UNIVERSITÀ
DI TRENTO

Faster coercion-resistant e-voting by encrypted sorting

Michele Battagliola[1], Diego Aranha[2], Lawrence Roy[2]

[1]Università di Trento, [2]Aarhus University

8th February 2024

Introduction

E-Voting

Electronic voting (e-voting) can be divided into two categories: physical and remote e-voting:

- *Physical e-voting* assumes trusted human supervision of the voters, procedures, hardware and software in polling places.
- *Remote e-voting* does not assume trusted supervision of polling places. A secure remote e-voting scheme is harder to achieve since it has to rely on stronger security assumptions and voters are often required to register in person.

Security Requirements

- *Privacy:* given voter it is impossible to deduce its vote.
- *Universal Verifiability:* the correctness of elections results can be verified by all observers.
- *Cast-as-intended verifiability:* every voter can check that their vote was correctly cast.
- *Tallied-as-recorded verifiability:* anyone can check that cast votes were correctly tallied.

Coercion Resistance

- *Coercion-resistant* protocols should defend voters from attackers that pressure them to vote in a specific way, either through threats or rewards.
- Internet voting increases the threats compared to voting at the polling station.

How To Achieve Coercion Resistance?

- The most common way to achieve it is with *fake credentials*, introduced by the JCJ protocol[1].
- Informally, an adversary must not be able to distinguish whether a credential is real or fake and whether a vote was cast with a real or fake one.

[1] Ari Juels, Dario Catalano, and Markus Jakobsson. 2005. Coercion-resistant electronic elections. In Proceedings of the 2005 ACM workshop on Privacy in the electronic society (WPES '05).

Security Game

Algorithm 3 Real

Require: $\mathcal{A}, k, n_T, t, n_V, n_{\mathcal{A}}, n_C, \mathcal{B}$

$BB \leftarrow \emptyset$
$\mathsf{pk}, \mathsf{sk}_i, h_i \leftarrow \mathtt{Setup}^{\mathcal{A}}(k, n_T, t)$
$V_{\mathcal{A}} \leftarrow A()$
$\{s^i\}_{i\in[1,n_V]}, R \leftarrow \mathtt{Register}(\mathsf{k}, \mathsf{pk}, \mathsf{n_V})$
$(j, \beta) \leftarrow \mathcal{A}(\{s^i\}_{i\in V}], R)$
if $|V| \neq n_A$ or $j \notin [1, n_V] \setminus V_{\mathcal{A}}$ or $\beta \notin [1, n_C] \cup \{\emptyset\}$ **then**
    return 0
end if
$B \leftarrow \mathcal{B}(n_V - n_{\mathcal{A}}, n_C)$
for $(i, *) \in B, i \notin [1, n_V]$ **do**
    $s^i \leftarrow \mathtt{FakeCred}(\mathsf{s}^1)$
end for
$b \xleftarrow{\$} \{0, 1\}$
$\tilde{s} \leftarrow s^j$
if $b == 1$ **then**
    Remove all $(j, *)$ from B
else
    Remove all $(j, *)$ from B but the last
    Replace it with (j, β)
    $\tilde{s} \leftarrow \mathtt{FakeCred}(s^j)$
end if
$\mathcal{A}(\tilde{s})$
for $(i, \alpha) \in B$ **do**
    $M \leftarrow \mathcal{A}(BB)$
    $BB \leftarrow BB \cup \{m \in M | m \text{ valid}\}$
    $BB \leftarrow \{\mathtt{Vote}(c_i, \alpha, \mathsf{pk})\}$
end for
$M \leftarrow \mathcal{A}(BB)$
$BB \leftarrow BB \cup \{m \in M | m \text{ valid}\}$
$X, \Pi \leftarrow \mathtt{Tally}^{\mathcal{A}}(BB, R, \mathsf{pk}, \{h_i, s_i\}, t)$
$b' \leftarrow \mathcal{A}()$
return $b == b'$

Algorithm 4 Ideal

Require: $\mathcal{A}, k, n_V, n_{\mathcal{A}}, n_C, \mathcal{B}$

$V_{\mathcal{A}} \leftarrow A()$
$(j, \beta) \leftarrow \mathcal{A}()$
if $|V| \neq n_A$ or $j \notin [1, n_V] \setminus V_{\mathcal{A}}$ or $\beta \notin [1, n_C] \cup \{\emptyset\}$ **then**
    return 0
end if
$B \leftarrow \mathcal{B}(n_V - n_{\mathcal{A}}, n_C)$
$b \xleftarrow{\$} \{0, 1\}$
if $b == 1$ **then**
    Remove all $(j, *)$ from B
else
    Remove all $(j, *)$ from B but the last
    Replace it with (j, β)
end if
$(v_i)_{i\in V_{\mathcal{A}}}, \beta' \leftarrow \mathcal{A}(|B|)$
if $b == 1$ and $\beta \neq \emptyset$ **then**
    $B \leftarrow B \cup \{(j, \beta')\}$
end if
$B \leftarrow B \cup \{(i, v_i) | i \in V_{\mathcal{A}}, v_i \in [1, n_C]\}$
$X \leftarrow \mathtt{result}(\mathtt{cleanse}(B))$
$b' \leftarrow \mathcal{A}(X)$
return $b == b'$

Actors

- *Trustees:* a set of n_T authorities that performs the cleansing and the tally. It is assumed that there are at most t dishonest trustees, where t of the encryption protocol used.
- *Registrars:* a second set of n_R authorities that provide credentials to voters. For coercion resistance it is assumed that all of them are honest. For verifiability only one of them need to be honest.
- *Public Board:* an append-only list of data, where all the other participants can write. The contents of the board can be read by anyone at any time, and the board is assumed to be honest.

Protocol Overview

- Voters receive their credential from the registrars.
- The registrars encrypt all the authorized credentials and publish them on a public register.
- Voters cast their vote on the bulletin board, encrypting the credential and their choice.
- During the tally, votes with the same credential and votes with a credential not in the public register are discarded.

JCJ and CHide

- JCJ protocol leaks the number of revotes and votes with wrong credentials separately. *This allows attacks.*
- CHide[2] solves the issue using bitwise encrypted credentials, that allows for more flexible computation.
- Both of them[3] have quadratic complexity in the tally, since they need to compare each credential with all the authorized ones and the other ones in the bulletin board.

[2] Véronique Cortier, Pierrick Gaudry and Quentin Yang. 2023. Is the JCJ voting system really coercion-resistant?

[3] CHide authors independently proposed an improvement to solve the issue.

Our protocol

Overview

- Our protocol maintains the same registration and voting phase of CHide.
- The cleansing and tally phase is drastically improved, to a complexity of $O(n \log(n))$, thanks to a preliminary sorting.
- We use ElGamal encryption as well as a verifiable mixnet.

To reduce the complexity we can sort the votes based on the credentials. In this way we need only to compute one comparison for each credential.

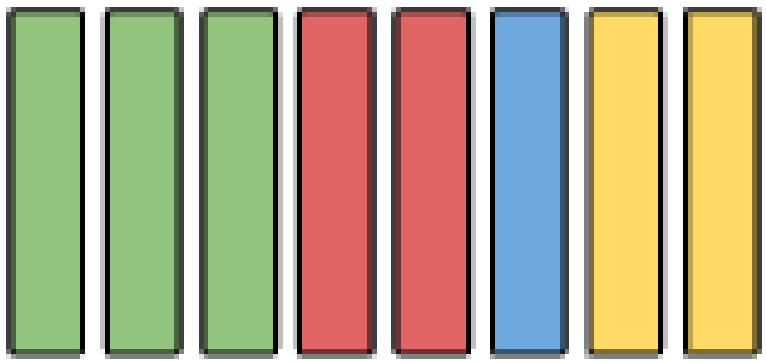

Figure: The ballot board ordered by credential.

Including Registered Credentials

To reduce the complexity even more, we can also include the registered credentials in the sorting. To distinguish them from real votes we add a *flag*: an encryption of 1 at the end of every registered credential and an encryption of 0 at the end of every vote credential.

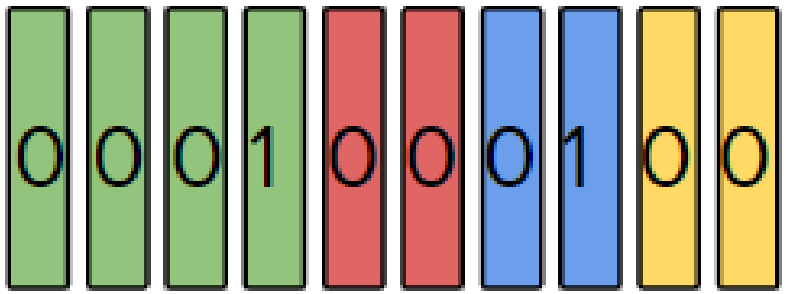

Figure: Final ordered list.

Cleansing

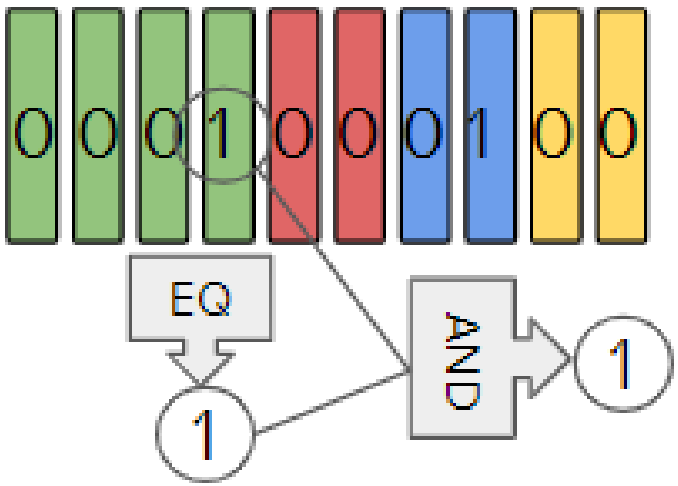

Figure: A vote with an authorized credential to keep.

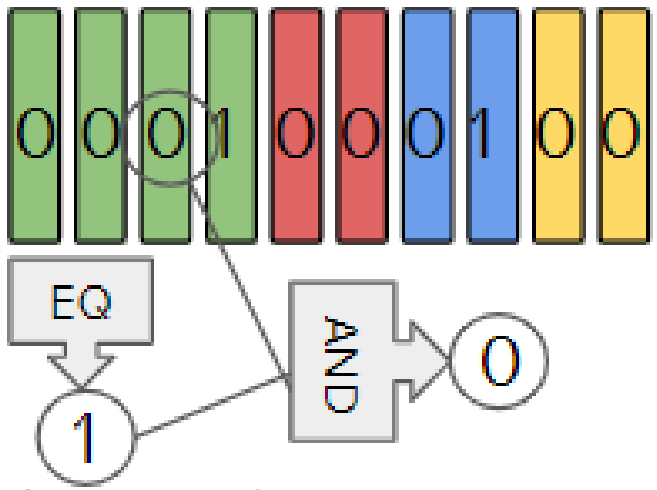

Figure: A duplicate vote to discard.

Algorithm 1 CGate protocol

Require: X, Y encryptions of x, y, with $y \in \{0, 1\}$, number of participants a.
Ensure: Z an encryption of xy.

1: Compute $Y_0 = E_{-1} \cdot Y^2$ and set $X_0 = X, a = t + 1$
2: **for** $i = 1$ to a **do**
3: Participant P_i picks $r_1, r_2 \in \mathbb{Z}_q$ and $s \in \{-1, 1\}$ randomly
4: P_i computes $X_i = \mathsf{ReEnc}(X_{i-1}^s, \mathsf{pk}; r_1)$ and $Y_i = \mathsf{ReEnc}(Y_{i-1}^s, \mathsf{pk}; r_2)$
5: P_i produces a ZKP π_i that X_i and Y_i are well formed
6: P_i reveals X_i, Y_i and π_i
7: **end for**
8: $P_1, ..., P_a$ verify all the proofs. Let $\Pi = (X_1, Y_1, \pi_1)||...||(X_a, Y_a, \pi_a)$.
9: $P_1, ..., P_a$ jointly rerandomize X_a, Y_a to get X', Y', producing transcript Π^{ReEnc}
10: $P_1, ..., P_a$ jointly compute $y_a = \mathsf{Dec}(Y')$ and transcript Π^{Dec}
11: **return** $Z = (XX'^{y_a})^{\frac{1}{2}}$ and verification transcript $(y_a, \Pi^{\mathsf{Dec}})||(X', Y', \Pi^{\mathsf{ReEnc}})||\Pi$

CGate

- When both x, y are bits, $\texttt{CGate}(X, Y)$ outputs an encryption of $\texttt{AND}(x, y)$.
- The $\texttt{NOT}$ operator can be computed as $\texttt{NOT}(X) = \texttt{Enc}(1)X^{-1}$.
- With $\texttt{AND}$ and $\texttt{NOT}$ we can compute equality and the less-than operator: $\texttt{Eq}(X, Y) = \texttt{NOT}(XY/\texttt{CGate}(X, Y)^2)$, $\texttt{Less}(X, Y) = Y/\texttt{CGate}(X, Y)$.
- Let a, b be two values and $A_1, ..., A_k$ and $B_1, ..., B_k$ their bitwise encryptions. To compute the encryption of $a < b$ we do: $L_0 = 0$, $L_i = \texttt{Less}(A_i, B_i) \cdot \texttt{CGate}(L_{i-1}, \texttt{Eq}(A_i, B_i))$. At the end L_k is the encryption of $a < b$.

Cleansing

For each vote, to decide whether to keep or discard it, authorities:

- Compute the equality between the vote credential and the vote credential of the next vote.
- Compute the `CGate` between the result of the previous step and the *flag* in the vote credential of the next vote.
- If the result is an encryption of 1 the vote is counted, otherwise it is discarded.

Security

- A verifiable mixnet is used both before the sorting and after the cleansing phase.
- Thanks to the mixnets the result of each comparison in the sorting can be decrypted.
- The real execution is indistinguishable from the ideal one.
- A lot of ZKPs for mixnets, `CGate` and decryption.

What's next?

- Bitwise encrypted credentials are very long.
- (Threshold) Class Group Encryption[4] has promising properties and allows for multiplication of plaintexts.

[4] Lennart Braun, Ivan Damgård, and Claudio Orlandi. 2023. Secure Multiparty Computation from Threshold Encryption Based on Class Groups. In Advances in Cryptology – CRYPTO 2023.

The key idea of Class Group Encryption[5] is to use a group in which the decisional Diffie-Hellman problem is hard, whereas it contains a subgroup in which the discrete logarithm problem is easy.

[5] Guilhem Castagnos, Fabien Laguillaumie. 2015. Linearly Homomorphic Encryption from DDH. Topics in Cryptology CT-RSA 2015.

Centralized Protocol

Algorithm $\mathsf{KeyGen}(1^\lambda)$

1. $(B, n, p, s, g, f, G, F) \xleftarrow{\$} \mathsf{Gen}(1^\lambda, 1^\mu)$
2. Pick[a] $x \xleftarrow{\$} \{0, \ldots, Bp-1\}$ and set $h \leftarrow g^x$
3. Set $pk \leftarrow (B, p, g, h, f)$ and $sk \leftarrow x$.
4. Return (pk, sk)

Algorithm $\mathsf{Encrypt}(1^\lambda, pk, m)$

1. Pick $r \xleftarrow{\$} \{0, \ldots, Bp-1\}$
2. Compute $c_1 \leftarrow g^r$
3. Compute $c_2 \leftarrow f^m h^r$
4. Return (c_1, c_2)

[a] As n will be unknown in the sequel, x is picked at random in $\{0, \ldots, Bp-1\}$

Algorithm $\mathsf{Decrypt}(1^\lambda, pk, sk, (c_1, c_2))$

1. Compute $M \leftarrow c_2 / c_1^x$
2. $m \leftarrow \mathsf{Solve}(p, g, f, G, F, M)$
3. Return m

Algorithm $\mathsf{EvalSum}(1^\lambda, pk, (c_1, c_2), (c_1', c_2'))$

1. Compute $c_1'' \leftarrow c_1 c_1'$ and $c_2'' \leftarrow c_2 c_2'$
2. Pick $r \xleftarrow{\$} \{0, \ldots, Bp-1\}$
3. Return $(c_1'' g^r, c_2'' h^r)$

Algorithm $\mathsf{EvalScal}(1^\lambda, pk, (c_1, c_2), \alpha)$

1. Compute $c_1' \leftarrow c_1^\alpha$ and $c_2' \leftarrow c_2^\alpha$
2. Pick $r \xleftarrow{\$} \{0, \ldots, Bp-1\}$
3. Return $(c_1' g^r, c_2' h^r)$

Threshold Encryption

Exploiting the *easy subrgroup*, it is possible to design a threshold variant of the protocol that allows for multiplication of ciphertexts without needing to decrypt[6].

[6] Lennart Braun, Ivan Damgård, Claudio Orlandi. 2023. Secure Multiparty Computation from Threshold Encryption Based on Class Groups. CRYPTO 2023.

Current Work

- We are confident about all the ZKP needed.
- We designed a protocol slightly more efficient than classic JCJ, with complexity $n * R$.
- Better efficiency seems still far away.

Thank you

Thank you for your the attention!

michele.battagliola@unitn.it