Formal Analysis of Electronic Voting Systems

Mark Ryan
University of Birmingham

joint work with

Ben Smyth
Steve Kremer

Imperial College
21 April 2010

Outline

Electronic voting: potential

Electronic voting potentially offers

- Efficiency
  - higher voter participation
  - greater accuracy
  - lower costs
- Better security
  - vote-privacy even in presence of corrupt election authorities
  - voter verification, i.e. the ability of voters and observers to check the declared outcome against the votes cast.

Governments world over have been trialling e-voting, e.g. USA, UK, Canada, Brasil, the Netherlands and Estonia.

Can also be useful for smaller-scale elections (student guild, shareholder voting, trade union ballots, local government).

Current situation

The potential benefits have turned out to be hard to realise.

In UK

- May 2007 elections included 5 local authorities that piloted a range of electronic voting machines.
- Electoral Commission report concluded that the implementation and security risk was significant and unacceptable and recommends that no further e-voting take place until a sufficiently secure and transparent system is available.

In USA:

- Diebold controversy since 2003 when code leaked on internet.
  - Kohno/Stubblefield/Rubin/Wallach analysis concluded Diebold system far below even most minimal security standards. Voters without insider privileges can cast unlimited votes without being detected.

Current situation in USA, continued

- In 2007, Secr. of State for California commissioned "top-to-bottom" review by computer science academics of the four machines certified for use in the state. Result is a catalogue of vulnerabilities, including
  - appalling software engineering practices, such as hardcoding crypto keys in source code; bypassing OS protection mechanisms, ...
  - susceptibility of voting machines to viruses that propogate from machine to machine, and that could maliciously cause votes to be recorded incorrectly or miscounted
  - "weakness-in-depth", architecturally unsound systems in which even as known flaws are fixed, new ones are discovered.

  In response to these reports, she decertified all four types of voting machine for regular use in California, on 3 August 2007.

Current situation in Estonia

- Estonia is a tiny former Soviet republic (pop. 1.4M), nicknamed "e-Stonia" because of its tech-savvy character.
- Oct. 2005 local election allowed voters to cast ballots on internet. There were 9,317 electronic votes cast out of 496,336 votes in total (1.9%) participated online.
- Officials hailed the experiment a success. Said no reports of hacking or flaws. System based on linux.
- Voters need special ID smartcard, a $24 device that reads the card, and a computer with internet access. About 80% of Estonian voters have the cards anyway, also used since 2002 for online banking and tax records.

- Feb. 2007 general election: 30,275 voters used internet voting.

Internet voting and coercion resistance

The possibility of coercion (e.g. by family members) seems very hard to avoid for internet voting.

In Estonia, the threat is somewhat mitigated:

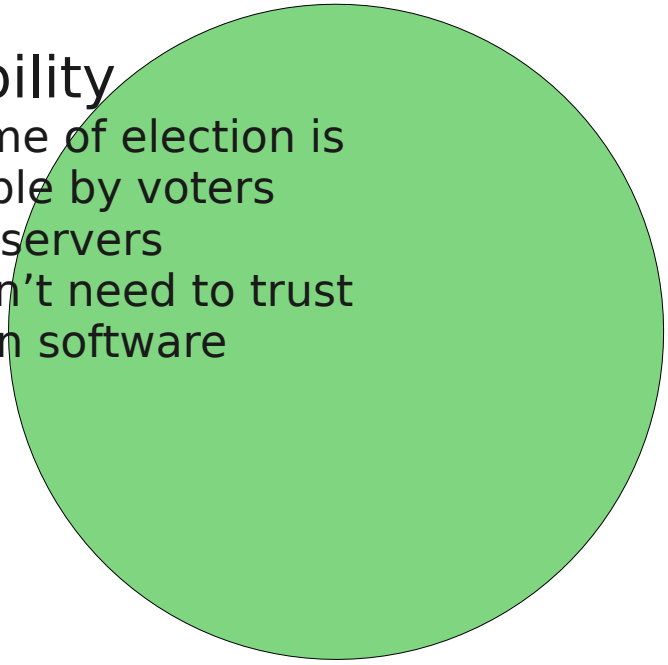
- Election system allows multiple online votes to be cast by the same person during the days of advance voting, with each vote cancelling the previous one.
- System gives priority to paper ballots; a paper ballot cancels any previous online ballot by the same person.

Where are we?

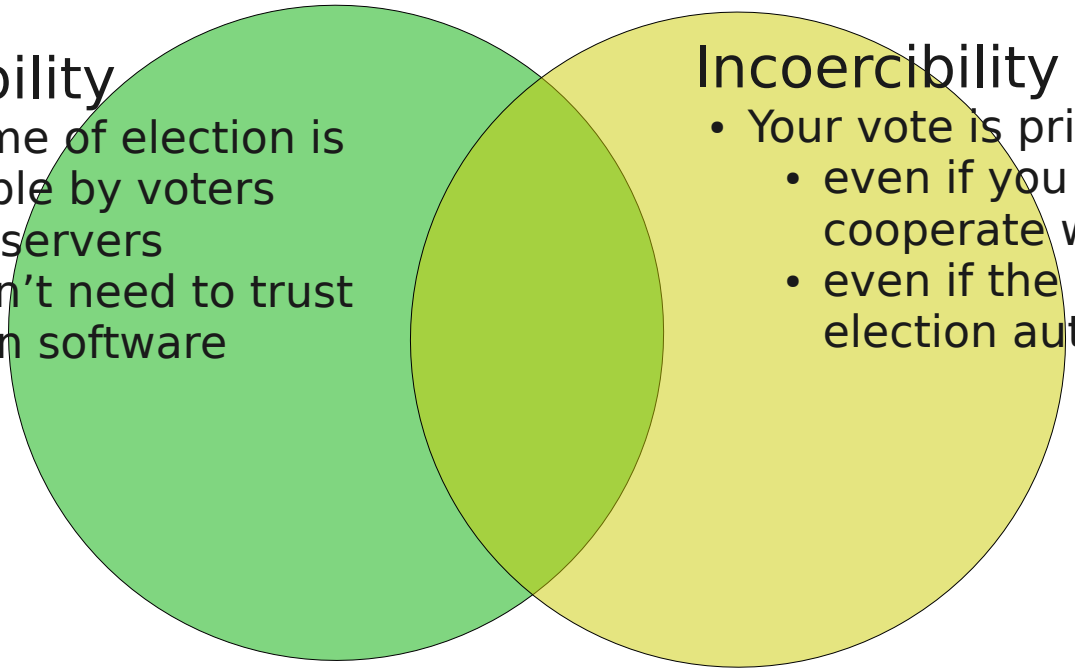
Desired properties

Verifiability

- Outcome of election is verifiable by voters and observers
- You don’t need to trust election software



Desired properties



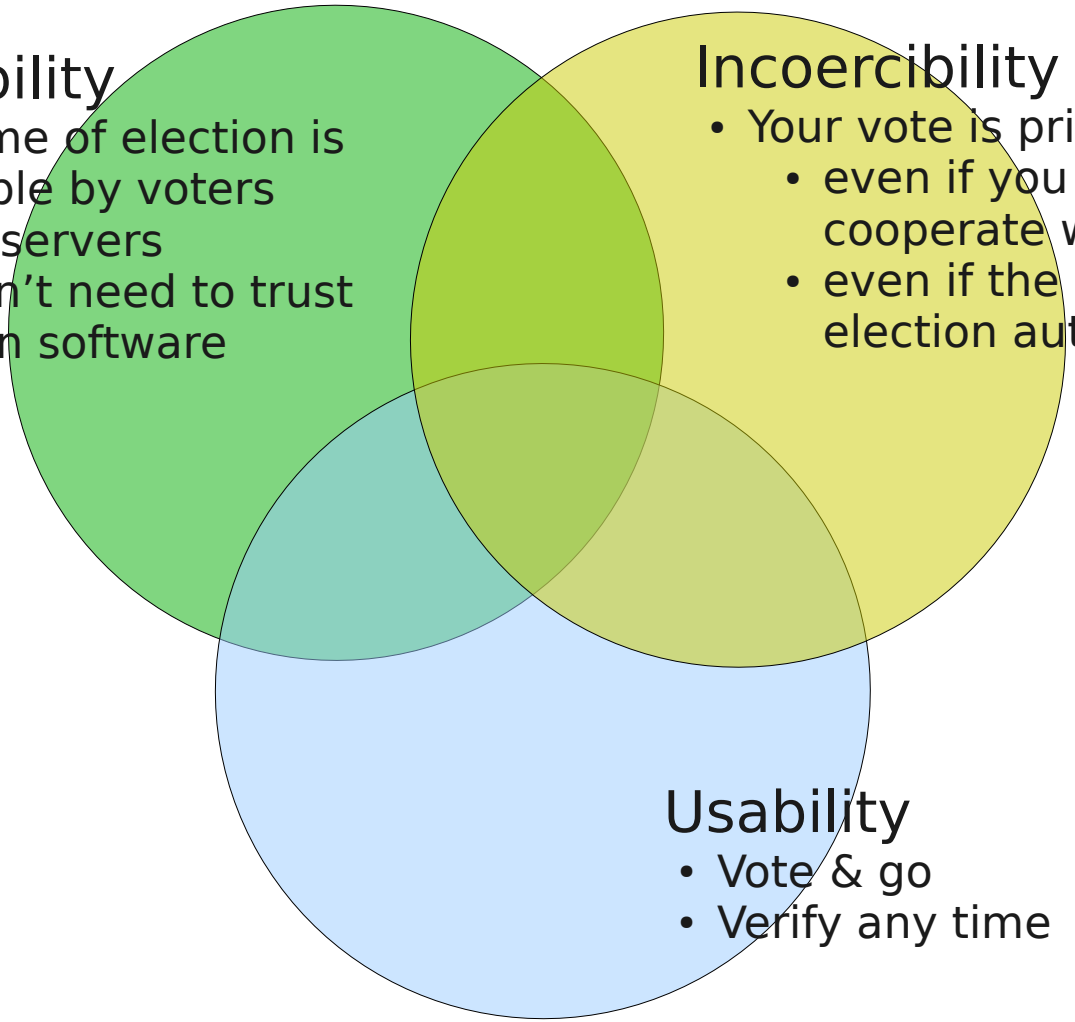
Verifiability

- Outcome of election is verifiable by voters and observers
- You don’t need to trust election software

Incoercibility

- Your vote is private
  - even if you try to cooperate with a coercer
  - even if the coercer is the election authorities

Desired properties



Verifiability

- Outcome of election is verifiable by voters and observers
- You don’t need to trust election software

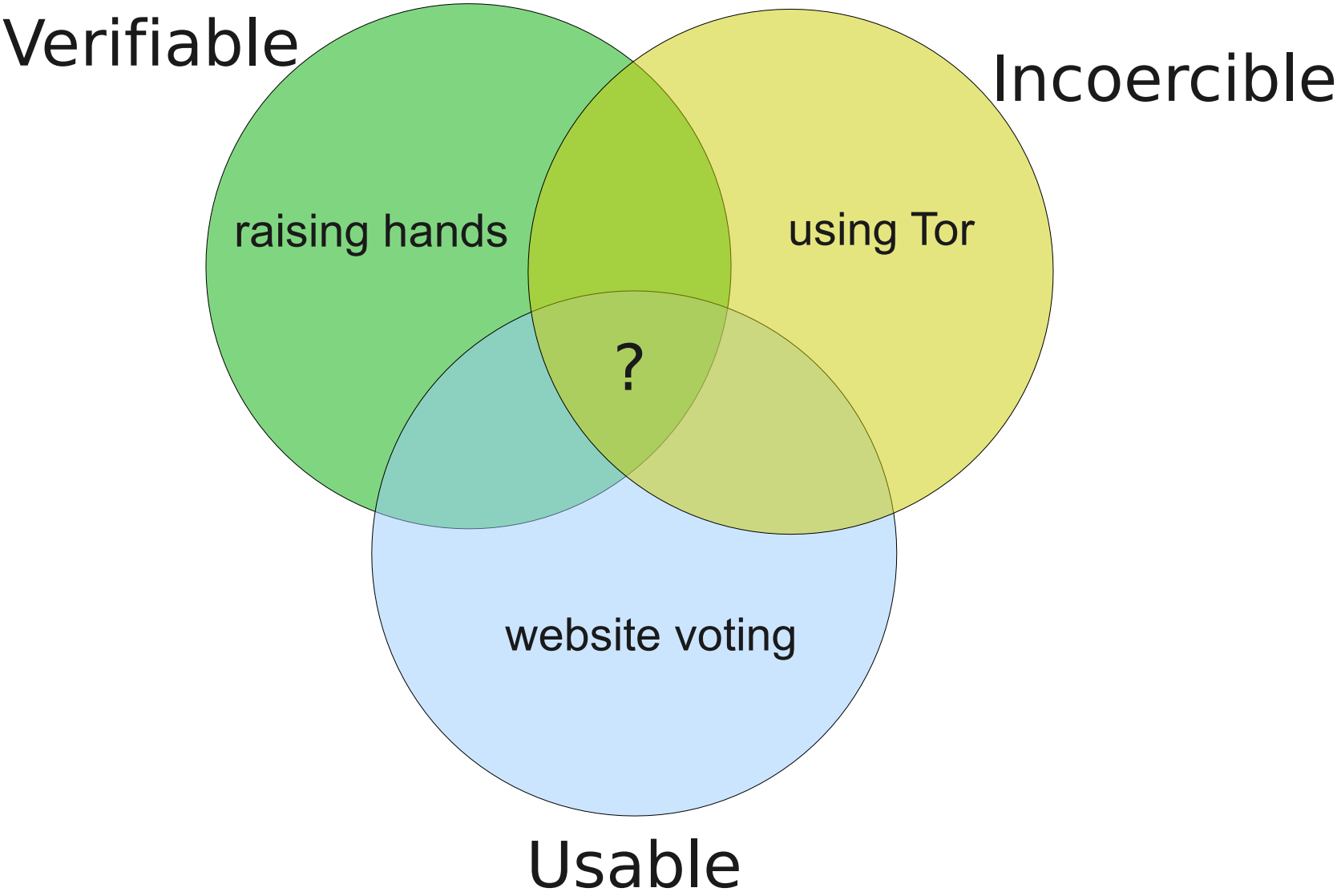
Incoercibility

- Your vote is private
  - even if you try to cooperate with a coercer
  - even if the coercer is the election authorities

Usability

- Vote & go
- Verify any time

Examples


Verifiable
Incoercible
raising hands
using Tor
?
website voting
Usable

How could it be secure?

Security by trusted client software

$\rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow$

- trusted by user
- does not need to be trusted by authorities or other voters

- not trusted by user
- doesn't need to be trusted by anyone

Where are we?

Election of president at University of Louvain

The election

- Helios 2.0
- 25,000 potential voters
  - 5000 registered, 4000 voted
  - Educated, but not technical
- 30% voters checked their vote
  - No valid complaints
- Verifiability
  - Anyone can write code to verify the election
  - Sample python code provided

No coercion resistance

- Only recommended for low-coercion environments
- Re-votes are allowed, but don't help w.r.t. "insider" coercer

[Adida/deMarneffe/Pereira/-Quisquater 09]

OPEN-AUDIT OF THE RESULTS OF THE RECTOR ELECTION 2009

The voting system used for this election provides *universally verifiable elections*. This means that:

1. a voter can verify that her ballot is cast as intended (her ballot reflects her own opinion),
2. a voter can verify that her ballot is included *unmodified* in the collection of ballots to be used at tally time,
3. anyone can verify that the election result is consistent with that collection of ballots.

Helios 2.0

$\rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow$

- User prepares ballot (encrypted vote) on her computer, together with ZKPs.
- Cut-and-choose auditability provides assurance of correctness
- Not much guarantee of privacy on client side.

- Ballots are checked & homomorphically combined into a single encrypted outcome.
- Outcome is decrypted by threshold of talliers.
- Proof of correct decryption.

Helios 2.0

$\rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow$

- User prepares ballot (encrypted vote) on her computer, together with ZKPs.
- Cut-and-choose auditability provides assurance of correctness
- Not much guarantee of privacy on client side.

- Ballots are checked & homomorphically combined into a single encrypted outcome.
- Outcome is decrypted by threshold of talliers.
- Proof of correct decryption.

Where are we?

Applied pi calculus and ProVerif

- The applied pi calculus is a language for describing concurrent processes and their interactions
  - Developed explicitly for modelling security protocols
  - Similar to spi calculus; with more general cryptography
- ProVerif is a leading software tool for automated reasoning
  - Takes applied pi processes and reasons about observational equivalence, correspondence assertions and secrecy

History of applied pi calculus and ProVerif

1970s:	Milner's *Calculus of Communicating Systems* (CCS)
1989:	Milner *et al.* extend CCS to *pi calculus*
1999:	Abadi & Gordon introduce *spi calculus*, variant of pi
2001:	Abadi & Fournet generalise spi to *applied pi calculus*
2000s:	Blanchet develops *ProVerif* to enable automated reasoning for applied pi calculus processes

Applied pi calculus: Grammar

Terms

$$
\begin{array}{lll}
L, M, N, T, U, V ::= & & \\
\quad a, b, c, k, m, n, s, t, r, \ldots & & \text{name} \\
\quad x, y, z & & \text{variable} \\
\quad g(M_1, \ldots, M_l) & & \text{function}
\end{array}
$$

Equational theory

Suppose we have defined nullary function `ok`, unary function `pk`, binary functions `enc`, `dec`, `senc`, `sdec`, `sign`, and ternary function `checksign`.

```
sdec( x, senc(x, y) )   =   y
dec( x, enc(pk(x), y) )   =   y
checksign( pk(x), y, sign(x, y) )   =   ok
```

Applied pi calculus: Grammar

Processes

$P, Q, R ::=$	processes	$A, B, C ::=$	extended processes
0	null process	P	plain process
$P \mid Q$	parallel comp.	$A \mid B$	parallel comp.
$!P$	replication	$\nu\, n.A$	name restriction
$\nu\, n.P$	name restriction	$\nu\, x.A$	variable restriction
$u(x).P$	message input	$\{^{M}/_{x}\}$	active substitution
$\overline{u}\langle M\rangle.P$	message output		
$\text{if } M = N \text{ then } P \text{ else } Q$	cond'nl		

Example

$$\nu\, k.(\overline{c}\langle \mathsf{senc}(k, a)\rangle.\, \overline{c}\langle \mathsf{senc}(k, b)\rangle \mid \{^{\mathsf{h}(k)}/_{x}\})$$

Modelling Helios 2.0: equational theory

$$\mathsf{dec}(x_{\mathsf{sk}}, \mathsf{penc}(\mathsf{pk}(x_{\mathsf{sk}}), x_{\mathsf{rand}}, x_{\mathsf{text}})) = x_{\mathsf{text}}$$

$$\mathsf{dec}(\mathsf{decKey}(x_{\mathsf{sk}}, \mathit{ciph}), \mathit{ciph}) = x_{\mathsf{plain}}$$
$$\text{where } \mathit{ciph} = \mathsf{penc}(\mathsf{pk}(x_{\mathsf{sk}}), x_{\mathsf{rand}}, x_{\mathsf{plain}})$$

$$\mathsf{penc}(x_{\mathsf{pk}}, y_{\mathsf{rand}}, y_{\mathsf{text}}) * \mathsf{penc}(x_{\mathsf{pk}}, z_{\mathsf{rand}}, z_{\mathsf{text}}) = \mathsf{penc}(x_{\mathsf{pk}}, y_{\mathsf{rand}} \circ z_{\mathsf{rand}}, y_{\mathsf{text}} + z_{\mathsf{text}})$$

$$\mathsf{checkBallotPf}(x_{\mathsf{pk}}, \mathit{ballot}, \mathsf{ballotPf}(x_{\mathsf{pk}}, x_{\mathsf{rand}}, s, \mathit{ballot})) = \mathsf{true}$$
$$\text{where } \mathit{ballot} = \mathsf{penc}(x_{\mathsf{pk}}, x_{\mathsf{rand}}, s)$$

$$\mathsf{checkDecKeyPf}(\mathsf{pk}(x_{\mathsf{sk}}), \mathit{ciph}, \mathit{dk}, \mathsf{decKeyPf}(x_{\mathsf{sk}}, \mathit{ciph}, \mathit{dk})) = \mathsf{true}$$
$$\text{where } \mathit{ciph} = \mathsf{penc}(\mathsf{pk}(x_{\mathsf{sk}}), x_{\mathsf{rand}}, x_{\mathsf{plain}})$$
$$\text{and } \mathit{dk} = \mathsf{decKey}(x_{\mathsf{sk}}, \mathit{ciph})$$

Modelling trusted & untrusted components

Depending on the property to be analysed, some components are required to be trusted & some not.

If it is	Then it is
Required to be trusted	Coded as a process
Auditable	Coded as a process
Not required to be trusted	Not coded as a process. The attacker can do what it wants.

Modelling Helios 2.0: processes

Definition

A *voting process specification* is a tuple $\langle V, A \rangle$ where V is a plain process without replication and A is a closed evaluation context such that $\mathrm{fv}(V) = \{v\}$ and $\mathrm{rv}(V) = \emptyset$.

Given a voting process specification $\langle V, A \rangle$, integer $n \in \mathbb{N}$, and names $s_1, \ldots, s_n$ we can build the voting process

$$\mathrm{VP}_n(s_1, \ldots, s_n) = A[V_1 \mid \cdots \mid V_n]$$

where $V_i = V\{^{s_i}/_{v}\}$. Intuitively, $\mathrm{VP}_n(s_1, \ldots, s_n)$ models the protocol with n voters casting votes for candidates $s_1, \ldots, s_n$.

Definition

The voting process specification $\langle V_{\mathsf{helios}}, A_{\mathsf{helios}} \rangle$ is defined where

$$\begin{aligned}
V_{\mathsf{helios}} &\triangleq d(x_{\mathsf{pid}}).\ \overline{d}\langle v \rangle.\ d(x_{\mathsf{ballot}}).\ d(x_{\mathsf{ballotpf}}).\overline{c}\langle (\!| w, x_{\mathsf{ballot}}, x_{\mathsf{ballotpf}} |\!) \rangle \\
A_{\mathsf{helios}}[_] &\triangleq \nu sk, d.\ \left(\overline{c}\langle \mathsf{pk}(sk) \rangle \mid (!\nu pid.\ \overline{d}\langle pid \rangle) \mid (!B) \mid T \mid _\ \right) \\
B &\triangleq \nu m.\ d(x_{\mathsf{vote}}).\overline{d}\langle \mathsf{penc}(\mathsf{pk}(sk), m, x_{\mathsf{vote}}) \rangle. \\
&\qquad \overline{d}\langle \mathsf{ballotPf}(\mathsf{pk}(sk), m, x_{\mathsf{vote}}, \mathsf{penc}(\mathsf{pk}(sk), m, x_{\mathsf{vote}})) \rangle \\
T &\triangleq c(x_{\mathsf{tally}}).\ \overline{c}\langle (\!| \mathsf{decKey}(sk, x_{\mathsf{tally}}), \mathsf{decKeyPf}(sk, x_{\mathsf{tally}}, \mathsf{decKey}(sk, x_{\mathsf{tally}})) |\!) \rangle
\end{aligned}$$

Where are we?

Election verifiability

Individual verifiability

A voter can check her own vote is included in the ballot collection.

Universal verifiability

Anyone can check that the declared outcome corresponds to the ballot collection.

Eligibility verifiability

Anyone can check that only eligible votes are included in the ballot collection.

Remark

- Verifiability $\neq$ correctness

Individual and universal verifiability

Individual test

We require a test

$$\Phi^{IV}(\text{my\_vote}, \text{my\_data}, \text{bb\_entry})$$

that a voter can apply after the election.
The test succeeds iff the bulletin board entry corresponds to the voter's vote and data.

Universal test

We require a test

$$\Phi^{UV}(\text{decl\_outcome}, \text{bb\_entries}, \text{pf})$$

that an observer can apply after the election.
The test succeeds iff the declared outcome is correct w.r.t. the bb entries and the proof.

Individual and universal verifiability

$\Phi^{IV}(\text{my\_vote}, \text{my\_data}, \text{bb\_entry})$ $\quad$ $\Phi^{UV}(\text{decl\_outcome}, \text{bb\_entries}, \text{pf})$

Acceptability conditions for Φ^{IV} and Φ^{UV}

Soundness, For all BBs σ:

$$\forall i, j. \quad \Phi^{IV}(v_i, r_i, y) \wedge \Phi^{IV}(v_j, r_j, y) \Rightarrow i = j \tag{1}$$

$$\Phi^{UV}(\tilde{v}, \tilde{y}, p) \wedge \Phi^{UV}(\tilde{v}', \tilde{y}, p) \Rightarrow \tilde{v} \simeq \tilde{v}' \tag{2}$$

$$\bigwedge_{1 \leq i \leq n} \Phi^{IV}(v_i, r_i, y_i) \wedge \Phi^{UV}(\tilde{v}', \tilde{y}, p) \Rightarrow \tilde{v} \simeq \tilde{v}' \tag{3}$$

Effectiveness There exists a BB σ s.t.

$$\bigwedge_{1 \leq i \leq n} \Phi_i^{IV}(v_i, r_i, y_i)\sigma \wedge \Phi^{UV}(\tilde{v}, \tilde{y}, p) \tag{4}$$

Helios 2.0: Verifiability

The untrusted server is assumed to publish the election data. When the protocol is executed as expected the resulting frame should have substitution σ such that

$$
\begin{aligned}
x_{pk}\sigma &= \mathsf{pk}(sk) \\
y_i\sigma &= (\!|\, pid_i, \mathsf{penc}(\mathsf{pk}(sk), m_i, v_i), \\
&\qquad \mathsf{ballotPf}(\mathsf{pk}(sk), m_i, v_i, \mathsf{penc}(\mathsf{pk}(sk), m_i, v_i)) \,|\!) \\
z_{\mathsf{tally}}\sigma &= \pi_2(y_1) * \cdots * \pi_2(y_n)\sigma \\
z_{\mathsf{decKey}}\sigma &= \mathsf{decKey}(sk, z_{\mathsf{tally}})\sigma \\
z_{\mathsf{decKeyPf}}\sigma &= \mathsf{decKeyPf}(sk, z_{\mathsf{tally}}, z_{\mathsf{decKey}})\sigma
\end{aligned}
$$

$$
\begin{aligned}
\Phi^{IV} &\mathrel{\widehat{=}} y =_E (\!|\, r_{pid}, r_{ballot}, r_{ballotpf} \,|\!) \\
\Phi^{UV} &\mathrel{\widehat{=}} z_{\mathsf{tally}} =_E \pi_2(y_1) * \cdots * \pi_2(y_n) \\
&\quad \wedge \textstyle\bigwedge_{i=1}^{n} (\mathsf{checkBallotPf}(x_{\mathsf{pk}}, \pi_2(y_i), \pi_3(y_i)) =_E \mathsf{true}) \\
&\quad \wedge \mathsf{checkDecKeyPf}(x_{\mathsf{pk}}, z_{\mathsf{tally}}, z_{\mathsf{decKey}}, z_{\mathsf{decKeyPf}}) =_E \mathsf{true} \\
&\quad \wedge v_1 + \cdots + v_n =_E \mathsf{dec}(z_{\mathsf{decKey}}, z_{\mathsf{tally}})
\end{aligned}
$$

Election verifiability

Individual verifiability

A voter can check her own vote is included in the ballot collection.

Universal verifiability

Anyone can check that the declared outcome corresponds to the ballot collection.

Eligibility verifiability

Anyone can check that only eligible votes are included in the ballot collection.

Remark

- Verifiability $\neq$ correctness

Individual and universal verifiability

Individual test

We require a test

$$\Phi^{IV}(\text{my\_vote}, \text{my\_data}, \text{bb\_entry})$$

that a voter can apply after the election.
The test succeeds iff the bulletin board entry corresponds to the voter's vote and data.

Universal test

We require a test

$$\Phi^{UV}(\text{decl\_outcome}, \text{bb\_entries}, \text{pf})$$

that an observer can apply after the election.
The test succeeds iff the declared outcome is correct w.r.t. the bb entries and the proof.

Eligibility test

We require a test

$$\Phi^{EV}(\text{pub\_creds}, \text{bb\_entries}, \text{pf})$$

that an observer can apply after the election. The test succeeds iff each the votes bb_entries was created by an owner of a pub_cred.

Individual, universal and eligibility verifiability

$$\Phi^{IV}(v, w, r, y) \qquad \Phi^{UV}(\tilde{v}, \tilde{y}, p) \qquad \Phi^{EV}(\tilde{w}, \tilde{y}, p)$$

Acceptability conditions for Φ^{IV}, Φ^{UV} and Φ^{EV}

(1), (2), (3), (4), and:

$$\Phi^{EV}(\tilde{w}, \tilde{y}, p) \wedge \Phi^{EV}(\tilde{w}', \tilde{y}, p) \Rightarrow \tilde{w} \simeq \tilde{w}' \tag{5}$$

$$\bigwedge_{1 \leq i \leq n} \Phi^{IV}(v_i, r_i, y_i) \wedge \Phi^{EV}(\tilde{w}', \tilde{y}, p) \Rightarrow \tilde{w} \simeq \tilde{w}' \tag{6}$$

JCJ-Civitas

- Voter prepares encrypted ballot on a trusted computer, and submits it.
  - Similar to Helios 2.0, except that cut-and-choose auditability of ballot isn't necessary.
- Ballots are submitted to a re-encryption mixnet that is able to prove it correctly mixed the ballots.
- Ballots are threshold decrypted with proof of correct decryption, and counted.

JCJ-Civitas: Eligibility verifiability

- Voters construct a secret credential through interaction with multiple registrars.
- A public part of the credential is published on the electoral register, for public scrutiny.
- Voters submit their vote with a differently randomised public part of the credential.

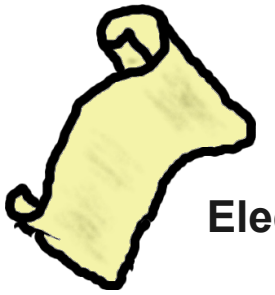
- This means that any observer can verify that all the votes cast are cast by eligible voters.

JCJ-Civitas

Voter with credential d

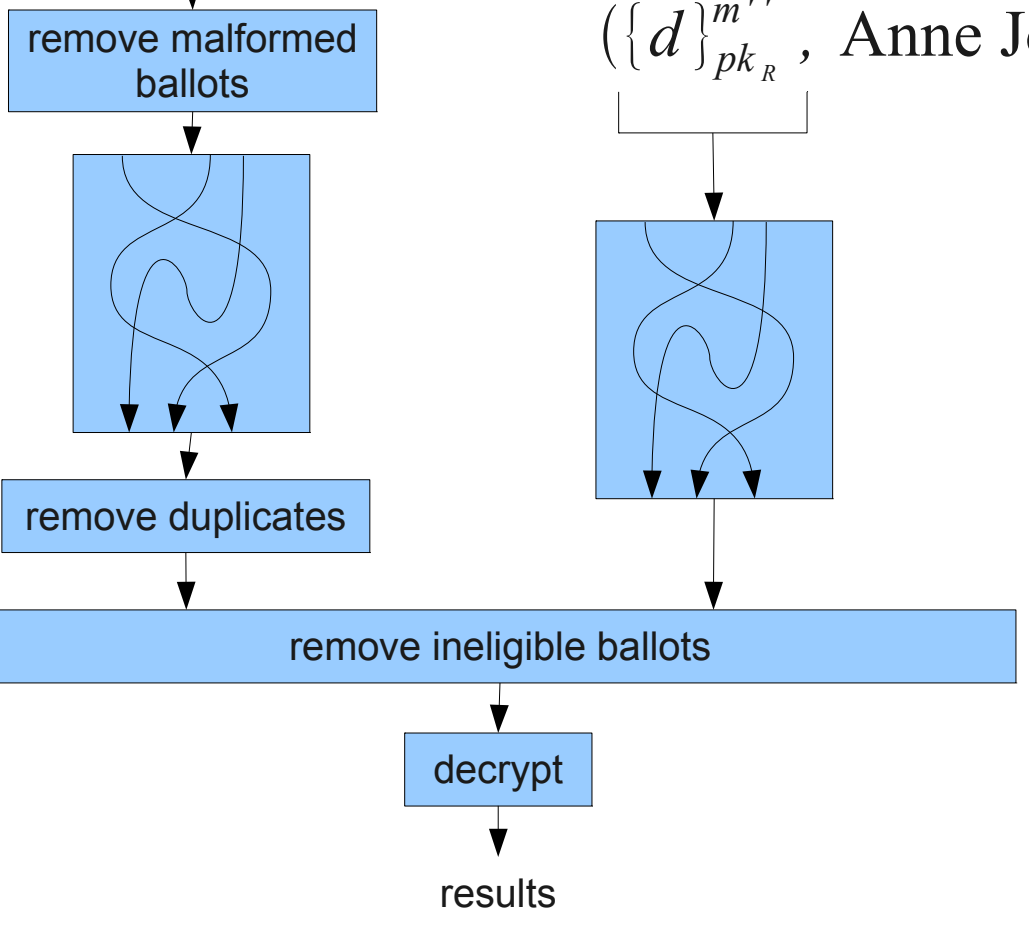
Ballot

$$(\{v\}^{m}_{pk_T}, \{d\}^{m'}, zkp)$$



Electoral register

$$(\{d\}^{m''}_{pk_R}, \text{Anne Jones})$$



JCJ-Civitas: equational theory

$$\mathsf{checkBallot}(\mathsf{ballotPf}(x_{\mathsf{pk}}, x_{\mathsf{rand}}, x_{\mathsf{text}}, x'_{\mathsf{pk}}, x'_{\mathsf{rand}}, x'_{\mathsf{text}}), \mathsf{penc}(x_{\mathsf{pk}}, x_{\mathsf{rand}}, x_{\mathsf{text}}), \mathsf{penc}(x'_{\mathsf{pk}}, x'_{\mathsf{rand}}, x'_{\mathsf{text}})) = \mathsf{true}$$

$\mathsf{pet}(\mathsf{petPf}(x_{\mathsf{sk}}, \mathit{ciph}, \mathit{ciph}'), \mathit{ciph}, \mathit{ciph}') = \mathsf{true}$ where
$\mathit{ciph} \hat{=} \mathsf{penc}(\mathsf{pk}(x_{\mathsf{sk}}), x_{\mathsf{rand}}, x_{\mathsf{text}})$ and $\mathit{ciph}' \hat{=} \mathsf{penc}(\mathsf{pk}(x_{\mathsf{sk}}), x'_{\mathsf{rand}}, x_{\mathsf{text}})$.

$$\mathsf{renc}(y_{\mathsf{rand}}, \mathsf{penc}(\mathsf{pk}(x_{\mathsf{sk}}), x_{\mathsf{rand}}, x_{\mathsf{text}})) = \mathsf{penc}(\mathsf{pk}(x_{\mathsf{sk}}), f(x_{\mathsf{rand}}, y_{\mathsf{rand}}), x_{\mathsf{text}}).$$

For each permutation χ on $\{1, \ldots, n\}$:

$$\mathsf{checkMix}(\mathsf{pfMix}(x_{\mathsf{ciph},1}, \ldots, x_{\mathsf{ciph},n}, \mathit{ciph}_1, \ldots, \mathit{ciph}_n, z_{\mathsf{rand},1}, \ldots, z_{\mathsf{rand},n}), x_{\mathsf{ciph},1}, \ldots, x_{\mathsf{ciph},n}, \mathit{ciph}_1, \ldots, \mathit{ciph}_n) = \mathsf{true}$$

where $\mathit{ciph}_i \hat{=} \mathsf{renc}(z_{\mathsf{rand},i}, x_{\mathsf{ciph},\chi(i)})$.

JCJ-Civitas: model

Definition

The voting process specification $A_{\text{jcj}}, V_{\text{jcj}}$ is defined where:

$$
\begin{array}{rcl}
A_{\text{jcj}} & \hat{=} & \nu\, a, ssk_R.(!R \mid \{\text{pk}(sk_R)/_{x_{\text{pk}_\text{R}}}, \text{pk}(ssk_R)/_{x_{spk_R}}, \text{pk}(sk_T)/_{x_{\text{pk}_\text{T}}}\} \mid \_) \\
V_{\text{jcj}} & \hat{=} & \nu\, m, m'.a(x_{cred}). \\
 & & \text{let } ciph = \text{penc}(x_{pk_T}, m, v) \text{ in} \\
 & & \text{let } ciph' = \text{penc}(x_{pk_R}, m', \pi_1(x_{cred})) \text{ in} \\
 & & \text{let } zkp = \text{ballotPf}(x_{pk_T}, m, v, x_{pk_R}, m', \pi_1(x_{cred})) \text{ in} \\
 & & \overline{c}\langle(\!| ciph, ciph', zkp |\!)\rangle \\
R & \hat{=} & \nu\, d, m''.\, \text{let } sig = \text{sign}(ssk_R, \text{penc}(x_{pk_R}, m'', d)) \text{ in } \overline{a}\langle(\!| d, sig |\!)\rangle.\overline{c}\langle sig\rangle
\end{array}
$$

JCJ-Civitas: verifiability

$$
\begin{array}{lcl}
w_i\sigma & = & \mathsf{sign}(ssk_R, c_i'') \\
x_{pk_R}\sigma & = & \mathsf{pk}(sk_R) \\
x_{spk_R}\sigma & = & \mathsf{pk}(ssk_R) \\
x_{pk_T}\sigma & = & \mathsf{pk}(sk_T) \\
y_i\sigma & = & (\!|c_i, c_i', \mathsf{ballotPf}(\mathsf{pk}(sk_T), m_i, s_i, \mathsf{pk}(sk_R), m_i', d_i)|\!) \\
z_{\mathsf{bal},i}\sigma & = & (\!|\mathsf{renc}(\hat{m}_i, c_{\chi(i)}), \mathsf{renc}(\hat{m}_i', c'_{\chi(i)})|\!) \\
z_{\mathsf{pfMixPair}}\sigma & = & \mathsf{pfMixPair}((\!|c_1, c_1'|\!), \ldots, (\!|c_n, c_n'|\!), (\!|\mathsf{renc}(\hat{m}_1, c_{\chi(1)}), \mathsf{renc}(\hat{m}_1', c'_{\chi(1)})|\!), \\
 & & \ldots, (\!|\mathsf{renc}(\hat{m}_n, c_{\chi(n)}), \mathsf{renc}(\hat{m}_n', c'_{\chi(n)})|\!), (\!|\hat{m}_1, \hat{m}_1'|\!), \ldots, (\!|\hat{m}_n, \hat{m}_n'|\!)) \\
z_{\mathsf{decKey},i}\sigma & = & \mathsf{decKey}(sk_T, \mathsf{renc}(\hat{m}_i, c_{\chi(i)})) \\
z_{\mathsf{decPf},i}\sigma & = & \mathsf{decKeyPf}(sk_T, \mathsf{renc}(\hat{m}_i, c_{\chi(i)}), \mathsf{decKey}(sk_T, \mathsf{renc}(\hat{m}_i, c_{\chi(i)}))) \\
z_{\mathsf{cred},i}\sigma & = & \mathsf{renc}(\hat{m}_i'', c''_{\chi'(i)}) \\
\hat{z}_{\mathsf{cred},i}\sigma & = & \mathsf{renc}(\hat{m}''_{\chi(\chi'^{-1}(i))}, c''_{\chi(i)}) \\
z_{\mathsf{credMixPf}}\sigma & = & \mathsf{pfMix}(c_1'', \ldots, c_n'', \mathsf{renc}(\hat{m}_1'', c''_{\chi'(1)}), \ldots, \mathsf{renc}(\hat{m}_n'', c''_{\chi'(n)}), \hat{m}_1'', \ldots, \hat{m}_n'') \\
z_{\mathsf{petPf},i}\sigma & = & \mathsf{petPf}(sk_R, \mathsf{renc}(\hat{m}_i', c'_{\chi(i)}), \mathsf{renc}(\hat{m}''_{\chi(\chi'^{-1}(i))}, c''_{\chi(i)}))
\end{array}
$$

where $c_i \hat{=} \mathsf{penc}(\mathsf{pk}(sk_T), m, s_i)$, $c_i' \hat{=} \mathsf{penc}(\mathsf{pk}(sk_R), m', d_i)$, $c_i'' \hat{=} \mathsf{penc}(\mathsf{pk}(sk_R), m'', d_i)$ and χ, χ' are permutations on $\{1, \ldots, n\}$.

JCJ-Civitas: verifiability

$$
\begin{aligned}
\Phi^{IV} \mathrel{\widehat{=}}\ & y =_E (\!|\mathsf{penc}(x_{\mathsf{pk_T}}, r_m, v), \mathsf{penc}(x_{\mathsf{pk_R}}, r_{m'}, \pi_1(r_{cred})),\\
& \qquad \mathsf{ballotPf}(x_{\mathsf{pk_T}}, r_m, v, x_{\mathsf{pk_R}}, r_{m'}, \pi_1(r_{cred}))|\!) \wedge w = \pi_2(r_{cred})\\
\Phi^{UV} \mathrel{\widehat{=}}\ & \mathsf{checkMixPair}(z_{\mathsf{pfMixPair}}, (\!|\pi_1(y_1), \pi_2(y_1)|\!), \ldots, (\!|\pi_1(y_n), \pi_2(y_n)|\!),\\
& \qquad z_{\mathsf{bal},1}, \ldots, z_{\mathsf{bal},n}) =_E \mathsf{true}\\
& \wedge \textstyle\bigwedge_{i=1}^{n} \mathsf{dec}(z_{\mathsf{decKey},i}, \pi_1(z_{\mathsf{bal},i})) =_E v_i\\
& \wedge \textstyle\bigwedge_{i=1}^{n} \mathsf{checkDecKeyPf}(x_{\mathsf{pk_T}}, \pi_1(z_{\mathsf{bal},i}), z_{\mathsf{decKey},i}, z_{\mathsf{decPf},i}) =_E \mathsf{true}\\
\Phi^{EV} \mathrel{\widehat{=}}\ & \textstyle\bigwedge_{i=1}^{n} \mathsf{checkBallot}(\pi_3(y_i), \pi_1(y_i), \pi_2(y_i))\\
& \wedge \mathsf{checkMixPair}(z_{\mathsf{pfMixPair}}, (\!|\pi_1(y_1), \pi_2(y_1)|\!), \ldots, (\!|\pi_1(y_n), \pi_2(y_n)|\!),\\
& \qquad z_{\mathsf{bal},1}, \ldots, z_{\mathsf{bal},n}) =_E \mathsf{true}\\
& \wedge \textstyle\bigwedge_{i=1}^{n} \mathsf{pet}(z_{\mathsf{petPf},i}, \pi_2(z_{\mathsf{bal},i}), \hat{z}_{\mathsf{cred},i}) =_E \mathsf{true}\\
& \wedge (\!|z_{\mathsf{cred},1}, \ldots, z_{\mathsf{cred},n}|\!) \simeq (\!|\hat{z}_{\mathsf{cred},1}, \ldots, \hat{z}_{\mathsf{cred},n}|\!)\\
& \wedge \mathsf{checkMix}(z_{\mathsf{credMixPf}}, \mathsf{getmsg}(w_1), \ldots, \mathsf{getmsg}(w_n), z_{\mathsf{cred},1}, \ldots, z_{\mathsf{cred},n}) =_E \mathsf{true}\\
& \wedge \textstyle\bigwedge_{i=1}^{n} \mathsf{checksign}(x_{spk_R}, w_i)
\end{aligned}
$$

XXX: CHECK mixPr and mixPairPf

Where are we?

Formalisation of vote-privacy

Classically modeled as observational equivalences between two slightly different processes P_1 and P_2, but

- changing the identity does not work, as identities are revealed
- changing the vote does not work, as the votes are revealed at the end

$\hookrightarrow$ consider two honest voters and swap their votes

Definition (Privacy)

A voting protocol respects privacy if

$$S[V_A\{^a/_v\} \mid V_B\{^b/_v\}] \approx_\ell S[V_A\{^b/_v\} \mid V_B\{^a/_v\}].$$

Receipt-freeness: leaking secrets to the coercer

To model receipt-freeness we need to specify that a coerced voter cooperates with the coercer by leaking secrets on a channel ch

$P ::=$
0
$P \mid P$
$\nu n.P$
$u(x).P$
$\overline{u}\langle M\rangle.P$
if $M = N$ then P else P
$!P$
$\ldots$

P^{ch} in terms of P

- $0^{ch} = 0$
- $(P \mid Q)^{ch} = P^{ch} \mid Q^{ch}$
- $(\nu n.P)^{ch} = \nu n.\overline{ch}\langle n\rangle.P^{ch}$
- $(u(x).P)^{ch} = u(x).\overline{ch}\langle x\rangle.P^{ch}$
- $(\overline{u}\langle M\rangle.P)^{ch} = \overline{u}\langle M\rangle.P^{ch}$
- $\ldots$

We denote by $P^{\backslash out(chc,\cdot)}$ the process $\nu chc.(P \mid !chc(x))$.

Lemma: $(P^{ch})^{\backslash out(chc,\cdot)} \approx_{\ell} P$

Receipt-freeness: definition

Intuition

There exists a process V' which

- votes a,
- leaks (possibly fake) secrets to the coercer,
- and makes the coercer believe she voted c

Definition (Receipt-freeness)

A voting protocol is receipt-free if there exists a process V', satisfying

- $V'^{\backslash out(chc,\cdot)} \approx_\ell V_A\{^a/_v\}$,
- $S[V_A\{^c/_v\}^{chc} \mid V_B\{^a/_v\}] \approx_\ell S[V' \mid V_B\{^c/_v\}]$.

Case study: Lee *et al.* protocol
We prove receipt-freeness by

- exhibiting V'
- showing that $V'^{\backslash out(chc,\cdot)} \approx_\ell V_A\{^a/_v\}$
- showing that $S[V_A\{^c/_v\}^{chc} \mid V_B\{^a/_v\}] \approx_\ell S[V' \mid V_B\{^c/_v\}]$

Coercion resistance: talking with the coercer

Like receipt-freness, but: voter interacts with the coercer during the protocol (instead of just supplying data at the end).

- The *voting booth* makes coercion resistance possible.

Interactively communicating with the coercer:

P^{c_1,c_2} in terms of P

- $0^{c_1,c_2} = 0$,
- $(P \mid Q)^{c_1,c_2} = P^{c_1,c_2} \mid Q^{c_1,c_2}$
- $(\nu n.P)^{c_1,c_2} = \nu n.\overline{c_1}\langle n\rangle.P^{c_1,c_2}$
- $(u(x).P)^{c_1,c_2} = u(x).\overline{c_1}\langle x\rangle.P^{c_1,c_2}$
- $(\overline{u}\langle M\rangle.P)^{c_1,c_2} = c_2(x).\overline{u}\langle x\rangle.P^{c_1,c_2}$
- $(!P)^{c_1,c_2} = !P^{c_1,c_2}$,
- (if $M = N$ then P else $Q)^{c_1,c_2} = c_2(x).$ if $x =$ true then P^{c_1,c_2} else Q^{c_1,c_2}

Coercion resistance: definition

Definition (Coercion resistance)

VP is coercion resistant if there exists a process V' such that for any $C = \nu c_1.\nu c_2.(\_ \mid P)$ satisfying

- $\tilde{n} \cap fn(C) = \emptyset$
- $S[C[V_A\{^?/_v\}^{c1,c2}] \mid V_B\{^a/_v\}] \approx_\ell S[V_A\{^c/_v\}^{chc} \mid V_B\{^a/_v\}]$

we have

- $C[V']^{\backslash out(chc,\cdot)} \approx_\ell V_A\{^a/_v\}$,
- $S[C[V_A\{^?/_v\}^{c_1,c_2}] \mid V_B\{^a/_v\}] \approx_\ell S[C[V'] \mid V_B\{^c/_v\}]$.

Intuitively, C together with the environment represent the coercer. The definition says there's a strategy V' for the voter such that

- if the coercer is trying to force A to vote c

then

- A can do V', which will result in an a vote, but will satisfy the coercer.

Doesn't take account of *fault attacks* (cf. Küsters/Truderung).

Privacy properties

Proposition

Let VP be a voting protocol. Then

$$VP \text{ is coercion-resistant}$$
$$\Downarrow$$
$$VP \text{ is receipt-free}$$
$$\Downarrow$$
$$VP \text{ respects privacy}$$

Where are we?

Conclusions

Electronic voting

- Ongoing issues
  - securability
  - usability
  - adoptability
- Comparison with digital cash

Process calculus analysis

- Powerful
- Appropriate
- Abstraction level
  - separation of concerns
  - may miss attacks

Trustworthy Voting Systems

- EPSRC project 2009-2013
- Birmingham, Surrey, Newcastle/Luxembourg
- Opt2Vote, Electoral Reform Services
- Ministry of Justice

[This slide has been intentionally left blank.]

[This slide has been intentionally left blank.]

[This slide has been intentionally left blank.]

[This slide has been intentionally left blank.]

[This slide has been intentionally left blank.]

[This slide has been intentionally left blank.]

[This slide has been intentionally left blank.]

[This slide has been intentionally left blank.]

[This slide has been intentionally left blank.]

[This slide has been intentionally left blank.]