How (not) to Achieve both Coercion Resistance and Cast as Intended Verifiability in Remote eVoting*

Tamara Finogina[1], Javier Herranz[2], and Enrique Larraia[3]

[1] Scytl Election Technologies, Barcelona, Spain tamara.finogina@scytl.com
[2] Dpt. Matemàtiques, Universitat Politècnica de Catalunya, Barcelona, Spain javier.herranz@upc.edu
[3] nChain, Zug, Switzerland e.larraia@nchain.com

Abstract. We consider the problem of achieving, at the same time, cast-as-intended verifiability and coercion resistance, in remote electronic voting systems where there are no secure channels through which voters can receive secret information/credentials before the voting phase.
We discuss why some simple solutions fail to achieve the two desired notions and we propose (a bit) more involved solutions that are satisfactory. Part of the discussion is closely related to the gap "full versus honest-verifier" when defining the zero-knowledge property of cryptographic zero-knowledge systems.

Keywords: electronic voting, coercion resistance, cast as intended verifiability, zero-knowledge systems

1 Introduction

Consider the following situation, very common in electronic voting. A voter interacts with a voting device VD (which may be a webpage or a voting terminal) to cast its intention m. The result of the interaction is typically a public-key encryption $C = \mathsf{Enc}_{\mathsf{pk}}(m; r)$ of the intention m, computed using randomness r. This ciphertext will be sent to the ballot box of the election, and then it will be decrypted (typically after a privacy-preserving operation, like shuffling). Suppose the voter does not trust the voting device (VD) performing the encryption. Casting a vote consists of two steps: (1) the voter sends its option m to VD, (2) VD prepares a ciphertext C and sends it to the ballot box. How can the voter be sure that C actually contains encryption of its choice m? If the system provides some way for the voter to be convinced, then it achieves the property of cast-as-intended verifiability (CAI, for short). One possibility is that the voter obtains a proof that its vote has not been changed, that is that C is indeed correct encryption of m. This proof can be either human-verifiable (e.g. a choice return code, a tracking number, etc.) or require the help of a verification device

* Work done while the 3rd author was at Scytl Election Technologies.

(such a mobile phone, for instance, to run some mathematical/cryptographic computations). However, in both cases, the sole existence of such proof might open a door to intentional vote selling or coercion.

In this work we focus on the following scenario: there are no secure channels that allow voters to receive some secret information or credential, but each voter has a personal (trusted) device to run mathematical/cryptographic computations, in the voting phase. In such a scenario, is it possible to achieve the two properties, namely CAI verifiability and coercion resistance? The second one means a coercer cannot distinguish a (real) execution of the voting phase between the voter and VD with input m from a (simulated) execution of the voting phase with any other input m^*, possibly chosen by the coercer, even if the coercer forces the voter to use some specific (distribution of) values during the voting phase. Of course, this notion of coercion resistance (CR, for short) makes sense only if one assumes that the coercer cannot control the voter during the execution of the voting phase (otherwise, the coercer becomes the voter, and there is not much to do). Please note, that our informal definition of CR does not account for forced-abstention attacks (i.e. the coercer forces voter not to participate in the election), since their mitigation requires [9] anonymous voting channels, which are hard to achieve in practice.

Contributions and organization of the paper. After recalling some necessary cryptographic notions in Section 2, we discuss in Section 3 possible solutions to the CAI+CR problem that are not satisfactory. Then in Section 4 we present two solutions that are satisfactory. The gap between unsatisfactory and satisfactory solutions is closely related to the gap between the honest-verifier and full zero-knowledge property of cryptographic zero-knowledge systems. In particular, we show that a solution with one or two rounds of communication between the voter and the VD cannot be secure. The two solutions of Section 4 are generic and could be described for general binary relations, zero-knowledge systems, etc. However, for the sake of clarity and simplicity, we directly describe specific instantiations of the generic solutions, for the particular case where voting options m are encrypted using the ElGamal.

We want to stress that this (short) paper is intentionally written in a not-fully formalized way, without detailed definitions of the security notions and without security proofs at all. Our main goal here is to present the main conclusions of our study in a clear way, to the broadest possible (maybe non-cryptographic) audience. The missing details and formalization will be added in a future extended version of this (ongoing) work.

2 Cryptographic Preliminaries

2.1 ElGamal Public Key Encryption

ElGamal public key encryption scheme [5] works as follows:

- The key generation protocol takes as input a security parameter κ and generates a prime number q and a cyclic group $G = \langle g \rangle$ of order q. The secret key sk is chosen at random from $\mathbb{Z}_q$, the matching public key is $\mathsf{pk} = g^{sk}$.

– The encryption protocol takes as input a public key pk and a message $m \in G$; then a random value $r \in \mathbb{Z}_q$ is chosen, and the ciphertext $C = (c_1, c_2)$ is computed as $c_1 = g^r$ and $c_2 = m \cdot \mathsf{pk}^r$.
– The decryption protocol takes as input the secret key sk and a ciphertext $C = (c_1, c_2) \in G \times G$, and outputs $c_2/c_1^{sk} = m$.

2.2 Zero-Knowledge Proof Systems

In a zero-knowledge interactive system, a prover P has some secret witness w of the fact that some public element x belongs to some language $\mathcal{L}_{\mathcal{R}}$, where $\mathcal{R} \subset \mathcal{W} \times \mathcal{X}$ is a binary relation and $\mathcal{L}_{\mathcal{R}} = \{x \in \mathcal{X} \mid \exists w \in \mathcal{W}\ s.t. (x, w) \in \mathcal{R}\}$. In an execution of the protocol, such a prover P and a verifier V interact by sending and receiving information through a number of rounds. At the end, the goal is that P convinces V of the fact that P knows a secret witness w such that $(x, w) \in \mathcal{R}$. Typically, three properties are required for such a protocol:

– **Completeness:** if both P and V are honest and $(x, w) \in \mathcal{R}$, then the verifier always accepts the proof as valid.
– **Soundness:** if P is dishonest and $(x, w) \notin \mathcal{R}$, then the verifier does not accept the proof as valid.
– **Zero-Knowledge:** the execution of the protocol does not leak any information about the secret witness w. This is formalized requiring, for every verifier $\mathtt{V}^*$, the existence of a polynomial-time simulator $\mathtt{M}_{\mathtt{V}^*}$ s.t. $\forall (x, w) \in \mathcal{R}$ the output $\langle \mathtt{P}(x, w), \mathtt{V}^*(x) \rangle$ is identically distributed to the output $\mathtt{M}_{\mathtt{V}^*}(x)$.

If Zero-Knowledge holds for any possible verifier $\mathtt{V}^*$, then we say the protocol achieves *full zero-knowledge*. However, if it only holds for verifiers $\mathtt{V}^*$ that correctly follow the prescribed steps of the protocol, then we say the protocol achieves *honest-verifier zero-knowledge*. A Σ-protocol is an example of an interactive system with honest-verifier zero-knowledge; it is a three-move protocol producing transcripts with the form (a, e, z), where the first message a is sent by the prover, e is a value chosen by the verifier uniformly and at random from a suitable challenge space, and z is the answer computed by the prover.

3 CAI+CR: Discussion of some Unsatisfactory Solutions

Most e-voting systems are designed to provide CAI verifiability by outputting some proof, therefore it is assumed that a coercer is quite limited e.g. the voting or registration is done without a coercer's control, nor can he obtain private voter's data, credentials or impersonate the voter, etc. The only scheme that focuses on full coercion is Civitas [4], however, it does not deal with CAI as VD is assumed to be honest.

One of the simplest ways to provide CAI verifiability is via transferable proofs e.g. QR-codes that store encryption randomness, NIZK proof of encryption randomness knowledge, etc. However CR does not hold when voter is not trusted.

Another approach requires secure channels to deliver some secret information to a voter and assumes that the voter will not share it with anyone. For example,

return-code based schemes pre-deliver voting cards with the mapping between choices and return codes to each voter. Similarly, the voter can receive tracking numbers, secret keys, voting cards with vote-codes, pre-computed ballots, etc. However, in the settings without secure channels, this strategy would not work.

It seems the only way to achieve both CAI and CR, without trusting voters to keep CAI proof private or using secure channels, requires interaction. Next we discuss why some strategies fail in providing both CAI and CR.

Solution U_1: cast-or-challenge. This technique [3] is used by Helios [2] and its derivatives to achieve the CAI property: the VD sends to the voter the randomness r that was used to produce the ciphertext $C = \mathsf{Enc}_{\mathsf{pk}}(m; r)$. The voter can then use another device to re-encrypt m with r and check that the result is C. This process is repeated a random number k of times, until for the $k + 1$-th interaction the voter does not require the randomness and casts the ciphertext.

While this solution is quite popular and enjoys CR, however, it does not provide CAI strictly speaking, since the sent ballot is never audited. True, it gives some chance to detect VD misbehavior, but a malicious VD may guess the last verification attempt and cheat or prepare separate ballots for auditing and sending. Moreover, studies show that users do not understand this verification method and on average only 43% of users are able to verify their votes[1].

Solution U_2: using Σ-protocols. A different possibility (not explicitly proposed anywhere, as far as we know) is to consider an interactive Σ protocol between VD and the voter, to let VD convince the voter that C encrypts m.

Unfortunately, the zero-knowledge property of a Σ protocol is honest-verifier only; and in our application, the verifier (the voter) can be under control of a coercer. If the coercer forces the voter to choose the challenge e in a particular way, for instance as $e = \mathtt{Hash}(a)$, where a is the message voter sees before introducing the challenge, then the coercer can easily verify if the voter obeyed and voted for m^* or not. Note that for such e it is computationally infeasible to simulate a valid transcript (a', e', z') for a different voting option $m^* \neq m$ that satisfies the required distribution $e' = \mathtt{Hash}(a')$ (see for instance [10]), thus the voter must cast its vote for m^* in order to satisfy the coercer.

Solution U_3: using OR-proofs. There are CAI mechanisms proposed (implicitly or explicitly) in the literature [8,6], based on the idea of designated verification, that deal with the problem of coercion.

In the first round of the protocol, $\mathcal{V}$ generates an instance (public element and trapdoor witness) of a hard relation and sends to $\mathcal{P}$ the public element. For instance, if we consider the Discrete Logarithm relation, $\mathcal{V}$ chooses $x \in \mathbb{Z}_q$ at random, computes $y = g^x$ and sends y to $\mathcal{P}$. In the second round of the protocol, $\mathcal{P}$ computes a non-interactive zero-knowledge proof π for the language: "I know r such that $C = \mathsf{Enc}_{\mathsf{pk}}(m; r)$" OR "I know x such that $y = g^x$" . Such a non-interactive (one-shot) proof can be computed by applying the Fiat-Shamir transformation to a Σ-protocol for the OR language above.

CAI verification holds because VD$=\mathcal{P}$ has no way of knowing the trapdoor x, while for CR the voter can use the knowledge of x to generate fake proofs for the coercer. However, CR approach heavily relies on the fact that an (honest) $\mathcal{V}$ actually knows the witness x as clearly stated in [8,6]. Unfortunately this assumption is false in a scenario with strong coercion: the coercer can generate the pair (x, y), give y to $\mathcal{V}$ and force it to run the above protocol with that specific public element y. Since $\mathcal{V}$ does not know x, it cannot simulate proofs and the only valid proof π it can show to the coercer will reveal its vote m.

3.1 On the Necessary Number of Rounds.

We omit the rounds where the voter sends its voting option m to the VD and where VD publishes the ciphertext C. Our study starts at the point, where m and C are already known to both $\mathcal{V}$ and $\mathcal{P}$. In some particular protocols (like the ones that we will describe in Section 4), the exchange of m, C can be integrated with the rest of the interaction aimed at providing CR and CAI.

One round is not enough. Let us imagine a one-round protocol: it consists of $\mathcal{P}$ sending a single message π to $\mathcal{V}$; for the CR property, it should be impossible for the coercer to distinguish (C, m, π) from (C, m^*, π^*). But the same holds for the voter, who has not participated in the protocol at all, which breaks the CAI property because the voter gets exactly the same conviction for any possible voting option.

Two rounds are not enough. Essentially the same idea extends to CAI mechanisms with two rounds: $\mathcal{V}$ first sends some message a to $\mathcal{P}$, in the first round, and finally $\mathcal{P}$ replies with some message π, in the second round. A coercer may force $\mathcal{V}$ to use a specific $\hat{a}$ as the first message of the protocol, so that only transcripts $(\hat{a}, \pi)$ will be accepted by the coercer. In such a case, as in the previous situation with one round, CR property would imply that the protocol does not provide CAI: if the coercer cannot distinguish if $(m, C) \in \mathcal{L}_{\mathcal{R}}$ or if $(m^*, C) \in \mathcal{L}_{\mathcal{R}}$, due to the CR property, then the same holds for $\mathcal{V}$, which breaks the CAI (soundness) property. Here $\mathcal{L}_{\mathcal{R}} = \{(C, m) \mid C \text{ is an encryption of } m\}$ is the ElGamal language.

Therefore, at least three rounds of communication between $\mathcal{V}$ and $\mathcal{P}$ are needed in order to get both the CR and CAI properties. We have not been able to find a simple solution with three rounds of communication. In the next section we describe two solutions which, when implemented in the ElGamal ciphertext case, involve four rounds of communication.

4 CAI+CR: Two Solutions

We describe in this section two ways of achieving both CAI and CR in our considered setting for remote e-voting (without secure channels). For each of the solutions, we first describe it in a generic and informal way, and then we describe

the protocol in detail for the particular case of ElGamal ciphertexts that we are considering as the illustrative example through all this paper.

Solution S_1: committing to challenges. The departing point for the solution S_1 is the unsatisfactory solution U_2 (using a Σ-protocol). The problem with solution U_2 was that a Σ-protocol is only honest-verifier zero-knowledge, which opens the door to coercions like the one described when discussing solution U_2. The solution is easy: use an interactive protocol for the ElGamal language $\mathcal{L}_{\mathcal{R}} = \{(C, m) \mid C \text{ is an encryption of } m\}$ which is full zero-knowledge, and not just honest-verifier zero-knowledge. A way of obtaining such a protocol is to use a well-known technique, described for example in [7]: the verifier $\mathcal{V}$ starts the protocol by committing to the challenge e that he will use later in the Σ-protocol. The prover $\mathcal{P}$ must check that the challenge e is a valid opening of the commitment previously received from $\mathcal{V}$.

There are different possibilities for the commitment scheme employed in this generic solution. For our detailed description in the ElGamal case, we will use Pedersen commitment scheme [11], which needs that an additional generator $h \in G = \langle g \rangle$ of the cyclic group G is published as part of the common public parameters of the election.

Detailed Protocol for ElGamal Ciphertexts. We assume both $\mathcal{P}$ and $\mathcal{V}$ already have access to values q, G, g, h such that $G = \langle g \rangle = \langle h \rangle$ has prime order q. The sending of m from $\mathcal{V}$ to $\mathcal{P}$ and of C from $\mathcal{P}$ to $\mathcal{V}$ can be integrated in the four rounds of the protocol, which works as follows:

1. $\mathcal{V}$ chooses $e, w \in \mathbb{Z}_q$ uniformly at random, computes the commitment $Z = g^e \cdot h^w$ and sends Z and the voting option m to $\mathcal{P}$.
2. $\mathcal{P}$ chooses $r, t \in \mathbb{Z}_q$ uniformly at random, computes $C = (c_1, c_2)$ where $c_1 = g^r$ and $c_2 = m \cdot \mathsf{pk}^r$, and also the pair $a = (A_1, A_2) = (g^t, \mathsf{pk}^t)$. The two pairs C and a are sent to $\mathcal{V}$.
3. $\mathcal{V}$ sends both e and w to $\mathcal{P}$.
4. $\mathcal{P}$ checks if $Z = g^e \cdot h^w$. If this is not the case, $\mathcal{P}$ aborts the protocol. Otherwise, $\mathcal{P}$ computes the value $z = t + e \cdot r \bmod q$ and sends it to $\mathcal{V}$.

For the CAI property, the voter will be convinced if the two following equalities hold: $g^z = A_1 \cdot c_1^e$ and $\mathsf{pk}^z = A_2 \cdot \left(\frac{c_2}{m}\right)^e$.

For the CR property, since the value e must be committed before the pair a is obtained, the problem with unsatisfactory solution U_2 does not exist anymore: the only choices of the voter (or the coercer) are those of e, w in step 1. Even if the coercer requires some specific distribution for these two values, the voter can simulate a valid transcript (m^*, e, w, C, a, z) for the real ciphertext C and any option m^* (possibly selected by the coercer), by choosing e, w with the required distribution (by the coercer), then choosing $z \in \mathbb{Z}_q$ at random, computing $A_1 = g^z \cdot c_1^{-e}$ and $A_2 = \mathsf{pk}^z \cdot \left(\frac{c_2}{m^*}\right)^{-e}$ and finally defining $a = (A_1, A_2)$.

Observe that the above simulator also works for instances (m^*, C) not in the language (i.e. when C is not an encryption of m^*): the outputs of P and $\mathtt{M}$ are indistinguishable under the DDH assumption. This holds for any language

$\mathcal{L} = \{(C, m) \mid C \text{ is an encryption of } m\}$ where the encryption scheme is IND-CPA secure.

Solution S_2: augmented OR proofs. The departing point for the solution S_2 is the unsatisfactory solution U_3: a non-interactive proof for the OR language "C is encryption of m OR I know x such that $y = g^x$". We had already discussed that this idea provides full CR as long as the voter $\mathcal{V}$ really knows the trapdoor x, and convinces the VD of this fact. This proof itself must be simulatable (a Σ-protocol), it cannot be a non-interactive (one-shot) proof; otherwise, the coercer could compute in advance the hard instance (x, y) along with a non-interactive proof of knowledge π_1 of x, and give to the voter just y, π_1, and not the trapdoor x (which would forbid the voter from simulating valid transcripts for coerced voting options).

Detailed Protocol for ElGamal Ciphertexts. We assume that both $\mathcal{P}$ and $\mathcal{V}$ have access to values q, G, g such that $G = \langle g \rangle$ has prime order q, and a secure hash function $H : \{0,1\}^* \to \mathbb{Z}_q$. The protocol works as follows:

1. $\mathcal{V}$ chooses the trapdoor $x \in \mathbb{Z}_q$ uniformly at random and computes the associated public element $y = g^x$ on the one hand. Then he chooses $t \in \mathbb{Z}_q$ uniformly at random and computes $a = g^t$ (first message of the Σ-protocol to prove knowledge of x) on the other hand. The voter $\mathcal{V}$ sends y, a and the voting option m to $\mathcal{P}$.
2. $\mathcal{P}$ chooses $r, e \in \mathbb{Z}_q$ uniformly at random, computes $C = (c_1, c_2)$ where $c_1 = g^r$ and $c_2 = m \cdot \mathsf{pk}^r$, and sends ciphertext C and the challenge e to $\mathcal{V}$.
3. $\mathcal{V}$ computes $z = t + x \cdot e \bmod q$ and sends z to $\mathcal{P}$.
4. $\mathcal{P}$ checks if $g^z = a \cdot y^e$. If this is not the case, $\mathcal{P}$ aborts the protocol. Otherwise, $\mathcal{P}$ computes a non-interactive zero-knowledge proof $\pi = (z_1, z_2, h_1, h_2)$ for the OR language, as follows
   - choose $w_1, z_2, h_2 \in \mathbb{Z}_q$ uniformly at random and compute $A_1 = g^{w_1}$, $A_2 = \mathsf{pk}^{w_1}$ and $Z_2 = g^{z_2} \cdot y^{-h_2}$,
   - compute the challenge as the hash value $h = H(C, y, A_1, A_2, Z_2)$,
   - compute $h_1 = h - h_2 \bmod q$ and $z_1 = w_1 + h_1 \cdot r \bmod q$.

For the CAI property, the voter $\mathcal{V}$ accepts the interaction as convincing if the OR proof π is valid; that is, if the following equality holds:

$$h_1 + h_2 \bmod q \;=\; H\left(C, y,\; g^{z_1} \cdot c_1^{-h_1} \;,\; \mathsf{pk}^{z_1} \cdot \left(\frac{c_2}{m}\right)^{-h_1} \;,\; g^{z_2} \cdot y^{-h_2}\right)$$

For the CR property, the only influence a coercer can have on the voter is in the distribution of the two values x and t of step 1. To simulate an execution of the protocol with the same ciphertext C but for a different voting option m^*, the voter chooses x and t with that distribution. Steps 2 and 3 are performed as in the real protocol. Finally, the voter can simulate the proof π for the OR language by using the knowledge of the trapdoor x, as follows:

- choose $z_1, w_2, h_1 \in \mathbb{Z}_q$ uniformly at random and compute $A_1 = g^{z_1} \cdot c_1^{-h_1}$, $A_2 = \mathsf{pk}^{z_1} \cdot \left(\frac{c_2}{m^*}\right)^{-h_1}$ and $Z_2 = g^{w_2}$,
- compute the challenge as the hash value $h = H(C, y, A_1, A_2, Z_2)$,
- compute $h_2 = h - h_1 \bmod q$ and $z_2 = w_2 + h_2 \cdot x \bmod q$.

5 Conclusion and Remaining Work

This short paper presents the main (positive and negative) results that we have obtained until now when studying the problem of achieving at the same time coercion resistance and cast as intended verifiability. We are currently working on the formalization of these two properties and the formal proof that our solutions in Section 4 satisfy them, as well as on post-quantum secure instantiations (based on lattices) of our solutions.

Acknowledgements

The work is partially supported by the Spanish *Ministerio de Ciencia e Innovación (MICINN)*, under Project PID2019-109379RB-I00.

References

1. Claudia Z. Acemyan, Philip Kortum, Michael D. Byrne, and Dan S. Wallach. Usability of voter verifiable, end-to-end voting systems: Baseline data for helios, prêt à voter, and scantegrity II. In *EVT/WOTE 14 Workshop*, San Diego, CA, August 2014. USENIX Association.
2. Ben Adida. Helios: Web-based open-audit voting. In *Proceedings of the 17th USENIX Security Symposium*, pages 335–348, 2008.
3. Josh Benaloh. Ballot casting assurance via voter-initiated poll station auditing. In *USENIX/ACCURATE Electronic Voting Technology Workshop, EVT'07*, 2007.
4. Michael R. Clarkson, Stephen Chong, and Andrew C. Myers. Civitas: Toward a secure voting system. In *2008 IEEE Symposium on Security and Privacy (S&P 2008)*, pages 354–368, 2008.
5. Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inf. Theory*, 31(4):469–472, 1985.
6. Sandra Guasch and Paz Morillo. How to challenge and cast your e-vote. In *Financial Cryptography and Data Security*, pages 130–145, 2017.
7. Carmit Hazay and Yehuda Lindell. *Efficient Secure Two-Party Protocols: Techniques and Constructions*. Springer-Verlag, Berlin, Heidelberg, 1st edition, 2010.
8. Markus Jakobsson, Kazue Sako, and Russell Impagliazzo. Designated verifier proofs and their applications. In *Advances in Cryptology - EUROCRYPT '96*, pages 143–154. Springer, 1996.
9. Ari Juels, Dario Catalano, and Markus Jakobsson. Coercion-resistant electronic elections. In *Towards Trustworthy Elections, New Directions in Electronic Voting*, pages 37–63, 2010.
10. Tal Moran and Moni Naor. Receipt-free universally-verifiable voting with everlasting privacy. In Cynthia Dwork, editor, *Advances in Cryptology - CRYPTO 2006*, pages 373–392, 2006.
11. Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology - CRYPTO '91*, pages 129–140, 1991.