# E-voting with Vector Ballots : Homomorphic Encryption with Writeins and Shrink-and-Mix networks

Aggelos Kiayias
University of Connecticut
joint work with
Moti Yung
Columbia University

## Motivation.

## Three Basic Paradigms to Cryptographic E-voting

- The Mix-net Approach
  - D. Chaum, 1982.
- The Homomorphic Encryption Approach.
  - J. Benaloh, 1986.
- The Blind Signature Approach.
  - Fujiyoka, Ohta, Okamoto, 1992.

## Three+2 Basic Properties

- "Universal Verifiability"
  - Anybody (the voters and any interested party) can verify that the tally includes all submitted votes. (challenging even assuming robust voter-system interaction – no matter how implemented).
- "Efficient Tallying."
  - Tallying (and tally verification) does not take "too long." [tallying = post-ballot-casting process]
- "Writein Capability"
  - Voters are allowed to cast ballots with any candidate of their choice.

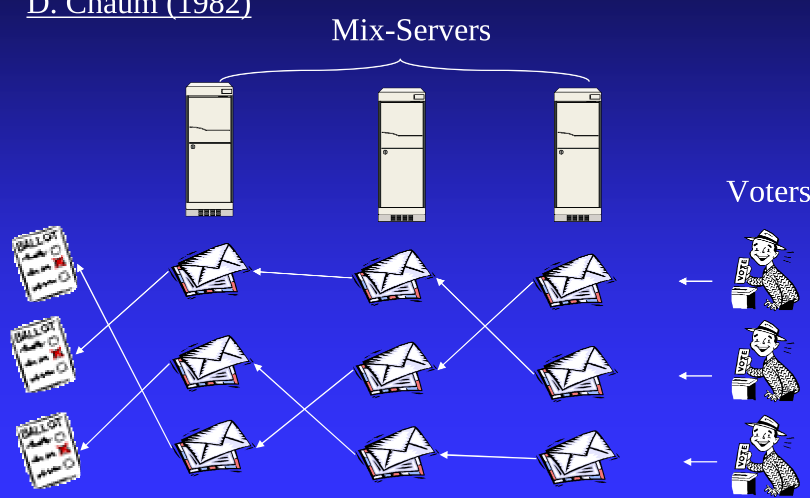
(also: Voter Privacy and prevention of Double Voting.)

# Question:

- How do the three basic approaches perform with respect to the three basic properties?

# Mix-net Approach

D. Chaum (1982)

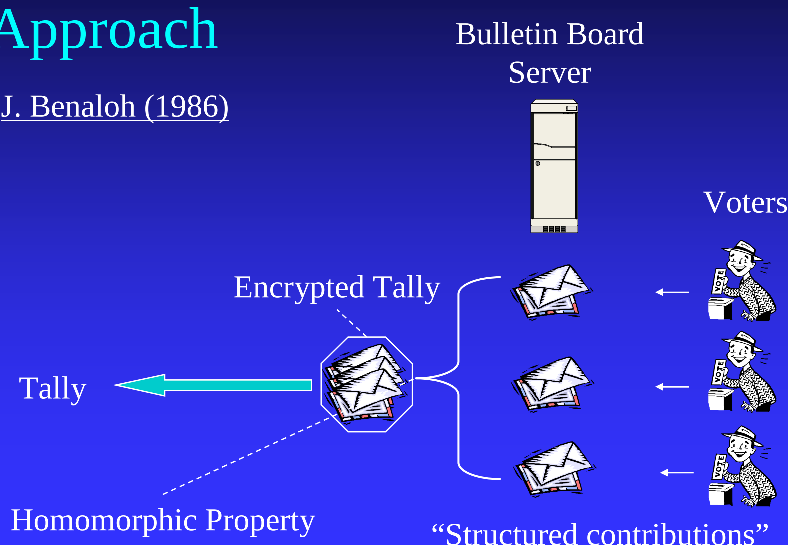


## Mix-net Approach, II

- voter privacy and double voting ok.
- The mix-net approach allows writeins naturally.
- It achieves universal verifiability by employing a robust mix:
  - Everytime you apply a mixer, the mixer has to prove that it didn't remove or modify any ballot.
- The bad news: mix-proofs are long / cumbersome to verify. Recent works on "partial verifying" promising but still not as efficient/ robust as non-mix approaches.

## Homomorphic Encryption Approach

J. Benaloh (1986)



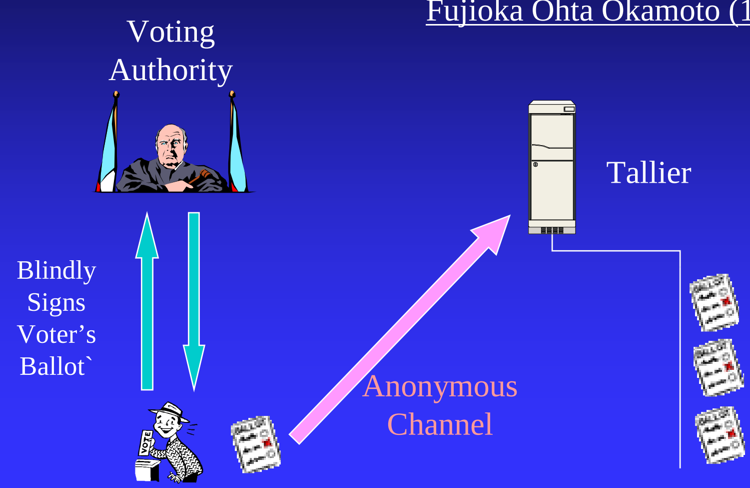
## Homomorphic Encryption, II

Voter Privacy and Double Voting ok.

- Efficient Tallying!
  - Compression operation very efficient.
- Universal Verifiability.
  - Based on voters’ proof and verification of the compression operation + proof of opening the ciphertext.
- The Bad news: no writeins.
  - Problem is inherent.
    information theoretic limitation of compressibility.

## Blind Signature Approach

Fujioka Ohta Okamoto (1992)



## Blind Signature Approach, II

- Double voting and voter privacy ok.
- Writeins are naturally allowed (the scheme is quite generic).
- Tallying is efficient (e.g. anonymous channel implementation through the employment of a non-robust mix is reasonably efficient).
- Bad news: universal verifiability is lacking…
  - Relies on voter for verifiability.
  - *how do I know that other voters check their votes off-line?*

## The state of things.

- No cryptographic e-voting approach beats the other two w.r.t. the properties of “efficient tallying”, “universal verifiability” and “writein capability.”

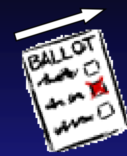
## Our solution

## The present work:

- Develops a new (cryptographic) e-voting approach that achieves the three properties.

- Key issue: understand the existing machinery.
  - Homomorphic encryption: good for fast tallying. Limited in terms of writein capability.
  - robust mix-nets: great for writeins votes but inefficient when applied to the total sum of votes.

## Vector Ballots




- Comprised out of three components:
  - The predetermined candidate component.
  - The Flag component.
  - The writein component.
- All encrypted.

## Vector Ballots, II

anatomy

Description of homomomorphic encryption function E

EXAMPLE: Voting among c candidates

$$Choices = \{1, M, M^2, \ldots, M^{c-1}\}$$

$$M > \#voters = N$$

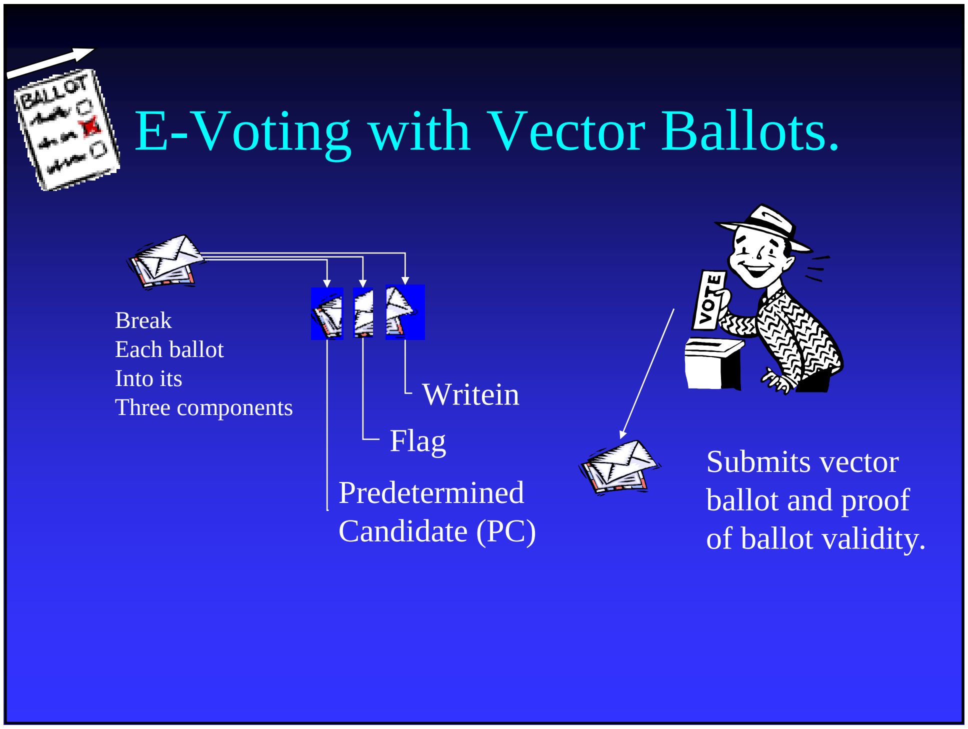
| Vote for $j$-th candidate | Writein vote |
|---|---|
| $E(M^{j-1}), E(0), E(0)$ | $E(0), E(1), E(writein)$ |

# Key Issues in Vector Ballots

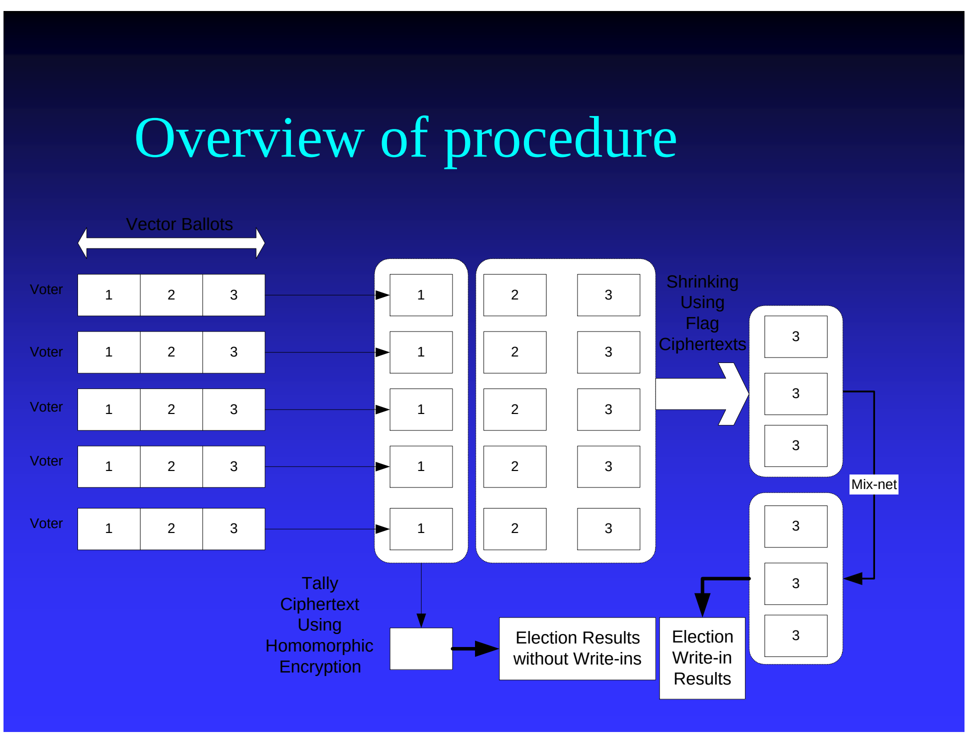
- Uniformity: Each vector-ballot should be indistinguishable (independently on the way the voters goes, predetermined or writein).
- Ballot Consistency (verification)
  - predetermined candidate component (PC) is in *Choices*
  - Make sure that in each ballot it is mutually exclusive for the voter to use the “” or the “writein” component.
  - If the writein component is used the predetermined candidate component must be 0.
  - If the predetermined candidate component is used the writein component must be 0.
  - Also the flag ciphertext should be 1 iff the writen component is used.
  - “0” is not a valid writein choice (sorry).

# How to deal with the key-issues:

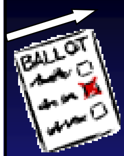
- For <u>uniformity</u> we rely on the semantic security of the underlying encryption mechanism.
- For <u>consistency</u> we develop the appropriate (NIHVZK) proofs of knowledge that the voter must append to his encrypted vector ballot.



E-Voting with Vector Ballots.
BALLOT
Break
Each ballot
Into its
Three components
Writein
Flag
Predetermined
Candidate (PC)
VOTE
Submits vector
ballot and proof
of ballot validity.

Overview of procedure
Vector Ballots
Voter
Voter
Voter
Voter
Voter
1
2
3
Shrinking
Using
Flag
Ciphertexts
Mix-net
Tally
Ciphertext
Using
Homomorphic
Encryption
Election Results
without Write-ins
Election
Write-in
Results



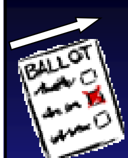

# E-voting with vector Ballots, II

Apply homomorphic encryption compression into the PC components (which essentially is adding the plaintexts)

observe:

$$\sum_{j=1,\ldots,N} v_j = d_0 + Md_1 + \ldots + M^{c-1} d_{c-1}$$

$d_j =$ # of votes won by $j$-th candidate.



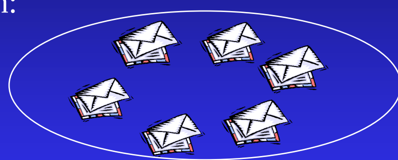

# E-voting with vector ballots, III

- PC results most likely reveal winner of the elections. Writein tallying reduced to an “off-line” operation
- This already makes system more efficient.
- But we can go even more efficient than that.

## Shrink and Mix networks

- New notion that is suitable for Vector-Ballot elections.
- Isolate the flag ciphertexts from each vector ballot.

Question:



Does
This batch of encrypted
Ballots contain
a writein ?

Authorities compress (relying on homomorphic encryption) a **batch** of flag ciphertexts and decrypt it: this allows to compute the # of writeins in a batch of voters.

Loss of privacy minimal (choose comfortably large enough batches) … Notion of “Security Perimeter”`

## Shrink and Mix Networks, II

- Clearly (in most elections) the majority of the ballots are of the PC type.
- SHRINKING : Authorities divide set of writein components into batches and throw away all batches that contain no writein.
- With writein probability 1/100 and batch size = 20, SHRINKING will throw away 81% of all (empty) writein components.

## Shrink and Mix Networks, III

- After the set of writein components is shrunk apply any robust mix-net that operates over the suggested encryption mechanism.
- Writein tallying time:
  - Significantly reduced because of shrinking.
  - An "off-line" operation anyway, the winner of the election already known from the PC tallying component.

## Other Interesting Issues

- Potentially the summation register is not large enough for all the candidates (could be the case for large # of candidates).
  - We call this the *capacity* of the hom. encryption function.

    $$\mathit{Capacity} > (\text{\# of voters})^{\text{\# of candidates}}$$

- We design an alternative vector ballot design, called "punch-hole" ballot that only requires

  $$\mathit{Capacity} > (\text{\# of voters})$$

- The punch-hole approach allows an exponential improvement for tallying in the instantiation of our approach over ElGamal encryption.

## Conclusion

- The "Vector Ballot" approach to E-Voting
  - Combines:
    - Writein capability.
    - Efficient tallying.
    - Universal verifiability.
  - Bridges H.E. approach and Mixnet
  - Sometimes bridging "technologies" also improves efficiency by their interaction (shrink-and-mix).
- Paper available:
  http://www.cse.uconn.edu/~akiayias/