OT. For the same range of parameters, this implies that realizing (p, q)-CRRT is impossible in the DE model.

(p, q)-CRRT when q is close to 1: When $q = 1$, both the pollster and the responder learn the poll result together. A $(p, 1)$-CRRT can be used as a protocol for strongly fair coin flipping with bias $p - \frac{1}{2}$. In a strongly fair coin flipping protocol with bias ϵ, the bias of an honest party's output is at most ϵ regardless of the other party's actions — even if the other party aborts prematurely. If q is close to 1, we can still construct a coin flipping protocol, albeit without perfect consistency. The protocol works as before, except that the responder outputs his best guess for the pollster's output: both will output the same bit with probability q.

A result by Cleve [8] shows that even if all the adversary can do is halt prematurely (and must otherwise follow the protocol exactly), any r-round protocol in which honest parties agree on the output with probability $\frac{1}{2} + \epsilon$ can be biased by at least $\frac{\epsilon}{4r+1}$. Cleve's proof works by constructing $4r + 1$ adversaries, each of which corresponds to a particular round. An adversary corresponding to round i follows the protocol until it reaches round i. It then halts immediately, or after one extra round. The adversary's decision is based only on what the ourput of an honest player would be in the same situation, should the *other* party halt after this round. Cleve shows that the average bias achieved by these adversaries is $\frac{\epsilon}{4r+1}$, so at least one of them must achieve this bias. The same proof also works in the DE model, since all that is required is that the adversary be able to compute what it would output should the other player stop after it sends the messages (and envelopes) for the current round. This calculation may require a party to open some envelopes (the problem being that this might prevent the adversary from continuing to the next round). However, an honest player would be able to perform the calculation in the next round, after sending this round's envelopes, so it cannot require the adversary to open any envelopes that may be sent in the next round.

Cleve's lower bound shows that a (p, q)-CRRT protocol must have at least $\frac{q-\frac{1}{2}}{4(p-\frac{1}{2})} - \frac{1}{4}$ rounds. Since a protocol with a large number of rounds is impractical for humans to implement, this puts a lower bound on the bias p (finding a CRRT protocol with a small p is important if we want to be able to repeat the poll while still preserving plausible deniability).

This result also implies that it is impossible to construct a $(p, 1)$-CRRT protocol in which there is a clear separation between the responder's choice and the final output. That is, the following functionality, which we call *p-CRRT with confirmation*, is impossible to implement in the DE model:

Vote b The issuer of this command is the responder. On receiving this command the functionality outputs "Ready?" to the pollster. When the pollster answers with "ok" the functionality tosses a weighted coin c, such that $c = 0$ with probability p. It then outputs $b \oplus c$ to the pollster and responder.

p-CRRT with confirmation is identical to $(p, 1)$-CRRT, except that the output isn't sent until the pollster is ready. The reason it is impossible to implement is that this functionality can be amplified by parallel repetition to give a strongly fair coin flipping protocol with arbitrarily small p. Since the amplification is in parallel, it does not increase the number of rounds required by the protocol, and thus contradicts Cleve's lower bound. Briefly, the amplified protocol works as follows: the responder chooses k inputs randomly, and sends each input to a separate (parallel) instance of p-CRRT with confirmation. The pollster waits until all the inputs have been sent (i.e., it receives the "Ready?" message from all the instances), then sends "ok" to all the instances. The final result will be the xor of the outputs of all the instances. Since the different instances act independently, the bias of the final result is exponentially small in k.

5 Discussion and Open Problems

Polling Protocols by Mail. The pollster-immune CRRT protocol requires only a single round; This makes it convenient to use in polls through the post (it only requires the poll to be sent to the responder, "filled out" and returned). The responder-immune protocol presents additional problems when used through the post. First, in this case the protocol realizes a slightly weaker functionality than in the face-to-face implementation. Second, it requires two rounds, and begins with the responder. This means, in effect, that it would require an extra half-round for the pollster to notify the responder about the existence of the poll. It would be interesting to find a one-round protocol for the responder-immune functionality as well. It may be useful, in this context, to differentiate between "information-only" communication (which can be conducted by phone or email), and transfer of physical objects such as envelopes (which require "real" mail).

Efficient Generalization to Arbitrary p. We describe efficient p-CRRT protocols for specific values of p: $p = \frac{3}{4}$ in the Pollster-Immune case, and $p = \frac{2}{3}$ in the Responder-Immune case. Our generalized protocols are not very efficient: for $p = \frac{1}{2} + \epsilon$ they require $\Omega(\frac{1}{\epsilon})$ envelopes. In a protocol meant to be implemented by humans, the efficiency of the protocol has great importance. It would be useful to find an efficient general protocol to approximate arbitrary values of p (e.g., logarithmic in the approximation error).

Side-Channel Attacks. The privacy of our protocols relies on the ability of the responder to secretly perform some actions. For instance, in the pollster-immune protocol we assume that the order in which the bubbles on the card were scratched remains secret. In practice, some implementations may be vulnerable to an attack on this assumption. For example, if the pollster uses a light-sensitive dye on the scratch-off cards that begins to darken when the coating is scratched off, he may be able to tell which of the bubbles was scratched first. Side-channel attacks are attacks on the model, not on the CRRT protocols themselves. As these attacks highlight, when implementing CRRT using a physical implementation of Distinguishable Envelopes, it is important to verify that this implementation actually does realize the required functionality.

Dealing With Human Limitations. Our protocols make two assumptions about the humans implementing them: that they can make random choices and that they can follow instructions. The former assumption can be relaxed: if the randomness "close to uniform" the security and privacy will suffer only slightly (furthermore, simple physical aids, such as coins or dice, make generating randomness much easier). The latter assumption is more critical; small deviations from the protocol can result in complete loss of privacy or security. Constructing protocols that are robust to human error could be very useful.

Practical Strong CRRT Protocols. As we discuss in Section 4.1, for a range of parameters p, q-CRRT is impossible, and for a different range of parameters it is impractical. For some very reasonable values, such as $\frac{3}{4}$-Strong CRRT, we can approximate the functionality using a large number of rounds, but do not know how to prove any lower bound on the number of rounds required. Closing this gap is an obvious open question. Alternatively, finding a physical model in which efficient Strong CRRT is possible is also an interesting direction.

Acknowledgements

We would like to thank Adi Shamir and Yossi Oren for pointing out possible side-channel attacks in the scratch-off card model, and the anonymous reviewers for many helpful comments.

References

1. A. Ambainis, M. Jakobsson, and H. Lipmaa. Cryptographic randomized response techniques. In *PKC '04*, volume 2947 of *LNCS*, pages 425–438, 2004.
2. J. Benaloh and D. Tuinstra. Receipt-free secret-ballot elections. In *STOC '94*, pages 544–553, 1994.
3. R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS '01*, pages 136–145, 2001.
4. R. Canetti, C. Dwork, M. Naor, and R. Ostrovsky. Deniable encryption. In *CRYPTO '97*, volume 1294 of *LNCS*, pages 90–104, 1997.
5. R. Canetti and R. Gennaro. Incoercible multiparty computation. In *FOCS '96*, pages 504–513, 1996.
6. A. Chaudhuri and R. Mukerjee. *Randomized Response: Theory and Techniques*, volume 85. Marcel Dekker, 1988.

7. D. Chaum. E-voting: Secret-ballot receipts: True voter-verifiable elections. *IEEE Security & Privacy*, 2(1):38–47, Jan./Feb. 2004.
8. R. Cleve. Limits on the security of coin flips when half the processors are faulty. In *STOC '86*, pages 364–369, 1986.
9. C. Crépeau and J. Kilian. Achieving oblivious transfer using weakened security assumptions. In *FOCS '88*, pages 42–52, 1988.
10. C. Crépeau and J. Kilian. Discreet solitary games. In *CRYPTO '93*, volume 773 of *LNCS*, pages 319–330, 1994.
11. I. B. Damgård, S. Fehr, K. Morozov, and L. Salvail. Unfair noisy channels and oblivious transfer. In *TCC '04*, volume 2951 of *LNCS*, pages 355–373, 2004.
12. I. B. Damgård, J. Kilian, and L. Salvail. On the (im)possibility of basing oblivious transfer and bit commitment on weakened security assumptions. In *Eurocrypt '99*, volume 1592 of *LNCS*, pages 56–73, 1999.
13. P. Diaconis, S. Holmes, and R. Montgomery. Dynamical bias in the coin toss, 2004. http://www-stat.stanford.edu/∼cgates/PERSI/papers/headswithJ.pdf.
14. J. A. Droitcour, E. M. Larson, and F. J. Scheuren. The three card method: Estimating sensitive survey items–with permanent anonymity of response. In *Proceedings of the American Statistical Association, Social Statistics Section [CD-ROM]*, 2001.
15. C. Dwork, M. Naor, and A. Sahai. Concurrent zero knowledge. In *STOC '98*, pages 409–418, New York, NY, USA, 1998. ACM Press.
16. R. Fagin, M. Naor, and P. Winkler. Comparing information without leaking it. *Commun. ACM*, 39(5):77–85, 1996.
17. H. Kikuchi, J. Akiyama, G. Nakamura, and H. Gobioff. Stochastic voting protocol to protect voters privacy. In *WIAPP '99*, pages 102–111, 1999.
18. T. Moran and M. Naor. Basing cryptographic protocols on tamper-evident seals. In *ICALP 2005*, volume 3580 of *LNCS*, pages 285–297, July 2005.
19. M. Naor, Y. Naor, and O. Reingold. Applied kid cryptography, Mar. 1999. http://www.wisdom.weizmann.ac.il/∼naor/PAPERS/waldo.ps.
20. M. Naor and B. Pinkas. Visual authentication and identification. In *CRYPTO '97*, volume 1294 of *LNCS*, pages 322–336, 1997.
21. M. Naor and A. Shamir. Visual cryptography. In *Eurocrypt '94*, volume 950 of *LNCS*, pages 1–12, 1995.
22. K. Sako and J. Kilian. Receipt-free mix-type voting schemes. In *EUROCRYPT '95*, volume 921 of *LNCS*, pages 393–403, 1995.
23. B. Schneier. The solitaire encryption algorithm, 1999. http://www.schneier.com/solitaire.html.
24. S. Stamm and M. Jakobsson. Privacy-preserving polling using playing cards. Cryptology ePrint Archive, Report 2005/444, December 2005.
25. S. Warner. Randomized response: a survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, pages 63–69, 1965.

A A Pollster-Immune $\frac{3}{4}$-CRRT Protocol

A.1 Formal Specification

Let $\mathcal{P}$ be the pollster and $\mathcal{R}$ the responder. Denote $\mathcal{P}$'s random bits p_0, p_1 and $\mathcal{R}$'s random bits r_0, r_1, r_2.

1. To implement **Query**: $\mathcal{P}$ creates two pairs of envelopes, each pair containing a 0 and a 1. The first pair contains $(p_0, 1 - p_0)$ and the second $(p_1, 1 - p_1)$. $\mathcal{P}$ sends both pairs to $\mathcal{R}$.

2. To implement **Vote** b: $\mathcal{R}$ opens a random envelope from each pair (the index of the first envelope opened is given by r_0 and the second by r_1. Denote the values of the opened envelopes $x_0 = p_0 \oplus r_0$ and $x_1 = p_1 \oplus r_1$.
   (a) If $x_0 = x_1$ (i.e., both the opened values are equal), $\mathcal{R}$ chooses a random pair and opens the remaining envelope in that pair (the first pair if $r_2 = 0$ and the second if $r_2 = 1$).
   (b) If $x_0 \neq x_1$, $\mathcal{R}$ opens the remaining envelope in the pair whose open envelope is not equal to b.
   (c) In both cases, $\mathcal{R}$ verifies that the envelopes in the completely opened pair contain different values (i.e., that the pair is valid). If so, $\mathcal{R}$ then sends all four envelopes back to $\mathcal{P}$, otherwise $\mathcal{R}$ halts.
3. If $\mathcal{R}$ halted in the previous step, $\mathcal{P}$ outputs $\boxtimes$ and halts. Otherwise, $\mathcal{P}$ verifies that exactly three of the four envelopes received from $\mathcal{R}$ are open. If so, $\mathcal{P}$ outputs the contents of the open envelope in the pair that contains the sealed envelope. If not, $\mathcal{P}$ outputs $\boxtimes$.

A.2 Proof of Security

In this section we give the proof that the protocol securely realizes Pollster-Immune $\frac{3}{4}$-CRRT in the UC model. The proof follows the standard outline for a UC proof: we describe an ideal adversary, $\mathcal{I}$, that works in the ideal world by simulating a real adversary, $\mathcal{A}$ (given black-box access to $\mathcal{A}$), along with the envelope functionalities used to implement the protocol in the real world. We then show that no environment machine, $\mathcal{Z}$ (which is allowed to set the parties' inputs) can distinguish between the case that it is communicating with $\mathcal{A}$ in the real world, and the case where it is communicating with $\mathcal{I}$ in the ideal world (for a more in-depth explanation of the UC model, see [3]). We'll deal separately with the case when $\mathcal{A}$ corrupts $\mathcal{P}$ and when it corrupts $\mathcal{R}$ (since we assume the corruption occurs as a first step). The proof that the views of $\mathcal{Z}$ in the real and ideal worlds are identical is by exhaustive case analysis.

$\mathcal{A}$ corrupts $\mathcal{P}$

1. $\mathcal{I}$ waits to receive c, the outcome of the poll from the ideal functionality. $\mathcal{I}$ now begins simulating $\mathcal{F}_{DE}$ and $\mathcal{R}$ (as if he were a real honest party). The simulation runs until $\mathcal{P}$ sends four envelopes as required by the protocol (up to this point $\mathcal{R}$ did not participate at all in the protocol).
2. If both pairs of envelopes are valid (contain a 0 and a 1), $\mathcal{I}$ chooses one of the pairs at random, and simulates opening the envelope in the pair that contains c and both envelopes in the other pair (there is an assignment to the random coins of $\mathcal{R}$ which would have this result in the real world). It then simulates the return of all four envelopes to $\mathcal{P}$.
3. If both pairs of envelopes are invalid, $\mathcal{I}$ simulates $\mathcal{R}$ halting (this would eventually happen in a real execution as well).
4. If exactly one pair of envelopes is invalid, denote the value in the invalid pair by z.

(a) If $c = z$, $\mathcal{I}$ simulates opening both envelopes in the valid pair, and a random envelope in the invalid pair (depending on the random coins of $\mathcal{R}$, this is a possible result in the real world). It then simulates the return of all four envelopes to $\mathcal{P}$

(b) If $c \neq z$, $\mathcal{I}$ simulates $\mathcal{R}$ halting (depending on the random coins of $\mathcal{R}$, this is also a possible result in the real world).

5. $\mathcal{I}$ continues the simulation until $\mathcal{A}$ halts.

Note that throughout the simulation, all simulated parties behave in a manner that is feasible in the real world as well. Thus, the only possible difference between the views of $\mathcal{Z}$ in the ideal and real worlds is the behavior of the simulated $\mathcal{R}$, which depends only on the contents of the four envelopes sent by $\mathcal{P}$ and the output of the ideal functionality (which in turn depends only on b). It is easy (albeit tedious) to go over all 32 combinations of envelopes and input, and verify that the distribution of $\mathcal{R}$'s output in both cases (the real and ideal worlds) are identical. We enumerate the basic cases below. All other cases are identical to one of the following by symmetry:

1. $\mathcal{A}$ sends two valid pairs of envelopes. Assume it sends $[(b, 1-b), (b, 1-b)]$ (the other combinations follow by symmetry). $\mathcal{I}$ returns the following distribution ("*" denotes a sealed envelope):
   (a) With probability $\frac{3}{4}$ $(c = b)$ it selects uniformly from $\{[(b, *), (b, 1-b)], [(b, 1-b), (b, *)]\}$
   (b) With probability $\frac{1}{4}$ $(c \neq b)$ it selects uniformly from $\{[(*, 1-b), (b, 1-b)], [(b, 1-b), (*, 1-b)]\}$
   In the real world, the order of envelopes opened by $\mathcal{R}$ would be distributed uniformly from one of the following sets (each with probability $\frac{1}{4}$):
   (a) $\{[(1, *), (3, 2)]\}$
   (b) $\{[(1, *), (2, 3)], [(1, 3), (2, *)]\}$
   (c) $\{[(3, 1), (2, *)]\}$
   (d) $\{[(3, 1), (*, 2)], [(*, 1), (3, 2)]\}$
   Note that the observed result is distributed identically in both cases.
2. $\mathcal{A}$ sends two invalid pairs of envelopes: in this case, in both the real and ideal worlds the adversary will see the responder halting with probability 1.
3. $\mathcal{A}$ sends one valid and one invalid pair of envelopes:
   (a) $\mathcal{A}$ sends $[(b, b), (b, 1-b)]$ (the other case where the invalid pair matches b is symmetric). The distribution of the returned envelopes in the ideal world is:
      i. With probability $\frac{3}{4}$ $(c = b)$ it selects uniformly from $\{[(b, *), (b, 1-b)], [(*, 1-b), (b, 1-b)]\}$
      ii. With probability $\frac{1}{4}$ $(c \neq b)$ it halts.
   In the real world, the order of envelopes opened by $\mathcal{R}$ would be distributed uniformly from one of the following sets (each with probability $\frac{1}{4}$); the sets marked with † lead to $\mathcal{R}$ halting:
      i. $\{[(1, *), (3, 2)]\}$
      ii. $\left\{[(1, *), (2, 3)], [(1, 3), (2, *)]^{\dagger}\right\}$
      iii. $\left\{[(*, 1), (2, 3)], [(3, 1), (2, *)]^{\dagger}\right\}$

iv. $\{[(*, 1), (3, 2)]\}$
Note that in both worlds $\mathcal{R}$ halts with probability $\frac{1}{4}$, and otherwise the returned envelopes are identically distributed.

(b) $\mathcal{A}$ sends $[(1-b, 1-b), (b, 1-b)]$ (the other case where the invalid pair matches $1-b$ is symmetric). The distribution of the returned envelopes in the ideal world is:
i. With probability $\frac{1}{4}$ ($c \neq b$) it selects uniformly from $\{[(1-b, *), (b, 1-b)], [(*, 1-b), (b, 1-b)]\}$
ii. With probability $\frac{3}{4}$ ($c = b$) it halts.
In the real world, the order of envelopes opened by $\mathcal{R}$ would be distributed uniformly from one of the following sets (each with probability $\frac{1}{4}$); the sets marked with † lead to $\mathcal{R}$ halting:
i. $\left\{[(1, *), (3, 2)], [(1, 3), (*, 2)]^{\dagger}\right\}$
ii. $\left\{[(1, 3), (2, *)]^{\dagger}\right\}$
iii. $\left\{[(*, 1), (3, 2)], [(3, 1), (*, 2)]^{\dagger}\right\}$
iv. $\left\{[(3, 1), (2, *)]^{\dagger}\right\}$
Note that in both worlds $\mathcal{R}$ halts with probability $\frac{3}{4}$, and otherwise the returned envelopes are identically distributed.

$\mathcal{A}$ corrupts $\mathcal{R}$

1. $\mathcal{I}$ waits to receive v and r from the ideal functionality (in response to the **Query** command sent by the ideal $\mathcal{P}$).
2. $\mathcal{I}$ simulates $\mathcal{R}$ receiving four envelopes. The remainder of the simulation depends on the values of v and r:
   (a) If $v = 1$, $\mathcal{I}$ chooses a uniformly random bit t. The first envelope $\mathcal{R}$ opens in the first pair will have the value t, and the first envelope opened in the second pair will have the value $1 - t$. The values revealed in the remaining envelopes will always result in a valid pair.
   (b) If $v = 0$, The first envelope $\mathcal{R}$ opens in each pair will have the value r, and the remaining envelopes the value $1 - r$.
3. $\mathcal{I}$ continues the simulation until $\mathcal{R}$ sends all four envelopes back to $\mathcal{P}$. If $\mathcal{R}$ opened exactly three envelopes, $\mathcal{I}$ sends **Vote** b to the ideal functionality, where b is calculated as by the pollster in the protocol description. If $\mathcal{R}$ did not open exactly three envelopes, $\mathcal{I}$ sends the **Halt** command to the ideal functionality.

Note that throughout the simulation, all simulated parties behave in a manner that is feasible in the real world as well. Furthermore, the outputs of the ideal and simulated $\mathcal{P}$ are always identical. Thus, the only possible difference between the views of $\mathcal{Z}$ in the ideal and real worlds is the contents of the envelopes opened by $\mathcal{R}$. In the real world, the envelope contents are random. In the ideal world, v and r are i.i.d. uniform bits. Therefore the order in which the envelopes are opened does not matter; any envelope in the first pair is independent of any envelope in the second. Hence, the distributions in the ideal and real worlds are identical.