Election verifiability in electronic voting protocols*

Steve Kremer[1], Mark Ryan[2], and Ben Smyth[2,3]

[1]LSV, ENS Cachan & CNRS & INRIA, France
[2]School of Computer Science, University of Birmingham, UK
[3]École Normale Supérieure, CNRS, INRIA, Paris, France

Technical Report CSR-10-06

April 9, 2010
(Revised: June 28, 2010)

Abstract

We present a symbolic definition of election verifiability for electronic voting protocols in the context of the applied pi calculus. Our definition is given in terms of boolean tests which can be performed on the data produced by an election. The definition distinguishes three aspects of verifiability, which we call individual verifiability, universal verifiability, and eligibility verifiability. It also allows us to determine precisely which aspects of the system's hardware and software must be trusted for the purpose of election verifiability. In contrast with earlier work our definition is compatible with a large class of electronic voting schemes, including those based on blind signatures, homomorphic encryption and mixnets. We demonstrate the applicability of our formalism by analysing two protocols which have been deployed; namely Helios 2.0, which is based on homomorphic encryption, and Civitas, which uses mixnets. In addition we consider the FOO protocol which is based on blind signatures.

1 Introduction

Electronic voting systems are being introduced, or trialled, in several countries to provide more efficient voting procedures with an increased level of security.

*This work has been partly supported by the EPSRC projects *UbiVal* (EP/D076625/2), *Trustworthy Voting Systems* (EP/G02684X/1) and *Verifying Interoperability Requirements in Pervasive Systems* (EP/F033540/1); the *ANR SeSur AVOTÉ* project; and the *Direction Générale pour l'Armement* (DGA).

However, the security of electronic elections has been seriously questioned [9, 19, 8, 24]. A major difference with traditional paper based elections is the lack of transparency. In paper elections it is often possible to observe the whole process from ballot casting to tallying, and to rely on robustness characteristics of the physical world (such as the impossibility of altering the markings on a paper ballot sealed inside a locked ballot box). By comparison, it is not possible to observe the electronic operations performed on data. Moreover, computer systems may alter voting records in a way that cannot be detected by either voters or election observers. For example, a voting terminal's software might be infected by malware which could change the vote entered by the user, or even execute a completely different protocol than the one expected. The situation can be described as *voting on Satan's computer*, analogously with [5]. Computer systems and election administrators should therefore be considered to be part of the adversary model.

The concept of *election verifiability* that has emerged in the academic literature, for example, [17, 18, 10, 3], aims to address this problem. It should allow voters and election observers to verify independently that votes have been recorded, tallied and declared correctly. To emphasise a voter's ability to verify the results of the entire election process, it is sometimes called *end-to-end* verifiability [20, 2]. The verification is performed using hardware and software of the verifier's own choice, and is completely independent of the hardware and software running the election. One generally distinguishes two aspects of verifiability.

- *Individual verifiability:* a voter can check that her own ballot is included in the bulletin board.
- *Universal verifiability:* anyone can check that the election outcome corresponds to the ballots published on the bulletin board.

We identify another aspect of verifiability which is sometimes included in universal verifiability.

- *Eligibility verifiability:* anyone can check that each vote in the election outcome was cast by a registered voter and there is at most one vote per voter.

We explicitly distinguish eligibility verifiability as a distinct property for compatibility with a larger class of protocols.

In this paper we present a symbolic definition of election verifiability for electronic voting protocols which captures the three desirable aspects. We model voting protocols in the applied pi calculus and formalise the different aspects of verifiability as a triple of boolean tests $\Phi^{IV}, \Phi^{UV}, \Phi^{EV}$. The test Φ^{IV} is intended to be checked by the individual voter who instantiates the test with her private information (for example, her vote and data derived during the execution of the protocol) and the public information available on the bulletin board. The tests Φ^{UV} and Φ^{EV} can be checked by any external observer and only rely on public information, that is, the contents of the bulletin board which may include, for example, the set of ballots cast by voters, the list of eligible voters and the declared outcome. Our definition requires that these tests satisfy several conditions on all possible executions of the protocol. The consideration of eligibility verifiability is particularly interesting because it is essential to provide an assurance that the election outcome corresponds to votes legitimately cast and hence provides a mechanism to detect ballot stuffing.

A further interesting aspect of our work is the clear identification of which parts of the voting system need to be trusted to achieve verifiability. As already discussed it is not reasonable to assume voting systems behave correctly. Accordingly, when modelling a voting protocol as a process, we only model the parts of the protocol that we need to trust for the purpose of verifiability; all the remaining parts of the system will be controlled by the adversarial environment. Ideally, such a process would only model the interaction between a *voter* and the voting terminal; *that is, the messages input by the voter.* In particular, the voter should not need to trust the election hardware or software. However, achieving absolute verifiability in this context is difficult and we sometimes need to make explicit trust assumptions about which parts of the voter and administrator processes need to be trusted. As an example, when showing that the protocol by Fujioka *et al.* [15] ensures individual and universal verifiability we model the protocol as $\nu r.\bar{c}\langle v\rangle.\bar{c}\langle r\rangle$: the voter needs to generate a fresh nonce r and then give her vote v and r to the voting terminal, which is part of the adversarial environment. When the protocol is executed correctly this nonce is used to compute a commitment to the vote. This can be checked by the tests that ensure verifiability. The fact that νr is part of the protocol model implies that the nonce needs to be fresh for verifiability to hold. Hence, in this example the voter either needs to have a means to provide a fresh nonce or trust some part of the process to generate it freshly. Such trust assumptions are motivated by the fact that parts of a protocol can be audited, or because they can be executed in a distributed manner amongst several different election officials. For example, in the Helios 2.0 voting protocol [3], ballot construction can be audited using a cast-or-audit mechanism. Since any third party software can be used to audit the ballots the voters are assured that the ballots cast were constructed according to the protocol specification with high probability. Whether these trust assumptions are reasonable depends on the context of the given election.

We also note that the tests Φ^{IV}, Φ^{UV} and Φ^{EV} are assumed to be verified in a trusted environment. Indeed, if a test is checked by malicious software that always evaluates the test to hold, it is not of great value. However, the verification of these tests, unlike the election, can be repeated sufficiently many times, on different machines and using different software, which could be provided by different stakeholders of the election. Another possibility to avoid this issue would be to have tests which are human-verifiable as discussed for instance in [2, Chapter 5].

We demonstrate the applicability of our definition with three case studies: the protocol by Fujioka, Okamoto and Ohta [15]; the Helios 2.0 protocol [4] which was effectively used in recent university elections in Belgium; and the protocol by Juels, Catalano and Jakobsson [18], which has been implemented by Clarkson, Chong and Myers as Civitas [13, 12]. Among other properties we show that the Helios protocol does not guarantee eligibility verifiability and is therefore vulnerable to ballot stuffing by dishonest administrators. As the protocol description does not mandate this property we do not claim this to be an attack, but simply clarify which aspects of verifiability are satisfied.

1.1 Our contribution

Our contribution is as follows:

1. A symbolic definition of election verifiability that considers a large class of protocols; including schemes based on: mixnets, homomorphic encryption and blind signatures. (In contrast, our preliminary work presented in [22] only considers blind signature schemes.)

2. Sound and intuitive consideration for eligibility verifiability. (A property which has been largely neglected and which our earlier work [22] provided only limited scope for.)

3. Formal treatment of trust assumptions for the purpose of verifiability.

In addition, the applicability of our work is demonstrated with respect to three case studies; namely, Helios 2.0, Civitas and FOO. The consideration of Helios 2.0 and Civitas is of particular interest since these systems have been implemented and deployed.

1.2 Related work

Juels *et al.* [17, 18] present a definition of universal verifiability in the provable security model. Their definition assumes voting protocols produce non-interactive zero-knowledge proofs of knowledge demonstrating the correctness of tallying. Here we consider definitions in a symbolic model. Universal verifiability was also studied by Chevallier-Mames *et al.* [11] with the aim of showing an incompatibility result: protocols cannot satisfy verifiability and vote privacy in an unconditional way (without relying on computational assumptions). But as witnessed by [17, 18], weaker versions of these properties can hold simultaneously. Our case studies demonstrate that our definition allows privacy and verifiability to coexist (see [14, 6] for a study of privacy properties in the applied pi calculus). Baskar *et al.* [7] and subsequently Talbi *et al.* [23] have formalised individual and universal verifiability with respect to the protocol by Fujioka *et al.* [15]. Their definitions are tightly coupled to that particular protocol and cannot easily be generalised. Moreover, their definitions characterise individual executions as verifiable or not; whereas such properties should be considered with respect to every execution (that is, the entire protocol).

In our earlier work [22] a preliminary definition of election verifiability was presented with support for automated reasoning. However, that definition is too strong to hold on protocols such as [18, 4]. In particular, our earlier definition was only illustrated on a simplified version of [18] which did not satisfy privacy because we omitted the mixnets. Hence, this is the first general, symbolic definbition which can be used to show verifiability for many important protocols, such as the ones studied in this paper.

2 Applied pi calculus

The applied pi calculus [1, 21] is a language for modelling concurrent, communicating processes. It is an extension of the pi calculus which was explicitly designed for modelling cryptographic protocols. For this purpose, the applied pi calculus allows processes to send terms constructed over a signature rather than just names. This term algebra can be used to model cryptographic primitives.

2.1 Syntax

The calculus assumes an infinite set of names $a, b, c, k, m, n, s, t, \ldots$, an infinite set of variables $v, x, y, z, \ldots$ and a finite signature Σ, that is, a finite set of function symbols each with an associated arity. A function symbol of arity 0 is a constant. We use metavariables u, w to range over both names and variables. Terms L, M, N, T, U, V are built by applying function symbols to names, variables and other terms. Tuples $u_1, \ldots, u_l$ and $M_1, \ldots, M_l$ are occasionally abbreviated $\tilde{u}$ and $\tilde{M}$. We write $\{^{M_1}/_{x_1}, \ldots, ^{M_l}/_{x_l}\}$ for substitutions that replace variables $x_1, \ldots, x_l$ with terms $M_1, \ldots, M_l$.

The applied pi calculus relies on a simple sort system. Terms can be of sort $\mathsf{Channel}$ for channel names or Base for the payload sent out on these channels. In addition we assume an infinite set of *record variables*. Function symbols can only be applied to, and return, terms of sort Base. A term is ground when it does not contain variables.

The grammar for processes is shown in Figure 1 where u is either a name or variable of channel sort. Plain processes are standard constructs, except for the *record message* $\mathsf{rec}(r, M).P$ construct which we discuss below. Extended processes introduce *active substitutions* which generalise the classical let construct: the process $\nu\, x.(\{^M/_x\} \mid P)$ corresponds exactly to the process let $x = M$ in P. As usual names and variables have scopes which are delimited by restrictions and by inputs. All substitutions are assumed to be cycle-free.

A *frame* φ is an extended process built from 0 and active substitutions $\{^M/_x\}$; which are composed by parallel composition and restriction. The *domain* of a frame φ is the set of variables that φ exports. Every extended process A can be mapped to a frame $\phi(A)$ by replacing every plain process in A with 0.

The record message construct $\mathsf{rec}(r, M).P$ introduces the possibility to enter special entries in frames. We suppose that the sort system ensures that r is a variable of record sort, which may only be used as a first argument of the rec construct or in the domain of the frame. Moreover, we make the global assumption that a record variable has a unique occurrence in each process. Intuitively, this construct will be used to allow a voter to privately record some

Figure 1 Applied pi calculus grammar

P, Q, R ::=	processes	A, B, C ::=	extended processes
0	null process	P	plain process
$P \mid Q$	parallel	$A \mid B$	parallel
$!P$	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		
$\mathsf{rec}(r, M).P$	record message		
if $M = N$ then P else Q	conditional		

information which she may later use to verify the election; for example, nonces constructed during an execution of the protocol and/or messages received as input.

The sets of free and bound names, respectively variables, in process A are denoted by $\mathrm{fn}(A)$, $\mathrm{bn}(A)$, $\mathrm{fv}(A)$, $\mathrm{bv}(A)$. We also write $\mathrm{fn}(M)$, $\mathrm{fv}(M)$ for the names, respectively variables, in term M. Similarly, we write $\mathrm{rv}(A)$ and $\mathrm{rv}(M)$ for the set of record variables in a process, respectively a term. An extended process A is *closed* if it has no free variables. A *context* $C[\_]$ is an extended process with a hole. We obtain $C[A]$ as the result of filling $C[\_]$'s hole with A. An *evaluation context* is a context whose hole is not under a replication, a conditional, an input, or an output.

The signature Σ is equipped with an equational theory E, that is, a finite set of equations of the form $M = N$. We define $=_E$ as the smallest equivalence relation on terms, that contains E and is closed under application of function symbols, substitution of terms for variables and bijective renaming of names.

Example 1. *Let* $\Sigma = \{\mathsf{pair}(\cdot,\cdot), \mathsf{fst}(\cdot), \mathsf{snd}(\cdot)\}$ *and* E *be defined over the equations*

$$\mathsf{fst}(\mathsf{pair}(x, y)) = x \qquad \mathsf{snd}(\mathsf{pair}(x, y)) = y$$

That is, the theory that models pairing and projection. Hence we have that $\mathsf{fst}(\mathsf{snd}(\mathsf{pair}(a, \mathsf{pair}(b, c)))) =_E b$.

In this paper we tacitly assume that all signatures and equational theories contain the function symbols $\mathsf{pair}(\cdot,\cdot), \mathsf{fst}(\cdot), \mathsf{snd}(\cdot)$ and equations for pairing as well as some constant $\bot$. As a convenient shortcut we then write $(\!| T_1, \ldots T_n |\!)$ for $\mathsf{pair}(T_1, \mathsf{pair}(\ldots, \mathsf{pair}(T_n, \bot)))$ and $\pi_i(T)$ for $\mathsf{fst}(\mathsf{snd}^{i-1}(T))$.

2.2 Semantics

We now define the operational semantics of the applied pi calculus by the means of three relations: structural equivalence, internal reductions and labelled reduction.

Structural equivalence ($\equiv$) is the smallest equivalence relation closed under α-conversion of both bound names and variables and application of evaluation contexts such that:

$$\begin{array}{llcl}
\text{PAR-0} & A \mid 0 & \equiv & A \\
\text{PAR-A} & A \mid (B \mid C) & \equiv & (A \mid B) \mid C \\
\text{PAR-C} & A \mid B & \equiv & B \mid A \\
\text{REPL} & !P & \equiv & P \mid !P \\
\\
\text{NEW-0} & \nu n.0 & \equiv & 0 \\
\text{NEW-C} & \nu u.\nu w.A & \equiv & \nu w.\nu u.A \\
\text{NEW-PAR} & A \mid \nu u.B & \equiv & \nu u.(A \mid B) \\
 & & & \text{if } u \notin \mathrm{fn}(A) \cup \mathrm{fv}(A) \\
\\
\text{ALIAS} & \nu x.\{^M/_x\} & \equiv & 0 \\
\text{SUBST} & \{^M/_x\} \mid A & \equiv & \{^M/_x\} \mid A\{^M/_x\} \\
\text{REWRITE} & \{^M/_x\} & \equiv & \{^N/_x\} \\
 & & & \text{if } M =_E N
\end{array}$$

Internal reduction ($\rightarrow$) is the smallest relation closed under structural equivalence, application of evaluation contexts and such that:

$$\begin{array}{ll}
\text{REC} & \mathsf{rec}(r, M).P \rightarrow P \mid \{^M/_r\} \\
\text{COMM} & \overline{c}\langle x\rangle.P \mid c(x).Q \rightarrow P \mid Q \\
\text{THEN} & \text{if } N = N \text{ then } P \text{ else } Q \rightarrow P \\
\text{ELSE} & \text{if } L = M \text{ then } P \text{ else } Q \rightarrow Q \\
 & \quad \text{for ground terms } L, M \text{ where } L \neq_E M
\end{array}$$

Labelled reduction ($\xrightarrow{\alpha}$) extends internal reduction and enables the environment to interact with the processes using the rules defined below. The label α is either an input, or the output of a channel name or a variable of base sort.

$$a(x).P \xrightarrow{a(M)} P\{^M/_x\} \qquad \mathrm{rv}(M) = \emptyset$$

$$\overline{a}\langle u\rangle.P \xrightarrow{\overline{a}\langle u\rangle} P$$

$$\frac{A \xrightarrow{\overline{a}\langle u\rangle} A' \qquad u \neq a}{\nu u.A \xrightarrow{\nu u.\overline{a}\langle u\rangle} A'}$$

$$\frac{A \xrightarrow{\alpha} A' \qquad u \text{ does not occur in } \alpha}{\nu u.A \xrightarrow{\alpha} \nu u.A'}$$

$$\frac{A \xrightarrow{\alpha} A' \quad bv(\alpha) \cap fv(B) = bn(\alpha) \cap fn(B) = \emptyset}{A \mid B \xrightarrow{\alpha} A' \mid B}$$

$$\frac{A \equiv B \quad B \xrightarrow{\alpha} B' \quad A' \equiv B'}{A \xrightarrow{\alpha} A'}$$

We write $\Longrightarrow$ for $(\rightarrow^* \xrightarrow{\alpha} \rightarrow^*)^*$, that is, the reflexive transitive closure of the labelled reduction. We will not discuss these semantics in detail but give an example illustrating them (Figure 2).

Figure 2 A sequence of reductions in the applied pi semantics

Let $P = \nu a, b.\mathsf{rec}(r, a).\overline{c}\langle (\!| a, b |\!) \rangle.c(x).\text{if } x = a \text{ then } \overline{c}\langle f(a) \rangle$. Then we have that

$$\begin{array}{lcl}
P & \rightarrow & \nu a, b.(\overline{c}\langle (\!| a, b |\!) \rangle.c(x).\text{if } x = a \text{ then } \overline{c}\langle f(a) \rangle \mid \{^{a}/_{r}\}) \\
 & \equiv & \nu a, b.(\nu y_1.(\overline{c}\langle y \rangle.c(x).\text{if } x = a \text{ then } \overline{c}\langle f(a) \rangle \mid \{^{(\!| a,b |\!)}/_{y_1}\}) \mid \{^{a}/_{r}\}) \\
 & \xrightarrow{\nu x.\overline{c}\langle x \rangle} & \nu a, b.(c(x).\text{if } x = a \text{ then } \overline{c}\langle f(a) \rangle \mid \{^{(\!| a,b |\!)}/_{y_1}\} \mid \{^{a}/_{r}\}) \\
 & \xrightarrow{\nu x.c(\pi_1(y))} & \nu a, b.(\text{if } a = a \text{ then } \overline{c}\langle f(a) \rangle \mid \{^{(\!| a,b |\!)}/_{y_1}\} \mid \{^{a}/_{r}\}) \\
 & \rightarrow & \nu a, b.(\overline{c}\langle f(a) \rangle \mid \{| \ \{^{(\!| a,b |\!)}/_{y_1}\} \mid \{^{a}/_{r}\}) \\
 & \xrightarrow{\nu y_2.\overline{c}\langle y_2 \rangle} & \nu a, b.(\text{if } a = a \text{ then } \overline{c}\langle f(a) \rangle \mid \{^{(\!| a,b |\!)}/_{y_1}\} \mid \{^{f(a)}/_{y_2}\} \mid \{^{a}/_{r}\}
\end{array}$$

Observe that each labelled output is done by reference and extends the domain of the process's frame.

3 Formalising voting protocols

As discussed in the introduction we want to explicitly specify the parts of the election protocol which need to be trusted (that is, those parts of the system for which no verifiable proof of correct behaviour is provided). Formally the trusted parts of the voting protocol can be captured using a voting process specification.

Definition 1 (Voting process specification)**.** *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 $fv(V) = \{v\}$ and $rv(V) = \emptyset$.*

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

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

where $V_i = V\{^{s_i}/_{v}\}$. Intuitively, $\mathsf{VP}_n(s_1, \ldots, s_n)$ models the protocol with n voters casting votes for candidates $s_1, \ldots, s_n$. Note that the votes $s_1, \ldots, s_n$ are not required to be distinct (several voters may cast votes for the same candidate).

Example 2. *Consider the following simple* raising hands *protocol. Every voter simply outputs her signed vote. We suppose that a trusted administrator first distributes keying material and outputs a list of signed public keys corresponding to the public credentials of eligible voters. Signatures are modeled by the equations*

$$\mathsf{checksign}(\mathsf{pk}(x), \mathsf{sign}(x, y)) = \mathsf{true} \qquad \mathsf{getmsg}(\mathsf{sign}(x, y)) = y$$

The administrator generating and distributing keys via a private channel d is modelled by the following context.

$$A \mathrel{\widehat{=}} \nu d.\nu skA.(!\nu skv.\overline{d}\langle skv \rangle.\overline{c}\langle \mathsf{sign}(skA, \mathsf{pk}(skv)) \rangle \mid \{^{\mathsf{pk}(skA)}/_{x_{pkA}}\} \mid \_)$$

The active substitution $\{^{\mathsf{pk}(skA)}/_{x_{pkA}}\}$ *models the fact that the administrator's public key is known, e.g. published on the election bulletin board. The voter, whom receives his private key and then outputs his signed vote is modelled by the process:*

$$V \mathrel{\hat{=}} d(x_{skv}).\overline{c}\langle (\!| \mathsf{pk}(x_{skv}), \mathsf{sign}(x_{skv}, v) |\!) \rangle$$

We will prove that this protocol trivially satisfies individual and universal verifiability in Section 4; and eligibility verifiability in Section 5.

For the purposes of individual verifiability the voter may be reliant on some data derived during the execution of the protocol. We must therefore keep track of all such values. Definition 2 achieves this objective using the record message construct.

Definition 2. *Let* rv *be an infinite list of distinct* record variables. *We define the function* R *on a finite process* P *without replication as* $\mathsf{R}(P) = \mathsf{R}_{\mathsf{rv}}(P)$ *and, for all lists* rv*:*

$$\begin{array}{lcl}
\mathsf{R}_{rv}(0) & \hat{=} & 0 \\
\mathsf{R}_{rv}(P \mid Q) & \hat{=} & \mathsf{R}_{odd(rv)}(P) \mid \mathsf{R}_{even(rv)}(Q) \\
\mathsf{R}_{rv}(\nu\, n.P) & \hat{=} & \nu\, n.\mathsf{rec}(head(rv), n).\mathsf{R}_{tail(rv)}(P) \\
\mathsf{R}_{rv}(u(x).P) & \hat{=} & u(x).\mathsf{rec}(head(rv), x).\mathsf{R}_{tail(rv)}(P) \\
\mathsf{R}_{rv}(\overline{u}\langle M\rangle.P) & \hat{=} & \overline{u}\langle M\rangle.\mathsf{R}_{rv}(P) \\
\mathsf{R}_{rv}(\textit{if } M = N \textit{ then } P \textit{ else } Q) & \hat{=} & \textit{if } M = N \textit{ then } \mathsf{R}_{rv}(P) \textit{ else } \mathsf{R}_{rv}(Q)
\end{array}$$

where the functions head and tail are the usual ones for lists, and odd (resp. even) returns the list of elements in odd (resp. even) position.

In the above definition odd and even are used as a convenient way to split an infinite list into two infinite lists. A voting process can now be constructed such that the voter V records the values constructed and input during execution.

Definition 3. *Given a voting process specification* $\langle V, A\rangle$*, integer* $n \in \mathbb{N}$*, and names* $s_1, \ldots, s_n$ *, we build the augmented voting process*

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

where $V_i^+ = \mathsf{R}(V)\{^{s_i}/_v\}\{^{r_i}/_r \mid r \in rv(\mathsf{R}(V))\}$.

For notational purposes, given a sequence of record variables $\tilde{r}$, we denote by $\tilde{r}_i$ the sequence of variables obtained by indexing each variable in $\tilde{r}$ with i. The process $\mathsf{VP}_n^+(s_1, \ldots, s_n)$ models the voting protocol for n voters casting votes $s_1, \ldots, s_n$, who privately record the data that may be needed for verification using record variables $\tilde{r}_i$.

4 Election verifiability

We formalize election verifiability using three tests Φ^{IV}, Φ^{UV}, Φ^{EV}. Formally, a test is built from conjunctions and disjunctions of *atomic tests* of the form

$(M =_E N)$ where M, N are terms. Tests may contain variables and will need to hold on frames arising from arbitrary protocol executions. The test Φ^{IV} has record variables which will be substituted by the records stored in the frame; and variables expected to correspond to the voter's ballot and other public information, which will be other variables in the domain of the frame. The tests Φ^{UV}, Φ^{EV} substitute only public information, that is, (plain) variables in the frame's domain and hence are suitable for the use by election observers. The designers of electronic voting protocols need not explicitly specify cryptographic tests Φ^{IV}, Φ^{UV}, Φ^{EV} since our definition assumes the existence of tests (perhaps devised after design) which satisfy our conditions. Now we recall the purpose of each test and assume some conventions about how variables are named in the tests.
Individual verifiability: The test Φ^{IV} allows a voter to identify her ballot in the bulletin board. The test has:

- a variable v referring to a voter's vote.
- a variable w referring to a voter's public credential.
- some variables $x, \bar{x}, \hat{x}, \ldots$ expected to refer to global public values pertaining to the election, for example, public keys belonging to election administrators.
- a variable y expected to refer to the voter's ballot on the bulletin board.
- some record variables $r_1, \ldots, r_k$ referring to the voter's private data.

Universal verifiability: The test Φ^{UV} allows an observer to check that the election outcome corresponds to the ballots in the bulletin board. The test has:

- a tuple of variables $\tilde{v} = (v_1, \ldots, v_n)$ referring to the declared outcome.
- some variables $x, \bar{x}, \hat{x}, \ldots$ as above.
- a tuple $\tilde{y} = (y_1, \ldots, y_n)$ expected to refer to all the voters' ballots on the bulletin board.
- some variables $z, \bar{z}, \hat{z}, \ldots$ expected to refer to outputs generated during the protocol used for the purposes of universal and eligibility verification.

Eligibility verifiability: The test Φ^{EV} allows an observer to check that each ballot in the bulletin board was cast by a unique registered voter. The test has:

- a tuple $\tilde{w} = (w_1, \ldots, w_n)$ referring to public credentials of eligible voters.
- some variables $x, \bar{x}, \hat{x}, \ldots$ as above.
- a tuple $\tilde{y}$ as above.
- some variables $z, \bar{z}, \hat{z}, \ldots$ as above.

The remainder of this section will focus on the individual and universal aspects of our definition; eligibility verifiability will be discussed in Section 5.

4.1 Individual and universal verifiability

The tests suitable for the purposes of election verifiability have to satisfy certain conditions: if the tests succeed, then the data output by the election is indeed valid (*soundness*); and there is a behaviour of the election authority which produces election data satisfying the tests (*effectiveness*). Formally these requirements are captured by the definition below. We use the notation $\tilde{T} \simeq \tilde{T}'$ to denote that the tuples $\tilde{T}$ and $\tilde{T}'$ are a permutation of each others modulo the equational theory, that is, we have $\tilde{T} = T_1, \dots T_n$, $\tilde{T}' = T'_1, \dots T'_n$ and there exists a permutation χ on $\{1, \dots, n\}$ such that for all $1 \leq i \leq n$ we have $T_i =_E T'_{\chi(i)}$.

Definition 4 (Individual and universal verifiability)**.** *A voting specification $\langle V, A \rangle$ satisfies* individual and universal verifiability *if for all $n \in \mathbb{N}$ there exist tests Φ^{IV}, Φ^{UV} such that $fn(\Phi^{IV}) = fn(\Phi^{UV}) = rv(\Phi^{UV}) = \emptyset$, $rv(\Phi^{IV}) \subseteq rv(\mathsf{R}(V))$, and for all names $\tilde{s} = (s_1, \dots, s_n)$ the conditions below hold. Let $\tilde{r} = rv(\Phi^{IV})$ and $\Phi_i^{IV} = \Phi^{IV}\{^{s_i}/_{v}, ^{\tilde{r}_i}/_{\tilde{r}}\}$.*

Soundness. *For all contexts C and processes B such that $C[\mathsf{VP}_n^+(s_1, \dots, s_n)] \Longrightarrow B$ and $\phi(B) \equiv \nu\tilde{n}.\sigma$, we have:*

$$\forall i, j. \quad \Phi_i^{IV}\sigma \wedge \Phi_j^{IV}\sigma \Rightarrow i = j \tag{1}$$

$$\Phi^{UV}\sigma \wedge \Phi^{UV}\{^{\tilde{v}'}/_{\tilde{v}}\}\sigma \Rightarrow \tilde{v}\sigma \simeq \tilde{v}'\sigma \tag{2}$$

$$\bigwedge_{1 \leq i \leq n} \Phi_i^{IV}\{^{y_i}/_{y}\}\sigma \wedge \Phi^{UV}\sigma \ \Rightarrow\ \tilde{s} \simeq \tilde{v}\sigma \tag{3}$$

Effectiveness. *There exists a context C and a process B, such that $C[\mathsf{VP}_n^+(s_1, \dots, s_n)] \Longrightarrow B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$ and*

$$\bigwedge_{1 \leq i \leq n} \Phi_i^{IV}\{^{y_i}/_{y}\}\sigma \wedge \Phi^{UV}\sigma \tag{4}$$

We now discuss how voters and observers use these tests and what are the guarantees given by the conditions stated in Definition 4.

An individual voter should verify that the test Φ^{IV} holds when instantiated with her vote s_i, the information $\tilde{r}_i$ recorded during the execution of the protocol and some bulletin board entry (which she needs to identify in some way, maybe by testing all of them). Indeed, Condition (1) ensures that the test will hold for at most one bulletin board entry. This allows the voter to convince herself that her ballot has been counted. (To understand the way the condition is encoded, notice that in the first conjunct, the test succeeds with the ith voter's data and a ballot $y\sigma$ provided by the context $C[\_]$; in the second conjunct, the test succeeds with j's data and the same ballot.) The fact that her ballot contains her vote will be ensured by Φ^{UV} which should also be tested by the voter.

An observer will instantiate the test Φ^{UV} with the bulletin board entries $\tilde{y}$ and the declared outcome $\tilde{v}$. Condition (2) ensures the observer that Φ^{UV} only holds for one outcome. (In the first and second conjuncts, the test succeeds with declared outcomes $\tilde{v}\sigma$ and $\tilde{v}'\sigma$ respectively, where both $\tilde{v}\sigma$ and $\tilde{v}'\sigma$ are provided by the context $C[\_]$.)

Condition (3) ensures that if a bulletin board contains the ballots of voters who voted $s_1, \ldots, s_n$ then Φ^{UV} only holds if the declared outcome is (a permutation of) these votes.

Finally, Condition (4) ensures that there exists an execution where the tests hold. In particular this allows us to verify whether the protocol can satisfy the tests when executed as expected. This also avoids tests which are always false and would make Conditions (1)-(3) vacuously hold.

Example 3. *We show that the raising hands protocol (Example 2) satisfies our definition. Note that in the augmented voting process, the voter will record his private key; we will denote the ith voter's private key with the record variable* r_{skv_i}*. For all* $n \in \mathbb{N}$ *we define the tests*

$$\Phi^{IV} \mathrel{\hat{=}} y =_E (\!|\mathsf{pk}(r_{skv}), \mathsf{sign}(r_{skv}, v)|\!) \qquad \Phi^{UV} \mathrel{\hat{=}} \bigwedge_{1 \le i \le n} \mathsf{getmsg}(\pi_2(y_i)) =_E v_i$$

We now show that Conditions (1)–(3) of Definition 4 are satisfied.

(1) Suppose that $\Phi_i^{IV}\sigma$ *and* $\Phi_j^{IV}\sigma$ *hold, that is,*

$$\begin{array}{lll} y\sigma & =_E & (\!|\mathsf{pk}(r_{skv_i}\sigma), \mathsf{sign}(r_{skv_i}\sigma, s_i)|\!) \\ y\sigma & =_E & (\!|\mathsf{pk}(r_{skv_j}\sigma), \mathsf{sign}(r_{skv_j}\sigma, s_j)|\!) \end{array}$$

From the equational theory it follows that $r_{skv_i}\sigma =_E r_{skv_j}\sigma$*. Moreover, it follows from the voting process specification and the semantics of the applied pi calculus that for every* σ*, such that* $C[\mathsf{VP}_n^+(s_1, \ldots, s_n)] \Longrightarrow B$ *and* $\phi(B) \equiv \nu\tilde{n}.\sigma$*,* $i \neq j$ *implies that* $r_{skv_i}\sigma \neq_E r_{skv_j}\sigma$*. Hence we conclude that Condition (1) holds.*

(2) For any substitution σ*, the premise of Condition (2) implies* $\bigwedge_{1\le i\le n} v_i\sigma =_E v_i'\sigma$ *and hence* $v\sigma \simeq v'\sigma$*.*

(3) For any substitution σ*, the premise of Condition (3) implies* $\bigwedge_{1\le i\le n} s_i =_E \pi_1(y_i\sigma) \wedge \pi_1(y_i\sigma) =_E v_i\sigma$ *and hence* $\tilde{s} \simeq \tilde{v}\sigma$*.*

To see that Condition (4) holds let $C \mathrel{\hat{=}} \_$*. It is easy to see that* $\mathsf{VP}_n^+(s_1, \ldots, s_n) \Longrightarrow B$*, such that*

$$\phi(B) \equiv \nu skA, skv_1 \ldots skv_n.\{^{\mathsf{pk}(skA)}/_{x_{pkA}}, ^{(\!|\mathsf{pk}(skv_1), \mathsf{sign}(skv_1, s_1)|\!)}/_{y_1}, ^{skv_1}/_{r_{skv_1}}, \ldots, ^{(\!|\mathsf{pk}(skv_n), \mathsf{sign}(skv_n, s_n)|\!)}/_{y_n}, ^{skv_n}/_{r_{skv_n}}\}$$

and that $\Phi^{IV}\sigma \wedge \Phi^{UV}\sigma$ *hold.*

Example 4. *Consider the* postal vote *protocol whereby all voters simply send their vote to an administrator who publishes the list of cast votes. The voting process specification is simply* $\langle \bar{c}\langle v\rangle, \_\rangle$*. Such a protocol is obviously not verifiable and violates our definition. It is indeed not possible to design a test* Φ^{IV} *such that Condition (1) holds when* $s_i = s_j$ *for some* $i \neq j$*.*

4.2 Case study: FOO

The protocol by Fujioka, Okamoto and Ohta [15], FOO for short, was an early protocol based on blind signatures and has been influential for the design of later protocols.

How FOO works. The FOO protocol involves voters, a registrar and a tallier. The voter first computes her ballot as a commitment to her vote $m' = \mathsf{commit}(rnd, v)$ and sends the signed blinded ballot $\mathsf{sign}(sk_V, \mathsf{blind}(rnd', m'))$ to the registrar. The registrar checks that the signature belongs to an eligible voter and returns $\mathsf{sign}(sk_R, \mathsf{blind}(rnd', m'))$ the blind signed ballot. The voter verifies that this input corresponds to the registrar's signature and unblinds the message to recover her ballot signed by the registrar $m = \mathsf{sign}(sk_R, m')$. The voter then posts her signed ballot to the bulletin board. Once all votes have been cast the tallier verifies all the entries and appends an identifier l to each valid entry. The voter then checks the bulletin board for her entry, the triple (l, m', m), and appends the commitment factor *rnd*. Finally, using *rnd* the tallier opens all of the ballots and announces the declared outcome.

Equational theory. We model blind signatures and commitment as follows.

$$\begin{array}{rclrcl}
\mathsf{checksign}(\mathsf{pk}(x), \mathsf{sign}(x,y)) & = & \mathsf{true} & \mathsf{getmsg}(\mathsf{sign}(x,y)) & = & y \\
\mathsf{unblind}(y, \mathsf{sign}(x, \mathsf{blind}(y,z))) & = & \mathsf{sign}(x,z) & \mathsf{unblind}(x, \mathsf{blind}(x,y)) & = & y \\
\mathsf{open}(x, \mathsf{commit}(x,y)) & = & y & & &
\end{array}$$

Model in applied pi. As discussed in the introduction, the parts of the protocol that need to be trusted for achieving verifiability are surprisingly simple. The name *rnd* models the randomness that is supposed to be used to compute the commitment of the vote. All a voter needs to ensure is that the randomness used for the commitment is fresh. To ensure verifiability, all other operations such as computing the commitment, blinding and signing can be performed by the untrusted terminal.

Definition 5. *The voting process specification* $\langle V_{\mathsf{foo}}, A_{\mathsf{foo}} \rangle$ *is defined where*

$$V_{\mathsf{foo}} \mathrel{\hat{=}} \nu rnd.\bar{c}\langle v\rangle.\bar{c}\langle rnd\rangle \quad \textit{and} \quad A_{\mathsf{foo}}[\_] \mathrel{\hat{=}} \_$$

The name *rnd* models the randomness that is supposed to be used to compute the commitment of the vote. All a voter needs to ensure is that the randomness used for the commitment is fresh. To ensure verifiability, all other operations such as computing the commitment, blinding and signing can be performed by the untrusted terminal. The augmented voting process $\mathsf{VP}_n^+(s_1, \ldots, s_n)$ is $\nu\ rnd.\mathsf{rec}(r_1, rnd).\bar{c}\langle s_1\rangle.\bar{c}\langle rnd\rangle \mid \ldots \mid \nu\ rnd.\mathsf{rec}(r_n, rnd).\bar{c}\langle s_n\rangle.\bar{c}\langle rnd\rangle$.

Individual and universal verifiability. We define the tests

$$\Phi^{IV} \mathrel{\hat{=}} y =_E (\!| r, \mathsf{commit}(r,v) |\!) \qquad \Phi^{UV} \mathrel{\hat{=}} \bigwedge_{1 \le i \le n} v_i =_E \mathsf{open}(\pi_1(y), \pi_2(y))$$

Intuitively, a bulletin board entry y should correspond to the pair formed of the random generated by voter i and commitment to her vote.

Theorem 1. $\langle V_{\mathsf{foo}}, A_{\mathsf{foo}} \rangle$ *satisfies individual and universal verifiability.*

Proof. We show that the Conditions (1)–(3) of Definition 4 hold.

(1) Suppose C, B, i, j are such that $C[\mathsf{VP}_{\mathsf{n}}^{+}(s_1, \ldots, s_n)] \Longrightarrow B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$, $\Phi_i^{IV}\sigma$ and $\Phi_j^{IV}\sigma$. Then $\pi_2(y)\sigma = r_i\sigma$ by $\Phi_i^{IV}\sigma$, and $\pi_2(y)\sigma = r_j\sigma$ by $\Phi_j^{IV}\sigma$, so $r_i\sigma = r_j\sigma$. But since these are randoms freshly generated by the processes V_i and V_j, it follows that $i = j$. (This can be easily shown by induction on the derivation which produces B.)

(2) For any σ we have for all $1 \leq i \leq n$ that

$$v_i\sigma =_E \mathsf{open}(\pi_1(y_i)\sigma, \pi_2(y_i)\sigma) \wedge v_i'\sigma =_E \mathsf{open}(\pi_1(y_i\sigma), \pi_2(y_i\sigma)) \\ \Rightarrow v_i\sigma =_E v_i'\sigma$$

(3) It follows from the equational theory that for all $1 \leq i \leq n$ and substitution σ that

$$y_i\sigma =_E (\!| r_i\sigma, \mathsf{commit}(r_i\sigma, s_i) |\!) \wedge v_i\sigma =_E \mathsf{open}(\pi_1(y_i\sigma), \pi_2(y_i\sigma)) \\ \Rightarrow \tilde{s} =_E \tilde{v}\sigma$$

It is also easy to see that a context modelling the entire FOO protocol would satisfy effectiveness (Condition (4)). One may for instance slightly adapt the modelling of the FOO protocol given in [14] for this purpose. □

Our model of FOO does not rely on the blind signatures. While this part is crucial for privacy properties it does not contribute to verifiability. Similarly, the voter's signature on the blinded committed vote and the confidentiality of the secret signing key are not required for individual and universal verifiability; they are however essential for eligibility.

4.3 Case study: Helios 2.0

Helios 2.0 [4] is an open-source web-based election system, based on homomorphic tallying of encrypted votes. It allows the secret election key to be distributed amongst several trustees, and supports distributed decryption of the election result. It also allows independent verification by voters and observers of election results. Helios 2.0 was successfully used in March 2009 to elect the president of the Catholic University of Louvain, an election that had 25,000 eligible voters.

How Helios works. An election is created by naming a set of trustees and running a protocol that provides each of them with a share of the secret part of a public key pair. The public part of the key is published. Each of the eligible voters is also provided with a private pseudo-identity. The steps that participants take during a run of Helios are as follows.

1. To cast a vote, the user runs a browser script that inputs her vote and creates a ballot that is encrypted with the public key of the election. The ballot includes a ZKP that the ballot represents an allowed vote (this is needed because the ballots are never decrypted individually).

2. The user can audit the ballot to check if it really represents a vote for her chosen candidate; if she elects to do this, the script provides her with the random data used in the ballot creation. She can then independently verify that the ballot was correctly constructed, but the ballot is now invalid and she has to create another one.

3. When the voter has decided to cast her ballot, the voter's browser submits it along with her pseudo-identity to the server. The server checks the ZKPs of the ballots, and publishes them on a bulletin board.

4. Individual voters can check that their ballots appear on the bulletin board. Any observer can check that the ballots that appear on the bulletin board represent allowed votes, by checking the ZKPs.

5. The server homomorphically combines the ballots, and publishes the encrypted tally. Anyone can check that this tally is done correctly.

6. The server submits the encrypted tally to each of the trustees, and obtains their share of the decryption key for that particular ciphertext, together with a proof that the key share is well-formed. The server publishes these key shares along with the proofs. Anyone can check the proofs.

7. The server decrypts the tally and publishes the result. Anyone can check this decryption.

Equational theory. We use a signature in which $\mathsf{penc}(x_{\mathsf{pk}}, x_{\mathsf{rand}}, x_{\mathsf{text}})$ denotes the encryption with key x_{pk} and random x_{rand} of the plaintext x_{text}, and $x_{\mathsf{ciph}} * y_{\mathsf{ciph}}$ denotes the homomorphic combination of ciphertexts x_{ciph} and y_{ciph} (the corresponding operation on plaintexts is written $+$ and on randoms $\circ$). The term $\mathsf{ballotPf}(x_{\mathsf{pk}}, x_{\mathsf{rand}}, s, x_{\mathsf{ballot}})$ represents a proof that the ballot x_{ballot} contains some name s and random x_{rand} with respect to key x_{pk}; $\mathsf{decKey}(x_{\mathsf{sk}}, x_{\mathsf{ciph}})$ is a decryption key for x_{ciph} w.r.t. public key $\mathsf{pk}(x_{\mathsf{sk}})$; and $\mathsf{decKeyPf}(x_{\mathsf{sk}}, x_{\mathsf{ciph}}, x_{\mathsf{dk}})$ is a proof that x_{dk} is a decryption key for x_{ciph} w.r.t. public key $\mathsf{pk}(x_{\mathsf{sk}})$. We use the equational theory that asserts that $+, *, \circ$ are commutative and associative, and includes the equations:

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

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

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

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

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

Note that in the equation for checkBallotPf we have that s is a name and not a variable. As the equational theory is closed under bijective renaming of names this equation will hold for any name, but will fail if one replaces the name by a term, for example, $s + s$. We suppose that all names are possible votes but give the possibility to check that a voter does not include a term $s + s$ which would allow her to add an additional vote to the outcome.

Model in applied pi. The parts of the system that are not verifiable are:

- The browser script that constructs the ballot. Although the voter cannot verify it, the trust in this script is motivated by the fact that she is able to audit it. She does that by creating as many ballots as she likes and checking all but one of them, and then casting the one she didn't verify.
- The trustees. Although the trustees' behaviour cannot be verified, voters and observers may want to trust them because trust is distributed among them.

We model these two components as trusted parts, by including them in the context A_{helios} of our voting process specification.

Definition 6. *The voting process specification* $\langle V_{\mathsf{helios}}, A_{\mathsf{helios}}\rangle$ *is defined where*

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

We suppose that the recording function records the inputs of x_{pid}, x_{ballot} and x_{ballotpf} in record variables r_{pid}, r_{ballot} and $r_{ballotpf}$ respectively. The voter V_{helios} receives her voter id *pid* on a private channel. She sends her vote on the channel to A_{helios}, which creates the ballot for her. She receives the ballot and sends it (paired with *pid*) to the server. A_{helios} represents the parts of the system that are required to be trusted. It publishes the election key and issues voter ids. It includes the ballot creation script B, which receives a voter's vote, creates a random m and forms the ballot, along with its proof, and returns it to the voter. A_{helios} also contains the trustee T, which accepts a tally ciphertext and returns a decryption key for it, along with the proof that the decryption key is correct. We assume the trustee will decrypt any ciphertext (but only one). In practice, of course, the trustee should ensure that the ciphertext is the right one, namely, the homomorphic addition of all the ballots posted to the bulletin board.

The untrusted server is assumed to publish the election data. In our formalism, we expect the frame to have a substitution σ that defines the election public key as x_{pk} and the individual *pid*'s and ballots as y_i for each voter i. It also contains the homomorphic tally z_{tally} of the encrypted ballots, and the decryption key z_{decKey} and its proof of correctness z_{decKeyPf} obtained from the trustees. When the protocol is executed as expected the resulting frame should have substitution σ such that

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

The server then decrypts the tally to obtain the outcome of the election.

Individual and universal verifiability. For the purposes of individual and universal verifiability, the tests Φ^{IV} and Φ^{UV} are introduced. Accordingly, given $n \in \mathbb{N}$ we define:

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

The test Φ^{IV} checks that the voter's ballot is recorded on the bulletin board. The test Φ^{UV} checks that the tally is correctly computed; it checks the proof for the decryption key; and it checks the decrypted tally corresponds to the declared outcome $\tilde{v}$.

Theorem 2. *$\langle V_{\mathsf{helios}}, A_{\mathsf{helios}} \rangle$ satisfies individual and universal verifiability.*

Proof. Suppose $n \in \mathbb{N}$ and test Φ^{IV}, Φ^{UV} are given above. We will now show that for all $\tilde{s} = (s_1, \ldots, s_n)$ that the conditions of Definition 4 are satisfied.

(1) Suppose C, B, i, j are such that $C[\mathsf{VP}^{+}_{\mathsf{n}}(s_1, \ldots, s_n)] \Longrightarrow B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$, and $\Phi^{IV}_i\sigma$ and $\Phi^{IV}_j\sigma$ hold. Then $\pi_2(y)\sigma = r_{ballot,i}\sigma$ by $\Phi^{IV}_i\sigma$, and

$\pi_2(y)\sigma = r_{ballot,j}\sigma$ by $\Phi^{IV}_j\sigma$, so $r_{ballot,i}\sigma = r_{ballot,j}\sigma$. But since these are randoms freshly generated for the processes V_i and V_j, it follows that $i = j$.

(2) Let σ be any substitution and suppose that $\Phi^{UV}\sigma$ and $\Phi^{UV}\{\tilde{v}'/\tilde{v}\}\sigma$. Then $(v_1 + \cdots + v_n)\sigma = (v'_1 + \cdots + v'_n)\sigma = \mathsf{dec}(z_{\mathsf{decKey}}, z_{\mathsf{tally}})\sigma$. Moreover, $\Phi^{UV}\sigma$ we have that $\bigwedge_{i=1}^{n}(\mathsf{checkBallotPf}(x_{\mathsf{pk}}, \pi_2(y_i), \pi_3(y_i)))\sigma$ which implies that each $v_i\sigma$ and $v'_i\sigma$ is a name. Hence $\tilde{v}\sigma \simeq \tilde{v}'\sigma$.

(3) Let σ be any substitution and suppose that $\bigwedge_{1\leq i\leq n}\Phi^{IV}_i\{y_i/y\}\sigma$ and $\Phi^{UV}\sigma$. From each $\Phi^{IV}_i\{y_i/y\}\sigma$, we have that $\pi_2(y_i)\sigma = \mathsf{penc}(\mathsf{pk}(sk), m_i, s_i)\sigma$ for some m_i. From $\Phi^{UV}\sigma$, we have $z_{\mathsf{tally}}\sigma = (\pi_2(y_1) * \cdots * \pi_2(y_n))\sigma$, and by the equation for homomorphic encryption, this is $\mathsf{penc}(\mathsf{pk}(sk), m_1 \circ \cdots \circ m_n, s_1 + \cdots + s_n)$. From the decryption key proof and the decryption, we have $(v_1 + \cdots + v_n)\sigma = (s_1 + \cdots + s_n)$, and from the ballot proofs, we can conclude that $\tilde{s} \simeq \tilde{v}\sigma$.

(4) The context C must marshal the election data on the frame in such a way that $x_{pk}\sigma$, $y_i\sigma$, $z_{\mathsf{tally}}\sigma$, $z_{\mathsf{decKey}}\sigma$, and $z_{\mathsf{decKeyPf}}\sigma$ are as defined above. Moreover, it finds some names $t_1, \ldots, t_n$ such that $z_{\mathsf{tally}}\sigma = t_1 + \cdots + t_n$, and sets the declared outcome $\tilde{v}\sigma$ to be $(\!| t_1, \ldots, t_n |\!)$. □

5 Eligibility verifiability

In order to fully capture *election verifiability*, the tests Φ^{IV} and Φ^{UV} must be supplemented by a test Φ^{EV} that checks eligibility of the voters whose votes have been counted in the outcome. We suppose that the public voter credentials appear on the bulletin board. Moreover, these credentials actually belong to eligible voters; verifying this is beyond the scope of this paper. One approach may involve publishing the list of credentials alongside the real names and addresses of the electorate, the validity of this list can then be scrutinised by the observer. The test Φ^{EV} allows an observer to check that only these individuals (that is, those in posession of credentials) cast votes, and at most one vote each. The test is instantiated with the list of public credentials, and other public outputs of the election process, such as the public keys, the voters' ballots, and any other outputs such as proofs. We use the variable naming convention introduced in the previous section.

Definition 7 (Election verifiability)**.** *A voting specification $\langle V, A\rangle$ satisfies* election verifiability *if for all $n \in \mathbb{N}$ there exist tests $\Phi^{IV}, \Phi^{UV}, \Phi^{EV}$ such that $fn(\Phi^{IV}) = fn(\Phi^{UV}) = fn(\Phi^{EV}) = rv(\Phi^{UV}) = rv(\Phi^{EV}) = \emptyset$, $rv(\Phi^{IV}) \subseteq rv(\mathsf{R}(V))$, and for all names $\tilde{s} = (s_1, \ldots, s_n)$ we have:*

1. *The tests Φ^{IV} and Φ^{UV} satisfy each of the conditions of Definition 4;*
2. *The additional conditions 5, 6, 7 and 8 below hold.*

Let $\tilde{r} = rv(\Phi^{IV})$, $\Phi^{IV}_i = \Phi^{IV}\{s_i/v, \tilde{r}_i/\tilde{r}, y_i/y\}$ and $X = fv(\Phi^{EV})\backslash dom(\mathsf{VP}^+_{\mathsf{n}}(s_1, \ldots, s_n))$

Soundness. *For all contexts C and processes B such that $C[\mathsf{VP}_\mathsf{n}^+(s_1,\ldots,s_n)] \Longrightarrow B$ and $\phi(B) \equiv \nu\tilde{n}.\sigma$, we have:*

$$\Phi^{EV}\sigma \wedge \Phi^{EV}\{^{x'}/_{x} \mid x \in X\backslash\tilde{y}\}\sigma \Rightarrow \tilde{w}\sigma \simeq \tilde{w}'\sigma \tag{5}$$

$$\bigwedge_{1\le i\le n} \Phi_i^{IV}\sigma \wedge \Phi^{EV}\{^{\tilde{w}'}/_{\tilde{w}}\}\sigma \quad \Rightarrow \quad \tilde{w}\sigma \simeq \tilde{w}'\sigma \tag{6}$$

$$\Phi^{EV}\sigma \wedge \Phi^{EV}\{^{x'}/_{x} \mid x \in X\backslash\tilde{w}\}\sigma \Rightarrow \tilde{y}\sigma \simeq \tilde{y}'\sigma \tag{7}$$

Effectiveness. *There exists a context C and a process B such that $C[\mathsf{VP}_\mathsf{n}^+(s_1, \ldots, s_n)] \Longrightarrow B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$ and*

$$\bigwedge_{1\le i\le n} \Phi_i^{IV}\sigma \wedge \Phi^{UV}\sigma \wedge \Phi^{EV}\sigma \tag{8}$$

The test Φ^{EV} is instantiated by an observer with the bulletin board. Condition (5) ensures that, given a set of ballots $\tilde{y}\sigma$, provided by the environment, Φ^{EV} succeeds only for one list of voter public credentials. Condition (6) ensures that if a bulletin board contains the ballots of voters with public credentials $\tilde{w}\sigma$ then Φ^{EV} only holds on a permutation of these credentials. Condition (7) ensures that, given a set of credentials $\tilde{w}$, only one set of bulletin board entries $\tilde{y}$ are accepted by Φ^{EV} (observe that for such a strong requirement to hold we expect the voting specification's frame to contain a public key, to root trust). Finally, the effectiveness condition is similar to Condition (4) of the previous section.

Example 5. *The raising hands protocol satisfies eligibility verifiability. Let*

$$\Phi^{EV} \mathrel{\hat{=}} \bigwedge_{1\le i\le n} \big(\mathsf{checksign}(x_{pkA}, w_i) =_E \mathsf{true} \wedge \pi_1(y_i) =_E \mathsf{getmsg}(w_i) \\ \wedge\ \mathsf{checksign}(\pi_1(y_1), \pi_2(y_i)) =_E \mathsf{true}\big)$$

We also need to slightly strengthen Φ^{IV} which is defined as

$$\Phi^{IV} \mathrel{\hat{=}} y =_E (\!|\mathsf{pk}(r_{skv}), \mathsf{sign}(r_{skv}, v)|\!) \wedge \mathsf{getmsg}(w) =_E \mathsf{pk}(r_{skv}) \\ \wedge\ \mathsf{checksign}(x_{pkA}, w) =_E \mathsf{true}$$

Conditions (1) – (4) can be proved as previously (Example 3). It remains to show that Conditions (5) – (8) hold.

(5) Suppose $\Phi^{EV}\sigma$ and $\Phi^{EV}\{^{x'}/_{x} \mid x \in X\backslash\tilde{y}\}\sigma$ hold; for all $1 \le i \le n$ we have

$$\mathsf{getmsg}(w_i)\sigma =_E \pi_1(y_i)\sigma \wedge \pi_1(y_i)\sigma =_E \mathsf{getmsg}(w_i')\sigma \\ \wedge\ \mathsf{checksign}(w_i, x_{pkA})\sigma =_E \mathsf{true} \wedge \mathsf{checksign}(w_i', x_{pkA})\sigma =_E \mathsf{true}$$

By inspection of the equational theory we have that $\tilde{w}_i\sigma =_E \tilde{w}_i'\sigma$.

(6) Suppose that $\bigwedge_{1\leq i\leq n}\Phi_i^{IV}\sigma$ and $\Phi^{EV}\sigma$ hold; hence for all $1 \leq i \leq n$ we have

$$\begin{aligned}&\mathsf{getmsg}(w_i)\sigma =_E \mathsf{getmsg}(w_i')\sigma\\ &\qquad\wedge\ \mathsf{checksign}(x_{pkA}, w_i)\sigma =_E \mathsf{checksign}(w_i', x_{pkA})\sigma =_E \mathsf{true}\end{aligned}$$

Again we have $\tilde{w}_i\sigma =_E \tilde{w}_i'\sigma$.

(7) Suppose $\Phi^{EV}\sigma \wedge \Phi^{EV}\{^{x'}/_x \mid x \in X\backslash\tilde{w}\}\sigma$ hold; for all $1 \leq i \leq n$ we have

$$\begin{gathered}\pi_1(y_i)\sigma =_E \mathsf{getmsg}(w_i)\sigma =_E \pi_1(y_i')\sigma \wedge \mathsf{checksign}(x_{pkA}, w_i)\sigma =_E \mathsf{true}\ \wedge\\ \mathsf{checksign}(\pi_1(y_i), \pi_2(y_i))\sigma =_E \mathsf{true} \wedge \mathsf{checksign}(\pi_1(y_i'), \pi_2(y_i'))\sigma =_E \mathsf{true}\end{gathered}$$

For any σ such that $C[\mathsf{VP}_\mathsf{n}^+(s_1, \ldots, s_n)] \Longrightarrow B$ and $\phi(B) \equiv \nu\tilde{n}.\sigma$ we have that if $\mathsf{checksign}(x_{pkA}, w_i)\sigma =_E \mathsf{true}$ then $\mathsf{getmsg}(w_i)\sigma =_E \mathsf{pk}(skv_j)$ for some $j \in [1..n]$. Moreover, if $\mathsf{checksign}(\mathsf{pk}(skv_j), \pi_2(x))\sigma =_E \mathsf{true}$ then $\mathsf{getmsg}(\pi_2(y_i))\sigma =_E s_j$. Hence we have that for all $1 \leq i \leq n$ that $\pi_2(y_i)\sigma =_E \pi_2(y_i')\sigma$. Finally we conclude $\tilde{y}\sigma =_E \tilde{y}'\sigma$.

Case studies: FOO and Helios 2.0. Neither FOO nor Helios use public voting credentials in a manner suitable for eligibility verifiability. In FOO, the administrator is responsible for ensuring eligibility, that is, checking the validity of the voter's ballots; whereas in Helios, there are no public voting credentials. It follows immediately that Condition (7), in particular, cannot be satisfied.

5.1 Case study: JCJ-Civitas

The protocol due to Juels, Catalano & Jakobsson [18] is based on mixnets and has been implemented by Clarkson, Chong & Myers [13, 12] as an open-source voting system called Civitas. The schemes, which we call JCJ-Civitas, are the first to provide election verifiability.

How JCJ-Civitas works. An election is created by naming a set of registrars and talliers. The protocol is divided into four phases: *setup*, *registration*, *voting* and *tallying*. We now detail the steps of the protocol, starting with the setup phase.

1. The registrars (respectively talliers) run a protocol which constructs a public key pair and distributes a share of the secret part amongst the registrars' (respectively talliers'). The public part $\mathsf{pk}(sk_T)$ (respectively $\mathsf{pk}(sk_R)$) of the key is then published. In addition, the registrars construct a distributed signing key pair ssk_R, $\mathsf{pk}(ssk_R)$.

The registration phase then proceeds as follows.

2. The registrars generate and distribute voter credentials: a private part d and a public part $\mathsf{penc}(\mathsf{pk}(sk_R), m'', d)$ (the probabilistic encryption of d under the registrars' public key $\mathsf{pk}(sk_R)$). This is done in a distributed manner, so that no registrar learns the value of any private credential d.

3. The registrars publish the signed public voter credentials.

4. The registrars announce the candidate list $\tilde{t} = (t_1, \ldots, t_l)$.

The protocol then enters the voting phase.

5. Each voter selects her vote $s \in \tilde{t}$ and computes two ciphertexts $M = \mathsf{penc}(\mathsf{pk}(sk_T), m, s)$ and $M' = \mathsf{penc}(\mathsf{pk}(sk_R), m', d)$ where m, m' are nonces. M contains her vote and M' her credential. In addition, the voter constructs a non-interactive zero-knowledge proof of knowledge demonstrating the correct construction of her ciphertexts and validity of the candidate ($s \in \tilde{t}$). (The ZKP provides protection against coercion resistance, by preventing forced abstention attacks via a *write in*, and binds the two ciphertexts for eligibility verifiability.) The voter derives her ballot as the triple consisting of her ciphertexts and zero-knowledge proof and posts it to the bulletin board.

After some predefined deadline the tallying phase commences in order to compute the election outcome.

6. The talliers read the n' ballots posted to the bulletin board by voters (that is, the triples consisting of the two ciphertexts and the zero-knowledge proof) and discards any entries for which the zero-knowledge proof does not hold.

7. The elimination of re-votes is performed on the ballots using pairwise *plaintext equality tests* (PET) on the ciphertexts containing private voter credentials. (A PET [16] is a cryptographic predicate which allows a keyholder to provide a proof that two ciphertexts contain the same plaintext.) Re-vote elimination is performed in a verifiable manner with respect to some publicly defined policy, *e.g.*, by the order of ballots on the bulletin board.

8. The talliers perform a verifiable re-encryption mix on the ballots (ballots consist of a vote ciphertext and a public credential ciphertext; the link between both is preserved by the mix.) The mix ensures that a voter cannot trace her vote, allowing the protocol to achieve coercion-resistance.

9. The talliers perform a verifiable re-encryption mix on the list of public credentials published by the registrar. This mix anonymises public voter credentials, breaking any link with the voter for privacy purposes.

10. Ballots based on invalid credentials are weeded using PETs between the mixed ballots and the mixed public credentials. Both have been posted to the bulletin board. (Using PETs the correctness of weeding is verifiable.)

11. Finally, the talliers perform a verifiable decryption and publish the result.

Equational theory. The protocol uses a variant of the ElGamal encryption scheme [18]. Accordingly we adopt the signature and associated equational theory from the Helios case study. The zero-knowledge proof demonstrating correct construction of the voter's ciphertexts is modelled by the equation

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

(For simplicity the zero-knowledge proof does not demonstrate that the voter's vote s is a valid vote, that is, $s \in \tilde{t}$; this is of importance for privacy properties, not verifiability.) Plaintext equivalence tests are modelled by the equation

$$\mathsf{pet}(\mathsf{petPf}(x_{\mathsf{sk}}, ciph, ciph'), ciph, ciph') = \mathsf{true}$$

where $ciph \mathrel{\hat{=}} \mathsf{penc}(\mathsf{pk}(x_{\mathsf{sk}}), x_{\mathsf{rand}}, x_{\mathsf{text}})$ and $ciph' \mathrel{\hat{=}} \mathsf{penc}(\mathsf{pk}(x_{\mathsf{sk}}), x'_{\mathsf{rand}}, x_{\mathsf{text}})$. Re-encryption is defined with respect to the standard equation

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

In addition we consider verifiable re-encryption mixnets and introduce for each permutation χ on $\{1, \dots, n\}$ the equation:

$$\begin{aligned}&\mathsf{checkMix}(\mathsf{mixPf}(x_{\mathsf{ciph},1}, \dots, x_{\mathsf{ciph},n},\\ &\qquad\qquad ciph_1, \dots, ciph_n, z_{\mathsf{rand},1}, \dots, z_{\mathsf{rand},n}),\\ &\qquad\qquad\qquad x_{\mathsf{ciph},1}, \dots, x_{\mathsf{ciph},n}, ciph_1, \dots, ciph_n) = \mathsf{true}\end{aligned}$$

where $ciph_i \mathrel{\hat{=}} \mathsf{renc}(z_{\mathsf{rand},i}, x_{\mathsf{ciph},\chi(i)})$. We also define re-encryption with respect to pairs of ciphertexts and introduce for each permutation χ on $\{1, \dots, n\}$ the equation:

$$\begin{aligned}&\mathsf{checkMixPair}(\mathsf{mixPairPf}((x_1, x'_1), \dots, (x_n, x'_n),\\ &\qquad (c_1, c'_1), \dots, (c_n, c'_n), (z_1, z'_1), \dots, (z_n, z'_n)),\\ &\qquad\qquad (x_1, x'_1), \dots, (x_n, x'_n), (c_1, c'_1), \dots, (c_n, c'_n)) = \mathsf{true}\end{aligned}$$

where $c_i \mathrel{\hat{=}} \mathsf{renc}(z_i, x_{\chi(i)})$ and $c'_i \mathrel{\hat{=}} \mathsf{renc}(z'_i, x'_{\chi(i)})$.

The following lemmata demonstrate useful properties of our equational theory. We make use of the notation $\tilde{M} \overset{\bullet}{\simeq} \tilde{M}'$ to denote that the ciphertext tuples $\tilde{M}$, $\tilde{M}'$ are defined over the same plaintexts with respect to some public key K, that is, we have $\tilde{M} =_E (\mathsf{penc}(K, R_1, N_1), \dots \mathsf{penc}(K, R_n, N_n))$, $\tilde{M}' =_E (\mathsf{penc}(K, R'_1, N'_1), \dots \mathsf{penc}(K, R'_n, N'_n))$ for some tuples $\tilde{N}, \tilde{N}', \tilde{R}, \tilde{R}'$ and there exists a permutation χ defined over $\{1, \dots, n\}$ such that for all $1 \le i \le n$ we have $N_i =_E N'_{\chi(i)}$. The relation $\overset{\bullet}{\simeq}$ is trivially seen to be an equivalence relation. Moreover, if $\tilde{M} \overset{\bullet}{\simeq} \tilde{N}$ and $\tilde{M} \simeq \tilde{M}'$, then $M' \overset{\bullet}{\simeq} \tilde{N}$.

Lemma 1. *Given terms* L, M, N, *if* $\mathsf{pet}(L, M, N) =_E \mathsf{true}$, *then* $M \stackrel{\bullet}{\simeq} N$.

Lemma 2. *Given terms* $L, \tilde{M}, \tilde{N}$, *if* $\mathsf{checkMix}(L, \tilde{M}, \tilde{N}) =_E \mathsf{true}$, *then* $\tilde{M} \stackrel{\bullet}{\simeq} \tilde{N}$.

Lemma 3. *Given terms* $L, \tilde{M}, \tilde{N}$, *if* $\mathsf{checkMixPair}(L, \tilde{M}, \tilde{N}) =_E \mathsf{true}$, *then* $(\pi_i(M_1), \ldots, \pi_i(M_{|\tilde{M}|})) \stackrel{\bullet}{\simeq} (\pi_i(N_1), \ldots, \pi_i(N_{|\tilde{N}|}))$.

Model in applied pi. We make the following trust assumptions for verifiability:

- The voter is able to construct her ballot; that is, she is able to generate nonces m, m', construct a pair of ciphertexts and generate a zero-knowledge proof.
- The registrar constructs distinct credentials d for each voter and constructs the voter's public credential correctly. (The latter assumption can be dropped if the registrar provides a proof that the public credential is correctly formed [18].) The registrar also keeps the private part of the signing key secret.

Although neither voters nor observers can verify that the registrars adhere to such expectations, they trust them because trust is distributed. The trusted components are modelled by the voting process specification $\langle A_{\mathsf{jcj}}, V_{\mathsf{jcj}} \rangle$ (Definition 8). The context A_{jcj} publishes public keys and defines a sub-process R to model the registrar. The registrar R constructs a fresh private credential d and sends the private credential along with the signed public part (that is, $\mathsf{sign}(ssk_R, \mathsf{penc}(x_{pk_R}, m'', d))$) to the voter; the registrar also publishes the signed public credential on the bulletin board. The voter V_{jcj} receives the private and public credentials from the registrar and constructs her ballot; that is, the pair of ciphertexts and a zero-knowledge proof demonstrating their correct construction.

Definition 8. *The voting process specification* $A_{\mathsf{jcj}}, V_{\mathsf{jcj}}$ *is defined where:*

$$\begin{array}{rcl}
A_{\mathsf{jcj}} & \hat{=} & \nu\, a, ssk_R.(!R \mid \{^{\mathsf{pk}(sk_R)}/_{x_{\mathsf{pk_R}}}, {}^{\mathsf{pk}(ssk_R)}/_{x_{spk_R}}, {}^{\mathsf{pk}(sk_T)}/_{x_{\mathsf{pk_T}}}\} \mid \_) \\
V_{\mathsf{jcj}} & \hat{=} & \nu\, m, m'.a(x_{cred}). \\
 & & \quad \mathit{let}\ ciph = \mathsf{penc}(x_{pk_T}, m, v)\ \mathit{in} \\
 & & \quad \mathit{let}\ ciph' = \mathsf{penc}(x_{pk_R}, m', \pi_1(x_{cred}))\ \mathit{in} \\
 & & \quad \mathit{let}\ zkp = \mathsf{ballotPf}(x_{pk_T}, m, v, x_{pk_R}, m', \pi_1(x_{cred}))\ \mathit{in} \\
 & & \quad \overline{c}\langle (\!| ciph, ciph', zkp |\!) \rangle \\
R & \hat{=} & \nu\, d, m''.\ \mathit{let}\ sig = \mathsf{sign}(ssk_R, \mathsf{penc}(x_{pk_R}, m'', d))\ \mathit{in}\ \overline{a}\langle (\!| d, sig |\!) \rangle.\overline{c}\langle sig \rangle
\end{array}$$

At the end of the election the bulletin board is represented by the frame. In our formalism we expect the frame to contain the substitution σ which defines the voters' public credentials as $w_1, \ldots, w_n$, public keys of the registrars as x_{pk_R}, x_{spk_R} and talliers' public key as x_{pk_T}. Triples $y_1, \ldots, y_n$ consisting of each voter's ciphertexts and zero-knowledge proofs. The mixed re-encryptions of the voter's ciphertexts $z_{\mathsf{bal},1}, \ldots, z_{\mathsf{bal},n}$ along with a proof $z_{\mathsf{mixPairPf}}$ that the mix was performed correct. For verifiable decryption we assume $z_{\mathsf{decKey},i}$ is defined as a decryption key associated with the proof $z_{\mathsf{decPf},i}$. For the purposes of eligibility verifiability we also expect the mixed re-encryptions of the voter's public credentials $z_{\mathsf{cred},1}, \dots, z_{\mathsf{cred},1}$ along with a proof of correctness z_{mixPf}. For convenience a reordering $\hat{z}_{\mathsf{cred},1}, \dots, \hat{z}_{\mathsf{cred},n}$ of these re-encryptions is also computed. Finally, we expect PET proofs $z_{\mathsf{petPf},1}, \dots, z_{\mathsf{petPf},n}$ for the reencryption of the ciphertext constructed by the voter on her private credential (that is, the output of the verifiable mix in Step 8 of the protocol) and the reencryption of the voter's public credential constructed by the registrars (that is, the output of the mix in Step 9); such that the PET holds, that is, the pair of ciphertexts contain the same private credential. Accordingly we expect σ to be such that for all $1 \leq i \leq n$:

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

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

Election verifiability. For the purpose of election verifiability we introduce the tests $\Phi^{IV}, \Phi^{UV}, \Phi^{EV}$. Without loss of generality suppose the recording function uses record variables $\tilde{r} = (r_{cred}, r_m, r_{m'}) = \mathrm{rv}(\mathsf{R}(V))$ (corresponding to the variable x_{cred} and names m, m' appearing in the process V). Accordingly, given $n \in \mathbb{N}$ we define:

$$
\begin{array}{lcl}
\Phi^{IV} & \hat{=} & y =_E (\!| \mathsf{penc}(x_{\mathsf{pk_T}}, r_m, v), \mathsf{penc}(x_{\mathsf{pk_R}}, r_{m'}, \pi_1(r_{cred})), \\
 & & \qquad \mathsf{ballotPf}(x_{\mathsf{pk_T}}, r_m, v, x_{\mathsf{pk_R}}, r_{m'}, \pi_1(r_{cred})) |\!) \wedge w = \pi_2(r_{cred}) \\
\Phi^{UV} & \hat{=} & \mathsf{checkMixPair}(z_{\mathsf{mixPairPf}}, (\!| \pi_1(y_1), \pi_2(y_1) |\!), \dots, (\!| \pi_1(y_n), \pi_2(y_n) |\!), \\
 & & \qquad\qquad z_{\mathsf{bal},1}, \dots, z_{\mathsf{bal},n}) =_E \mathsf{true} \\
 & & \wedge \bigwedge_{i=1}^{n} \mathsf{dec}(z_{\mathsf{decKey},i}, \pi_1(z_{\mathsf{bal},i})) =_E v_i \\
 & & \wedge \bigwedge_{i=1}^{n} \mathsf{checkDecKeyPf}(x_{\mathsf{pk_T}}, \pi_1(z_{\mathsf{bal},i}), z_{\mathsf{decKey},i}, z_{\mathsf{decPf},i}) =_E \mathsf{true} \\
\Phi^{EV} & \hat{=} & \bigwedge_{i=1}^{n} \mathsf{checkBallot}(\pi_3(y_i), \pi_1(y_i), \pi_2(y_i)) \\
 & & \wedge\ \mathsf{checkMixPair}(z_{\mathsf{mixPairPf}}, (\!| \pi_1(y_1), \pi_2(y_1) |\!), \dots, (\!| \pi_1(y_n), \pi_2(y_n) |\!), \\
 & & \qquad\qquad z_{\mathsf{bal},1}, \dots, z_{\mathsf{bal},n}) =_E \mathsf{true}
\end{array}
$$

$\wedge \bigwedge_{i=1}^{n} \mathsf{pet}(z_{\mathsf{petPf},i}, \pi_2(z_{\mathsf{bal},i}), \hat{z}_{\mathsf{cred},i}) =_E \mathsf{true}$
$\wedge \langle\!| z_{\mathsf{cred},1}, \ldots, z_{\mathsf{cred},n} |\!\rangle \simeq \langle\!| \hat{z}_{\mathsf{cred},1}, \ldots, \hat{z}_{\mathsf{cred},n} |\!\rangle$
$\wedge\ \mathsf{checkMix}(z_{\mathsf{mixPf}}, \mathsf{getmsg}(w_1), \ldots, \mathsf{getmsg}(w_n), z_{\mathsf{cred},1}, \ldots, z_{\mathsf{cred},n}) =_E \mathsf{true}$
$\wedge \bigwedge_{i=1}^{n} \mathsf{checksign}(x_{spk_R}, w_i)$

The test Φ^{IV} checks that the voter's ballot and public credential are recorded on the bulletin board. The test Φ^{UV} checks that the tally is correctly computed; that is, the mix is checked, the validity of decryption keys have been verified and the decrypted tally corresponds to the declared outcome. Finally, the test Φ^{EV} checks that only eligible ballots are considered; that is, ballots are correctly formed, mixes have been handled in suitable manner, PETs have been verified and only authentic public voter credentials are considered.

Theorem 3. *$\langle A_{\mathsf{jcj}}, V_{\mathsf{jcj}}\rangle$ satisfies election verifiability.*

Proof. Suppose $n \in \mathbb{N}$ and the tests $\Phi^{IV}, \Phi^{UV}, \Phi^{EV}$ are given above. We will now show that for all names $\tilde{s} = (s_1, \ldots, s_n)$ that the conditions of Definition 7 hold.

(1) Suppose C is a context, B is a process and i, j are integers such that $C[\mathsf{VP}_{\mathsf{n}}^{+}(s_1, \ldots, s_n)] \Longrightarrow B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$ and $\Phi^{IV}\{^{s_i}/_{v}, ^{\tilde{r}_i}/_{\tilde{r}}\}\sigma \wedge \Phi^{IV}\{^{s_j}/_{v}, ^{\tilde{r}_j}/_{\tilde{r}}\}\sigma$. It follows that $\pi_1(y)\sigma =_E \mathsf{penc}(x_{\mathsf{pk_T}}, r_{m,i}, s_i)\sigma =_E \mathsf{penc}(x_{\mathsf{pk_T}}, r_{m,j}, s_j)\sigma$ and by inspection of the equational theory it is the case that $r_{m,i}\sigma = r_{m,j}\sigma$. Since the record variables $r_{m,i}$, $r_{m,j}$ are handles for fresh nonces created by name restriction in the voter process it follows immediately from $r_{m,i}\sigma = r_{m,j}\sigma$ that $i = j$.

(2) We prove a stronger result, namely for any σ the condition holds. Suppose $\Phi^{UV}\sigma \wedge \Phi^{UV}\{^{\tilde{v}'}/_{\tilde{v}}\}\sigma$ and hence

$$\bigwedge_{i=1}^{n} \mathsf{dec}(z_{\mathsf{decKey},i}, \pi_1(z_{\mathsf{bal},i}))\sigma =_E v_i\sigma =_E v_i'\sigma.$$

It follows immediately that $\tilde{v}\sigma =_E \tilde{v}'\sigma$.

(3) Again, we will show that the condition holds for all substitutions σ. Suppose $\Phi^{IV}\{^{s_i}/_{v}, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_{y}\}\sigma$ holds for $1 \le i \le n$ and hence

$$\bigwedge_{1\le i\le n} \pi_1(y_i)\sigma =_E \mathsf{penc}(x_{\mathsf{pk_T}}, r_{m,i}, s_i)\sigma.$$

Moreover suppose $\Phi^{UV}\sigma$ holds and therefore

$$\mathsf{checkMixPair}(z_{\mathsf{mixPairPf}}, \langle\!| \pi_1(y_1), \pi_2(y_1) |\!\rangle, \ldots, \langle\!| \pi_1(y_n), \pi_2(y_n) |\!\rangle, z_{\mathsf{bal},1}, \ldots, z_{\mathsf{bal},n})\sigma =_E \mathsf{true}$$

holds. By inspection of the equational theory we have

$$\pi_1(z_{\mathsf{bal},i})\sigma =_E \mathsf{penc}(x_{\mathsf{pk_T}}, f(r_{m,\chi(i)}, R_i), s_{\chi(i)})\sigma$$

for some permutation χ defined over $\{1,\dots,n\}$ and terms $R_1,\dots,R_n$ (note $R_1,\dots,R_n$ appear in $z_{\mathsf{mixPairPf}}\sigma$). By our hypothesis, we also have for all $1 \le i \le n$ that

$$\mathsf{checkDecKeyPf}(x_{\mathsf{pk_T}}, \pi_1(z_{\mathsf{bal},i}), z_{\mathsf{decKey},i}, z_{\mathsf{decPf},i})\sigma =_E \mathsf{true}$$

and hence $z_{\mathsf{decKey},i}\sigma$ is a decryption key for $\pi_1(z_{\mathsf{bal},i})\sigma$. It follows that

$$\bigwedge_{1\le i\le n} \mathsf{dec}(z_{\mathsf{decKey},i}, \pi_1(z_{\mathsf{bal},i}))\sigma =_E s_{\chi(i)}$$

Finally, by hypothesis, we also have

$$\bigwedge_{1\le i\le n} \mathsf{dec}(z_{\mathsf{decKey},i}, \pi_1(z_{\mathsf{bal},i}))\sigma =_E v_i\sigma$$

and hence it follows that $\tilde{s} \simeq \tilde{v}$.

(4) We prove a stronger result, namely Condition 8 below.

(5) Suppose C is a context and B is a process such that $C[\mathsf{VP}^+_\mathsf{n}(s_1,\dots,s_n)] \Longrightarrow B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$, and $\Phi^{EV}\sigma \wedge \Phi^{EV}\{^{x'}/_x \mid x \in X\backslash\tilde{y}\}\sigma$. We have for all $1 \le i \le n$ that $\mathsf{checkBallot}(\pi_3(y_i), \pi_1(y_i), \pi_2(y_i))\sigma =_E \mathsf{true}$ and it follows by inspection of the equational theory that

$$\pi_2(y_i)\sigma =_E \mathsf{penc}(K_i, S_i, M_i)$$

for some terms K_i, S_i, M_i. Since $\mathsf{checkMixPair}(z_{\mathsf{mixPairPf}}, \llparenthesis\pi_1(y_1), \pi_2(y_1)\rrparenthesis, \dots, \llparenthesis\pi_1(y_n), \pi_2(y_n)\rrparenthesis, z_{\mathsf{bal},1}, \dots, z_{\mathsf{bal},n})\sigma =_E \mathsf{true}$ and $\mathsf{checkMixPair}(z_{\mathsf{mixPairPf}}{}', \llparenthesis\pi_1(y_1), \pi_2(y_1)\rrparenthesis, \dots, \llparenthesis\pi_1(y_n), \pi_2(y_n)\rrparenthesis, z'_{\mathsf{bal},1}, \dots, z'_{\mathsf{bal},n})\sigma =_E \mathsf{true}$, it follows by Lemma 3 and transitivity of $\stackrel{\bullet}{\simeq}$ that

$$(\pi_2(z_{\mathsf{bal},1}), \dots, \pi_2(z_{\mathsf{bal},n}))\sigma \stackrel{\bullet}{\simeq} (\pi_2(z'_{\mathsf{bal},1}), \dots, \pi_2(z'_{\mathsf{bal},n}))\sigma.$$

Moreover, we have for all $1 \le i \le n$ that $\mathsf{pet}(z_{\mathsf{petPf},i}, \pi_2(z_{\mathsf{bal},i}), \hat{z}_{\mathsf{cred},i})\sigma =_E \mathsf{true}$ and $\mathsf{pet}(z'_{\mathsf{petPf},i}, \pi_2(z'_{\mathsf{bal},i}), \hat{z}'_{\mathsf{cred},i})\sigma =_E \mathsf{true}$; by Lemma 1 it follows that

$$(\hat{z}_{\mathsf{cred},1}, \dots, \hat{z}_{\mathsf{cred},n})\sigma \stackrel{\bullet}{\simeq} (\hat{z}'_{\mathsf{cred},1}, \dots, \hat{z}'_{\mathsf{cred},n})\sigma.$$

We have $\llparenthesis z_{\mathsf{cred},1}, \dots, z_{\mathsf{cred},n}\rrparenthesis\sigma \simeq \llparenthesis \hat{z}_{\mathsf{cred},1}, \dots, \hat{z}_{\mathsf{cred},n}\rrparenthesis\sigma$, $\llparenthesis z'_{\mathsf{cred},1}, \dots, z'_{\mathsf{cred},n}\rrparenthesis\sigma \simeq \llparenthesis \hat{z}'_{\mathsf{cred},1}, \dots, \hat{z}'_{\mathsf{cred},n}\rrparenthesis\sigma$ and hence we trivially derive

$$(z_{\mathsf{cred},1}, \dots, z_{\mathsf{cred},n})\sigma \stackrel{\bullet}{\simeq} (z'_{\mathsf{cred},1}, \dots, z'_{\mathsf{cred},n})\sigma.$$

Since $\mathsf{checkMix}(z_{\mathsf{mixPf}}, \mathsf{getmsg}(w_1), \dots, \mathsf{getmsg}(w_n), z_{\mathsf{cred},1}, \dots, z_{\mathsf{cred},n})\sigma =_E \mathsf{true}$ and $\mathsf{checkMix}(z_{\mathsf{mixPf}}{}', \mathsf{getmsg}(w'_1), \dots, \mathsf{getmsg}(w'_n), z'_{\mathsf{cred},1}, \dots, z'_{\mathsf{cred},n})\sigma =_E \mathsf{true}$; it follows by Lemma 2 that

$$(\mathsf{getmsg}(w_1), \dots, \mathsf{getmsg}(w_n))\sigma \stackrel{\bullet}{\simeq} (\mathsf{getmsg}(w'_1), \dots, \mathsf{getmsg}(w'_n))\sigma.$$

We have for all $1 \leq i \leq n$ that $\mathsf{checksign}(x_{spk_R}, w_i)\sigma =_E \mathsf{true}$ and $\mathsf{checksign}(x_{spk_R}, w'_i)\sigma =_E \mathsf{true}$ where $x_{spk_R}\sigma = \mathsf{pk}(ssk_R)$ and $ssk_R \in \tilde{n}$. By inspection of the equational theory it is the case that $w_i\sigma =_E \mathsf{sign}(ssk_R, M_i)\sigma$ and $w'_i\sigma =_E \mathsf{sign}(ssk_R, M'_i)\sigma$ for some terms M_i, M'_i. Furthermore, since for all $1 \leq i \leq n$ we have $\mathsf{getmsg}(w_i)\sigma =_E M_i$, $\mathsf{getmsg}(w'_i)\sigma =_E M'_i$ and because $(\mathsf{getmsg}(w_1), \ldots, \mathsf{getmsg}(w_n))\sigma \overset{\bullet}{\simeq} (\mathsf{getmsg}(w'_1), \ldots, \mathsf{getmsg}(w'_n))\sigma$, it follows that $\tilde{M} \overset{\bullet}{\simeq} \tilde{M}'$. Now, since the signing key is under restriction, and by inspection of the voting process and its possible outputs, it follows that for all $1 \leq i \leq n$ we have

$$\begin{aligned} &\mathsf{getmsg}(w_i)\sigma =_E \mathsf{penc}(\mathsf{pk}(sk_R), m''_{\chi(i)}, d_{\chi(i)}) \\ &\mathsf{getmsg}(w'_i)\sigma =_E \mathsf{penc}(\mathsf{pk}(sk_R), m''_{\chi'(i)}, d_{\chi'(i)}) \end{aligned}$$

where d_i, m''_i are names under restriction in the registrar process R, $x_{pk_R}\sigma =_E \mathsf{pk}(sk_R)$ and χ, χ' are permutations defined over $\{1, \ldots, n\}$. Finally we conclude $\tilde{w}\sigma \simeq \tilde{w}'\sigma$.

(6) Suppose C is a context and B is a process such that $C[\mathsf{VP}^+_\mathsf{n}(s_1, \ldots, s_n)] \Longrightarrow B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$, and $\bigwedge_{1 \leq i \leq n} \Phi^{IV}_i\sigma \wedge \Phi^{EV}\{^{\tilde{w}'}/_{\tilde{w}}\}\sigma$ holds. We have for all $1 \leq i \leq n$ that $w_i\sigma = \pi_2(r_{cred_i})\sigma$ and by inspection of the voting process we have

$$\begin{aligned} \tilde{w}\sigma =_E (&\mathsf{sign}(ssk_R, \mathsf{penc}(\mathsf{pk}(sk_R), m''_1, d_1)), \\ &\ldots, \mathsf{sign}(ssk_R, \mathsf{penc}(\mathsf{pk}(sk_R), m''_n, d_n))). \end{aligned}$$

In addition we have $\pi_2(y_i)\sigma =_E \mathsf{penc}(\mathsf{pk}(sk_R), m'_i, d_i)$ for all $1 \leq i \leq n$ and by similar reasoning to the above (see Condition 5.1) we derive $\tilde{w}\sigma \simeq \tilde{w}'\sigma$.

(7) Suppose C is a context and B is a process such that $C[\mathsf{VP}^+_\mathsf{n}(s_1, \ldots, s_n)] \Longrightarrow B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$, and $\Phi^{EV}\sigma \wedge \Phi^{EV}\{^{x'}/_x \mid x \in X \backslash \tilde{w}\}\sigma$. We have for all $1 \leq i \leq n$ that $\mathsf{checksign}(x_{spk_R}, w_i)\sigma =_E \mathsf{true}$ where $x_{spk_R}\sigma = \mathsf{pk}(ssk_R)$ and $ssk_R \in \tilde{n}$. By inspection of the equational theory it is the case that

$$w_i\sigma =_E \mathsf{sign}(ssk_R, M_i)\sigma$$

for some term M_i. Since the signing key is under restriction, and by inspection of the voting process, it follows that for all $1 \leq i \leq n$ we have

$$M_i =_E \mathsf{penc}(\mathsf{pk}(sk_R), m''_i, d_i)$$

where d_i, m''_i are names under restriction in the registrar process R and $x_{pk_R}\sigma =_E \mathsf{pk}(sk_R)$. Since $\mathsf{checkMix}(z_{\mathsf{mixPf}}, \mathsf{getmsg}(w_1), \ldots, \mathsf{getmsg}(w_n), z_{\mathsf{cred},1}, \ldots, z_{\mathsf{cred},n})\sigma =_E \mathsf{true}$, $\mathsf{checkMix}(z_{\mathsf{mixPf}}', \mathsf{getmsg}(w_1), \ldots, \mathsf{getmsg}(w_n), z'_{\mathsf{cred},1}, \ldots, z'_{\mathsf{cred},n})\sigma =_E \mathsf{true}$ and for all $1 \leq i \leq n$ we have $\mathsf{getmsg}(w_i)\sigma =_E M_i$ it follows that

$$(z_{\mathsf{cred},1}, \ldots, z_{\mathsf{cred},n})\sigma \overset{\bullet}{\simeq} (z'_{\mathsf{cred},1}, \ldots, z'_{\mathsf{cred},n})\sigma$$

by Lemma 2. We have $(\!|z_{\mathsf{cred},1},\dots,z_{\mathsf{cred},n}|\!)\sigma \simeq (\!|\hat{z}_{\mathsf{cred},1},\dots,\hat{z}_{\mathsf{cred},n}|\!)\sigma$ and $(\!|z'_{\mathsf{cred},1},\dots,z'_{\mathsf{cred},n}|\!)\sigma \simeq (\!|\hat{z}'_{\mathsf{cred},1},\dots,\hat{z}'_{\mathsf{cred},n}|\!)\sigma$; it trivially follows that

$$(\hat{z}_{\mathsf{cred},1},\dots,\hat{z}_{\mathsf{cred},n})\sigma \overset{\bullet}{\simeq} (\hat{z}'_{\mathsf{cred},1},\dots,\hat{z}'_{\mathsf{cred},n})\sigma.$$

Moreover, we have for all $1 \le i \le n$ that $\mathsf{pet}(z_{\mathsf{petPf},i},\pi_2(z_{\mathsf{bal},i}),\hat{z}_{\mathsf{cred},i})\sigma =_E \mathsf{true}$ and $\mathsf{pet}(z'_{\mathsf{petPf},i},\pi_2(z'_{\mathsf{bal},i}),\hat{z}'_{\mathsf{cred},i})\sigma =_E \mathsf{true}$; hence by Lemma 1 it follows that

$$(\pi_2(z_{\mathsf{bal},1}),\dots,\pi_2(z_{\mathsf{bal},n}))\sigma \overset{\bullet}{\simeq} (\pi_2(z'_{\mathsf{bal},1}),\dots,\pi_2(z'_{\mathsf{bal},n}))\sigma.$$

By $\mathsf{checkMixPair}(z_{\mathsf{mixPairPf}},(\!|\pi_1(y_1),\pi_2(y_1)|\!),\dots,(\!|\pi_1(y_n),\pi_2(y_n)|\!),z_{\mathsf{bal},1},\dots,z_{\mathsf{bal},n})\sigma =_E \mathsf{true}$, $\mathsf{checkMixPair}(z_{\mathsf{mixPairPf}}',(\!|\pi_1(y'_1),\pi_2(y'_1)|\!),\dots,(\!|\pi_1(y'_n),\pi_2(y'_n)|\!),z'_{\mathsf{bal},1},\dots,z'_{\mathsf{bal},n})\sigma =_E \mathsf{true}$ and Lemma 3 we have

$$(\pi_2(y_1),\dots,\pi_2(y_n))\sigma \overset{\bullet}{\simeq} (\pi_2(y'_1),\dots,\pi_2(y'_n))\sigma.$$

We have for all $1 \le i \le n$ that $\mathsf{checkBallot}(\pi_3(y_i),\pi_1(y_i),\pi_2(y_i))\sigma =_E \mathsf{true}$ and $\mathsf{checkBallot}(\pi_3(y'_i),\pi_1(y'_i),\pi_2(y'_i))\sigma =_E \mathsf{true}$. By inspection of the equational theory and because $(\pi_2(y_1),\dots,\pi_2(y_n))\sigma \overset{\bullet}{\simeq} (\pi_2(y'_1),\dots,\pi_2(y'_n))\sigma \overset{\bullet}{\simeq} (\mathsf{penc}(\mathsf{pk}(sk_R),m''_1,d_1),\dots,\mathsf{penc}(\mathsf{pk}(sk_R),m''_n,d_n))$ it is the case that

$$\begin{aligned}\pi_3(y_i)\sigma &=_E \mathsf{ballotPf}(PK_{T_i},R_i,N_i,\mathsf{pk}(sk_R),S_i,d_{\chi(i)})\\ \pi_3(y'_i)\sigma &=_E \mathsf{ballotPf}(PK'_{T_i},R'_i,N'_i,\mathsf{pk}(sk_R),S'_i,d_{\chi'(i)})\end{aligned}$$

for some terms $PK_{T_i},R_i,N_i,S_i,PK'_{T_i},R'_i,N'_i,S'_i$ and permutations χ,χ' defined over $\{1,\dots,n\}$. Since for all $1 \le i \le n$ the name d_i is under restriction in the voting process specification, it follows that

$$\begin{aligned}\pi_3(y_i)\sigma &=_E \mathsf{ballotPf}(\mathsf{pk}(sk_T),m_{\chi(i)},s_{\chi(i)},\mathsf{pk}(sk_R),m'_{\chi(i)},d_{\chi(i)})\\ \pi_3(y'_i)\sigma &=_E \mathsf{ballotPf}(\mathsf{pk}(sk_T),m_{\chi'(i)},s_{\chi'(i)},\mathsf{pk}(sk_R),m'_{\chi'(i)},d_{\chi'(i)})\end{aligned}$$

(that is, $\pi_3(y_i)\sigma$, $\pi_3(y'_i)\sigma$ are the zero-knowledge proofs output by the voters) and moreover by the validity of the proof, we have

$$\begin{aligned}\pi_1(y_i)\sigma &=_E \mathsf{penc}(\mathsf{pk}(sk_T),m_{\chi(i)},s_{\chi(i)})\\ \pi_1(y'_i)\sigma &=_E \mathsf{penc}(\mathsf{pk}(sk_T),m_{\chi'(i)},s_{\chi'(i)})\end{aligned}$$

$$\begin{aligned}\pi_2(y_i)\sigma &=_E \mathsf{penc}(\mathsf{pk}(sk_R),m'_{\chi(i)},d_{\chi(i)})\\ \pi_2(y'_i)\sigma &=_E \mathsf{penc}(\mathsf{pk}(sk_R),m'_{\chi'(i)},d_{\chi'(i)})\end{aligned}$$

Finally we conclude $\tilde{y}\sigma \simeq \tilde{y}'\sigma$. (Formally we should also show that $|\tilde{y}| = |\tilde{y}'|$. We omitted this detail from our test Φ^{EV} for simplicity, however, in this instance it could be incorporated with the additional conjunct $y = (\!|\pi_1(y),\pi_2(y),\pi_3(y)|\!)$.)

(8) This can be witnessed by modelling the complete JCJ-Civitas protocol as the context $C[\_]$. □

6 Conclusion

We present a symbolic definition of election verifiability which allows us to precisely identify which parts of a voting system need to be trusted for verifiability. The suitability of systems can then be evaluated and compared on the basis of trust assumptions. We also consider eligibility verifiability, an aspect of verifiability that is often neglected and satisfied by only a few protocols, but nonetheless an essential mechanism to detect ballot stuffing. We have applied our definition to three protocols: FOO, which uses blind signatures; Helios 2.0, which is based on homomorphic encryption, and JCJ-Civitas, which uses mixnets and anonymous credentials. For each of these protocols we discuss the trust assumptions that a voter or an observer needs to make for the protocol to be verifiable. Since Helios 2.0 and JCJ-Civitas have been implemented and deployed, we believe our formalisation is suitable for analysing real world election systems.

Acknowledgements

We are particularly grateful to Michael Clarkson for careful reading of an earlier draft, and for his perceptive questions and comments.

References

[1] M. Abadi and C. Fournet. Mobile values, new names, and secure communication. In *POPL'01: Proceedings of the 28th ACM Symposium on Principles of Programming Languages*, pages 104–115, New York, USA, 2001. ACM.

[2] B. Adida. *Advances in Cryptographic Voting Systems.* PhD thesis, MIT, 2006.

[3] B. Adida. Helios: Web-based open-audit voting. In *Proceedings of the Seventeenth Usenix Security Symposium*, pages 335–348. USENIX Association, 2008.

[4] B. Adida, O. de Marneffe, O. Pereira, and J.-J. Quisquater. Electing a university president using open-audit voting: Analysis of real-world use of Helios. In *Electronic Voting Technology/Workshop on Trustworthy Elections (EVT/WOTE)*, 2009.

[5] R. Anderson and R. Needham. Programming Satan's Computer. In Jan van Leeuwen, editor, *Computer Science Today: Recent Trends and Developments*, volume 1000 of *LNCS*, pages 426–440. Springer, 1995.

[6] M. Backes, C. Hritcu, and M. Maffei. Automated verification of remote electronic voting protocols in the applied pi-calculus. In *CSF'08: Proceed-*

ings of the 21st IEEE Computer Security Foundations Symposium, pages 195–209, Washington, USA, 2008. IEEE.

[7] A. Baskar, R. Ramanujam, and S. P. Suresh. Knowledge-based modelling of voting protocols. In *TARK'07: Proceedings of the 11th International Conference on Theoretical Aspects of Rationality and Knowledge*, pages 62–71, New York, USA, 2007. ACM.

[8] D. Bowen. Secretary of State Debra Bowen Moves to Strengthen Voter Confidence in Election Security Following Top-to-Bottom Review of Voting Systems. California Secretary of State, press release DB07:042 `http://www.sos.ca.gov/elections/voting_systems/ttbr/db07_042_ttbr_system_decisions_release.pdf`, August 2007.

[9] Bundesverfassungsgericht (Germany's Federal Constitutional Court). Use of voting computers in 2005 Bundestag election unconstitutional. Press release 19/2009 `http://www.bundesverfassungsgericht.de/en/press/bvg09-019en.html`, March 2009.

[10] D. Chaum, P. Y. A. Ryan, and S. Schneider. A practical, voter-verifiable election scheme. In *Proc. 10th European Symposium On Research In Computer Security (ESORICS'05)*, volume 3679 of *Lecture Notes in Computer Science*, pages 118–139. Springer, 2005.

[11] B. Chevallier-Mames, P.-A. Fouque, D. Pointcheval, J. Stern, and J. Traore. On Some Incompatible Properties of Voting Schemes. In *WOTE'06: Proceedings of the International Association for Voting Systems Sciences Workshop on Trustworthy Elections*, 2006.

[12] M. R. Clarkson, S. Chong, and A. C. Myers. Civitas: Toward a secure voting system. Technical Report 2007-2081, Cornell University, May 2007. Revised March 2008. `http://hdl.handle.net/1813/7875`.

[13] M. R. Clarkson, S. Chong, and A. C. Myers. Civitas: Toward a secure voting system. In *S&P'08: Proceedings of the 2008 IEEE Symposium on Security and Privacy*, pages 354–368, Washington, DC, USA, 2008. IEEE Computer Society.

[14] S. Delaune, S. Kremer, and M. D. Ryan. Verifying privacy-type properties of electronic voting protocols. *Journal of Computer Security*, 2009. To appear.

[15] A. Fujioka, T. Okamoto, and K. Ohta. A Practical Secret Voting Scheme for Large Scale Elections. In *ASIACRYPT'92: Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques*, pages 244–251, London, 1992. Springer.

[16] M. Jakobsson and A. Juels. Mix and match: Secure function evaluation via ciphertexts. In *ASIACRYPT '00: Proceedings of the 6th International*

Conference on the Theory and Application of Cryptology and Information Security, pages 162–177, London, UK, 2000. Springer.

[17] A. Juels, D. Catalano, and M. Jakobsson. Coercion-Resistant Electronic Elections. Cryptology ePrint Archive, Report 2002/165, 2002.

[18] A. Juels, D. Catalano, and M. Jakobsson. Coercion-resistant electronic elections. In *WPES '05: Proceedings of the 2005 ACM workshop on Privacy in the electronic society*, pages 61–70, New York, NY, USA, 2005. ACM. See also `http://www.rsa.com/rsalabs/node.asp?id=2860`.

[19] Ministerie van Binnenlandse Zaken en Koninkrijksrelaties (Netherland's Ministry of the Interior and Kingdom Relations). Stemmen met potlood en papier (Voting with pencil and paper). Press release `http://www.minbzk.nl/onderwerpen/grondwet-en/verkiezingen/nieuws--en/112441/stemmen-met-potlood`, May 2008.

[20] Participants of the Dagstuhl Conference on Frontiers of E-Voting. Dagstuhl accord. `http://www.dagstuhlaccord.org/`, 2007.

[21] M. D. Ryan and B. Smyth. Applied pi calculus. In V. Cortier and S. Kremer, editors, *Formal Models and Techniques for Analyzing Security Protocols*, chapter 6. IOS Press, 2010. To appear.

[22] B. Smyth, M. D. Ryan, S. Kremer, and M. Kourjieh. Towards automatic analysis of election verifiability properties. In *Joint Workshop on Automated Reasoning for Security Protocol Analysis and Issues in the Theory of Security (ARSPA-WITS'10)*, Lecture Notes in Computer Science. Springer, 2010. To appear.

[23] M. Talbi, B. Morin, V. V. T. Tong, A. Bouhoula, and M. Mejri. Specification of Electronic Voting Protocol Properties Using ADM Logic: FOO Case Study. In *ICICS'08: Proceedings of the 10th International Conference on Information and Communications Security Conference*, pages 403–418, London, 2008. Springer.

[24] UK Electoral Commission. Key issues and conclusions: May 2007 electoral pilot schemes. `http://www.electoralcommission.org.uk/elections/pilots/May2007`.