Secure and Efficient Boardroom Voting with Malleable Proof Systems and Batch Proofs

Stephan Neumann – Security, Usability, and Society (TU Darmstadt)

CASED
Usable Security Lab
Crypto Lab

TECHNISCHE UNIVERSITÄT DARMSTADT

SECUSO
SECURITY · USABILITY · SOCIETY

TECHNISCHE UNIVERSITÄT DARMSTADT
Fraunhofer SIT
h_da HOCHSCHULE DARMSTADT UNIVERSITY OF APPLIED SCIENCES
LOEWE Exzellente Forschung für Hessens Zukunft

Motivation

- Boardroom voting
  - No server setup (tallying authorities, bulletin board, ...)
  - Implementation on smartphones
- Ensure security properties
  - Ballot secrecy
  - Verifiability
  - Robustness
  - Dispute-freeness
- Efficient in terms of complexity
  - computational
  - communication
  - round

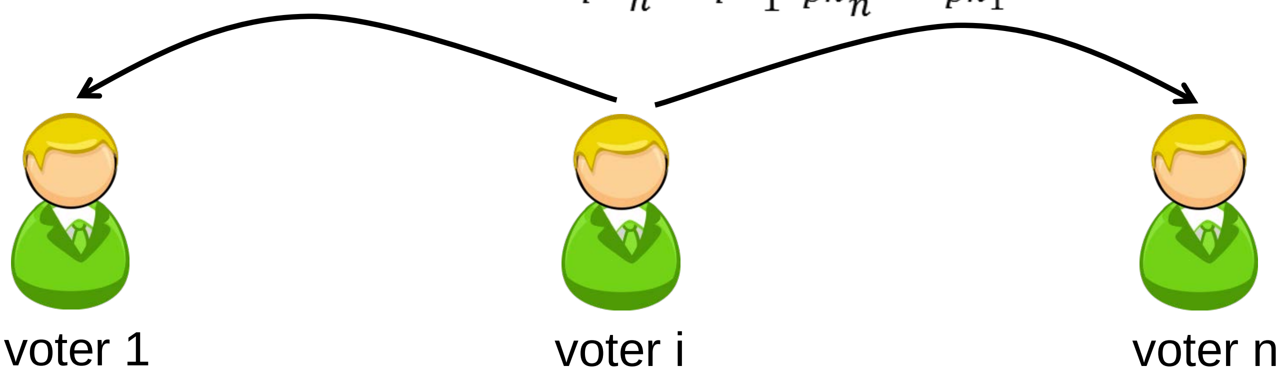
First Approach

Initial Voting Step

- DeMillo et al. 1982, Volkamer et al. 2005, Meletiadou 2007-2009

- Each voter makes her unique selection and encrypts her vote twice

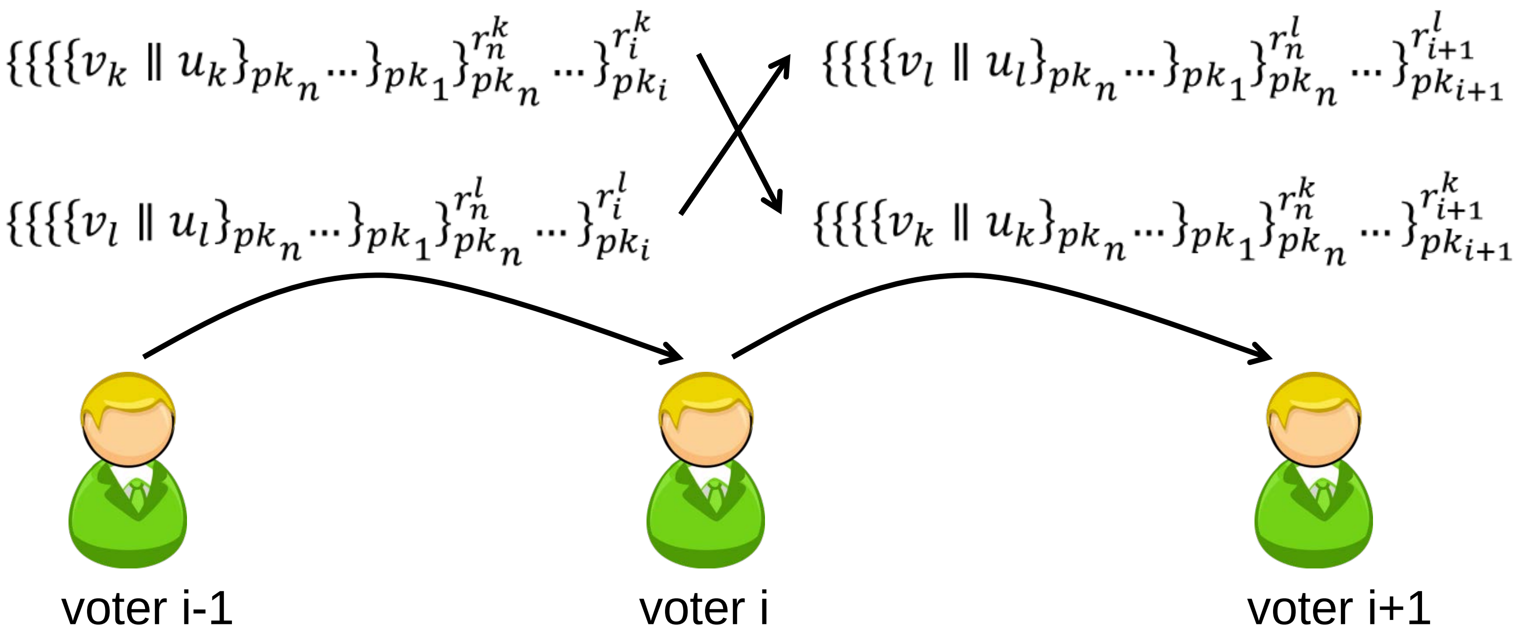
$$\{\{\{\{v_i \parallel u_i\}_{pk_n} \cdots\}_{pk_1}\}^{r_n}_{pk_n} \cdots\}^{r_1}_{pk_1}$$



Anonymization Phase

CASED

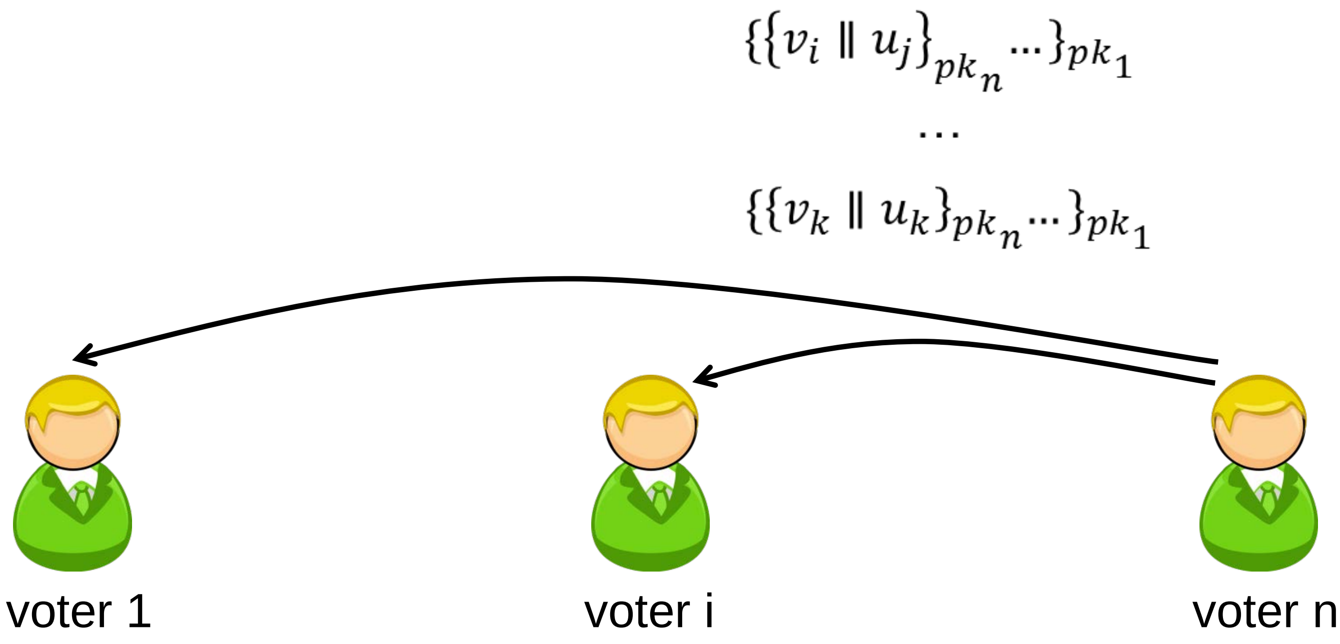
- Each voter strips off the outmost layer, permutes the ciphertexts, and forwards the partially anyonymized ciphertext to the next voter.



SECUSO SECURITY · USABILITY · SOCIETY

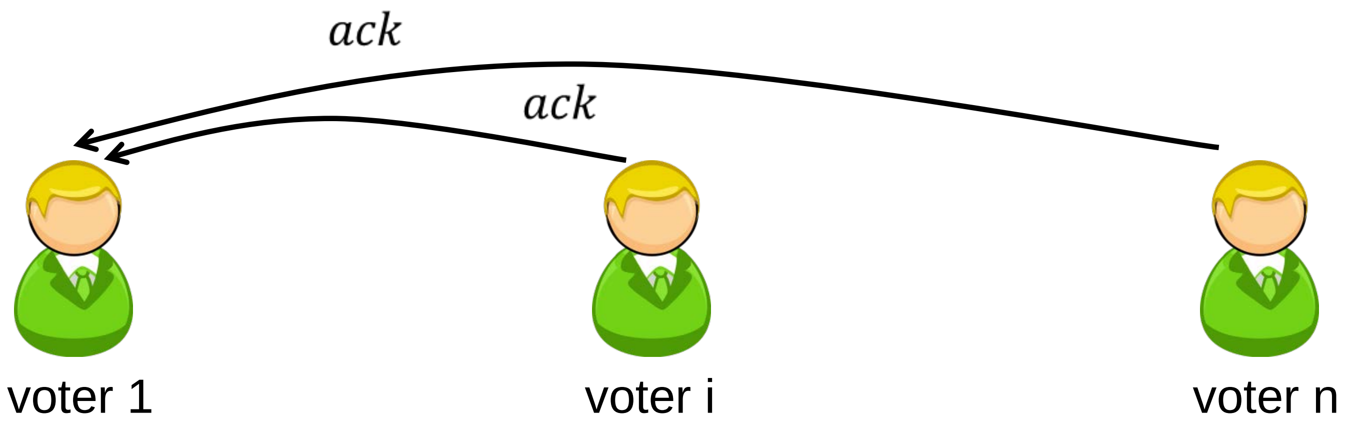
Anonymization Phase

- After the last voter stripped off her layer, the set of anonymized ciphertexts is sent to all other voters



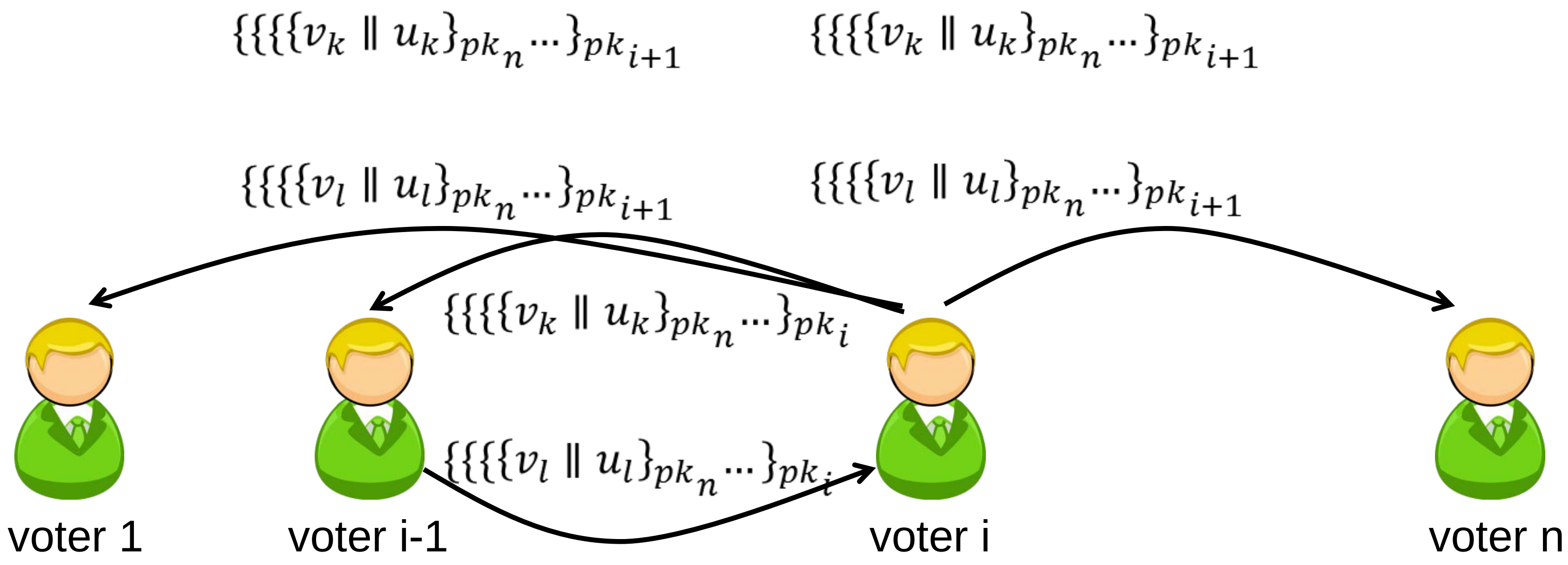
Anonymization Phase

- Each voter verifies the presence of her vote and acknowledges to the first voter.



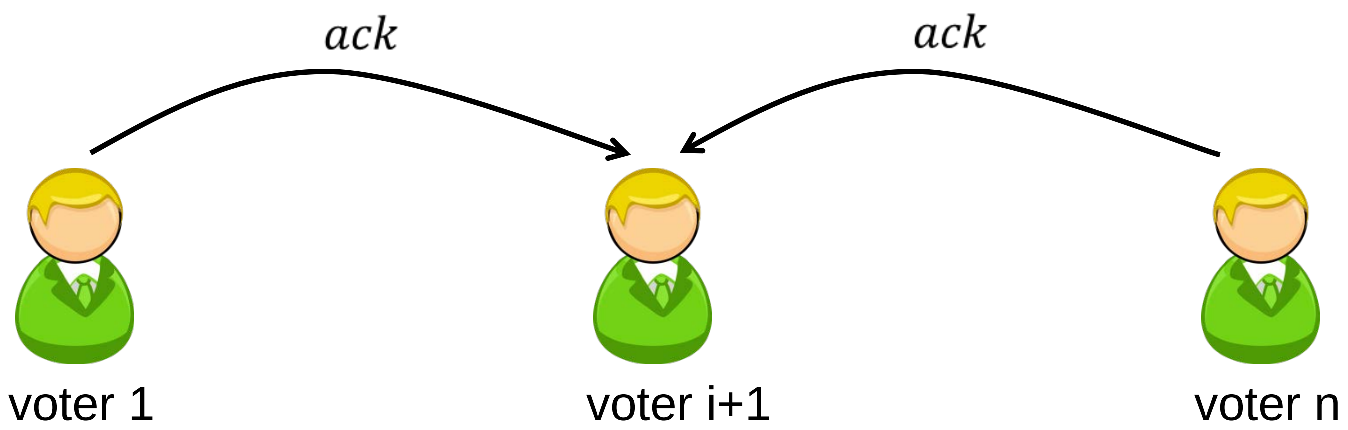
Decryption Phase

- The i-th voter receives the set of partially decrypted votes, stripps off the outmost layer and broadcasts all partially decrypted votes to all other voters.



Decryption Phase

- All voters acknowledge the correct processing to the (i+1)-th voter that proceeds with the decryption process.



Existing Approaches (DeMillo 1982, DuD 2005, Meletiadou 2007-2009)

Security Analysis:

- Robustness not given (due to decryption shuffle)
- Verifiability not given (malicious device can accept dishonest behavior)
- Weak form of receipt-freeness

Complexity:

- Computational Complexity:

$$n^3 * ExpCost\left(\left||(p_{RSA}-1)*(q_{RSA}-1)|\right|\right) + 3n * ExpCost\left(\left||q|\right|\right) + n$$

- Network Complexity:

$$(8n+2n^2) * \left||p|\right| + (n+1) * s(ack)$$

- Round Complexity:

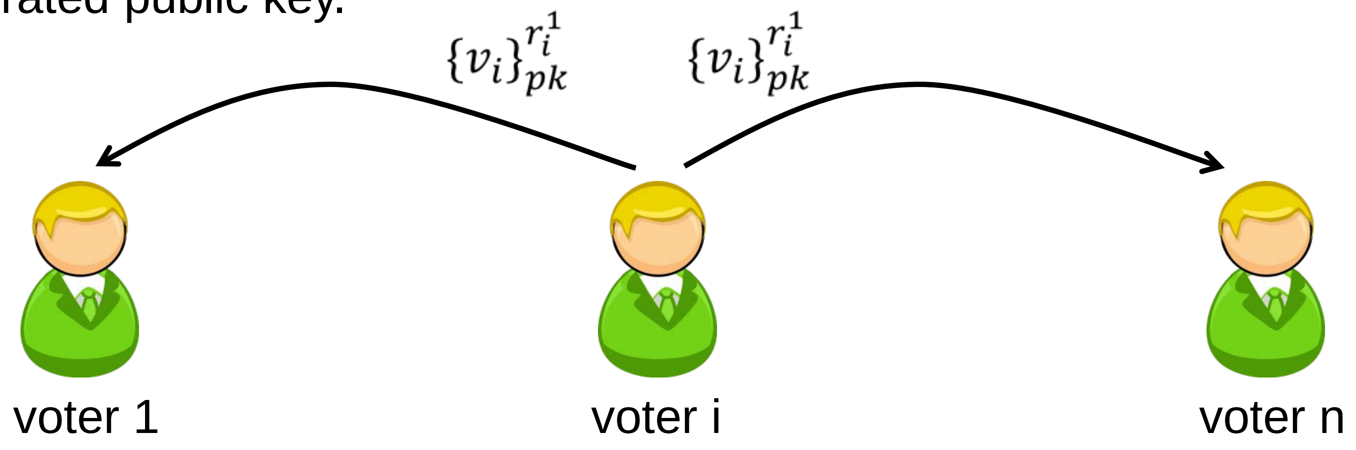
$$3n+6$$

A Naive Improvement

Initial Voting Step

- A public key is generated distributively by all voters such that each voter holds a secret key share (e.g., Joint-Feldman DKG).

- Each voter makes her selection and encrypts her vote with commonly generated public key.



Initial Voting Step

- Each voter holds all encrypted votes

$\{v_1\}_{pk}^{r_1^1}, \ldots, \{v_n\}_{pk}^{r_n^1}$

voter 1

$\{v_1\}_{pk}^{r_1^1}, \ldots, \{v_n\}_{pk}^{r_n^1}$

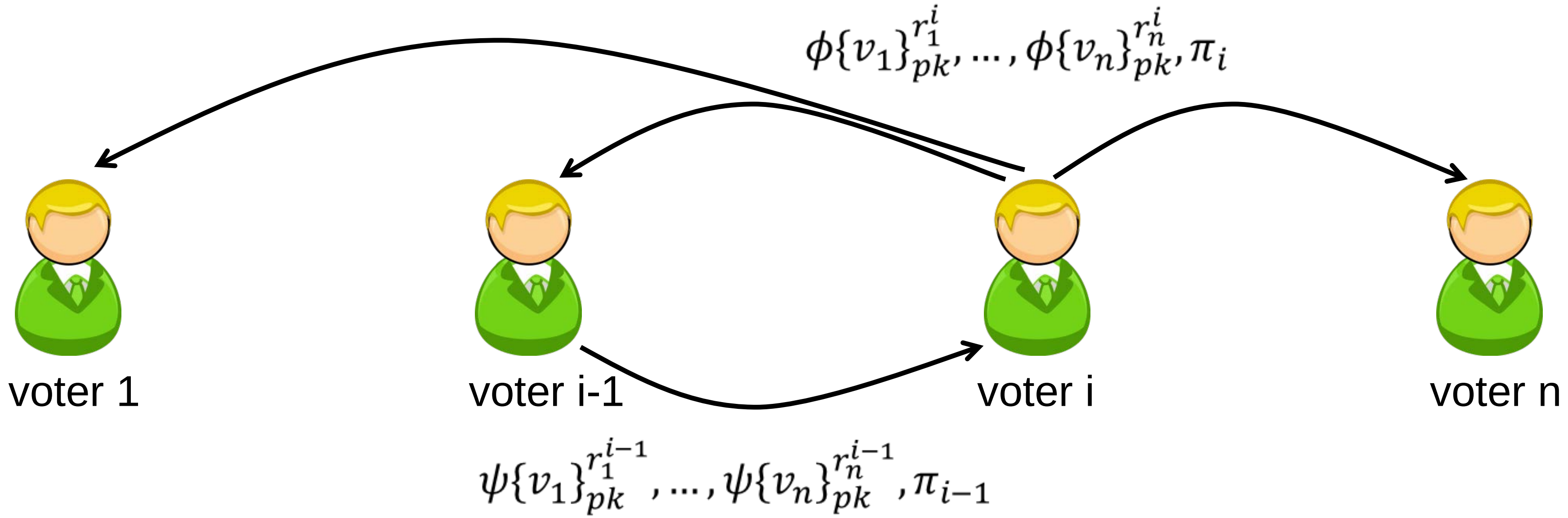
voter i

$\{v_1\}_{pk}^{r_1^1}, \ldots, \{v_n\}_{pk}^{r_n^1}$

voter n

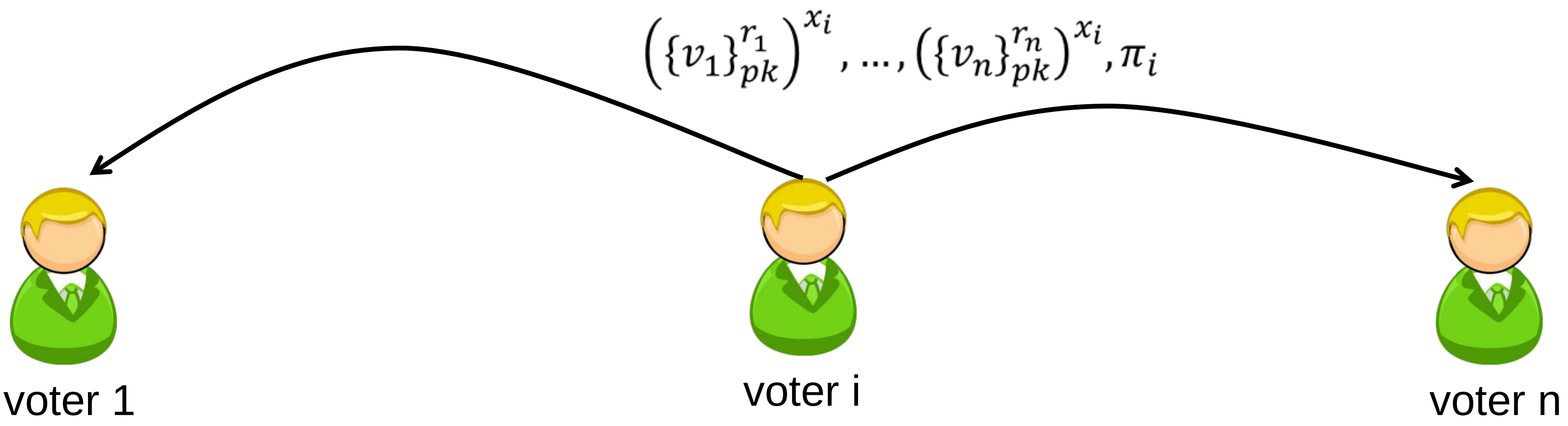
Anonymization Phase

- Each voter permutes the received votes, re-encrypts them and broadcasts a proof of correct proceeding to all other voters that have to acknowledge.



Decryption Phase

- Each voter partially decrypts the set of encrypted votes and broadcasts the partial decryption together with a proof of correct proceeding to all other voters.



A Naive Improvement

Security Analysis:

- Robustness given (re-encryption substitutes decryption shuffle)
- Verifiability given (all steps universally verifiable)
- Stronger form of receipt-freeness

Complexity:

- Computational Complexity:

$$(10n^2 + 11n + 2) * ExpCost(||q||) + 4n^2 + 3n + 1$$

- Network Complexity:

$$6n^2||p|| + 4n^2||q|| + n^2 s(ack)$$

- Round Complexity:

$$2n^2 + 5n + 1$$

A Distributed Voting System

General Idea of this Work

- Initial broadcasting of each encrypted vote
- Improve computational, communication, and round complexity due to
  - Shuffle proof chain (Eurocrypt 2012)
  - Decryption proof batching (ACNS 2004)
- Final broadcasting of anonymized and decrypted votes
- Integrity of both phases universally verifiable

Controlled Malleable Proof Systems

Idea: Prove particular statements relying on proofs of related statements

- **Definitions** for a proof system that is
  - **malleable** wrt. to set of transformations (valid transformations):

    Given proofs for

    $$(x_1, w_1) \in R, \dots, (x_n, w_n) \in R$$

    these proofs can be transformed into valid proof for

    $$(T_x(x_1, \dots, x_n), T_w(w_1, \dots, w_n)) \in R$$

  - **derivation private**: Transformed proofs cannot be distinguished from fresh proofs for a statement

Verifiable Shuffle Construction

Procedure (k-th server):

- Obtain$\left(\{c_i\}, \{c_i'\}, \pi, \{pk_j\}\right)$ and check validity of π
- Pick $\{r_i\}$ and permutation ϕ_i and compute
$$\{c_i''\} \leftarrow ReRand(pk, \phi_i\{c_i'\}; \{r_i\})$$
- Based on valid transformation (specified in the paper), a valid proof is generated
$$\pi' \leftarrow ZKEval(\sigma_{crs}, T, \left(pk, \{c_i\}, \{c_i'\}, \{pk_j\}\right), \pi))$$

This proof shows that $\{c_i''\}$ is a valid shuffle of $\{c_i\}$ by voters in possession of $(sk_1, \dots, sk_k)$

- Output
$$(\{c_i\}, \{c_i''\}, \pi', \{pk_j\} \cup pk_k)$$

Partial ElGamal Decryption

Given ElGamal ciphertext $(c_1, c_2) = (g^r, y^r \cdot m)$ of message m under public key

(p, g, y) and randomness $r \leftarrow \{1, \dots, p-2\}$

- Each voter i computes

$$c_{1,i} = {c_1}^{x_i}$$

and proves the equality of discrete logarithms

$$\log_g y_i = \log_{c_1} c_{1,i}$$

Proof of Equality of Discrete Logarithms

- Sigma protocol due to Chaum and Pedersen (1992)
- Given $x = g^l, y = h^l$, a prover wants to convince a verifier about the fact

$$\log_g x = \log_h y = l$$

- Computational Cost for decryption of n ciphertexts:
  - Prover: $2n * ExpCost(||q||) + n$
  - Verifier: $4n * ExpCost(||q||) + 2n$

Batch Proof Generation and Verification

Batch Theorem:

Given two large primes p, q with $p = 2q + 1$, a security parameter l with $2^l < q, t_j \leftarrow \{1, \ldots, 2^l\}$, a set of n ciphertexts c_k, voter's i public key y_i, n corresponding partial decryptions $c_{k,1,i}$, then the following holds with probability more than $1 - 2^{-l}$:

$$\exists k \in \{1, \ldots, n\}\, s.t.\ |{c_{k,1}}^{\log_g y_i}| \neq |c_{k,1,i}| \Longrightarrow \left(\prod_{k=1}^{n} \left(c_{k,1}\right)^{t_k}\right)^{\log_g y_i} \neq \prod_{k=1}^{n} \left(c_{k,1,i}\right)^{t_k}$$

Linear Encryption

- **Motivation:** Move from DDH to DLIN assumption.
- **Key Generation**: The user randomly chooses $x_1, x_2 \leftarrow Z_p$, and computes $y_1 = g^{x_1}$ and $y_2 = g^{x_2}$. The secret key is $sk = (x_1, x_2)$ and the public key is $pk = (y_1, y_2)$
- **Encryption**: In order to encrypt message m, two values $r_1, r_2 \leftarrow Z_p$ are randomly drawn and the ciphertext is computed as follows:

$$(c_1, c_2, c_3) = ({y_1}^{r_1}, {y_2}^{r_2}, m * g^{r_1+r_2})$$

- **Decryption**: Ciphertext (c_1, c_2, c_3) is decrypted with (x_1, x_2)

$$m = \frac{c_3}{{c_1}^{\frac{1}{x_1}} * {c_2}^{\frac{1}{x_2}}}$$

Discussion

- Can distributed key generation and distributed decryption be adapted to Linear Encryption?
- Can the corresponding proofs still be batched?
- Can distributed ElGamal decryption proofs be cm-NIZK?

Thank you for your Attention!

Faleminderit
Kosovan & Albanian

ありがとうございました
Japanese

Спасибо
Russian

كل اركش
Arabic

תודה
Hebrew

Bedankt
Dutch

Grazie
Italian

Obrigado
Portuguese

Danke
German

Tack
Swedish

Gracias
Spanish

Thank you
English

Takk
Norwegain

Hvala
Croation

Çok teşekkür ederiz
Turkish

Merci
French

Dziękuję
Polish

Благодаря
Bulgarian

Tashakkur
Afghan

Distributed ElGamal Key Generation

- Each voter i generates
$$x_i \leftarrow Z_q$$
- Each voter i generates a polynom
$$f_i(x) = f_{i0} + f_{i1} \cdot x + \cdots + f_{i(t-1)} \cdot x^{t-1}$$
  with
$$f_i(0) = x_i = f_{i0}$$
- Each voter i commits on the generated polynom by broadcasting
$$F_{ij} = g^{f_{ij}} \bmod p$$
- Each voter i sends to voter j
$$s_{ij} = f_i(j) \bmod q$$
- Each voter i verifies received shares by
$$g^{s_{ji}} = \prod_{l=0}^{t-1} F_{jl}^{i^l} \bmod p$$

Distributed ElGamal Key Generation

- Each voter i computes shares s_i of private key x

$$s_i = \sum_{j=1}^{n} s_{ji} \mod q$$

- The public key can be publicly computed

$$h = \prod_{i=1}^{n} F_{i0} = \prod_{i=1}^{n} g^{x_i} \mod p$$

  and public shares

$$h_j = g^{\sum_{i=1}^{n} f_i(j)}$$

- For each s_i a commitment p_i can be publicly computed

$$p_i = \prod_{j=1}^{n} g^{s_{ji}} = \prod_{j=1}^{n} (h_j \cdot \prod_{l=1}^{t-1} {F_{jl}}^{i^l}) = g^{\sum_{j=1}^{n} s_{ji}} \mod p$$