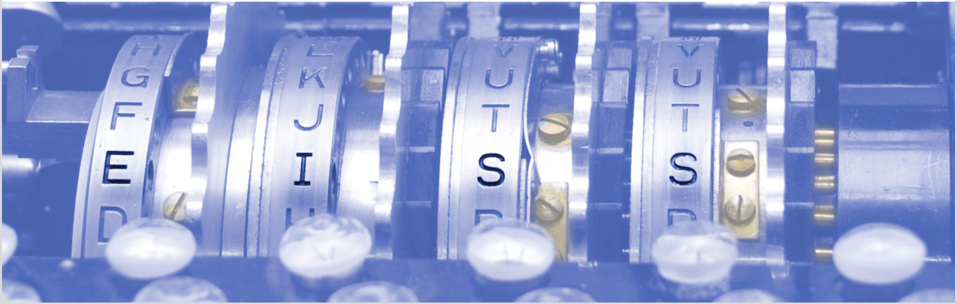
Coercion-Resistant Electronic Elections with Write-In Candidates

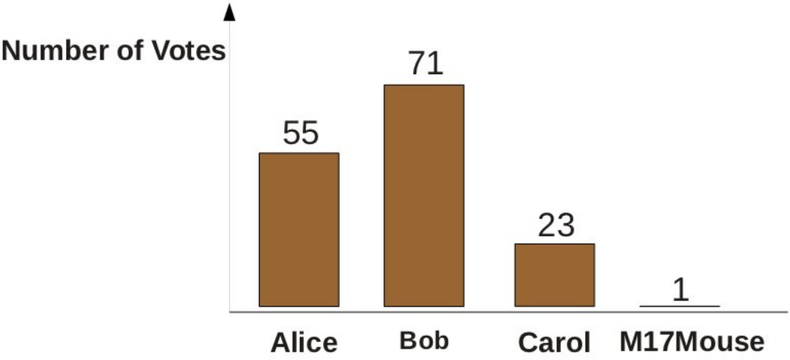
2012-08-06

Carmen Kempka

INSTITUTE OF CRYPTOGRAPHY AND SECURITY

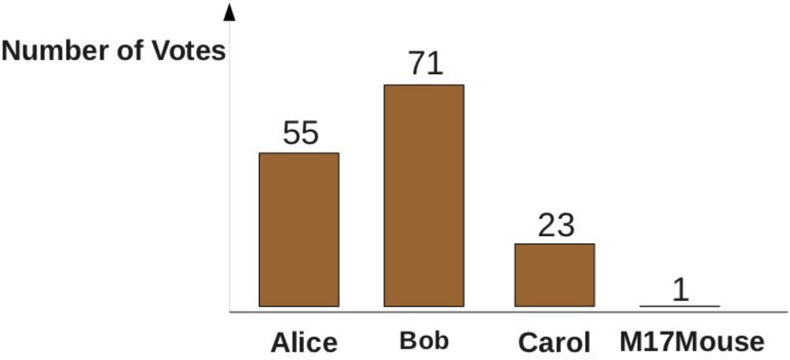


Election with Write-In Candidates



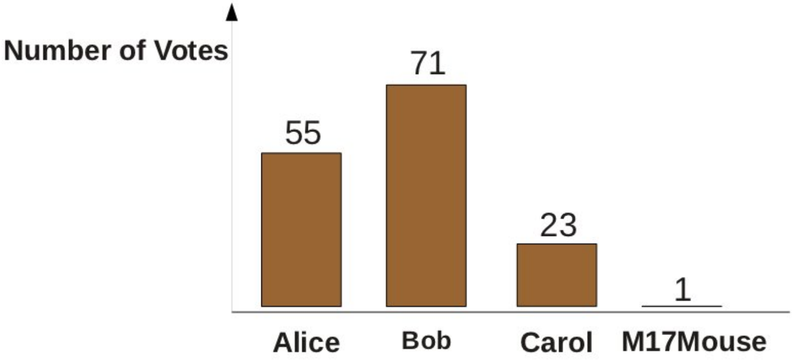
If the tally is published directly, forced abstention attacks are always possible. ⇒ Coercion-resistance is unachievable with write-in candidates.

Election with Write-In Candidates



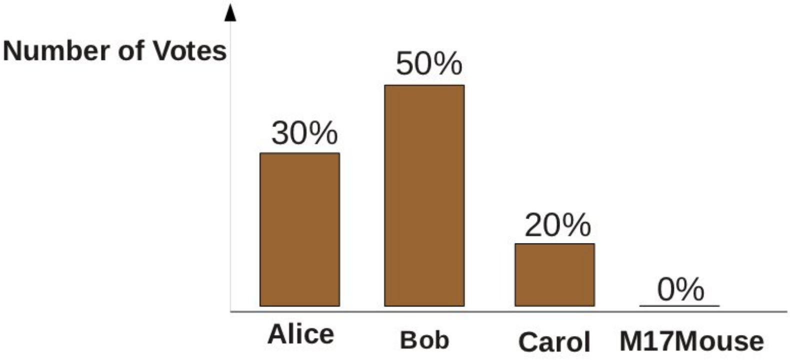
If the tally is published directly, forced abstention attacks are always possible. ⇒ Coercion-resistance is unachievable with write-in candidates.

Election with Write-In Candidates



If the tally is published directly, forced abstention attacks are always possible. ⇒ Coercion-resistance is unachievable with write-in candidates.

Election with Write-In Candidates



If the tally is represented in a fuzzy way, coercion resistance becomes possible.

Contribution of this Work

- A formalization of fuzziness
- A construction of election schemes that provide end-to-end verifiability without revealing the exact tally
- Coercion-resistant write-in support for Bingo Voting

Some Notions

- A **tally** T is represented by a **representation** R.
- A representation R represents the tallies in a set T_R
- A set $\mathcal{R}$ of representations is **complete**, if every tally has a representation $R \in \mathcal{R}$.
- Two tallies are δ-**neighboured**, if each candidate's numbers of votes in the two tallies do not differ by more than δ.

μ, δ-Fuzzable Election

μ, δ*-fuzzable*:
Let $\mathcal{T}$ be the set of all possible tallies of a given election. We call this election μ, δ-*fuzzable* if there is a complete set $\mathcal{R}$ of representations such that for each representation $R \in \mathcal{R}$ and its set of represented tallies $T_R \subseteq \mathcal{T}$:

1. All elements in T_R are δ-neighbored to each other.
2. For each candidate there are at least μ different values for his number of votes in T_R.

μ, δ-Fuzzy Election Scheme

- $\mathcal{T}$: set of all possible tallies
- $\mathcal{R}$: complete set of representations
- T_R: subset of $\mathcal{T}$ represented by $R \in \mathcal{R}$

μ, δ***-fuzzy*****:**
An election scheme is μ, δ-*fuzzy*, if applied to a μ, δ-fuzzable election, for each representation $R \in \mathcal{R}$:

1. The scheme's proof of correctness proves that the tally lies in T_R.
2. No other information is revealed by this proof or any published data.

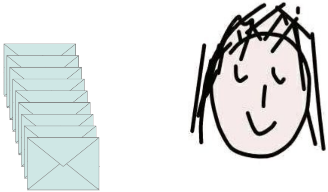
General Construction of a Fuzzy Election Scheme

Preconditions:

- Shuffle-based or homomorphic underlying election scheme
- Existence of a trusted authority
- Votes are cast encrypted
- The trusted authority can open those encrypted votes

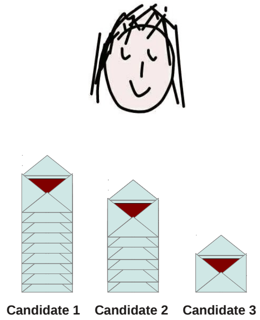
Fuzzy Election Scheme

A trusted authority gets the encrypted votes



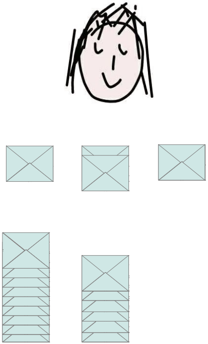
Fuzzy Election Scheme

The trusted authority computes the tally in secret



Fuzzy Election Scheme

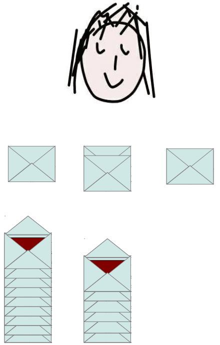
The trusted authority chooses a fuzzines vector $f = (f_1, f_2, \ldots)$, according to which she takes aside a small amount of votes.



	Candidate 1	Candidate 2	Candidate 3
Votes:	9	7	1
Fuzz:	1	2	1

Fuzzy Election Scheme

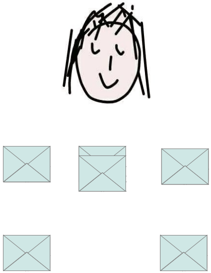
The trusted authority openes and publishes the remaining votes, according to the underlying election scheme.



	Candidate 1	Candidate 2	Candidate 3
Votes:	9	7	1
Fuzz:	1	2	1

Fuzzy Election Scheme

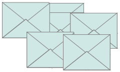
The trusted authority adds $\mu - f_i$ new votes for each candidate.



	Candidate 1	Candidate 2	Candidate 3
Votes:	9	7	1
Fuzz:	1	2	1
New:	1	0	1

Fuzzy Election Scheme

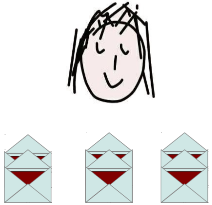
The trusted authority proves that there are μ for each candidate.



	Candidate 1	Candidate 2	Candidate 3
Votes:	9	7	1
Fuzz:	1	2	1
New:	1	0	1

Fuzzy Election Scheme

The trusted authority proves that there are μ for each candidate.



	Candidate 1	Candidate 2	Candidate 3
Votes:	9	7	1
Fuzz:	1	2	1
New:	1	0	1
Repr.:	8-10	5-7	0-2

μ, μ-Fuzzy Bingo Voting

Bingo Voting:

- End-to-end-verifiable voting scheme
- Relies on a trusted random number generator
- The voting machine has to be trusted for secrecy, but not for correctness
- The original scheme does not support write-in candidates

Pre-Voting Phase

For each candidate, dummy votes are created (and kept secret) that are later used on the receipt to conceal the real vote:

$com(213)$	$com(683)$	$com(172)$
$com(769)$	$com(579)$	$com(413)$
$com(145)$	$com(123)$	$com(756)$
Alice	**Bob**	**Write-In**

Published on a public bulletin board:

- Commitments to those dummy votes
- A proof that each candidate got the same amount of dummy votes

Pre-Voting Phase

For each candidate, dummy votes are created (and kept secret) that are later used on the receipt to conceal the real vote:

$com(213)$	$com(683)$	$com(172)$
$com(769)$	$com(579)$	$com(413)$
$com(145)$	$com(123)$	$com(756)$
Alice	**Bob**	**Write-In**

Published on a public bulletin board:

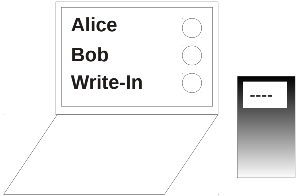
- Commitments to those dummy votes
- A proof that each candidate got the same amount of dummy votes

Voting Phase

The dummy votes are known to the voting machine:

213	683	172
769	579	413
145	123	756
Alice	**Bob**	**Write-In**

Now the voter votes:

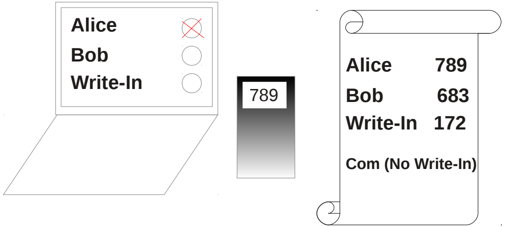


Voting Phase

The dummy votes are known to the voting machine:

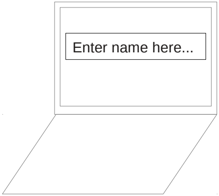
213		
769	579	413
145	123	756
Alice	**Bob**	**Write-In**

Now the voter votes:



Voting Phase

The voter can also input names to be included in the representation of the tally without voting for them.



Post-Voting Phase

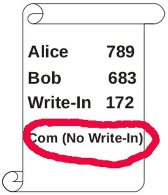
The tally is mirrored by the remaining dummy votes:

$com(769)$		
$com(145)$	$com(123)$	$com(756)$
Alice	**Bob**	**Write-In**

Tally of the regular candidates (non-fuzzy):

- Unveil and publish all remaining dummy votes
- Publish all receipts
- Publish a proof that on each receipt, exactly one random number is fresh and the others are dummy votes

Post-Voting Phase



Tally of the write-in candidates (non-fuzzy):

- Mix and open the commitments to the write-in names
- Publish a proof of correct mixing

Post-Voting Phase

Tally of the regular candidates (fuzzy):

1. Step 1: Publish a proof that on each receipt, exactly one random number is fresh and the others are dummy votes.
2. Step 2: Compute and publish a representation of the tally
3. Step 3: Prove the correctness of the tally representation

Post-Voting Phase

Step 2: Publish a fuzzy representation of the tally:

1. Compute tally $T = (t_1, t_2, t_3)$ in secret.
2. Choose vector $f = (f_1, f_2, f_3)$ with:
   - $\sum_{i=1}^{n} f_i \geq \mu$
   - $0 \leq f_i \leq \min t_i, \mu\}$ for all i
3. Publish $R = (t_1 - f_1, t_2 - f_2, t_3 - f_3)$ as the representation of the tally

Post-Voting Phase

Step 3: Prove correctness of the fuzzy tally representation:

1. For all i: Open $t_i - f_i$ unused commitments to dummy votes of the ith candidate
2. For all i: Create $\mu - f_i$ new commitments for the ith candidate, contents indistinguishable to the dummy votes.
3. Shuffle and open the remaining unopend dummy votes and the new commitments with a proof of correct shuffling. This proves that:
   - There are exactly μ commitments for each candidate
   - Each candidate i got between R_i and and $R_i + \mu$ votes

Post-Voting Phase

Karlsruhe Institute of Technology

Tally of the write-in votes: Analogous to the regular candidates

Conclusion

- Coercion resistance with write-in candidates is possible.

Open Problems and Future Work

Karlsruhe Institute of Technology

What is not covered by this notion of fuzziness:

- Group Coercion
- Pattern Voting

Future work:

- Fuzzy schemes that need no trusted authority
- Relation between fuzziness and database anonymity

Thank you

Thank you

Paper

Paper to this talk:
"Coercion-Resistant Electronic Elections with Write-In Candidates", Carmen Kempka, EVT/WOTE 2012