Thoughts on JCJ05

Reto E. Koenig

Swissvote

15. April 2010

What it is all about

For an e-voting system to be secure, it has to function without vulnerabilities in potentially insecure environments such as the internet. For this, it has to be implemented according to an intrinsically secure design. Despite the complexity of designing and implementing such a system, some criteria seem to be unanimously accepted as the core security requirements for e-voting systems.

What this means: A system offers...

Accuracy: if casted votes can not be altered (integrity), valid votes can not be eliminated from the final tally (completeness), and invalid votes are not counted in the final tally (soundness).

What this means: A system offers...

Accuracy: if casted votes can not be altered (integrity), valid votes can not be eliminated from the final tally (completeness), and invalid votes are not counted in the final tally (soundness).

Democracy: if only authorized voters can vote (eligibility) and eligible voters can only vote once (uniqueness).

What this means: A system offers...

Accuracy: if casted votes can not be altered (integrity), valid votes can not be eliminated from the final tally (completeness), and invalid votes are not counted in the final tally (soundness).

Democracy: if only authorized voters can vote (eligibility) and eligible voters can only vote once (uniqueness).

Privacy: if no casted vote can be linked to its voter, neither by voting authorities nor anyone else (anonymity), and no voter can prove that he or she voted in a particular way (receipt-freeness).

What this means: A system offers...

Accuracy: ...

Democracy: ...

Privacy: ...

Verifiability: individually verifiable if voters can independently verify that their own votes have been counted correctly in the final tally and universally verifiable, if everyone can do that.

What this means: A system offers...

- **Accuracy:** ...
- **Democracy:** ...
- **Privacy:** ...
- **Verifiability:** individually verifiable if voters can independently verify that their own votes have been counted correctly in the final tally and universally verifiable, if everyone can do that.
- **Fairness:** if no intermediate results can be obtained before the voting period ends.

E-Banking < E-Voting

What makes E-Banking so easy and E-Voting so hard:

E-Banking: ctrl-z A wrong transaction can be undone (no privacy)

E-Voting: ctrl-z A wrong transaction can not be undone (privacy)

E-Banking The bank wants to have a trustworthy system

E-Voting The party in power wants to stay in power

Enemies of an E-Voting system

- Voters
- External parties
- Internal parties
- The party that rules the country
- The men running the e-voting system

How to gain Power

- Buy
- Bribe
- Coerce

But 2005 privacy and Coercion-Resistance has been redefined:

Privacy < Coercion-Resistance

privacy Even if someone observes the voter (passively) during the actual voting process, the voters will should not be unveiled.

coercion-resistance Even if someone actively interacts with the voter during the voting process, the voters will should not be unveiled.

Even: Neither the observer / coercer **nor the voter** should be able to unveil the voters will

Grail-Elements

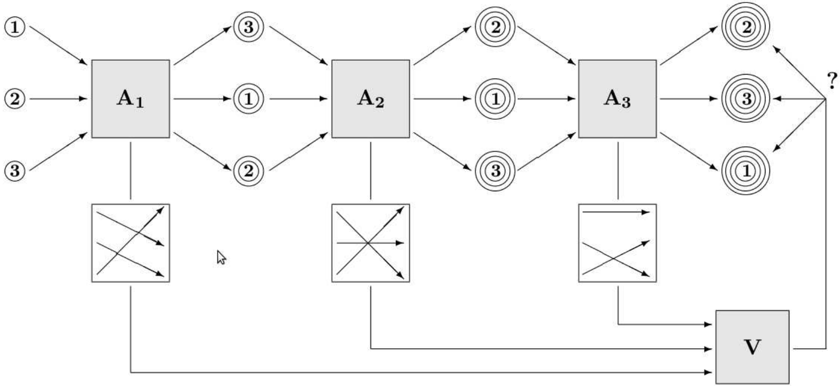
Homomorphic calculations E-Voting systems based homomorphic schemes were introduced in 1994 by Benaloh and Tuinstra

Grail-Elements

Homomorphic calculations E-Voting systems based homomorphic schemes were introduced in 1994 by Benaloh and Tuinstra

Non-Transferable-Verifiability An important security aspect within E-Voting systems: Voter can not prove the will on the casted vote to a third party. (Also known as receipt). First working system showed in 2000 by Hirt and Sako.

Hrt00



Unacceptable Constraint

- The system requires an untappable channel [a]...

[a] physically secure; not achievable by cryptographic means

Unacceptable Constraint

- The system requires an untappable channel [a]...
- ...during **voting** phase.

[a] physically secure; not achievable by cryptographic means

Unacceptable Constraint

- The system requires an untappable channel [a]...
- ...during **voting** phase.
- not coercer resistant

[a] **physically secure; not achievable by cryptographic means**

What a Coercer Can Achieve

Forced abstention You shall not vote

What a Coercer Can Achieve

Forced abstention You shall not vote

To Render the System Coercion Resistant

- The system has to disable the voter to prove the act of a vote casting to a third party.

What a Coercer Can Achieve

Forced abstention You shall not vote

Forced Randomization You shall vote like a dice

What a Coercer Can Achieve

Forced abstention You shall not vote

Forced Randomization You shall vote like a dice

To Render the System Coercion Resistant

- The system has to disable the voter to prove the act of a vote casting to a third party.
- The system has to allow the voter to fake-vote (not to be confused with re-vote)

What a Coercer Can Achieve

Forced abstention You shall not vote

Forced Randomization You shall vote like a dice

Forced Simulation / Impersonation I am You

What a Coercer Can Achieve

Forced abstention You shall not vote

Forced Randomization You shall vote like a dice

Forced Simulation / Impersonation I am You

To Render the System Coercion Resistant

- The system has to disable the voter to prove the act of a vote casting to a third party.
- The system has to allow the voter to fake-vote (not to be confused with re-vote)
- The system has to allow the voter to distribute false identity-mark (credentials)

In 2005 Jules, Catalano and Jakobsson introduced a system providing the following:

Grail-Elements

- Homomorphic calculations

In 2005 Jules, Catalano and Jakobsson introduced a system providing the following:

Grail-Elements

- Homomorphic calculations
- Non-Transferable-Verifiability

In 2005 Jules, Catalano and Jakobsson introduced a system providing the following:

Grail-Elements

- Homomorphic calculations
- Non-Transferable-Verifiability
- Coercion resistant

Players and Their Abbreviation

R Registration Authority (multiple instances)

[a]The sender can verify (designated), The receiver does not know the origin
[b]The sender looses track, the receiver is allowed to know the sender

Players and Their Abbreviation

R Registration Authority (multiple instances)

T Tallying Authority (multiple instances)

[a] The sender can verify (designated), The receiver does not know the origin

[b] The sender looses track, the receiver is allowed to know the sender

Players and Their Abbreviation

R Registration Authority (multiple instances)

T Tallying Authority (multiple instances)

V_i Voter i

---

[a] The sender can verify (designated), The receiver does not know the origin

[b] The sender looses track, the receiver is allowed to know the sender

Players and Their Abbreviation

R	Registration Authority (multiple instances)
T	Tallying Authority (multiple instances)
V_i	Voter i
BB	Bulletin Board (multiple instances)

[a] The sender can verify (designated), The receiver does not know the origin
[b] The sender looses track, the receiver is allowed to know the sender

Players and Their Abbreviation

- R Registration Authority (multiple instances)
- T Tallying Authority (multiple instances)
- V_i Voter i
- BB Bulletin Board (multiple instances)
- AN Anonymous Net (Preserving 'shape' and temporal order)[a]

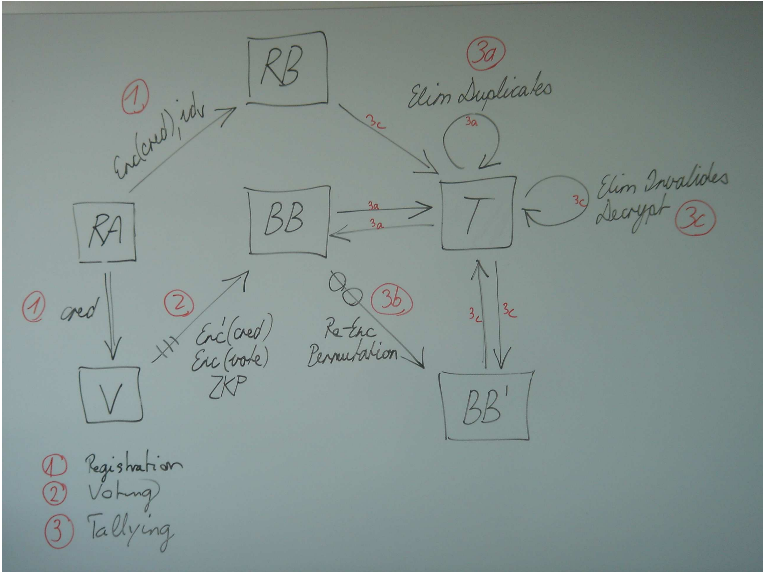
[a] The sender can verify (designated), The receiver does not know the origin
[b] The sender looses track, the receiver is allowed to know the sender

Players and Their Abbreviation

R	Registration Authority (multiple instances)
T	Tallying Authority (multiple instances)
V_i	Voter i
BB	Bulletin Board (multiple instances)
AN	Anonymous Net (Preserving 'shape' and temporal order)[a]
MN	Mixed Net (changing 'shape' and temporal order)[b]

[a] The sender can verify (designated), The receiver does not know the origin
[b] The sender looses track, the receiver is allowed to know the sender

Overview



RB
3a
Elim Duplicates
1
Enc(cred), idv
3c
3a
RA
BB
3a
3a
T
3c
Elim Invalides
Decrypt
3c
1
cred
2
Enc'(cred)
Enc(vote)
ZKP
3b
Re-Enc
Permutation
3c
3c
V
BB'
1 Registration
2 Voting
3 Tallying

General Protocol description

1. Setup If not already available, key pairs are generated by R and T . The candidate slate C is published by R (or T) with appropriate integrity protection.

General Protocol description

1. Setup — If not already available, key pairs are generated by R and T . The candidate slate C is published by R (or T) with appropriate integrity protection.
2. Registration — Given successful verification of the id and eligibility, an individual becomes a registered voter V_i, receiving from R a credential permitting participation in elections. R publishes a signed voter roll L on BB

General Protocol description

1. Setup If not already available, key pairs are generated by R and T . The candidate slate C is published by R (or T) with appropriate integrity protection.
2. Registration Given successful verification of the id and eligibility, an individual becomes a registered voter V_i, receiving from R a credential permitting participation in elections. R publishes a signed voter roll L on BB
3. Voting Referring to C and the credentials Vs cast their ballots to BB via anonymous channel.

General Protocol description

1. **Setup** If not already available, key pairs are generated by R and T . The candidate slate C is published by R (or T) with appropriate integrity protection.
2. **Registration** Given successful verification of the id and eligibility, an individual becomes a registered voter V_i, receiving from R a credential permitting participation in elections. R publishes a signed voter roll L on BB
3. **Voting** Referring to C and the credentials Vs cast their ballots to BB via anonymous channel.
4. **Tallying** The T processes the contents of BB so as to produce a tally vector X specifying the outcome of the election, along with a proof of correctness P of the tally.

General Protocol description

1. **Setup** If not already available, key pairs are generated by R and T . The candidate slate C is published by R (or T) with appropriate integrity protection.
2. **Registration** Given successful verification of the id and eligibility, an individual becomes a registered voter V_i, receiving from R a credential permitting participation in elections. R publishes a signed voter roll L on BB
3. **Voting** Referring to C and the credentials Vs cast their ballots to BB via anonymous channel.
4. **Tallying** The T processes the contents of BB so as to produce a tally vector X specifying the outcome of the election, along with a proof of correctness P of the tally.
5. **Verification** Any player, whether or not a participant in the election, can refer to BB, P and L to verify the correctness of the tally produced by T in the previous phase.

Acceptable constraint

- The system requires an untappable channel channel [a]...

[a] physically secure; not achievable by cryptographic means

Acceptable constraint

- The system requires an untappable channel channel [a]...
- ...during **registration** phase

[a] **physically secure; not achievable by cryptographic means**

Setup

R Key pair SK_R, PK_R is generated; PK_R is published[a]

[a] Created on distributed threshold manner
[b] Created on distributed threshold manner

Setup

R Key pair SK_R, PK_R is generated; PK_R is published[a]

T Key pair SK_T, PK_T is generated; PK_T is published[b]

[a]Created on distributed threshold manner

[b]Created on distributed threshold manner

Setup

R Key pair SK_R, PK_R is generated; PK_R is published[a]

T Key pair SK_T, PK_T is generated; PK_T is published[b]

$[MN]$ Is set up

[a] Created on distributed threshold manner

[b] Created on distributed threshold manner

Setup

R Key pair SK_R, PK_R is generated; PK_R is published[a]

T Key pair SK_T, PK_T is generated; PK_T is published[b]

[MN] Is set up

[AN] Is set up and keys of the servers are known

[a] Created on distributed threshold manner

[b] Created on distributed threshold manner

Setup

R	Key pair SK_R, PK_R is generated; PK_R is published[a]
T	Key pair SK_T, PK_T is generated; PK_T is published[b]
	[MN] Is set up
	[AN] Is set up and keys of the servers are known
BB	Is set up

[a] Created on distributed threshold manner
[b] Created on distributed threshold manner

Credentials

1. V_i goes to (=untappable channel) the R and proofs eligibility

σ_i can be used for multiple voting-sessions

[a] Created on distributed threshold manner with multiple R
[b] L is maintained digitally signed on the bulletin board BB

Credentials

1. V_i goes to (=untappable channel) the R and proofs eligibility
2. R generates String $\sigma_i \in_R G$ [a]

σ_i can be used for multiple voting-sessions

[a] Created on distributed threshold manner with multiple R
[b] L is maintained digitally signed on the bulletin board BB

Credentials

1. V_i goes to (=untappable channel) the R and proofs eligibility
2. R generates String $\sigma_i \in_R G$ [a]
3. V_i generates $S_i = E_{PK_T}(\sigma_i)$ and sends it to R

σ_i can be used for multiple voting-sessions

[a] Created on distributed threshold manner with multiple R
[b] L is maintained digitally signed on the bulletin board BB

Credentials

1. V_i goes to (=untappable channel) the R and proofs eligibility
2. R generates String $\sigma_i \in_R G$ [a]
3. V_i generates $S_i = E_{PK_T}(\sigma_i)$ and sends it to R
4. R encrypts or re-encrypts $S_i = EPK_T(\sigma_i)$ and sends it to R

σ_i can be used for multiple voting-sessions

[a] Created on distributed threshold manner with multiple R
[b] L is maintained digitally signed on the bulletin board BB

Credentials

1. V_i goes to (=untappable channel) the R and proofs eligibility
2. R generates String $\sigma_i \in_R G$ [a]
3. V_i generates $S_i = E_{PK_T}(\sigma_i)$ and sends it to R
4. R encrypts or re-encrypts $S_i = EPK_T(\sigma_i)$ and sends it to R
5. R puts id of V_i and S_i to the voter roll L [b]

σ_i can be used for multiple voting-sessions

[a] Created on distributed threshold manner with multiple R
[b] L is maintained digitally signed on the bulletin board BB

Afterthought on S_i generation

- It is most important that the final S_i is not encrypted by V_i.

Afterthought on S_i generation

- It is most important that the final S_i is not encrypted by V_i.
- Otherwise V_i always could decrypt S_i to prove the valid σ (ElGamal Trapdoor 2)

Afterthought on S_i generation

- It is most important that the final S_i is not encrypted by V_i.
- Otherwise V_i always could decrypt S_i to prove the valid σ (ElGamal Trapdoor 2)
- If R re-encrypts S_i with a dedicated proof, it is absolutely essential...

Afterthought on S_i generation

- It is most important that the final S_i is not encrypted by V_i.
- Otherwise V_i always could decrypt S_i to prove the valid σ (ElGamal Trapdoor 2)
- If R re-encrypts S_i with a dedicated proof, it is absolutely essential...
- ... that V_i can get a proof for ANY credential in p-time (and within 'usability')

Afterthought on S_i generation

- It is most important that the final S_i is not encrypted by V_i.
- Otherwise V_i always could decrypt S_i to prove the valid σ (ElGamal Trapdoor 2)
- If R re-encrypts S_i with a dedicated proof, it is absolutely essential...
- ... that V_i can get a proof for ANY credential in p-time (and within 'usability')
- This problem-domain is discussed in Schweisgut07

Definitions

c_j Candidate choice.

And not as expected $h = g_1^{S_T} g_2^{S_T}$

Definitions

c_j Candidate choice.

σ_i Credential of V_i.

And not as expected $h = g_1^{s_T} g_2^{s_T}$

Definitions

c_j Candidate choice.

σ_i Credential of V_i.

a_1, a_2 $\in_R Z_q$.

And not as expected $h = g_1^{S_T} g_2^{S_T}$

Definitions

c_j Candidate choice.

σ_i Credential of V_i.

a_1, a_2 $\in_R Z_q$.

ElGamal Setup g_1, g_2

And not as expected $h = g_1^{S_T} g_2^{S_T}$

Definitions

c_j Candidate choice.

σ_i Credential of V_i.

a_1, a_2 $\in_R Z_q$.

ElGamal Setup g_1, g_2

ElGamal private Key S_T

And not as expected $h = g_1^{S_T} g_2^{S_T}$

Definitions

c_j Candidate choice.

σ_i Credential of V_i.

a_1, a_2 $\in_R Z_q$.

ElGamal Setup g_1, g_2

ElGamal private Key S_T

ElGamal public Key: $h = g_1^{S_T}$.

And not as expected $h = g_1^{S_T} g_2^{S_T}$

V voting

Candidate choice $E_1^{(i)} = (\alpha_1, \alpha_1', \beta_1) = (g_1^{a_1}, g_2^{a_1}, c_j h^{a_1})$,
NIZKPK of knowledge of c_j AND $c_j \in C$.

Voter Credential $E_2^{(i)} = (\alpha_2, \alpha_2', \beta_2) = (g_1^{a_2}, g_2^{a_2}, \sigma_i h^{a_2})$
NIZKPK of knowledge of σ_i

NIZKPK that α_i, α_i' have the same discrete log with respect to g_1 and g_2

1.NIZKPK: Otherwise V can be forced to submit an observable invalid choice

2.NIZKPK: Makes it (non-malleable) impossible to validly copy an encrypted credential for *BB* for a new ballot

Vote casting

- V sends the vote to AN

[a] This restriction should be investigated with the goal of getting rid of it

Vote casting

- V sends the vote to AN
- AN sends the vote to BB

[a] This restriction should be investigated with the goal of getting rid of it

Vote casting

- V sends the vote to AN
- AN sends the vote to BB
- In JCJ05 V is only able to read BB after the vote-casting phase is over.[a]

[a] This restriction should be investigated with the goal of getting rid of it

After-Thought BB

It seems as if BB can be made public all the time:
After a vote-casting, the casted vote can be identified and the credential can be reconstructed and proven by the vote-sender. But no proof can be forced to show that the credential in question is a valid credential.

Duplicate Vote Elimination

- T eliminates (pair-wise) votes with same credentials (PET)[a]
- ZKPK for each action

[a] It marks the 'last' vote

MixNet-Re-Encryption-Shuffling

- *MN* shuffles re-encrypts the 'last' votes

MixNet-Re-Encryption-Shuffling

- *MN* shuffles re-encrypts the 'last' votes
- A ZKPK for each re-encrypted vote proves its existence in the original 'last' vote list

MixNet-Re-Encryption-Shuffling

- *MN* shuffles re-encrypts the 'last' votes
- A ZKPK for each re-encrypted vote proves its existence in the original 'last' vote list
- If the size of the two lists are equal, this proves equality of the two lists.

Elimination of Invalid Votes

- T eliminates votes with invalid credentials. Done by a (PET)[a] with the Voter Slate
- ZkPK for each action

[a] Threshold decryption

Decryption

- T decrypts each vote[a]
- ZkPK for each action

[a] Threshold decryption

Questions?

Plaintext Equality Test

- ElGamal $(x_1, y_1)(x_2, y_2)$ encrypt same plaintext? (Without wanting to know plaintexts)

Plaintext Equality Test

- ElGamal $(x_1, y_1)(x_2, y_2)$ encrypt same plaintext? (Without wanting to know plaintexts)
- $(g^{r_1}, h^{r_1} m), (g^{r_2}, h^{r_2} m')$ with $m =^? m'$ where $h = g^s$

Plaintext Equality Test

- ElGamal $(x_1, y_1)(x_2, y_2)$ encrypt same plaintext? (Without wanting to know plaintexts)
- $(g^{r_1}, h^{r_1} m), (g^{r_2}, h^{r_2} m')$ with $m =^? m'$ where $h = g^s$
- $(x_1/x_2, y_1/y_2) = (x_3, y_3)$

Plaintext Equality Test

- ElGamal $(x_1, y_1)(x_2, y_2)$ encrypt same plaintext? (Without wanting to know plaintexts)
- $(g^{r_1}, h^{r_1} m), (g^{r_2}, h^{r_2} m')$ with $m =^? m'$ where $h = g^s$
- $(x_1/x_2, y_1/y_2) = (x_3, y_3)$
- $dec(x_3, y_3) = y_3 x_3^{-s} = (g^s)^{r_3} \frac{m}{m'} (g^{r_3})^{-s} = \frac{m}{m'} =^? 1$

Plaintext Equality Test

- ElGamal $(x_1, y_1)(x_2, y_2)$ encrypt same plaintext? (Without wanting to know plaintexts)
- $(g^{r_1}, h^{r_1} m), (g^{r_2}, h^{r_2} m')$ with $m =^? m'$ where $h = g^s$
- $(x_1/x_2, y_1/y_2) = (x_3, y_3)$
- $dec(x_3, y_3) = y_3 x_3^{-s} = (g^s)^{r_3} \frac{m}{m'} (g^{r_3})^{-s} = \frac{m}{m'} =^? 1$
- Problem: PkZk only works if $m = m'$ but fails otherwise

Plaintext Equality Test

- ElGamal $(x_1, y_1)(x_2, y_2)$ encrypt same plaintext? (Without wanting to know plaintexts)
- $(g^{r_1}, h^{r_1} m), (g^{r_2}, h^{r_2} m')$ with $m =^? m'$ where $h = g^s$
- $(x_1/x_2, y_1/y_2) = (x_3, y_3)$ ... (x_3^z, y_3^z) where $z \in_R$
- $dec(x_3^z, y_3^z) = y_3^z x_3^{-sz} = (g^s)^{r_3 z} \frac{m^z}{m'^z} (g^{r_3})^{-sz} = (\frac{m}{m'})^z =^? 1$
- Solution: PkZk works if $m = m'$ and $m \neq m'$

Threshold decryption

- $(x, y) = (g^{\alpha}, h^{\alpha} m) | h = g^{s}$

Threshold decryption

- $(x, y) = (g^{\alpha}, h^{\alpha} m) | h = g^{s}$
- s distributed on $T = (T_1, ..., T_n)$instances

Threshold decryption

- $(x, y) = (g^\alpha, h^\alpha m) | h = g^s$
- s distributed on $T = (T_1, ..., T_n)$instances
- Each T_j presents $z_j = x^{s_j}$, PkZk $\log_g(h_j) = \log_x(h_j)$

Threshold decryption

- $(x, y) = (g^{\alpha}, h^{\alpha} m) | h = g^{s}$
- s distributed on $T = (T_1, ..., T_n)$instances
- Each T_j presents $z_j = x^{s_j}$, PkZk $\log_g(h_j) = \log_x(h_j)$
- I represents the set of T_j where verification(PkZk) = true

Threshold decryption

- $(x, y) = (g^{\alpha}, h^{\alpha} m) | h = g^{s}$
- s distributed on $T = (T_1, \ldots, T_n)$instances
- Each T_j presents $z_j = x^{s_j}$, PkZk $\log_g(h_j) = \log_x(h_j)$
- I represents the set of T_j where verification(PkZk) = true
- $m = g^{-\alpha s} g^{\alpha s} m = (g^{\alpha})^{-\sum\limits_{j \in I} l_j(0) \cdot s_j} h^{\alpha} m$

Threshold decryption

- $(x, y) = (g^{\alpha}, h^{\alpha} m) | h = g^{s}$
- s distributed on $T = (T_1, ..., T_n)$instances
- Each T_j presents $z_j = x^{s_j}$, PkZk $\log_g(h_j) = \log_x(h_j)$
- I represents the set of T_j where verification(PkZk) = true
- $m = g^{-\alpha s} g^{\alpha s} m = (g^{\alpha})^{-\sum_{j \in I} l_j(0) \cdot s_j} h^{\alpha} m$
- This is feasible as long as $|J| \geq t$ where t is a certain threshold.

Threshold keyGeneration

986220914.pdf p. 59

Thoughts on Anonymous Channel / Re-Voting

- JCJ05 allows multiple vote-casting by V_i
- JCJ05 needs the order of the entries in BB (latest vote wins)
- Therefore no mix-Net approach with shuffling allowed or...
- ... some 'encrypted' timestamp within the ballot which makes the vote traceable
- But multiple vote casting does not strengthen the system against a coercer
- Weber eliminates this by counting the first (temporal) vote only

Shouldersurfing Resistance

- JCJ05 allows ONE correct credential σ and 'unlimitted' fake credentials
- if the voting legitimation $enc_T(\sigma)$ has to be calculated under observation...
- ...then Shouldersurfing resistance (over several Votings) requires that at least One correct and independent credential per Voting-session