Bern University of Applied Sciences

One Person, One Computer, One Vote

Theoretical and Practical Challenges of Designing Online Voting Systems

Rolf Haenni

February 6th, 2018

Outline

Bern University
of Applied Sciences

Introduction

netzwoche

NEWS STORYS MEINUNGEN STUDIEN DOS

NEWS

Zertifizierung erweitert

Post kann E-Voting für 50 Prozent der Stimmbürger anbieten

Mo 21.08.2017 - 10:23 Uhr | Aktualisiert 21.08.2017 - 10:23
von Christoph Grau

Bisher durften mit der E-Voting-Lösung der Post nur 30 Prozent der Stimmbürger elektronisch abstimmen. Die Bundeskanzlei hob die Grenze nun auf 50 Prozent an. In einem Jahr sollen es 100 Prozent werden.

HOME ABO E-PAPER/HEFTARCHIV EVENTS NEWSLETTER RSS-FEED

Computerworld

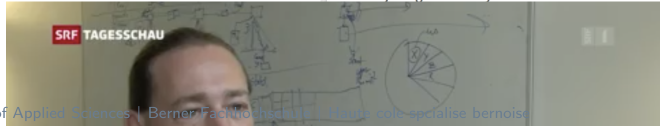
NEWS BUSINESSPRAXIS TESTS MARKTANALYSEN JOB & KARRIERE WHITEPAPERS SERVICE PARTNE

Home » News » Security

Twittern +1 Gefällt mir XING Share

E-Voting: Wie sicher sind die Schweize Lösungen?

Wie sicher sind die Schweizer E-Voting-Systeme? Diese Frage beschäftigt Politik, sondern auch die IT-Security-Szene. So widmete sich auch ein Th diesjährigen SwissCyberStorm in Luzern der elektronischen Stimmabga



It is enough that the people know there was an election. The people who cast the votes decide nothing. The people who count the votes decide everything.

Josef Stalin

If we are to bring computerization into our electoral processes, then we must do it in such a way as to preserve the integrity of the process and to prevent the concentration of power into the hands of the few who control the process.

Josh Benaloh, *Verifiable Secret-Ballot Elections*
PhD Thesis, Yale University, 1987

The introduction of verifiability is central to the new security requirements.

3rd Vote Electronique Report
Swiss Federal Council, 2013

Voters must be able to ascertain whether their vote has been manipulated or intercepted on the user platform or during transmission. [...] Voters must receive proof that the server system has registered the vote as it was entered by the voter on the user platform.

Federal Chancellery Ordinance on Electronic Voting
VEleS, Art.4, 2013

Auditors receive proof that the result has been ascertained correctly. They must evaluate the proof in a observable procedure. To do this, they must use technical aids that are independent of and isolated from the rest of the system.

Federal Chancellery Ordinance on Electronic Voting
VEleS, Art.5, 2013

Bern University
of Applied Sciences

Swiss Context

Direct Democracy in Switzerland

- Up to four election days per year
  - Elections
  - Mandatory referendums
  - Optional referendums ($>$50k signatures)
  - Popular initiatives ($>$100k signatures)
- Four different political levels
  - Federal
  - Cantonal
  - Municipal
  - Pastoral

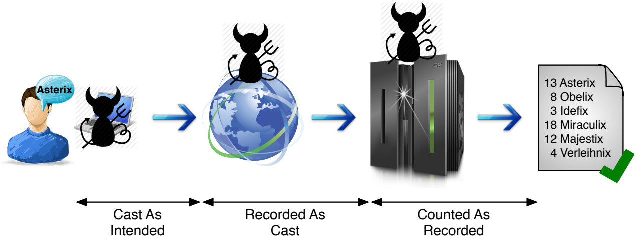
  (voters are not necessarily eligible on all four levels)
- Up to 10 different election topics per election day

E-Voting Tradition in Switzerland

- Classical voting channels
  - Polling station
  - Landsgemeinde
  - Postal voting (since 1994, approx. 90%)
- Non-verifiable “blackbox” e-voting systems (1st generation)
  - Canton of Geneva (since 2003)
  - Canton of Zürich (Unisys, 2004–2015)
  - Canton of Neuchâtel (Scytl, 2005–2015)
- Collaborations with 10 other cantons (since 2009)
- Target audience: Swiss citizens living abroad

Legal Ordinance on Electronic Voting

- Effective since December 2013
- Enhanced security requirements
  - End-to-end encryption
  - End-to-end verifiability (cast-as-intended, recorded-as-cast, counted-as-recorded)
  - Distribution of trust (shared decryption key, mix-net)

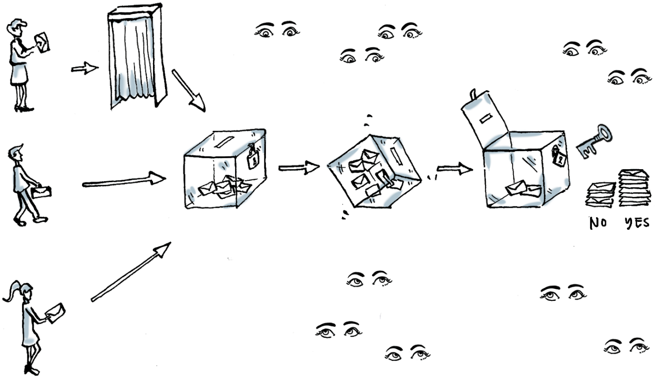


Stepwise Introduction

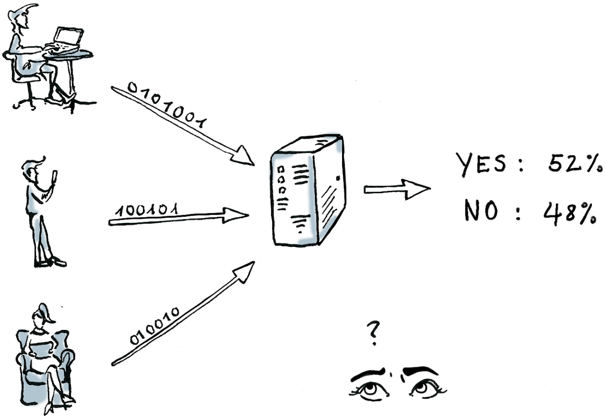
- Current systems: max. 10/30% of federal/cantonal electorate
- Two-step expansion
  - Step 1: max. 30/50% of federal/cantonal electorate
  - Step 2: 100% electorate
- Two competing 2nd generation projects
  - Swiss Post (Scytl):
    - Has reached Step 1 in 2017
    - Plans to reach Step 2 in 2019
  - Canton of Geneva (CHVote)
    - Plans to reach Step 2 in 2019

Verifiable Elections

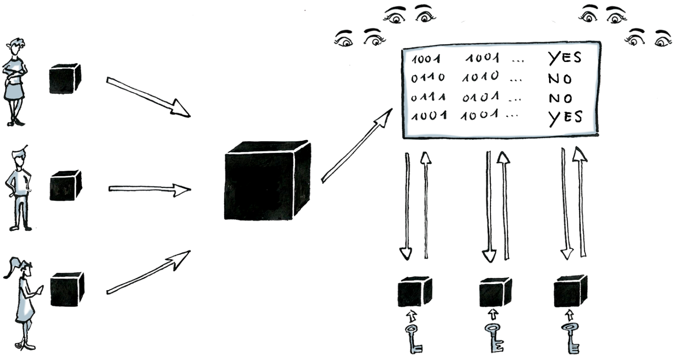
Traditional Paper-Based Voting



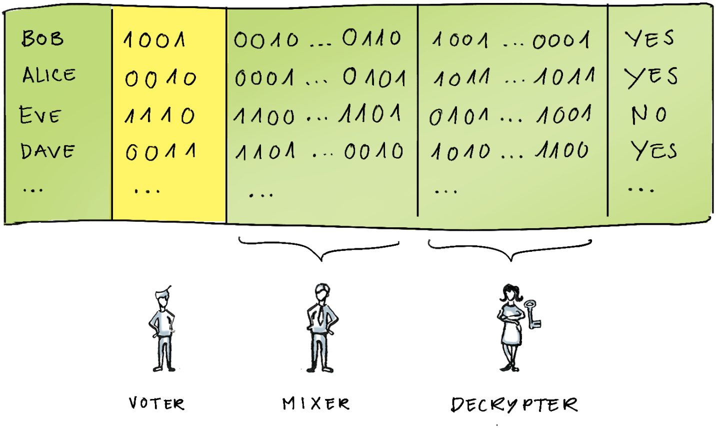
Remote Electronic Voting (Blackbox)



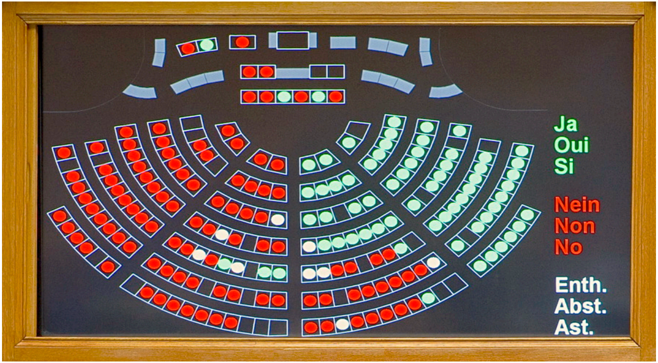
Verifiable Remote Electronic Voting



Bulletin Board



Bulletin Board



Voting panel, Swiss National Council, Bern, Switzerland (srf.ch)

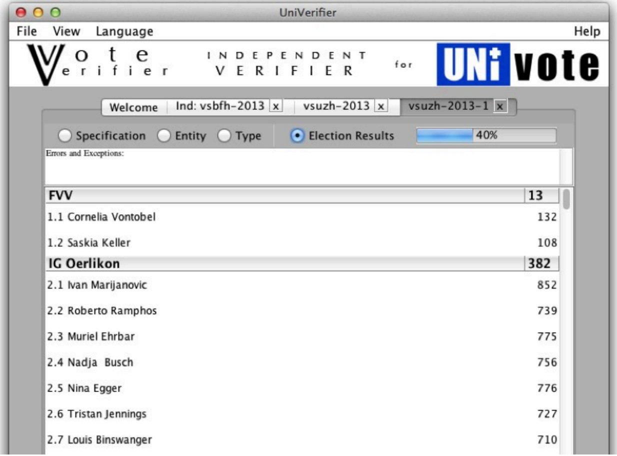
Bulletin Board

List of eligible voters, Erbil, Iraq (nzz.ch)

Bulletin Board

Landsgemeinde, Glarus, Switzerland (blick.ch)

Verification Software



Cryptographic Voting Protocols

The Secure MPC Perspective

- In secure multi-party computation (MPC), the voting problem can be formulated as follows:
  - Parties $P_1, \ldots, P_n$ with private inputs $v_i \in \{0, 1\}$
  - Common output $s = f(v_1, \ldots, v_n) = \sum_i v_i$
- Design a secure protocol to be executed among $P_1, \ldots, P_n$
  - Privacy: No party should learn anything more than $s = \sum_i v_i$
  - Correctness: Each party receives the correct output
  - Independent inputs: Parties choose their inputs independently
  - Output delivery
  - Fairness
- Formal security definition based on ideal/real-model paradigm

Cryptographic Voting Protocol

- General MPC protocols are not applicable to real-world elections
  - Protocols are not efficient enough for large n
  - Several preconditions are not met
- Therefore, e-voting research focuses on designing specialized cryptographic voting protocols
  - Election administration
  - Independent authorities (of which a majority is honest)
  - Append-only bulletin board
  - Voters
  - Verifiers (auditors)

Desirable Security Properties

- Privacy
  - Vote secrecy (everlasting?)
  - Participation secrecy
  - Receipt-freeness
- Correctness
  - Votes from ineligible voters are not counted
  - Eligible voters can vote at most once
  - All valid from eligible voters votes are counted
- E2E Verifibility
  - Individual (cast-as-intended, recorded-as-cast)
  - Universal (counted-as-recorded)
- Fairness: nobody learns partial election results during election
- Coercion-Resistance

Approach 1: Homomorphic Tallying

- Public-key encryption scheme
  - $(pk, sk) \leftarrow KeyGen()$
  - $e \leftarrow Enc_{pk}(m, r)$
  - $m \leftarrow Dec_{sk}(e)$
- Additively homomorphic encryption schemes

$$Enc_{pk}(m_1, r_1) * Enc_{pk}(m_2, r_2) = Enc_{pk}(m_1 + m_2, r_1 + r_2)$$

- Examples: Exponential ElGamal, Paillier

Approach 1: Homomorphic Tallying

- Step 1: Multiple authorities generate a common public key pk
- Step 2: Voters submit $e_i = Enc_{pk}(v_i, r_i)$ to bulletin board
- Step 3: The authorities jointly...
  - Retrieve $E = (e_1, \ldots, e_n)$ from bulletin board
  - Compute $e = \prod_i e_i$
  - Decrypt e into $s \leftarrow Dec_{sk}(e)$ using their shares of sk
  - Publish s on the bulletin board
- Non-interactive zero-knowledge proofs are added to prevent cheating voters and authorities

Approach 2: Re-Encryption Mixnet

- Re-encryption of $e = Enc_{pk}(m, r)$

$$ReEnc_{pk}(e, r') = e * Enc_{pk}(0, r') = Enc_{pk}(m, r + r')$$

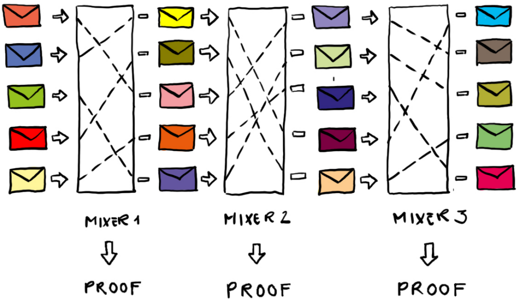
- A cryptographic shuffle transforms an list $E = (e_1, \ldots, e_n)$ of encryptions into $E' = (e'_1, \ldots, e'_n)$ such that

$$e'_j = ReEnc_{pk}(e_i, r'_j)$$

  holds for every i and j

- A series of cryptographic shuffles forms a re-encryption mixnet

Approach 2: Re-Encryption Mixnet



Approach 2: Re-Encryption Mixnet

- Step 1: Multiple authorities generate a common public key pk
- Step 2: Voters submit $e_i = Enc_{pk}(v_i, r_i)$ to bulletin board
- Step 3: The authorities perform a mixnet on $E = (e_1, \ldots, e_n)$
- Step 4: The authorities jointly...
  - Retrieve $E' = \{e'_1, \ldots, e'_n\}$ from bulletin board
  - Decrypt each e'_i into $v_i \leftarrow Dec_{sk}(e'_i)$ using their shares of sk
  - Publish $(v_1, \ldots, v_n)$ and $s = \sum_i v_i$ on bulletin board
- Non-interactive zero-knowledge proofs are added to prevent cheating authorities

CHVote Voting Protocol

CHVote Project

- Project goals
  - New implementation from scratch
  - Reach second expansion stage in one step (100% electorate)
  - Developed, hosted, operated entirely by the State of Geneva
- Strategy
  - Collaboration with academia
  - State-of-the-art technologies
  - Maximal transparency
  - High-quality open documentation
  - Open-source license (Affero GPL)
  - Invitation to public code reviewing

CHVote Voting Protocol

- Collaboration with Bern University of Applied Sciences
- Cast-as-intended verifiability à la Norway (see next slide)
- Key cryptographic ingredients
  - Distributed generation of codes
  - Oblivious transfer of selected codes
  - Verifiable re-encryption mix-net
  - Schnorr identification
  - Distributed decryption with shared ElGamal private key
- Scientific papers presented at E-Vote-ID'16, FC'17, FC'18

Cast-as-Intended Verification

- Prior to an election, a code sheet with different verification codes for each voting option is generated for every voter
- Verification codes are different on every code sheet
- Code sheets are sent to voters by postal mail

Code Sheet Nr.291

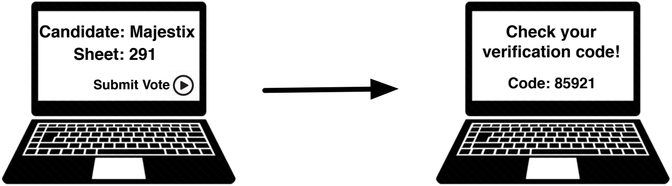
Candidates	Codes
Asterix	74494
Obelix	84443
Idefix	91123
Miraculix	63382
Majestix	85921
Verleihnix	79174

Code Sheet Nr.321

Candidates	Codes
Asterix	21344
Obelix	29173
Idefix	91123
Miraculix	72282
Majestix	18194
Verleihnix	53382

Cast-as-Intended Verification

- After submitting a vote, corresponding verification codes are displayed



- Matching codes imply that the vote has been cast as intended
- Otherwise, voters are instructed to vote by postal mail

Cast-as-Intended Verification

- Detectable malware attacks
  - Manipulated votes ✓
  - Suppressed votes ✓
  - Manipulated verification codes ✓
  - Suppressed verification codes ✓
- Unsolved malware attacks
  - Secrecy of vote ☒
  - Social engineering attack: "Please enter verification code" ☒
- Critical processes
  - Generation and printing of code sheets
  - Sending code sheet by postal mail

Liste de codes pour la carte n° 5874-8863-1400-8743			
Votation fédérale			
Question 1 Acceptez-vous l'arrêté fédéral du 20 juin 2013 portant règlement du financement et de l'aménagement de l'infrastructure ferroviaire (Contre-projet direct à l'initiative populaire "Pour les transports publics", qui a été retirée) ?	Oui A2B4	Non J5B9	Blanc Z8H5
Question 2 Acceptez-vous l'initiative populaire "Financer l'avortement est une affaire privée - Alléger l'assurance-maladie en radiant les coûts de l'interruption de grossesse de l'assurance de base" ?	Oui P8H3	Non X2A7	Blanc Q3L7
Votation cantonale			
Question 1 Acceptez-vous l'initiative 143 «Pour une véritable politique d'accueil de la Petite enfance» ?	Oui U6T4	Non P3D6	Blanc S6C2
Question 2 Acceptez-vous la loi constitutionnelle modifiant la constitution de la République et canton de Genève (Contreprojet à l'IN 143) (A 2 00 – 10895), du 15 décembre 2011 ?	Oui N4F2	Non M2A3	Blanc Q9L5
Question 3 **Question subsidiaire**: Si l'initiative (IN 143 «Pour une véritable politique d'accueil de la Petite enfance») et le contreprojet sont acceptés, lequel des deux a-t-il votre préférence ? Initiative 143 ? Contreprojet ?	IN K9W9	CP T3S6	Blanc Y2V4


VOTE ELECTRONIQUE
Identification
Rappel légal
Bulletin de vote
Récapitulatif
5
Vérification
6
Finalisation du vote
Il vous reste 29 minute(s) 18 seconde(s) pour confirmer votre vote
Codes de vérification
1) Consultez les codes de vérification fournis dans votre matériel de vote
2) Vérifiez que les codes pour chaque question soient les mêmes entre cette page web et ceux de votre matériel de vote
Où trouver les codes ?
VOTATION FÉDÉRALE
VOS CHOIX
VOS CODES
1 Acceptez-vous l'initiative populaire «Pour une économie durable et fondée sur une gestion efficiente des ressources (économie verte)»?
NON
M9F2
2 Acceptez-vous l'initiative populaire «AVSplus: pour une AVS forte»?
NON
L3M8
3 Acceptez-vous la loi fédérale du 25 septembre 2015 sur le renseignement (LRens)?
NON
X3T6
VOTATION CANTONALE
VOS CHOIX
VOS CODES
1 Acceptez-vous la loi constitutionnelle modifiant la constitution de la République et canton de Genève (Cst-GE) (Elections au système majoritaire) (A 2 00 - 11757), du 26 février 2016?
NON
V3Q3

CHVote Protocol Specification

- Published on April 20, 2017 (with prototype source code)
- Self-contained and comprehensive document ($\sim$120 pages)
  - Description of election use cases
  - Mathematical and cryptographic background
  - Details of encoding and hashing algorithms
  - Adversary and trust assumptions
  - Cryptographic and election parameters
  - Recommendations for group sizes, key lengths, code lengths
- Three main protocols (three sub-protocols each)
  - Pre-election
  - Election
  - Post-election
- About 60 pseudo-code algorithms

Phase	Election Admin.	Election Authority	Printing Authority	Voter	Voting Client	Bulletin Board	Protocol Nr.
1. Pre-Election	•	•	•	•		•	
1.1 Election Preparation	•	•				•	6.1
1.2 Printing of Code Sheets		•	•	•		•	6.2
1.3 Key Generation		•				•	6.3
2. Election		•		•	•	•	
2.1 Candidate Selection				•	•	•	6.4
2.2 Vote Casting		•			•	•	6.5
2.3 Vote Confirmation		•		•	•	•	6.6
3. Post-Election	•	•				•	
3.1 Mixing		•				•	6.7
3.2 Decryption		•				•	6.8
3.3 Tallying	•					•	6.9

Voting Client		Bulletin Board		Election Authority $j \in \{1, \ldots, s\}$
knows $i, X_i, \mathbf{k}_i, \mathbf{s}$		knows $\mathbf{pk}$		knows $pk, \mathbf{n}, \mathbf{K}, \mathbf{P}_j, \hat{\mathbf{x}}, B_j \leftarrow \langle\rangle$
	$\xleftarrow{\mathbf{pk}}$			
$pk \leftarrow \mathsf{GetPublicKey}(\mathbf{pk})$ $(\alpha, \mathbf{r}) \leftarrow \mathsf{GenBallot}(X_i, \mathbf{s}, pk)$				
	$\xrightarrow{i, \alpha}$			
			$\xrightarrow{i, \alpha}$	
				if $\neg\mathsf{CheckBallot}(i, \alpha, pk, \mathbf{K}, \hat{\mathbf{x}}, B)$ **abort** $(\beta_j, \mathbf{r}) \leftarrow \mathsf{GenResponse}(i, \mathbf{a}, pk, \mathbf{n}, \mathbf{K}, \mathbf{P}_j)$ $B_j \leftarrow B_j \parallel \langle(i, \alpha, \mathbf{r})\rangle$
			$\xleftarrow{i, \beta_j}$	
		$\boldsymbol{\beta}_i = (\beta_1, \ldots, \beta_s)$		
	$\xleftarrow{\boldsymbol{\beta}_i}$			
$\mathbf{P}_\mathbf{s} \leftarrow \mathsf{GetPointMatrix}(\boldsymbol{\beta}_i, \mathbf{k}_i, \mathbf{s}, \mathbf{r})$ $\mathbf{rc}_\mathbf{s} \leftarrow \mathsf{GetReturnCodes}(\mathbf{s}, \mathbf{P}_\mathbf{s})$				

Protocol 6.5: Vote Casting

Algorithm: $\mathsf{GenBallot}(X, \mathbf{s}, pk)$

Input: Voting code $X \in A_X^{\ell_X}$
Selection $\mathbf{s} = (s_1, \ldots, s_k)$, $1 \leqslant s_1 < \cdots < s_k$
Encryption key $pk \in \mathbb{G}_q \backslash \{1\}$

$x \leftarrow \mathsf{ToInteger}(X)$ // see Alg. 4.7
$\hat{x} \leftarrow \hat{g}^x \bmod \hat{p}$
$\mathbf{q} \leftarrow \mathsf{GetSelectedPrimes}(\mathbf{s})$ // $\mathbf{q} = (q_1, \ldots, q_k)$, see Alg. 7.19
$m \leftarrow \prod_{i=1}^{k} q_i$
if $m \geqslant p$ **then**
  return $\perp$ // (k, n) is incompatible with p
$(\mathbf{a}, \mathbf{r}) \leftarrow \mathsf{GenQuery}(\mathbf{q}, pk)$ // $\mathbf{a} = (a_1, \ldots, a_k)$, $\mathbf{r} = (r_1, \ldots, r_k)$, see Alg. 7.20
$a \leftarrow \prod_{i=1}^{k} a_i \bmod p$
$r \leftarrow \sum_{i=1}^{k} r_i \bmod q$
$b \leftarrow g^r \bmod p$
$\pi \leftarrow \mathsf{GenBallotProof}(x, m, r, \hat{x}, a, b, pk)$ // $\pi = (t, s)$, see Alg. 7.21
$\alpha \leftarrow (\hat{x}, \mathbf{a}, b, \pi)$
return $(\alpha, \mathbf{r})$ // $\alpha \in \mathbb{Z}_{\hat{q}} \times \mathbb{G}_q^k \times \mathbb{G}_q \times ((\mathbb{G}_{\hat{q}} \times \mathbb{G}_q^2) \times (\mathbb{Z}_{\hat{q}} \times \mathbb{G}_q \times \mathbb{Z}_q))$, $\mathbf{r} \in \mathbb{Z}_q^k$

Algorithm 7.18: Generates a ballot based on the selection $\mathbf{s}$ and the voting code X. The ballot includes an OT query $\mathbf{a}$ and a NIZKP π. The algorithm also returns the randomizations $\mathbf{r}$ of the OT query, which are required in Alg. 7.27 to derive the transferred messages from the OT response.

```java
/**
 * Algorithm 7.18: GenBallot
 *
 * @param upper_x the voting code
 * @param bold_s  voters selection (indices)
 * @param pk      the public encryption key
 * @return the combined ballot, OT query and random elements used
 */
public BallotQueryAndRand genBallot(String upper_x, List<Integer> bold_s, EncryptionPublicKey pk) {
    BigInteger x = conversion.toInteger(upper_x, publicParameters.getUpper_a_x());
    BigInteger x_circ = modExp(g_circ, x, p_circ);
    List<BigInteger> bold_q = computeBoldQ(bold_s);
    BigInteger m = computeM(bold_q, p);
    ObliviousTransferQuery query = genQuery(bold_q, pk);
    BigInteger a = computeA(query, p);
    BigInteger r = computeR(query, q);
    BigInteger b = modExp(g, r, p);
    NonInteractiveZKP pi = genBallotProof(x, m, r, x_circ, a, b, pk);
    BallotAndQuery alpha = new BallotAndQuery(x_circ, query.getBold_a(), b, pi);

    return new BallotQueryAndRand(alpha, query.getBold_r());
}
```

Crypto-Algorithms in Pseudo-Code

- Ideal interface between cryptographers, developers, auditors
  - Cryptographers can write, read, and check pseudo-code
  - Developers can derive real code from pseudo-code
  - Auditors can check if pseudo-code and real code match
  - Useful for security proofs
- Rarely used in . . .
  - cryptographic literature
  - electronic voting protocols
- Often used in standards (FIPS, RFC, PKCS, ...)

FIPS PUB 186-4: Digital Signature Standard (DSS)

A.2.3 Verifiable Canonical Generation of the Generator *g*

Input:

1.	p, q	The primes.
2.	*domain_parameter_seed*	The seed used during the generation of p and q.
3.	*index*	The index to be used for generating g. *index* is a bit string of length 8 that represents an unsigned integer.

Process:

1. If (*index* is incorrect), then return **INVALID.**
2. $N = \mathbf{len}(q)$.
3. $e = (p - 1)/q$.
4. $count = 0$.
5. $count = count + 1$.
6. If ($count = 0$), then return **INVALID**.
7. $U = domain\_parameter\_seed \mid\mid \text{"ggen"} \mid\mid index \mid\mid count$.
8. $W = \mathbf{Hash}(U)$.
9. $g = W^e \bmod p$.
10. If ($g < 2$), then go to step 5. Comment: If a generator has not been found.
11. Return **VALID** and the value of g.

Demo

NextGen Vote Visualization

- Bachelor thesis by Y. Denzer and K. Häni (January 2018)
- One-to-one implementation of CHVote specification
- Made for educational purpose only

https://chvote.virvum.ch

Bern University
of Applied Sciences

Conclusion

Conclusion

- Verifiability is central to making e-voting secure
- Various cryptographic protocols exist in scientific literature, e.g. based on homomorphic tallying or re-encryption mixnets
- The process of introducing e-voting in Switzerland is slow, but on the right track (legal ordinance VEleS)
- Challenges and open problems
  - Complexity of cryptographic protocols
  - Cryptography in web browser (JavaScript)
  - Vote secrecy on insecure platform
  - Vote buying and coercion
  - Everlasting privacy
  - Usability and "voter education"