Internet Voting System with Cast as Intended Verification

September 2011

VP R&D
Jordi Puiggali@scytl.com

scytl
Secure Electronic Voting

Index

- Introduction
- Proposal
- Security
- Conclusions

Introduction

- Client computers could be compromised by malicious software that could manipulate the voting options before being encrypted.

- The introduction of cast as intended mechanisms could allow voter to detect any manipulation of her vote before being cast

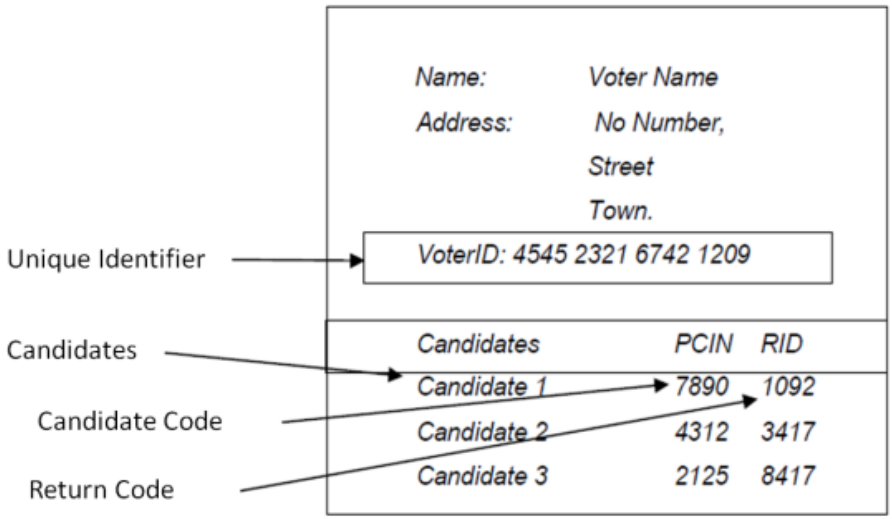
- This presentation describes Scytl's cast as intended initial proposal for the eValg 2011 project.

- This proposal was modified to meet specific usability and performance requirements proposed during the eValg 2011 project [1]

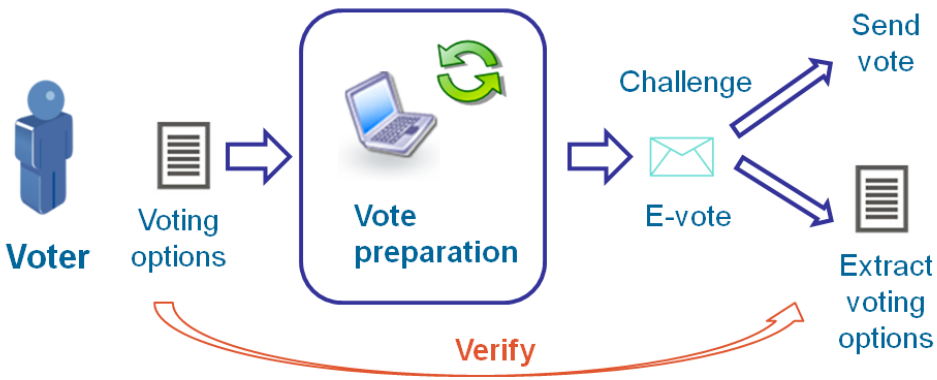
[1] Kristian Gjøsteen. Analysis of an internet voting protocol. Cryptology ePrint Archive, Report 2010/380, 2010. http://eprint.iacr.org/

Two main approaches

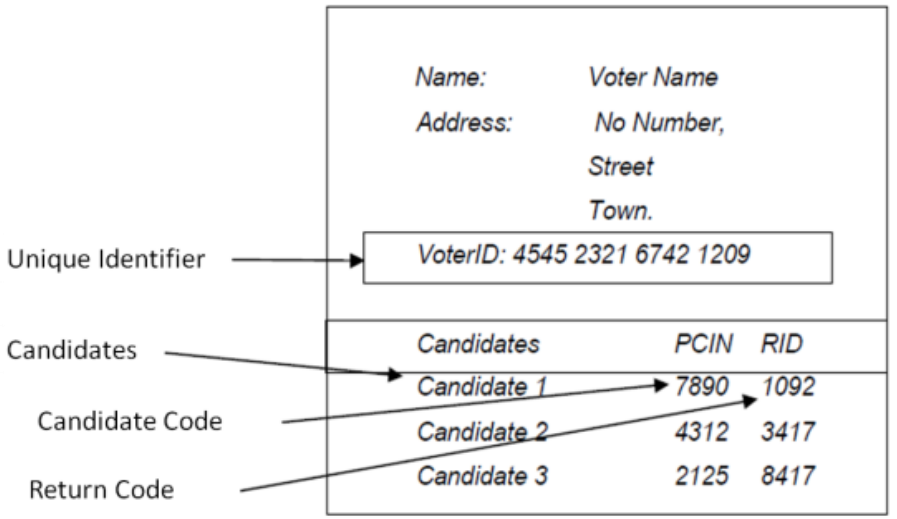
- Pollsterless or code voting
  - Voting Card per voter with two codes:
    - Candidate codes: used to select a specific candidate.
    - Return codes: used to verify that the server properly registered the selection



- Vote encryption challenge (Helios)
  - Application shows the secret random parameters used to encrypt the vote.
  - Voter uses the random parameters to verify if the encrypted vote contains her voter intent.
  - Verified votes are discarded



- Pollsterless or code voting:
  - Pros: Comparing codes in a sheet is intuitive and can be done without any support tool.
  - Cons: Casting votes by entering codes is not usable and error prone.



- Vote encryption challenge:
  - Pros: Friendly click & select interface for casting votes.
  - Cons: Independent voter verifiability process is less usable and cannot be done by humans means: requires an external tool.

Your Ballot:

{"answers": [{"choices": [{"alpha": "128...811", "beta": "
"individual_proofs": [[{"challenge": "323...", "commitment"
"695..."}, "response": "293..."}, {"challenge": "289...",
"715...", "B": "127..."}, "response": "596..."}], ...], "o
"answer": [1], "randomness": ["281...", ...]}], "election_
"OzxEVTC7SjQHiISorz8ehe/ENBE42BHHyVU+sZQyHgc", "election_u
"065a07e6-2893-11e0-b634-12313f025959"}

scytl Secure Electronic Voting

New proposal objectives

- Combine the usability features of both proposals:
  - Implement an usable verification process based on the same return codes concept of pollsterless systems.
  - Uses a click & select scheme for vote casting instead of sending candidate codes

Index

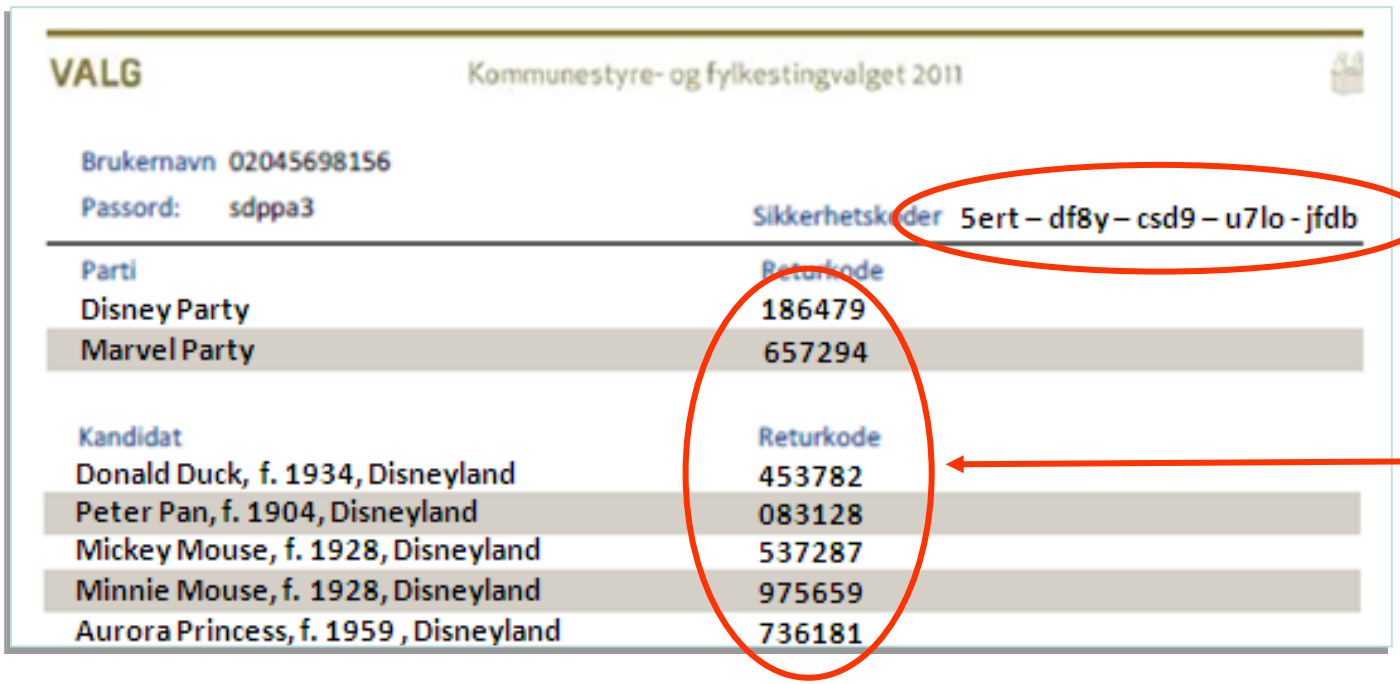
- Introduction
- Proposal
- Security
- Conclusions

Components

- Voting Client (VC):
  - Component executed in the voting terminal that implements the cryptographic processes in the client side
- Vote Collector Server (VCS):
  - Stores the accepted votes in the Ballot Box
  - Contributes in the operations that generate a deterministic value used for obtaining the Return Codes
- Return Code Generator (RCG):
  - Generates the Return Codes from the deterministic value generated by the VC and the RCG
  - Generates the voting receipt
- Voting Card:
  - It contains a unique Voting Card Identifier (VCID) and the Verification Code for each candidate/option
  - The VCID contributes on generating the Return Codes of the vote content

VC, VCS and RCG are executed in different environments managed by different actors. Voting Card is provided through a non electronic channel to voters.

Voting Card



VALG

Kommunestyre- og fylkestingvalget 2011

Brukernavn 02045698156

Passord: sdppa3

Sikkerhetskoder 5ert – df8y – csd9 – u7lo - jfdb

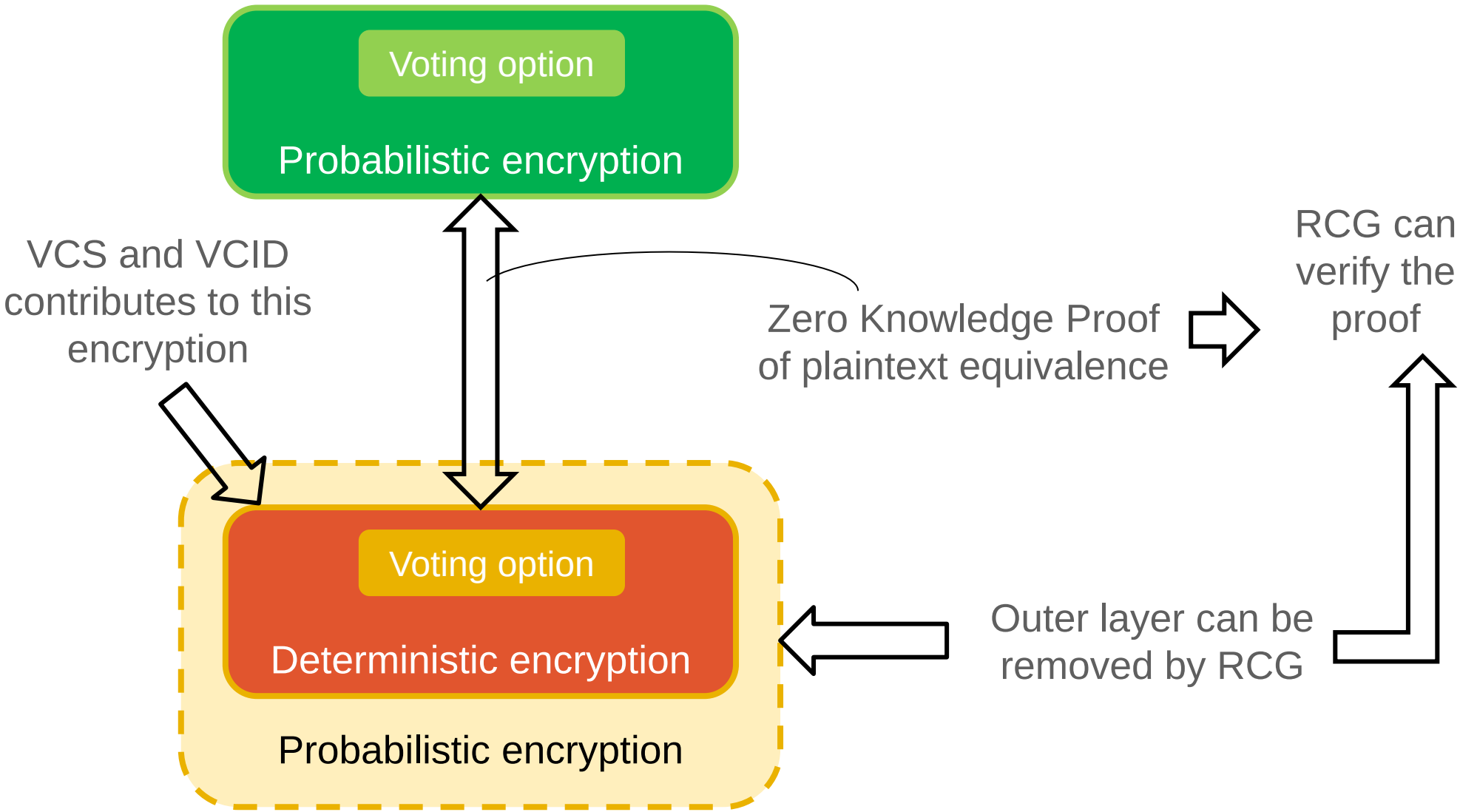
Parti	Returkode
Disney Party	186479
Marvel Party	657294

Kandidat	Returkode
Donald Duck, f. 1934, Disneyland	453782
Peter Pan, f. 1904, Disneyland	083128
Mickey Mouse, f. 1928, Disneyland	537287
Minnie Mouse, f. 1928, Disneyland	975659
Aurora Princess, f. 1959 , Disneyland	736181

Vote Card ID

Return code voting option

Probabilistic and deterministic encryption



Initial settings

- Participants of the voting process: voter *V*, voting client *VC*, voting server *VCS*, validation server *RCG.*

- Asymmetric encryption: ElGamal.
  - Election parameters:
    - *p* is a safe prime (*p=2q+1 where q is also prime)*.
    - *g* is a generator of *Gq,* a *q*-order subgroup of *Zp**.

  - Key pairs:
    - Election: private key x_e, public key h_e. $h_e \equiv g^{x_e} \bmod p$
    - RCG: private key x_{rcg}, public key h_{rcg}. $h_{rcg} \equiv g^{x_{rcg}} \bmod p$

- Symmetric keys
  - RCG: secret key K_{rcg}.
  - VCS: secret key K_{vcs}.
  - VCID: secret key specific of each voting card

Vote preparation (i)

1. Vote preparation for vote casting:
   - The voter (V) chooses her selections (v), and the Voting Client (VC) encrypts them using the ElGamal cryptosystem and a random exponent (r). A ZKP of plaintext independence ($proof_1$) is also generated.

     V -> VC: v

     VC: $c_{prob} = (v \cdot h_e^r, g^r) = (\alpha, \beta)$, $proof_1$

2. Vote preparation for verification (I):
   - The voter selections are encrypted again, using the ElGamal cryptosystem and a fixed exponent obtained from two values (a,d) calculated using the Voting Card Identifier (VCID) and the selections made by the voter:
     - Fixed exponent generated in the Voting Client.
       - VC: $a = H(VCID, v)$
     - Fixed exponent generated by the VCS.
       - VC -> VCS: $b = H(VCID)$
       - VCS -> VC: $d = H(K_{vcs}, b)$
     - C: $c_{det} = (v \cdot h_e^{a+d}, g^{a+d}) = (\alpha', \beta')$

These values could be hidden from VCS and potential observers by using a blind signature

2. Vote preparation for verification (II):
   - Generation of a ZKP of plaintext equivalence, showing that both encryptions have the same voting option encrypted
     - C: $proof_2$ relates:

$$c_{prob} = (v \cdot h_e^r, g^r) = (\alpha, \beta)$$

$$c_{det} = (v \cdot h_e^{a+d}, g^{a+d}) = (\alpha', \beta')$$

Both ciphertexts have the same *v* encrypted

     - C proves knowledge of the equivalent encryption exponent

$$(\alpha'/\alpha, \beta'/\beta) = (h_e^{(a+d)/r}, g^{(a+d)/r})$$

   - Re-encryption of the deterministic cipher text in order to prevent VCS from trying a brute-force attack over the value *a* or an observer from knowing about a voter voting twice the same option:
     - C: $c'_{det} = (v \cdot h_e^{a+d} \cdot h_{rcg}^r, g^{a+d} \cdot h_{rcg}^{r'}, g^{r'}) = (\alpha'', \beta'', \gamma)$

3. Vote Casting:
   - Both cipher texts and proofs are digitally signed in the Voting Client and sent to the VCS:
     - VC -> VCS: $\mathrm{Sig}((\alpha,\beta),(\alpha'',\beta'',\gamma),proof_1,proof_2;S_{voter})$
   - VCS verifies the ZKP of cipher text independence and the voter digital signature before forwarding the message to the RCG.

$$\mathrm{Sig}((\alpha,\beta),(\alpha'',\beta'',\gamma),proof_1,proof_2;S_{voter}) \quad P_{voter}$$

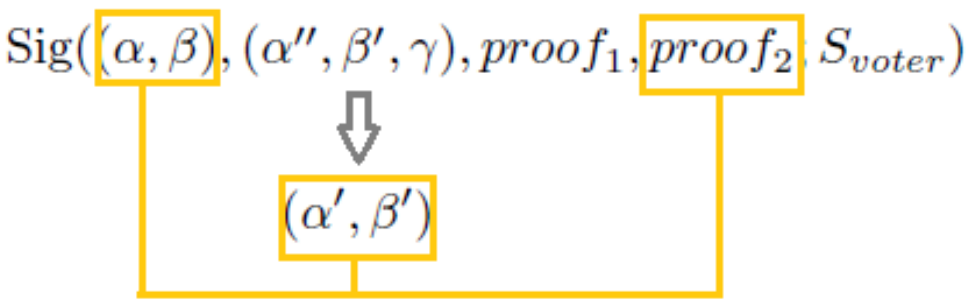
     - VCS -> RCG: $\mathrm{Sig}((\alpha,\beta),(\alpha'',\beta'',\gamma),proof_1,proof_2;S_{voter})$

4. Vote verification at RCG:
   - RCG verifies the ZKP of cipher text independence and the voter digital signature.

$$\text{Sig}((\alpha, \beta), (\alpha'', \beta'', \gamma), proof_1, proof_2; S_{voter}) \quad P_{voter}$$

   - RCG decrypts the second cipher text to obtain the deterministic encryption value generated in the Voting Client.
     - RCG: $(\alpha', \beta') = (\alpha'' \cdot (\beta^{-x_{rcg}}), \beta'' \cdot (\gamma^{-x_{rcg}})) = (v \cdot h_e^{a+d}, g^{a+d})$

   - RCG can then verify the ZKP of plaintext equivalence to ensure both cipher texts contain the same encrypted voting options.



Return Code Generation (ii)

5. Return Code generation:
   - RCG generates the Return Code from the deterministic cipher text using its secret key and sends it back to the voter by an alternative channel (e.g. by SMS).
     - RCG -> V: $\mathrm{RC} = H(\alpha'; K_{rcg})$
   - The voter can check that the Return Code value matches that assigned to her selection in the Voting Card.

6. Sending Voting Receipt:
   - RCG generates a Voting Receipt as a confirmation for the VCS that the operations have been successful, so that the VCS stores the vote in the Ballot Box.
     - RCG -> VCS: $\mathrm{Sig}(H(\alpha, \beta); S_{rcg})$

7. Storing the vote and sending confirmation:
   - VCS receives the Voting Receipt and verifies the digital signature.

$$\boxed{\mathrm{Sig}(H(\alpha,\beta);S_{rcg})} \quad \boxed{P_{rcg}}$$

   - VCS publishes the Voting Receipt in a Bulletin Board, stores the vote in the Ballot Box and forwards the receipt to the Voting Client, which shows it to the voter.
     - VCS -> Bulletin Board: $\mathrm{Sig}(H(\alpha,\beta);S_{rcg})$
     - VCS -> BB: Store $\mathrm{Sig}((\alpha,\beta),(\alpha'',\beta'',\gamma),proof_1,proof_2;S_{voter})$
     - VCS -> VC: $\mathrm{Sig}(H(\alpha,\beta);S_{rcg})$
     - VC -> V: $\mathrm{Sig}(H(\alpha,\beta);S_{rcg})$

   - The voter verifies that the vote has been stored in the Ballot Box by comparing her Voting Receipt with the contents of the Bulletin Board.

Index

- Introduction
- Proposal
- Security
- Conclusions

Risk Analysis (i)

- Return Code compromise:
  - Risks:
    - Manipulation of a vote: Requires compromise also VC and RCG
    - Voter privacy compromise: Requires compromise also RCG
  - Mitigations:
    - Return codes are sent through a different channel from the voting one
    - VC and RCG are not connected
    - Return Code sheets are not originally linked to voters
    - Voters can use different Return Code sheets (if multiple voting is allowed)

- VC compromise:
  - Risks:
    - Voter privacy compromise: VC could capture the voter intent
    - Manipulation of a vote: Requires compromise also return codes and RCG
  - Mitigations:
    - Multiple voting
    - VC and RCG are not connected
    - Voters can use different Return Code sheets (if multiple voting is allowed)
    - The hash of the ballot box votes are published in a bulletin board

Risk Analysis (ii)

- VCS compromise :
  - Risks:
    - Manipulation of a vote: Requires compromise also VC, RCG and Return Codes
    - Voter privacy compromise: Requires compromise RCG and perform brute force attack
  - Mitigations:
    - Return codes are sent to voter through a different channel
    - Deterministic encryption value has a length of 2048bits obtained using a KDF from a random 80 bits seed

- RCG compromise:
  - Risks:
    - Voter privacy compromise: Requires compromise Return Codes sheets
    - Manipulation of a vote: Requires compromise also Return Codes and VC
  - Mitigations:
    - Multiple voting
    - VC and RCG are not connected
    - Voters can use different Return Code sheets (if multiple voting is allowed)
    - The hash of the ballot box votes are published in a Bulletin Board

Index

- Introduction
- Proposal
- Security
- Conclusions

Conclusions

- The proposal implements a voter verifiability approach with the usability of pollsterless approach but without requiring candidate codes for casting votes (i.e., better usability and accessibility)

- The proposal is not vulnerable to a single component compromise

- The main risk of the proposal is the strength of the VCID to prevent attackers with access to the VCS and RCG keys from compromising the privacy of a vote using a brute force attack.

www.scytl.com