Amun: Securing E-Voting Against Over-the-Shoulder Coercion

Riccardo Longo[1] [0000-0002-8739-3091] and Chiara Spadafora[2] [0000-0003-3352-9210]

[1] `riccardolongomath@gmail.com`
[2] `c.spadaff@gmail.com`
Department of Mathematics, University Of Trento, 38123 Povo, Trento, Italy

Abstract. In an election where each voter may express P preferences among M possible choices, the Amun protocol allows to secure vote casting against over-the-shoulder adversaries, retaining privacy, transparency, end-to-end verifiability, and receipt-freeness.
Before the election, each voter receives a ballot containing valid and fake tokens: only valid tokens contribute in the final tally, but they remain indistinguishable from the fake ones. Since the voter is the only one who knows which tokens are valid (without being able to prove it to a coercer), over-the-shoulder attacks are thwarted.
We prove the security of the construction under the standard Decisional Diffie Hellman assumption.

Keywords: E-Voting · Over-the-Shoulder Attack · Coercion-Resistance · End-to-End Verifiability · Diffie-Hellman Assumption · Cryptography · Formal Proof of Security.

1 Introduction

Remote voting, thanks to advanced cryptographic techniques, may guarantee higher levels of security with respect to classical paper-based voting. However, being able to vote from home is a double-edged sword: on one hand it may improve turnout by easing the voting process, on the other hand it comes at the expense of the loss of privacy that only a voting booth can guarantee.

Although many systems protect against various adversaries trying to bribe electors, we found that it is more difficult to counter opponents that closely monitor voters during the voting phase (over-the-shoulder attacks). The main mitigation technique against coercion is the usage of fake credentials, which are indistinguishable from real ones but that do not produce valid votes. However, if the adversary keeps under control the voter until the end of the voting period, it becomes impossible to re-vote with the valid credential.

Here we present the Amun[3] protocol, which hides the real choice expressed by a ballot even if an adversary is physically monitoring the elector during

[3] Amun was a major ancient Egyptian deity. The name Amun 𓇋𓏠𓈖𓀭 meant something like "the hidden one" or "invisible".

vote casting. This feature protects the elector against over-the-shoulder attacks without the need to re-vote.

The Amun protocol aims to achieve end-to-end verifiability (also obtaining the cast-as-intended, recorded-as-cast, and tallied-as-recorded properties), transparency, privacy, receipt-freeness, vote-selling and coercion resistance. The base idea beneath the protocol is a generalization of [22], adding support for multiple candidates and more than one choice (which can be exploited to express blank or partial ballots), and forsakes the blockchain infrastructure in favor of a more traditional bulletin board.

In Amun, three authorities share the administration of the election: they setup the parameters, manage voters' registration, and compute the final tally at the end of the voting phase. Privacy is preserved even if an attacker colludes with one authority, limiting their power. Votes are cast by assigning some "voting tokens", generated during the registration, to the candidates. Among these tokens, only a few are valid and express the real preference of the voter, but they are indistinguishable from the other, fake, tokens. This trick disguises the actual choice made by the elector, even if the adversary is watching.

1.1 Related Work

Protocols for electronic election systems have been abundantly proposed in recent years, an incomplete selection may include *Caveat Coercitor* [9], *sElect* [16], *JCJ* [13], *Helios* [1], *Votor* [10], *Ordinos* [15], *VoteAgain* [18], *Remotegrity* [23], *Pretty Good Democracy* [21], *Prêt à Voter* [19].

Many protocols have addressed the problem of vote selling, and many definitions of coercion resistance have been given [7,11,13,14,17]. *Civitas* [4] deals with coercion by allowing voters to vote multiple times via a mechanism of *real* and *fake* credentials. *Selene* [20] associates to every vote a unique tracker, the idea is that every voter, in case of attack, is able to open up its commitment to a fake tracker in order to mock the adversary. *Bingo voting* [2,8] tries to mitigate coercion by using a trusted random number generator. *Belenios* [6] itself is not coercion resistant: voters can keep the randomness used to encrypt the ballot to prove how they voted. This limitation has been overcome with the deployment of *BeleniosRF* [3,5].

1.2 Organization

We present some preliminaries in Section 2, in particular in Section 2.1 we describe what we mean by the term *bulletin board*. We describe our protocol in Section 3 and we provide a proof of security in Section 4. Finally, in Section 5 we draw some conclusions.

2 Preliminaries

The algebraic preliminaries we need to build the protcol are the *Decisional Diffie-Hellmann Assumption*, the *Equality of discrete logarithms ZKP* and a commitment scheme formalized in [22].

For the sake of compactness we use the following notation for the indexes: $[n] = \{i \in \mathbb{N} : 1 \leq i \leq n\}$, $(t_j)_{j \in [m]} = (t_1, \dots, t_m)$.

2.1 Bulletin Board

The concept of *(Web) Bulletin Board* (BB) is well established in literature, as its use in e-voting. A BB can be thought of as an append-only secure broadcast mechanism: published information cannot be modified or removed, and everyone has a consistent view. A more detailed discussion could be found in [12].

2.2 General Requirements for Remote Voting Systems

A secure remote voting systems should satisfy certain requirements before being deployed. In [22] the properties of *Correctness, Fairness, Transparency, Privacy*, and *Verifiability* are formally defined. Concerning Coercion-Resistance, in [11] there is a critical analysis of various definitions. In this first version, the Amun protocol protects against coercers that wish to sway elections towards specific candidates, but is not very effective against the more subtle randomization and forced abstention attacks. In this simplified model, we adapt the definition of Coercion Resistance as follows:

Definition 1 (Vote-Coercion Resistance). *Let $\mathcal{A}$ be a coercer, V_c the set of coerced voters, and $(c_{i,1}, \dots, c_{i,P})$ the choices that $\mathcal{A}$ wants to impose to $v_i \in V_c$. Let Ψ_1 be the scenario in which $\mathcal{A}$ has access only to the final tally. Let Ψ_2 be the scenario in which $\mathcal{A}$ has access to the whole Bulletin Board, and can see all the actions performed by the voters in V_c, with the exception of the ones carried out in a protected environment (or through an untappable channel). A voting protocol is* Vote-Coercion Resistant *if the probability of $\mathcal{A}$ detecting that a voter in V_c has not followed its instruction is the same in Ψ_1 and Ψ_2.*

3 Multi-Candidate Voting System

This section presents our proposal for a remote e-voting protocol that manages an election with N voters, where each one expresses P preferences among M candidates (obviously $P < M$).

The basic idea is that every voter owns M voting tokens (*v-tokens*): P are valid, the others are fake, but only the voter knows which is which. When voting, voters express their preferences assigning the valid *v-tokens* to the chosen candidates and the fake ones to the others. The voter gets a transcript that reports the assignment of the v-tokens to the candidates. The protocol allows for re-voting, before tallying duplicate ballots (i.e. ballots with the same *v-tokens* regardless of their order) are discarded, keeping only the most recent. After the voting phase, when counting the votes, the fake *v-tokens* do not contribute to the tally, so only valid *v-tokens* are counted. The whole process is publicly auditable and fully verifiable, and preserves privacy as long as at most one authority is corrupt.

3.1 Protocol Description

The key components involved in the protocol are:

1. A finite set of voters $V = \{v_i\}_{i\in[N]}$ (where v_i is a pseudonymous id), with $N \in \mathbb{N}$ the number of eligible voters;
2. A finite set of candidates $C = \{c_\ell\}_{\ell\in[M]}$ with $M \in \mathbb{N}$ the number of candidates;
3. Three trusted authorities[4] $\mathcal{A}_0$, $\mathcal{A}_1$, and $\mathcal{A}_2$.
4. One ballot b_i (comprising M *v-tokens*) for every $i \in [N]$, i.e. one for each eligible voter.

Throughout the protocol we implicitly assume that every public value (including a description of the key components presented above) are published in the BB. The protocol is divided into four phases:

Setup The authority $\mathcal{A}_0$ selects a secure group $\mathbb{G}$ of prime order p in which the DDH assumption holds, along with a generator $g \in \mathbb{G}$. Then it publishes $\mathbb{G}$, g, and p. In this phase the authorities use a commitment scheme to commit to their values before publishing them, in order to improve security. We refer to [22] for more details on this primitive and its properties.

Then $\mathcal{A}_0$ performs the following operations:

1. chooses uniformly at random two values k and λ in $\mathbb{Z}_p^*$. $\mathcal{A}_0$ knows that the *v-tokens* computed using k are valid, while the ones computed using λ are fake, but this information is kept secret;
2. chooses uniformly at random $N \cdot M$ distinct values $\bar{z}_{i,\ell} \in \mathbb{Z}_p^*$, with $i \in [N]$, $\ell \in [M]$;
3. finally, $\mathcal{A}_0$ commits to the values g^k, g^λ, and for every $i \in [N]$ it commits to $\big(v_i, (g^{\bar{z}_{i,\ell}})_{\ell\in[M]}\big)$.

It is important that all the values $\bar{z}_{i,\ell}$, k, λ remain private.

The authority $\mathcal{A}_1$ performs the following operations:

1. chooses uniformly at random M distinct values $\alpha'_\ell \in \mathbb{Z}_p^*$, with $\ell \in [M]$, these will be the first half of the candidates' masks;

[4] We use a weak concept of trust here, since the conduct of these authorities can be checked by voters.

2. chooses uniformly at random N distinct values $x'_i \in \mathbb{Z}_p^*$, with $i \in [N]$;
3. chooses uniformly at random two sets of $N \cdot M$ distinct values $z'_{i,\ell}, y'_{i,\ell} \in \mathbb{Z}_p^*$, with $i \in [N], \ell \in [M]$;
4. finally, $\mathcal{A}_1$ commits to the values $g^{\alpha'_\ell}$, $\forall \ell \in [M]$, and for every $i \in [N]$ it commits to the tuple $\left(v_i, g^{x'_i}, (g^{z'_{i,\ell}})_{\ell \in [M]}, (g^{y'_{i,\ell}})_{\ell \in [M]}\right)$.

It is important that all the values α'_ℓ, x'_i, $z'_{i,\ell}$, $y'_{i,\ell}$ remain private.

The authority $\mathcal{A}_2$ performs the following operations:

1. chooses uniformly at random M distinct values $\alpha''_\ell \in \mathbb{Z}_p^*$, with $\ell \in [M]$, these will be the second half of the candidates' masks;
2. chooses uniformly at random N distinct values $x''_i \in \mathbb{Z}_p^*$, with $i \in [N]$;
3. chooses uniformly at random $N \cdot M$ distinct values $y''_{i,\ell} \in \mathbb{Z}_p^*$, with $i \in [N]$, $\ell \in [M]$;
4. Finally $\mathcal{A}_2$ commits to the values $g^{\alpha''_\ell}$, $\forall \ell \in [M]$, and for every $i \in [N]$ it commits to the tuple $\left(v_i, g^{x''_i}, (g^{y''_{i,\ell}})_{\ell \in [M]}\right)$.

It is important that all the values α''_ℓ, x''_i, $y''_{i,\ell}$ remain private.

Once that all the commitments have been published, the authorities can decommit the values:

- $\mathcal{A}_0$ publishes the decommitments for the values g^k, g^λ, alongside all the tuples $\left(v_i, (g^{\bar{z}_{i,\ell}})_{\ell \in [M]}\right) \forall i \in [N]$;
- $\mathcal{A}_1$ publishes the decommitments for the values $g^{\alpha'_\ell} \forall \ell \in [M]$, and the tuples $\left(v_i, g^{x'_i}, (g^{z'_{i,\ell}})_{\ell \in [M]}, (g^{y'_{i,\ell}})_{\ell \in [M]}\right) \forall i \in [N]$;
- $\mathcal{A}_2$ publishes the decommitments for the values $g^{\alpha''_\ell} \forall \ell \in [M]$, and the tuples $\left(v_i, g^{x''_i}, (g^{y''_{i,\ell}})_{\ell \in [M]}\right) \forall i \in [N]$.

To simplify notation we introduce the following definitions for aggregate values for all $i \in [N]$ and $\ell \in [M]$:

$$x_i = x'_i + x''_i, \qquad \alpha_\ell = \alpha'_\ell \cdot \alpha''_\ell, \qquad z_{i,\ell} = \bar{z}_{i,\ell} \cdot z'_{i,\ell}, \qquad y_{i,\ell} = y'_{i,\ell} \cdot y''_{i,\ell}. \quad (1)$$

Registrar Phase For every voter $v_i \in V$ the following steps are performed:

1. Let Alice be the person associated to the voter v_i, note that the authorities do not need to know this association. She goes in a safe and controlled environment where she is identified and authenticated as the eligible and not yet registered voter v_i. In this environment she can interact with all three authorities without fear that any adversary can eavesdrop or interfere.
2. Alice creates a signing key-pair (s_i, K_i) and gives K_i to the authorities proving the knowledge of s_i (e.g. by signing a challenge message), and the authorities associate K_i to v_i in their respective voters lists.

3. $\mathcal{A}_1$ and $\mathcal{A}_2$ prove to Alice with a ZKP the knowledge of the exponents x_i' and x_i'' respectively, corresponding to the values $g^{x_i'}$ and $g^{x_i''}$, that were publicly decommitted at the end of the setup phase;
4. $\mathcal{A}_0$ chooses, for every $i \in [N]$, a random subset $V_i \subset [M]$ such that its cardinality is exactly P, then sets:
$$\sigma_{i,\ell} = \begin{cases} k \iff \ell \in V_i \\ \lambda \iff \ell \notin V_i \end{cases} \tag{2}$$
i.e. the random choice of the V_i determines which tokens will be valid and which will be fake;
5. $\mathcal{A}_0$ takes the (publicly available) values $g^{x_i'}$ and $g^{x_i''}$ and creates the step 0 of the ballot $\bar{b}_{0,i} = (\bar{b}_{0,i,\ell})_{\ell\in[M]}$ where:
$$\bar{b}_{0,i,\ell} = \left(g^{\sigma_{i,\ell}} \cdot g^{x_i'} \cdot g^{x_i''}\right)^{\bar{z}_{i,\ell}} = g^{\bar{z}_{i,\ell}(\sigma_{i,\ell}+x_i)} \qquad \forall \ell \in [M]. \tag{3}$$
Then $\mathcal{A}_0$ sends to $\mathcal{A}_1$ the initial ballot $\bar{b}_{0,i}$ and sends to Alice $\bar{b}_{0,i}$ and V_i.
6. $\mathcal{A}_0$ proves the correctness of computations with the Schnorr ZKP presented in [22]:
   (a) $\mathcal{A}_0$ proves that the $g^{\bar{z}_{i,\ell}\sigma_{i,\ell}}$ are correct (using $\sigma_{i,\ell} = k$ or $\sigma_{i,\ell} = \lambda$) with:
$$\omega = \sigma_{i,\ell}, \quad u = g, \quad z = g^{\sigma_{i,\ell}}, \quad \bar{u} = g^{\bar{z}_{i,\ell}}, \quad \bar{z} = g^{\bar{z}_{i,\ell}\sigma_{i,\ell}} \qquad \forall \ell \in V_i, \tag{4}$$
   (b) then $\mathcal{A}_0$ proves that the $\bar{b}_{0,i,\ell}$ are correct using for all $\ell \in [M]$:
$$\omega = \bar{z}_{i,\ell}, \quad u = g, \quad z = g^{\bar{z}_{i,\ell}}, \quad \bar{u} = g^{\sigma_{i,\ell}} \cdot g^{x_i'} \cdot g^{x_i''}, \quad \bar{z} = \bar{b}_{0,i,\ell}. \tag{5}$$
7. $\mathcal{A}_1$ computes the step 1 of the ballot $\bar{b}_{1,i} = (\bar{b}_{1,i,\ell})_{\ell\in[M]}$ where:
$$\bar{b}_{1,i,\ell} = \left(\bar{b}_{0,i,\ell}\right)^{z_{i,\ell}'} = g^{z_{i,\ell}(\sigma_{i,\ell}+x_i)} \qquad \forall \ell \in [M] \tag{6}$$
and sends it to Alice and to $\mathcal{A}_2$.
8. $\mathcal{A}_1$ proves that the $\bar{b}_{1,i,\ell}$ are correct using:
$$\omega = z_{i,\ell}', \quad u = g, \quad z = g^{z_{i,\ell}'}, \quad \bar{u} = \bar{b}_{0,i,\ell}, \quad \bar{z} = \bar{b}_{1,i,\ell} \qquad \forall \ell \in [M]. \tag{7}$$
9. $\mathcal{A}_2$ chooses uniformly at random a permutation $\pi_i \in \mathrm{Sym}([M])$ and computes the step 2 of the ballot $\bar{b}_{2,i} = (\bar{b}_{2,i,\ell})_{\ell\in[M]}$ where:
$$\bar{b}_{2,i,\ell} = \left(\bar{b}_{1,i,\ell}\right)^{y''_{i,\pi_i^{-1}(\ell)}} = g^{z_{i,\ell}y''_{i,\pi_i^{-1}(\ell)}(\sigma_{i,\ell}+x_i)} \qquad \forall \ell \in [M] \tag{8}$$
and sends it to Alice and to $\mathcal{A}_0$, π_i is sent to Alice and $\mathcal{A}_1$.
10. $\mathcal{A}_2$ proves that the $\bar{b}_{2,i,\ell}$ are correct using:
$$\omega = y''_{i,\pi_i^{-1}(\ell)}, \ u = g, \ z = g^{y''_{i,\pi_i^{-1}(\ell)}}, \ \bar{u} = \bar{b}_{1,i,\ell}, \ \bar{z} = \bar{b}_{2,i,\ell} \quad \forall \ell \in [M]. \tag{9}$$

11. $\mathcal{A}_0$ computes the step 3 of the ballot $\bar{b}_{3,i} = (\bar{b}_{3,i,\ell})_{\ell\in[M]}$ where:

$$\bar{b}_{3,i,\ell} = \left(\bar{b}_{2,i,\ell}\right)^{\frac{1}{\bar{z}_{i,\ell}}} = g^{z'_{i,\ell} y''_{i,\pi_i^{-1}(\ell)} (\sigma_{i,\ell}+x_i)} \qquad \forall \ell \in [M] \tag{10}$$

and sends it to Alice and to $\mathcal{A}_1$.

12. $\mathcal{A}_0$ proves that the $\bar{b}_{3,i,\ell}$ are correct using:

$$\omega = \frac{1}{\bar{z}_{i,\ell}}, \quad u = g^{\bar{z}_{i,\ell}}, \quad z = g, \quad \bar{u} = \bar{b}_{2,i,\ell}, \quad \bar{z} = \bar{b}_{3,i,\ell} \qquad \forall \ell \in [M]. \tag{11}$$

13. $\mathcal{A}_1$ computes the final ballot $b_i = (b_{i,\ell})_{\ell\in[M]}$ where:

$$b_{i,\ell} = \left(\bar{b}_{3,i,\pi_i(\ell)}\right)^{\frac{y'_{i,\ell}}{z'_{i,\pi_i(\ell)}}} = g^{y_{i,\ell}(\sigma_{i,\pi_i(\ell)}+x_i)} \qquad \forall \ell \in [M] \tag{12}$$

and sends it to Alice and publishes on the BB the pair (K_i, b_i).

14. $\mathcal{A}_1$ proves that the $b_{i,\ell}$ are correct using:

$$\omega = \frac{y'_{i,\ell}}{z'_{i,\pi_i(\ell)}}, \; u = g^{z'_{i,\pi_i(\ell)}}, \; z = g^{y'_{i,\ell}}, \; \bar{u} = \bar{b}_{3,i,\pi_i(\ell)}, \; \bar{z} = b_{i,\ell} \quad \forall \ell \in [M]. \tag{13}$$

Note that Alice, thanks to the proofs and the knowledge of the intermediate values, knows which ones are a valid token (the ones with $\sigma_{i,\ell} = k$), but thanks to the random choices of V_i and π_i the authorities cannot distinguish the tokens unless they collude. Moreover the properties of the ZKP allow Alice to forge the transcript changing which tokens are valid, making them useless for proving the validity of a token.

A coercer could bypass the privacy given by the ZKP by forcing Alice to use a predetermined randomness, so that the transcripts are no longer falsifiable and she cannot disguise the valid tokens. For this reason it is advisable to source the randomness from the authorities not normally involved in the ZKP so that a coercer cannot force its choice. For example in Step 6 the randomness for the ZKP challenge should be the XOR of two random strings provided by $\mathcal{A}_1$ and $\mathcal{A}_2$ (note that the challenge remains random even if one of them is corrupt). If this mitigation is in place, it is effectively impossible for Alice to reliably prove the validity of her tokens.

Voting Phase Voters express their preferences by assigning the valid tokens to their chosen candidates, and the fake tokens to the others. This assignment is then signed by voter v_i with s_i, and sent to be published on the BB. Once the voting phase ends, duplicate, incomplete, and forged (i.e. not published by the authorities on the BB during the registration phase or with invalid signatures) ballots are discarded while the remaining are published on the BB, so voters can check that their votes have been correctly registered.

Tallying Once the voting phase is over, the tallying can start.

In order to count the votes, the authorities have to process the tokens received by each candidate, substituting the *voter's masks* $y_{i,\ell}$ with the appropriate *candidate mask* α_ℓ. Suppose that $T \leq N$ participants voted. Without loss of generality, we can assume that only the participants with index $i \in [T]$ voted, while the remaining $N - T$ abstained from voting.

For every $i \in [T]$ let $\phi_i : [M] \longrightarrow [M]$ be the bijective map that associates to each candidate index ℓ the index of the token $b_{i,\phi_i(\ell)}$ that the voter v_i sent to the candidate C_ℓ. Then, for every $i \in [T], \ell \in [M]$, the authorities process the token $b_{i,\phi_i(\ell)}$ by performing the following steps:

1. $\mathcal{A}_1$ computes and publishes the preliminary vote $\bar{t}_{\ell,i}$ as:

$$\bar{t}_{\ell,i} = \left(b_{i,\phi_i(\ell)}\right)^{\frac{\alpha'_\ell}{y'_{i,\phi_i(\ell)}}} = g^{\alpha'_\ell y''_{i,\phi_i(\ell)}(\sigma_{i,\pi_i(\phi_i(\ell))}+x_i)}. \tag{14}$$

2. Any observer could ask for a proof that this computation is correct. $\mathcal{A}_1$ proves that $\bar{t}_{\ell,i}$ is correct using:

$$\omega = \frac{\alpha'_\ell}{y'_{i,\phi_i(\ell)}}, \quad u = g^{y'_{i,\phi_i(\ell)}}, \quad z = g^{\alpha'_\ell}, \quad \bar{u} = b_{i,\phi_i(\ell)}, \quad \bar{z} = \bar{t}_{\ell,i}. \tag{15}$$

3. $\mathcal{A}_2$ then computes and publishes the final vote $t_{\ell,i}$ as:

$$t_{\ell,i} = (\bar{t}_{\ell,i})^{\frac{\alpha''_\ell}{y''_{i,\phi_i(\ell)}}} = g^{\alpha_\ell(\sigma_{i,\pi_i(\phi_i(\ell))}+x_i)}. \tag{16}$$

4. Again, any observer could ask for a proof that this computation is correct. $\mathcal{A}_2$ proves that $t_{\ell,i}$ is correct using:

$$\omega = \frac{\alpha''_\ell}{y''_{i,\phi_i(\ell)}}, \qquad u = g^{y''_{i,\phi_i(\ell)}}, \qquad z = g^{\alpha''_\ell}, \qquad \bar{t} = b_{\ell,i}, \qquad \bar{z} = t_{\ell,i}. \tag{17}$$

Once that all final votes have been computed, the actual tallying is performed.

Let R_ℓ be the number of valid tokens given to the ℓ-th candidate (i.e. the number of preferences received by said candidate), and let F_ℓ be the number of fake tokens given to the ℓ-th candidate. Clearly $T = R_\ell + F_\ell \quad \forall \ell \in [M]$. The count R_ℓ can be computed with the following steps:

1. Both $\mathcal{A}_1$ and $\mathcal{A}_2$ can compute g^{α_ℓ} (as $(g^{\alpha''_\ell})^{\alpha'_\ell}$ and $(g^{\alpha'_\ell})^{\alpha''_\ell}$ respectively). The correctness can be proved by $\mathcal{A}_1$ using:

$$\omega = \alpha'_\ell, \qquad u = g, \qquad z = g^{\alpha'_\ell}, \qquad \bar{u} = g^{\alpha''_\ell}, \qquad \bar{z} = g^{\alpha_\ell}, \tag{18}$$

and by $\mathcal{A}_2$ using:

$$\omega = \alpha''_\ell, \qquad u = g, \qquad z = g^{\alpha''_\ell}, \qquad \bar{u} = g^{\alpha'_\ell}, \qquad \bar{z} = g^{\alpha_\ell}. \tag{19}$$

Practically, each authority publishes half of the values.

2. $\mathcal{A}_0$ computes and publishes $g^{\alpha_\ell k} = (g^{\alpha_\ell})^k$ and $g^{\alpha_\ell \lambda} = (g^{\alpha_\ell})^\lambda$. then proves that $g^{\alpha_\ell k}$ is correct using:
$$\omega = k, \qquad u = g, \qquad z = g^k, \qquad \bar{u} = g^{\alpha_\ell}, \qquad \bar{z} = g^{\alpha_\ell k}, \tag{20}$$
and that $g^{\alpha_\ell \lambda}$ is correct using:
$$\omega = \lambda, \qquad u = g, \qquad z = g^\lambda, \qquad \bar{u} = g^{\alpha_\ell}, \qquad \bar{z} = g^{\alpha_\ell \lambda}. \tag{21}$$
3. $\mathcal{A}_1$ computes $\sum_{i=1}^T x'_i$, and publishes $g^{\alpha_\ell \sum_{i=1}^T x'_i}$.
4. Note that any observer can compute $g^{\sum_{i=1}^T x'_i} = \prod_{i=1}^T g^{x'_i}$, $\mathcal{A}_1$ can prove that $g^{\alpha_\ell \sum_{i=1}^T x'_i}$ is correct using:
$$\omega = \sum_{i=1}^T x'_i, \quad u = g, \quad z = g^{\sum_{i=1}^T x'_i}, \quad \bar{u} = g^{\alpha_\ell}, \quad \bar{z} = g^{\alpha_\ell \sum_{i=1}^T x'_i}. \tag{22}$$
5. Similarly, $\mathcal{A}_2$ computes $\sum_{i=1}^T x''_i$ and publishes $g^{\alpha_\ell \sum_{i=1}^T x''_i}$.
6. Again, any observer can compute $g^{\sum_{i=1}^T x''_i} = \prod_{i=1}^T g^{x''_i}$, and $\mathcal{A}_2$ can prove that $g^{\alpha_\ell \sum_{i=1}^T x''_i}$ is correct using:
$$\omega = \sum_{i=1}^T x''_i, \qquad u = g, \quad z = g^{\sum_{i=1}^T x''_i}, \quad \bar{u} = g^{\alpha_\ell}, \quad \bar{z} = g^{\alpha_\ell \sum_{i=1}^T x''_i}. \tag{23}$$
7. Note that any observer can compute the value:
$$g^{\alpha_\ell (\sum_{i=1}^T x_i + R_\ell k + F_\ell \lambda)} = \prod_{i=1}^T t_{\ell,i}. \tag{24}$$
Given that:
$$g^{\alpha_\ell \sum_{i=1}^T x_i} = g^{\alpha_\ell \sum_{i=1}^T (x'_i + x''_i)} = g^{\alpha_\ell \sum_{i=1}^T x'_i} \cdot g^{\alpha_\ell \sum_{i=1}^T x'_i}, \tag{25}$$
anyone can compute:
$$\left(g^{\alpha_\ell k}\right)^{R_\ell} \cdot \left(g^{\alpha_\ell \lambda}\right)^{F_\ell} = \left(g^{\alpha_\ell \sum_{i=1}^T x_i}\right)^{-1} \cdot g^{\alpha_\ell (\sum_{i=1}^T x_i + R_\ell k + F_\ell \lambda)}. \tag{26}$$
8. Finally, note that now R_ℓ and F_ℓ can be easily computed by brute force. In fact, given a positive integer $T \in \mathbb{N}$ it is possible to represent it in $T + 1$ ways as a sum of two non-negative integers, and the number of valid and fake votes must sum up to the number of actual voters T, so the effort is linear in the number of actual votes.

4 Security Analysis

The goal is to prove that an adversary cannot distinguish between valid and fake *v-tokens* and guess how voters cast their preferences. Since election results are obviously public, we have to avoid some trivial cases in which the adversary can deduce the votes by simply observing the results.

Therefore we assume that the adversary controls one authority and all but two voters, and that these two voters express distinct preferences, in particular we let the adversary select two distinct sets of preferences, but they are randomly assigned to the two voters. The adversary wins the security game if it guesses correctly which voter expressed which preferences.

4.1 Security Model

The security of the protocol will be proven in terms of vote indistinguishability (VI), as detailed in Definition 3.

The security of the protocol will be proved in the presence of a malicious authority, so the simulator in the proof will take on the roles of the two honest authorities and the two voters that the adversary does not control.

To simplify our analysis we assume that the adversary-controlled authority does not intentionally fail decommitments or ZKPs, so the protocol does not abort. This is a reasonable assumption considering the application context, however it is not necessary to attain security. In fact, if the adversary wins the security game with non-negligible advantage, then it must run the protocol smoothly with non-negligible probability (since it outputs its guess once the protocol has correctly terminated).

Definition 2 (Security Game). *The security game for the election protocol proceeds as follows:*

- ***Init.*** *The adversary $\mathcal{A}$ chooses the authority and the $N-2$ users that it will control. This means that the adversary knows which are the valid and fake* v-tokens *of these users. The remaining two users are called* free voters. *The challenger $\mathcal{C}$ takes the role of the other authorities and the free voters.*
- ***Phase 0.*** *$\mathcal{A}$ and $\mathcal{C}$ run the* Setup *and* Registrar *phases of the protocol, interacting as needed.*
- ***Phase 1.*** *The adversary votes with some or all of the voters it controls.*
- ***Challenge.*** *$\mathcal{A}$ selects two distinct sets of preferences $\tilde{P}_0 \neq \tilde{P}_1$, with $\tilde{P}_i \subset [M]$, $\#\tilde{P}_i = P$ for $i = 0, 1$. $\mathcal{C}$ flips a random coin $\mu \in \{0, 1\}$ to determine which preference set the first free voter will use, i.e. $P_1 = \tilde{P}_\mu$ (the second one uses the set $P_2 = \tilde{P}_{\mu\oplus 1}$). Then, $\mathcal{C}$ constructs two random ballot assignment maps $\tilde{\phi}_1, \tilde{\phi}_2 : [M] \longrightarrow [M]$ such that $\tilde{\phi}_i(\ell)$ refers to a valid token if and only if $\ell \in P_i$, for $i = 1, 2$. Finally, $\mathcal{C}$ votes by sending to the candidate C_ℓ the $\tilde{\phi}_1(\ell)$-th token of the first free voter and the $\tilde{\phi}_2(\ell)$-th token of the second free voter, $\forall \ell \in [M]$.*
- ***Phase 2.*** *The adversary votes with some or all of the voters it controls which did not vote in Phase 1.*
- ***Phase 3.*** *$\mathcal{A}$ and $\mathcal{C}$ run the* Tallying *phase of the protocol, and the election result is published. Note that the adversary can request the ZKPs of the correctness of the computations performed by $\mathcal{C}$, and vice versa.*

– ***Guess.*** *The adversary outputs a guess μ' of the coin flip that randomly assigned the voting preferences of the two free voters.*

$\mathcal{A}$ wins if $\mu' = \mu$.

Definition 3 (Vote Indistinguishability). *An E-Voting Protocol with security parameter θ is VI-secure if, for every probabilistic polynomial-time adversary $\mathcal{A}$ that outputs a guess μ' of the coin flip μ (as described in the security game of Definition 2), there exists a negligible function η such that:*

$$\mathbb{P}[\mu' = \mu] \leq \frac{1}{2} + \eta(\theta). \tag{27}$$

In the following theorem we prove our voting protocol VI-secure under the DDH assumption in the security game defined above.

Theorem 1. *If the DDH assumption holds, then the protocol described in Section 3.1 is VI-secure, as per Definition 3.*

Proof. Suppose there exists a polynomial time adversary $\mathcal{A}$, that can attack the scheme with advantage ε. We claim that a simulator $\mathcal{S}$ can be built to play the decisional DH game with advantage $\frac{\varepsilon}{2}$. The simulator starts taking in a DDH challenge:

$$(g, A = g^a, B = g^b, T), \tag{28}$$

with $T = g^{ab}$ or $T = R = g^{\xi}$.

First we consider the case in which the adversary controls $\mathcal{A}_0$, where the simulation proceeds as follows.

– **Init**. The adversary chooses the $N - 2$ users to control. Without loss of generality we may assume that the two free voters are v_1 and v_2.
– **Setup**. $\mathcal{S}$ chooses uniformly at random in $\mathbb{Z}_p^*$ the values $\tilde{x}_i$, $\tilde{\alpha}_\ell$, $\tilde{y}_{i,\ell}$, and $\tilde{z}_{i,\ell}$ for all $i \in [2]$, $\ell \in [M]$, and implicitly sets for all $i \in [2]$, $\ell \in [M]$:

$$x''_i = \tilde{x}_i + (-1)^i b, \qquad \alpha'_\ell = a \cdot \tilde{\alpha}_\ell, \qquad y'_{i,\ell} = a \cdot \tilde{y}_{i,\ell}, \qquad z'_{i,\ell} = a \cdot \tilde{y}_{i,\ell}. \tag{29}$$

$\mathcal{S}$ chooses the other values for the authorities $\mathcal{A}_1$ and $\mathcal{A}_2$ following the protocol. Notice that in the improbable case where $a = 0$ the DDH problem is easily solvable ($g^a = g^{ab} = 1$), otherwise since a and b come from an uniform distribution, then also these implicit values are uniform distributed, so the choices of the simulator are indistinguishable from a real protocol execution. Note also that $\mathcal{S}$ can compute all the values $g^{x''_i}$, $g^{\alpha'_\ell}$, $g^{y'_{i,\ell}}$, $g^{z'_{i,\ell}}$, either normally (when the parameter has been explicitly chosen) or as follows:

$$g^{x''_i} = g^{\tilde{x}_i} \cdot B^{(-1)^i}, \qquad g^{\alpha'_\ell} = A^{\tilde{\alpha}_\ell}, \qquad g^{y'_{i,\ell}} = A^{\tilde{y}_{i,\ell}}, \qquad g^{z'_{i,\ell}} = A^{\tilde{z}_{i,\ell}}. \tag{30}$$

for all $i \in [2]$, $\ell \in [M]$. Therefore, $\mathcal{S}$ can perfectly simulate the setup phase.

- **Registrar Phase**. For the voters v_i with $3 \leq i \leq N$, $\mathcal{S}$ can simulate this phase following the protocol normally (since all relevant parameters have been explicitly chosen), while for $i \in [2]$ $\mathcal{S}$ does the following:
  1. $\mathcal{S}$ proves the knowledge of x'_i, and simulates the ZKP for x''_i (see [22]),
  2. $\mathcal{A}$ computes the initial step of the ballot $\bar{b}_{0,i}$ on behalf of $\mathcal{A}_0$ and proves its correctness. From these proofs $\mathcal{S}$ extracts the values of k, λ, and $\bar{z}_{i,\ell}$ $\forall \ell \in [M]$ (see [22]). Moreover, since $\mathcal{A}_0$ communicates the set of indexes of valid tokens V_i to the voter v_i (that is controlled by the simulator), $\mathcal{S}$ can reconstruct the values of the $\sigma_{i,\ell} \forall \ell \in [M]$.
  3. $\mathcal{S}$ computes step 1 of the ballot $\bar{b}_{1,i} = (\bar{b}_{1,i,\ell})_{\ell \in [M]}$ as:

$$\bar{b}_{1,i,\ell} = A^{\bar{z}_{i,\ell}\tilde{z}_{i,\ell}(\sigma_{i,\ell}+x'_i+\tilde{x}_i)} \cdot T^{\bar{z}_{i,\ell}\tilde{z}_{i,\ell}(-1)^i} \overset{*}{=} g^{z_{i,\ell}(\sigma_{i,\ell}+x_i)} \qquad \forall \ell \in [M] \tag{31}$$

     where $\overset{*}{=}$ of Equation (31) holds iff $T = g^{ab}$ in the DDH challenge. Since it controls the voter v_i, $\mathcal{S}$ does not have to simulate the ZKPs.
  4. $\mathcal{S}$ can perform step 2 on behalf of $\mathcal{A}_2$ normally, then $\mathcal{A}$ computes step 3 on behalf of $\mathcal{A}_0$ and proves its correctness.
  5. Finally $\mathcal{S}$ computes the final ballot $b_i = (b_{i,\ell})_{\ell \in [M]}$ as:

$$b_{i,\ell} = A^{\tilde{y}_{i,\ell}y''_{i,\ell}(\sigma_{i,\pi_i(\ell)}+x'_i+\tilde{x}_i)} \cdot T^{\tilde{y}_{i,\ell}y''_{i,\ell}(-1)^i} \overset{*}{=} g^{y_{i,\ell}(\sigma_{i,\pi_i(\ell)}+x_i)} \tag{32}$$

     where again $\overset{*}{=}$ of Equation (32) holds if and only if $T = g^{ab}$ in the DDH challenge, and the ZKPs can be omitted.
- **Voting**: Phases 1, 2, and the Challenge are performed as in Definition 2.
- **Tallying**. Without loss of generality, suppose that only the v_i with $i \in [T]$ have voted. For $\ell \in [M]$, $\mathcal{S}$ carries on with the simulation as follows:
  1. $\mathcal{S}$ computes the preliminary and final votes on behalf of $\mathcal{A}_1$ and $\mathcal{A}_2$ following the protocol without problems. In fact, for $i \in [2]$, we have that

$$\frac{\alpha'_\ell}{y'_{i,\tilde{\phi}_i(\ell)}} = \frac{a\tilde{\alpha}_\ell}{a\tilde{y}_{i,\tilde{\phi}_i(\ell)}} = \frac{\tilde{\alpha}_\ell}{\tilde{y}_{i,\tilde{\phi}_i(\ell)}} \qquad \forall \ell \in [M], \tag{33}$$

     and these values are known to $\mathcal{S}$.
  2. $\mathcal{S}$ computes and publishes the values $g^{\alpha_\ell} = A^{\tilde{\alpha}_\ell \alpha''_\ell}$ $\forall \ell \in [M]$, and simulates the proofs of correctness.
  3. Finally note that $\mathcal{S}$ can compute:

$$\sum_{i=1}^{T} x''_i = \tilde{x}_1 - b + \tilde{x}_2 + b + \sum_{i=3}^{T} x''_i = \tilde{x}_1 + \tilde{x}_2 + \sum_{i=3}^{T} x''_i, \tag{34}$$

     so for the rest of the tallying phase $\mathcal{S}$ can follow the protocol.
- **Guess** Eventually $\mathcal{A}$ will output a guess μ' of the coin flip performed by $\mathcal{S}$ during the Challenge. $\mathcal{S}$ then outputs 0 to guess that $T = g^{ab}$ if $\mu' = \mu$, otherwise it outputs 1 to indicate that T is a random group element $R \in \mathbb{G}$.

The case in which the adversary controls $\mathcal{A}_1$ and the case in which the adversary controls $\mathcal{A}_2$, proceed similarly. A complete proof can be found in [*citation omitted for double-blind review*].

Essentially, in all three cases when T is not random the simulator $\mathcal{S}$ gives a perfect simulation. This means that the advantage is preserved, so it holds that:

$$\mathbb{P}[\mathcal{S}(g, A, B, T = g^{ab}) = 0] = \frac{1}{2} + \varepsilon. \tag{35}$$

On the contrary, when T is a random element $R \in \mathbb{G}$, every token and vote belonging to the free voters becomes independent from the values that would have been computed by following the protocol (since they are simulated using the random value R), so $\mathcal{A}$ can gain no information about the votes from them, while the tally is always correct. Since the security game is structured in such a way that the tally and the tokens of the other voters (i.e. the values where T is not used in the computation by $\mathcal{S}$) do not give any information about the coin flip μ, we have that:

$$\mathbb{P}[\mathcal{S}(g, A, B, T = R) = 0] = \frac{1}{2}. \tag{36}$$

Therefore, $\mathcal{S}$ can play the DDH game with non-negligible advantage $\frac{\varepsilon}{2}$. □

4.2 General properties of the protocol

The general properties of a vote system introduced in [22] (see Section 2.2), can all be proved for the protocol described in Section 3.1 in a very similar way, since many assumptions on the blockchain made in [22] are covered by our assumptions on the BB (see Section 2.1). Due to space reasons, we include here only the proof that the Amun protocol satisfies Vote-Coercion Resistance as per Definition 1.

Proposition 1 (Vote-Coercion Resistance). *If the DDH assumption holds, then the protocol is vote-selling and coercion resistant, as per Definition 1.*

Proof. In order to comply with the coercer's request, a voter $v_i \in V_c$ has to order the tokens so that the valid ones correspond to $(c_{i,1}, \ldots, c_{i,P})$. Since the Registrar Phase is performed in a protected environment, only v_i knows which tokens are valid, and cannot prove it to $\mathcal{A}$ as discussed in Section 3.1.

Thanks to Theorem 1, if the DDH assumption holds, the protocol has vote-indistinguishability and the only way to determine if a vote expresses a specific choice is to distinguish valid and fake tokens. Since $\mathcal{A}$ cannot do so, all the information that can be gained from the votes is given by the final tally. This means exactly that the probability of $\mathcal{A}$ detecting that a voter in V_c has not followed its instruction is the same in Ψ_1 and Ψ_2. □

5 Conclusions

In this paper we have generalized the two-candidates-one-preference e-voting protocol of [22] into an M-candidates-P-preferences protocol. We have preserved the general construction which uses a system of ZKPs to ensure transparency and full auditability of the whole process, but we have abandoned the blockchain infrastructure in favor of a more traditional bulletin board. The protocol achieves also the same extensive security properties of [22] (proven under the classical Decisional Diffie-Hellman Assumption), including *coercion* resistance.

The notion of coercion resistance is rather intricate, and the differences between definitions are subtle. In its strongest form, coercion resistance includes protection against forced abstention attacks and randomized voting. Although our definition seems weaker, we remark that the most prominent e-voting protocols with stronger defence against coercion assume that there is a moment during the voting phase when the voter is not under control of the attacker. The Amun protocol, instead, protects the voter even if there is constant surveillance from the coercer, thus may be preferable when the voting period is limited. In fact, in this scenario, it is more likely for the attacker to maintain continuous control. Moreover, we believe that randomized vote attacks are the least effective coercion for swaying an election result, a forced abstention may be more effective, but it would require the coercer a considerable effort, considering that more voters have to be controlled in order to achieve an impacting result. In fact, the attacker should identify every coerced voter by requesting a signature, in order to link the voter's identity with a public key and its ballot, as published in the BB.

Compared with the two-candidates protocol, our generalization introduces an additional authority, that is required in order to properly mask the multiple valid and fake tokens in each ballot, so that the system remains secure even if one authority is corrupt. Note that the authorities can perform the setup phase asynchronously, and possible DOS attacks may be mitigated with a long-lasting Registrar phase. We can also adopt the strategy of dividing the authorities in independent triplets that manage restricted pools of voters (much like how large-scale elections are divided in voting districts). This approach limits the damage in case more than one authority is corrupted, speeds up the final step of tallying (whose computational cost is linear in the number of votes), and enhances the overall efficiency by distributing the workload.

Many election systems allow voters to cast a blank ballot or to leave some of the P possible preferences unexpressed. This feature can be easily added to the protocol presented here by simply adding P *dummy* candidates that represent blank choices.

References

1. Adida, B.: Helios: Web-based open-audit voting. In: USENIX security symposium. vol. 17, pp. 335–348 (2008)
2. Bohli, J.M., Müller-Quade, J., Röhrich, S.: Bingo voting: Secure and coercion-free voting using a trusted random number generator. In: International Conference on E-Voting and Identity. pp. 111–124. Springer (2007)

3. Chaidos, P., Cortier, V., Fuchsbauer, G., Galindo, D.: Beleniosrf: A non-interactive receipt-free electronic voting scheme. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. pp. 1614–1625 (2016)
4. Clarkson, M.R., Chong, S., Myers, A.C.: Civitas: Toward a secure voting system. In: IEEE Symposium on Security and Privacy. pp. 354–368 (2008)
5. Cortier, V., Fuchsbauer, G., Galindo, D.: Beleniosrf: A strongly receipt-free electronic voting scheme. IACR Cryptol. ePrint Arch. p. 629 (2015)
6. Cortier, V., Gaudry, P., Glondu, S.: Belenios: a simple private and verifiable electronic voting system. In: Foundations of Security, Protocols, and Equational Reasoning, pp. 214–238. Springer (2019)
7. Gardner, R.W., Garera, S., Rubin, A.D.: Coercion resistant end-to-end voting. In: International Conference on Financial Cryptography and Data Security. pp. 344–361. Springer (2009)
8. Giustolisi, R., Bruni, A.: Privacy-preserving dispute resolution in the improved bingo voting. In: International Joint Conference on Electronic Voting. pp. 67–83. Springer (2020)
9. Grewal, G.S., Ryan, M.D., Bursuc, S., Ryan, P.Y.: Caveat coercitor: Coercion-evidence in electronic voting. In: 2013 IEEE Symposium on Security and Privacy. pp. 367–381. IEEE (2013)
10. Haines, T., Boyen, X.: Votor: conceptually simple remote voting against tiny tyrants. In: Proceedings of the Australasian Computer Science Week Multiconference. pp. 1–13 (2016)
11. Haines, T., Smyth, B.: Surveying definitions of coercion resistance. Cryptology ePrint Archive (2019)
12. Heather, J., Lundin, D.: The append-only web bulletin board. In: International Workshop on Formal Aspects in Security and Trust. pp. 242–256. Springer (2008)
13. Juels, A., Catalano, D., Jakobsson, M.: Coercion-resistant electronic elections. In: Towards Trustworthy Elections, pp. 37–63. Springer (2010)
14. Krips, K., Willemson, J.: On practical aspects of coercion-resistant remote voting systems. In: International Joint Conference on Electronic Voting. pp. 216–232. Springer (2019)
15. Küsters, R., Liedtke, J., Müller, J., Rausch, D., Vogt, A.: Ordinos: a verifiable tally-hiding e-voting system. In: 2020 IEEE European Symposium on Security and Privacy (EuroS&P). pp. 216–235. IEEE (2020)
16. Küsters, R., Müller, J., Scapin, E., Truderung, T.: select: A lightweight verifiable remote voting system. In: 2016 IEEE 29th Computer Security Foundations Symposium (CSF). pp. 341–354. IEEE (2016)
17. Kusters, R., Truderung, T., Vogt, A.: A game-based definition of coercion-resistance and its applications. In: 2010 23rd IEEE Computer Security Foundations Symposium (2010)
18. Lueks, W., Querejeta-Azurmendi, I., Troncoso, C.: VoteAgain: A scalable coercion-resistant voting system. In: 29th USENIX Security Symposium. pp. 1553–1570 (2020)
19. Ryan, P.Y.: A variant of the chaum voter-verifiable scheme. In: Proceedings of the 2005 Workshop on Issues in the Theory of Security. pp. 81–88 (2005)
20. Ryan, P.Y., Rønne, P.B., Iovino, V.: Selene: Voting with transparent verifiability and coercion-mitigation. In: International Conference on Financial Cryptography and Data Security. pp. 176–192. Springer (2016)
21. Ryan, P.Y., Teague, V.: Pretty good democracy. In: International Workshop on Security Protocols. pp. 111–130. Springer (2009)

22. Spadafora, C., Longo, R., Sala, M.: A coercion-resistant blockchain-based E-voting protocol with receipts (2021). In: Advances in Mathematics of Communications. American Institute of Mathematical Sciences, doi:10.3934/amc.2021005 (2021)
23. Zagórski, F., Carback, R.T., Chaum, D., Clark, J., Essex, A., Vora, P.L.: Remotegrity: Design and use of an end-to-end verifiable remote voting system. In: International Conference on Applied Cryptography and Network Security. pp. 441–457. Springer (2013)