Anonymity and CSP for Voting Systems

Murat Moran, James Heather, Steve Schneider
Department of Computing, Faculty of Engineering and Physical Science
University of Surrey
Guildford, United Kingdom
{m.moran, j.heather, s.schneider}@surrey.ac.uk

***Abstract*—In this paper we review a wide range of existing definitions for anonymity defined in different formal languages from the literature. Moreover, we discuss anonymity definitions for voting systems and model the specifications using Communicating Sequential Processes (CSP). In addition, we formalise conventional voting system with CSP and analyse whether our voting system design satisfies the anonymity requirement with two different definitions, *strong* and *weak* anonymity. Furthermore, we highlight the difference between strong and weak anonymity definitions regarding voting systems with a case study on conventional voting system. Finally, we conclude with the results taken from our model analysis, which expresses that the strong anonymity definition is too strong and the weak anonymity is more suitable specification for the voting processes.**

Keywords: anonymity, CSP, anonymous elections, conventional voting system, formal analysis, FDR2

I. Introduction

Whilst elections offer voters a chance to exercise their democratic rights, voters should also have the right to keep their vote secret. Some people may publicize their votes freely, however in some cases people might have to hide who they voted for, such as an employee when asked how she or he voted by their employers, who may fire them according to their political views. Thus, anonymity or in other words vote privacy is one of the most crucial requirements for voting systems. Many voting systems proposed over the last decades are claimed to have anonymity property sometimes without any proof but mostly with their own definitions of anonymity in a several different formal languages and proof methods, such as pi calculus, modular approach and epistemic logic. Recently, there have been quite a number of research that is trying to give precise formal definitions of desired properties of trustworthy voting systems. In this paper, we focus on anonymity definitions.

On the other hand, there have been significant improvements in trustworthy voting systems, which provide anonymity over decades, however there are many other countries, such as the United Kingdom, in which national and local elections are still run by using conventional voting system (CVS). It is the simple ballot-based voting system in which voters have to go to polling stations to vote using a booth and put their votes in a ballot box. We use CVS as a case study to test if our anonymity definition sounds.

We come up with the concise definition of the anonymity types namely; strong and weak anonymity after the discussion on previous definitions of anonymity using the process algebra CSP. We mainly focus on the work carried out by Schneider and Sidiropoulos [11], which is the only work that defines anonymity through formal analysis in CSP.

Before we give our anonymity definition, we present the notations of CSP language in Section II and we also present a variety of formal anonymity definitions from the literature regarding voting systems if possible in Section III. Moreover, in Section IV, we summarize the Schneider *et al.* definition of strong anonymity in CSP with a referendum scenario as an example protocol. Moreover, we define weak anonymity in terms of CSP for voting systems. In Section V we formalise our conventional voting system model and analyse it according to the anonymity definitions that we have given. Finally, in Section VI we conclude with a discussion on anonymity for voting systems and present future work.

II. CSP Notation

CSP is a formal language that is designed to describe concurrent systems in terms of components that interacts each other by means of message passing. It is a member of the process calculi family and was introduced by C.A.R Hoare in 1978 [5].

Systems in CSP are modelled in terms of *processes*, which interact with each other and the environment by means of synchronizations. The processes are defined in terms of a collection of *events* that the process can perform. In CSP an event can happen when all processes agree on executing it and the occurrence of an event should be regarded as an atomic action without time, which means it happens at the moment when it is inevitable. Thus, the set of all events that is visible is called Σ and the set of all internal events is written τ and the set of all events that the process P uses is called its *alphabet* and denoted αP. The simplest process is $STOP$, which fundamentally does nothing.

Given a process P and an event a in Σ, the process $a \rightarrow P$ is called *prefixing*, which is initially willing to perform an event a, therefore it waits until an a event happens and then behave like the process P.

CSP provides a couple of ways of describing "choice" events. One of them is named *guarded alternative* and denoted $'|'$ where distinct prefix events give options to the environment to choose any initial event and the behaviour of corresponding process. CSP also offers choice operations for the processes, which are called *external* and *internal* choice (nondeterministic choice) denoted as $\Box$ and $\sqcap$ respectively. Moreover, apart from

the conditional choice operator $if-then-else$, another choice operator is $b\&P$, which means that if the boolean condition b is true then behave as P else $STOP$.

Systems can be a collection of processes that run in parallel and synchronize with each other by means of events that they agree on. Thus, the processes $P \parallel Q$, which run in parallel have to synchronize on all common events. Moreover, operators that we frequently use in our analysis are: *hiding* denoted $' \setminus '$ and *renaming* shown as $[\![a/b]\!]$, i.e., the events a occurring in the process are replaced by the events b. Furthermore, we use traces in our refinement checks, which are finite or infinite sequences of the events that a process may perform and denoted $\langle\rangle$.

Finally, for the refinement checks we write that the process P is refined by the process Q if every behaviour of Q is also a behaviour of P and it is written $P \sqsubseteq Q$. That is, Q meets the specification P if Q is a refinement of P. In addition, we use Failures-Divergences Refinement (FDR2)[1] as model-checker software, which was designed by Formal Systems (Europe) Ltd to check formal models created with CSP formal language.

III. Previous work on anonymity definitions

The definitions of anonymity below are taken from literature so that we understand and analyse different notions of definitions and consider our definition of anonymity for voting systems and come up with the formal definition of it in CSP.

Definition 1 (Pfitzmann et al.): Anonymity is introduced in the paper [10] as a terminology to the literature. In a message sender-receiver settings, *anonymity is the state of being not identifiable within a set of subjects, the anonymity set, i.e. set of all possible subjects who might cause an action*. Moreover, the difference between anonymity and indistinguishability is expressed with the definition of indistinguishability, which is *the state of being indistinguishable from other elements of a set*. To summarize, if we consider sender anonymity as voter anonymity with respect to voter ID, and the messages being sent as votes, voter ID anonymity has the properties that a particular vote is not linkable to any voter, and that no vote is linkable to any voter ID.

Moreover, not being able to distinguish votes from a bunch of votes is indistinguishability. Furthermore, authors also describe *unobservability* as not being able to notice that a voter's voting within unobservability set, which is the set of voters.

Finally, the term pseudonymity is described as *the use of pseudonyms as IDs, which are identifiers of subjects*. Thus, we can consider a pseudonym as identifiers of the ballot papers, e.g. serial numbers, which can then be linked back to the voter IDs.

Definition 2 (Fournet and Abadi): Fournet and Abadi give a general privacy definition [4] in pi calculus with respect to private authentication protocols. In the decryption, observational equivalence is used to formalize, that is, given two user processes U_1 and U_2, they are indistinguishable in any context from the environment's point of view, which we call this kind of anonymity definition as *weak anonymity*.

Definition 3 (Mauw et al. [9]): Mauw *et al.* [9] gives a general description of anonymity based on Pfitzmann *et al.* in [10]. Their definition states anonymity to be in such a way that a coercer should not be able to distinguish the difference between these two behaviours in which a user u' is the user of the anonymity group of the user u, every behaviour of the system that can be attributed to u there is another system behaviour that can be attributed to u'.

Definition 4 (Shmatikov/Hughes): Hughes and Shmatikov gives a specification framework for anonymity and privacy based on a function view [6] in which system behaviour is described as a set of functions. Furthermore, the specification of the desired properties are defined in terms of observational equivalence using modular approach. In the paper, several forms of anonymity in terms of sender-receiver relation are described. We adopt some of those definitions, which are applicable to voting scenario;

1) *Absolute voter anonymity (strong anonymity)*: An attacker cannot tell anything about the voter's identity, as every voter is plausible for every observed vote. In this model, an attacker should not be able to link a pseudonym with a sender id, e.g., voter ID and serial number on ballot forms.
2) *Type-anonymity*: An attacker may learn the type of the voter, that is, in the case of a postal voting, if there is a small number of voters, who registered and cast their votes by postal voting, an attacker may reduce the number of the possible voters for a particular vote to a small group of voters, such as the voters using postal voters, or the voters going polling station to vote. Another example, if there is only two voters who signed up for the remote voting, then observing an instance of a remote voting is enough for the attacker to reduce the number of possible voters for this vote to 2.
3) *Session-level*: An attacker may know the entire set of voters and the votes, but unable to link the votes to the voters' identities. For instance, if an attacker is observing a polling station where only one vote has been cast, he may then deduce the voter's identity.
4) *Unobservability*: An attacker should not be able to identify that a particular voter has cast a vote, that is, a voting act should be unobservable.
5) *Untraceability (weak anonymity)*: An attacker or an observer should not be able to determine whether two votes have been cast by the same voter.

Definition 5 (Juels et al. [7]): Juels *et al.* describes anonymity as private election using provable security, which is defined in terms of an adversary who cannot interact with the voters during the election process. The description given by them is that the coercer/adversary cannot guess how a voter voted better than an adversarial algorithm whose only access is the final tally. Moreover, they also express that an election process is considered to be private even if all the

[1] An academic-use version of FDR2 is available for download from Formal Systems web page

voters vote for the same candidate.

Definition 6 (Kremer/Ryan [8] and Delaune et al. [2], [3]): These papers define privacy of the election adopting Fournet and Abadi's general privacy definition [4] in pi calculus to voting system protocols. Delaune *et al.* uses the term "vote privacy" as a synonym for anonymity and states that nobody has enough information to identify whether two voters swapped their votes. Therefore, if an observer cannot tell whether two arbitrary honest voters swapped their votes, then he cannot deduce information about how these voters cast their votes (weak anonymity). Moreover, Backes *et al.* [1] gives the same definition as Delaune *et al.* [2].

IV. Anonymity

In the CSP approach to anonymity, we consider identity of the agents performing events in the form of $x.i$, where i is the identity of the agent and x is the content of the event. Hence, the anonymity could be described as that a message could be originated from any other agents, i.e. an event $x.i$ could be in the form of $x.j$, where j is another agent in the group of USERS. Thus, the set of all the messages can be written as:

$$A = \{x.i | i \in USERS\}$$

Hence, if an observer has only access to the content of the message, that is if the identity of the agent is hidden from an observer, then the content could equally have been generated by any other agent from $USERS$.

Definition 7 (Strong Anonymity): A process P is strongly anonymous if:

$$f_A^{-1}(f_A(P)) = P \quad \text{with respect to traces, and}$$

$$f_A(x) = \begin{cases} \alpha & \text{if} \quad x \in \text{A} \\ x & \text{if} \quad x \notin \text{A} \end{cases}$$

where $\alpha \notin \Sigma$

$f_A^{-1}(f_A(P)) = P$ means in CSP language that the two processes P and $f_A^{-1}(f_A(P))$ are indistinguishable in terms of the traces model, which means both processes have the same traces.

If we rename every event from the alphabet A to a dummy event α, then whenever an event α is possible, any event from the original process should have been possible. Thus, in the renamed process $f_A(P)$, if an event α is possible then, any event in A is also possible. The process $f_A^{-1}(f_A(P))$ makes all events from A available whenever α is available in the abstracted process $f_A(P)$.

Second anonymity definition, which is a weaker condition than strong anonymity is given as;

Definition 8 (Weak Anonymity): Suppose two honest voters are v_a and v_b and their votes are $vote.v_a.s_a.c_a$ and $vote.v_b.s_b.c_b$. Thus, the anonymity definition for a system *SYS* with all possible behaviours of the voters, which is all possible $a's$ and $b's$, is:

$$SYS \equiv_{\mathrm{T}} [\![^{v_a, v_b}/_{v_b, v_a}]\!] \Leftrightarrow \begin{cases} SYS \sqsubseteq SYS[\![^{v_a, v_b}/_{v_b, v_a}]\!] \\ \quad \text{and} \\ SYS \sqsupseteq SYS[\![^{v_a, v_b}/_{v_b, v_a}]\!] \end{cases}$$

For clarity of the strong anonymity definition and the abstraction methods, which we mentioned previously, we can give this simple example of referendum, where there are only two possible voters with the serial numbers $s1$ and $s2$, and only one of them votes for or against a referendum. But they are certain on how they would vote, i.e., the first voter with serial number $s1$ always says yes and the other with $s2$ always says no. Then we describe the process *Obs* as external choice:

$$\begin{aligned} Obs \mathrel{\widehat{=}} & \; vote.s1 \rightarrow yes \rightarrow STOP \\ & \Box \; vote.s2 \rightarrow no \rightarrow STOP \end{aligned}$$

Thus, if we consider the case $f_A^{-1}(f_A(Obs)) = Obs$, for the set $A = \{vote.s1, vote.s2\}$, we will see the equality is not satisfied.

$$\begin{aligned} f_A^{-1}(f_A(Obs)) = & \; vote.s1 \rightarrow yes \rightarrow STOP \\ & \Box \; vote.s2 \rightarrow no \rightarrow STOP \\ & \Box \; vote.s1 \rightarrow no \rightarrow STOP \\ & \Box \; vote.s2 \rightarrow yes \rightarrow STOP \end{aligned}$$

This implies that the equality does not hold, which means traces of the each side are different such as $\langle vote.s1, no \rangle$, which is not possible for the process Obs. As a result, the process Obs is not anonymous as the vote no can give enough information about the voter. However, if we are to hide H the set of events yes and no from the referendum observer, which means the observer does not have any direct information about their occurrence, then the new process:

$$\begin{aligned} Obs \setminus H \mathrel{\widehat{=}} & \; vote.s1 \rightarrow STOP \\ & \Box \; vote.s2 \rightarrow STOP \end{aligned}$$

In this case, observer is not able to differentiate between the occurrence of events as he cannot see the events yes and no so the process now provides anonymity.

In some cases, the observer can see the occurrence of events, but he could be unable to identify which event. For example, the votes can be cast in an envelope. The abstraction that we use here is *Renaming*. Therefore, the process is *Obs2*:

$$\begin{aligned} Obs2 \mathrel{\widehat{=}} & \; Obs[\![^{yes, no}/_{envelope, envelope}]\!] \\ \mathrel{\widehat{=}} & \; vote.s1 \rightarrow envelope \rightarrow STOP \\ & \Box \; vote.s2 \rightarrow envelope \rightarrow STOP \end{aligned}$$

Thus, in the case of renaming, the equality $f_A^{-1}(f_A(f_R(Obs)) = f_R(Obs) = Obs2$ meets for the set $A = \{vote.s1, vote.s2\}$ and R is the set of abstracted (renamed) events namely; yes and no.

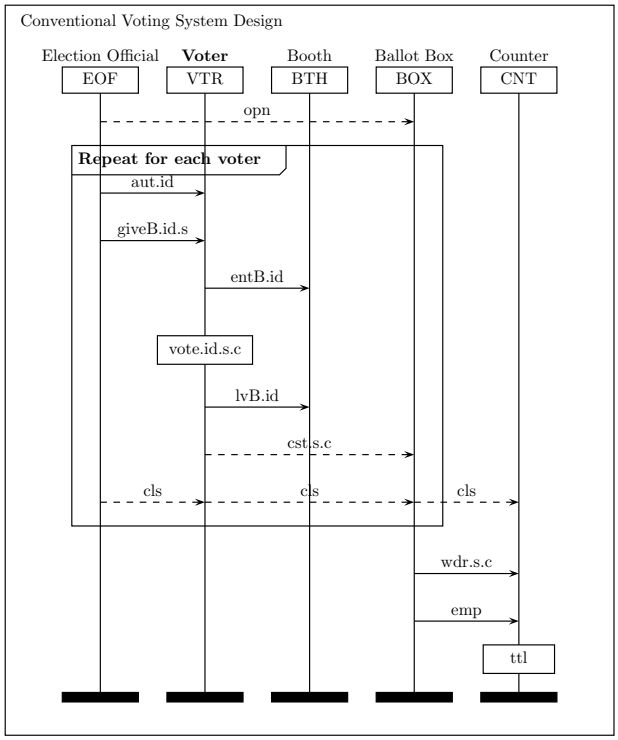

Figure 1. Conventional Voting System Design

Another abstraction method we may use is $masking$. In this abstraction method, events occurring during the protocol run could be from the voters we are interested in or could be from anywhere else from an observer point of view as the occurrence of all the other events acting as noise. Then, the process now can be written as parallel combination of Obs with $RUN(M)$ where M is the set of events to be abstracted, which are yes and no events. Therefore, the protocol $Obs \;|||\; RUN(M)$ can perform any event from the set M, thus the observer cannot tell the occurrence of such events due to Obs or $RUN(M)$. Thus, the new protocol where such events masked provides anonymity as $f_A^{-1}(f_A(Obs \;|||\; RUN(M)) = Obs \;|||\; RUN(M)$ with respect to traces.

V. Formalization and Anonymity Check of a Conventional Ballot-based Voting System

We have described anonymity requirement of a trustworthy election systems and anonymity has been defined in terms of CSP. The next step is to start anonymity specification check for our election system, which behaves as a conventional voting system and which we modelled in CSP. In our voting system model (see Figure 1 for event sharing), we describe each entity in terms of events that an election system can perform. Thus, we list each agent that we have in our model and which interact each other and describe them as follows;

A *Voter* goes to the polling station, authorizes herself to the electoral official, receives a ballot form with the serial number on it, goes in a booth, votes according to her preference with a pen or a stamp then leaves the booth, drops the ballot to the ballot box and leaves the polling station. Hence, the process with the sets serials, $\mathcal{S}$, candidates, $\mathcal{C}$ and voterids, $\mathcal{V}$;

$$VTR(id) \mathrel{\widehat{=}} aut.id \rightarrow \Box_{s \in \mathcal{S}} giveB.id.s \rightarrow entB!id \rightarrow \bigsqcap_{c \in \mathcal{C}} vote.id.s.c \rightarrow lvB!id \rightarrow cst.s.c \rightarrow cls \rightarrow VTR(id)$$

An *Election Official* working in a polling station authenticates the eligible voters with their identifications, and issues them ballot papers on which there is an arbitrary ballot serial number, which is different in each time and candidate lists in alphabetic order. In our simplistic model he can also open and close the election for a polling station. Hence;

$$EOF(ids, sns) \mathrel{\widehat{=}} opn \rightarrow OF(ids, sns)$$

$$OF(ids,sns) \mathrel{\widehat{=}} \Big(\Box_{id \in ids} aut.id \rightarrow \Big(\bigsqcap_{s \in sns} giveB.id.s \rightarrow OF(ids \backslash \{id\}), sns \backslash \{s\})\Big)\Big) \Box\; cls \rightarrow STOP$$

A $Booth$ is a private environment for the voters to be able to vote without any interaction with anyone else rather than a voter. Thus, in the model, the booth process can only allow one voter to go in, vote and leave at a time. Hence;

$$BTH \mathrel{\widehat{=}} entB?id \rightarrow lvB?id \rightarrow BTH$$

A $Ballot\ Box$ is a box where all cast votes are collected under the control of election official. We assume that there is a private untappable channel between a voter and a ballot box. In our model, a ballot box accepts the ballots from the voters and gathers them for the collection. Once the election is closed, the box can be opened and all the ballot papers can be withdrawn for the tallying. Hence;

$$BOX \mathrel{\widehat{=}} opn \rightarrow BOX'(\emptyset)$$

$$BOX'(vts) \mathrel{\widehat{=}} \Big(\Big(\Box_{\substack{s \in \mathcal{S} \\ c \in \mathcal{C}}} cst.s.c \rightarrow BOX'(vts \cup \{(s,c)\})\Big) \Box\; cls \rightarrow BOX''(vts)\Big)$$

$$BOX''(vts) \mathrel{\widehat{=}} \text{if } vts = \emptyset \text{ then } emp \rightarrow STOP \text{ else } \bigsqcap_{(s,c) \in vts} wdr.s.c \rightarrow BOX''(vts \backslash \{(s,c)\})$$

A $Counter$ is the official who withdraws all the ballots in the ballot box and counts them according to chosen candidates. Once, the ballot box is empty, he announces the total votes

that each candidate has. Hence;
$CNT \mathrel{\widehat{=}} cls \rightarrow CT(0,0,0)$

$$CT(i,j,k) \mathrel{\widehat{=}} \left(\begin{array}{l} \Box_{\substack{s\in\mathcal{S}\\ c\in\mathcal{C}}} \left(\begin{array}{l} wdr.s.c \rightarrow CT(i+1,j,k) \,\triangleleft c = c1 \triangleright\, STOP \\ \Box\ wdr.s.c \rightarrow CT(i,j+1,k) \,\triangleleft c = c2 \triangleright\, STOP \\ \Box\ wdr.s.c \rightarrow CT(i,j,k+1) \,\triangleleft c = c3 \triangleright\, STOP \end{array} \right) \\ \Box \\ emp \rightarrow ttl!c1!i \rightarrow ttl!c2!j \rightarrow ttl!c3!k \rightarrow SKIP \end{array} \right)$$

Hence, our conventional voting system model is a parallel composition of all the five process under correct alphabets. That is:

$$SYS = ((((VTRS \parallel EOF) \parallel BTH) \parallel BOX) \parallel CNT)$$

A. Strong Anonymity Analysis

We aim to understand the discussion about the anonymity definition and requirements made in [11]. Schneider states that different definitions are required for different situations depending on the requirements of the situation. For instance, in a voting system where the anonymity of the voter identity is required, the anonymity definition that we described previously is too strong. That is, different votes are required to be generated by different voters, which means the two votes are not entirely independent from each other. As a result, the anonymity check with their definition would fail in such systems, which requires each of the votes could independently have generated from the same voter. In our analysis, we proved that the definition is too strong for the conventional voting system that we modelled. As a result, the definition should be weakened for the voting scenario, which we see as a challenge for the rest of the research. We assume that there are at least two honest voters in case all the voters are dishonest but one in which case vote privacy could be revealed. In our model, we use three voters, candidates, and serials in order to restrict state exploration in FDR2 refinement checker, which cannot deal with millions of process states.

We approach anonymity in two scenarios, one from a counter point of view and the other from an election official point of view. Firstly, we decide what a counter can see and what should be hidden from him. Thus, a counter can observe occurrence of the events in his alphabet, which are denoted α*CNT*, and all the other events are hidden from the counter. Therefore, the strong anonymity for the vote events is checked by the trace refinement of the processes, *Cc1* and *Cc2*.

$$Cc1 \mathrel{\widehat{=}} SYS \setminus (\Sigma \backslash (\alpha CNT \cup \{|\, vote \,|\}))$$

$$Cc2 \mathrel{\widehat{=}} Cc1[\![^{vote.id.s.cnd1}/_{dmy}]\!][\![^{dmy}/_{vote.id2.j.cnd2}]\!]$$

We show that the assertion does not hold with an example trace $\langle vote.v3.s1.c3, vote.v2.s1.c1\rangle$ taken from FDR2, which means that the protocol is not strongly anonymous from a counter point of view as he has the knowledge of that the two votes with the same serial number cannot come up at the final tally proving that the definition is too strong for this particular situation.

Secondly, we analyse the strong anonymity of the system from an official point of view. Therefore, an official can only see what he is able to see which is his own alphabet, α*EOF* and we hide all the other events from him. Then, for the anonymity of the $vote$ events with the renaming function, we use the trace refinement of these two processes, i.e. *Ofc1* and *Ofc2*.

$$Ofc1 \mathrel{\widehat{=}} SYS \setminus (\Sigma \backslash (\alpha EOF \cup \{|\, vote \,|\}))$$

$$Ofc2 \mathrel{\widehat{=}} Ofc1[\![^{vote.id.s.cnd1}/_{dmy}]\!][\![^{dmy}/_{vote.id2.j.cnd2}]\!]$$

However, this process also does not meet the anonymity definition because of the trace;

$$\langle opn, aut.v3, giveB.v3.s2, vote.v2.s2.c1\rangle$$

This is a trace of the right hand side of the assertion but not a trace of the left hand side process violating the anonymity property in this case. The reason behind this is that the serial number $s2$ is given to $v3$, while in the final tally $v2$ votes with the same serial number. This behaviour is not allowed by the voting system.

As we see in both approach anonymity does not meet the strong anonymity definition in [11], which is too strong for the voting scenario. Hence, we need the weaker definition for anonymity, which we already defined it previously as *weak anonymity*.

B. Weak Anonymity Analysis

In our second analysis we use the weak anonymity definition formalised before. From the experimental results, we show that our conventional voting system model has the property of anonymity with respect to weak anonymity definition. We analyse the system comparing these two situations, the first one is in which the voters $v1$ and $v2$ vote however they like, and the second one is that they swap their votes.

Thus, in our conventional voting model, firstly from a counter point of view, the two systems *Cc3* and *Cc4* should not be distinguishable with the events that he can perform. In the normal system, we hide occurrence of all the other events from him rather than the events in his alphabet, α*CNT* and in the second system we swap the votes cast by $v1$ and $v2$. Thus, the two systems are defined as follows;

$$Cc3 \mathrel{\widehat{=}} SYS \setminus (\Sigma \backslash \alpha CNT)$$

$$Cc4 \mathrel{\widehat{=}} Cc3[\![^{vote.v1.s.c,\, vote.v2.s.c}/_{vote.v2.s.c,\, vote.v1.s.c}]\!]$$

Through our analysis of weak anonymity, we investigated whether these two systems are observationally equivalent from the counter point of view. We show that refinement checks, *Cc3* $\sqsubseteq_{\mathrm{T}}$ *Cc4* and *Cc4* $\sqsubseteq_{\mathrm{T}}$ *Cc3* hold, which means that the two systems refine each other, which concludes that the two systems are observationally equivalent with respect to traces and from the counter point of view. Therefore, the system has the weak anonymity with the observational equivalence *Cc3* $\equiv_{\mathrm{T}}$ *Cc4*.

Secondly, from an official point of view in the normal system, *Ofc3*, we hide the entire events occurrence from

him except the events he is capable of performing during a process which is the alphabet, α*EOF*. In addition, in the second system, *Ofc4* we swap the votes cast by $v1$ and $v2$. Thus, as in counter case, we show that the weak anonymity specification is satisfied by the conventional voting system with the observational equivalence *Ofc3* $\equiv_{\mathrm{T}}$ *Ofc4* and the system definitions;

$$\textit{Ofc3} \mathrel{\widehat{=}} \textit{SYS} \setminus (\Sigma \backslash \alpha \textit{EOF})$$

$$\textit{Ofc4} \mathrel{\widehat{=}} \textit{Ofc3}[\![^{vote.v1.s.c,\, vote.v2.s.c}/_{vote.v2.s.c,\, vote.v1.s.c}]\!]$$

VI. Discussion and Further Work

In this paper we have investigated strong and weak anonymity definitions, which other approaches seem to have similar characterization, for the voting systems using CSP language and FDR2 as a model checker. We have used conventional voting system as case study for our refinement checks. From our experiments and results, we conclude with the following discussion.

The first motivation of the paper was to explore why the strong anonymity is too strong for the voting protocols. Through our analysis we run a couple of experiments with the strong anonymity specification based on Schneider's definition, which we could not prove that our anonymous voting model is actually anonymous. The main reason for this is that intuitively a voting process should not allow two voters to use same serial number and cast their votes. Therefore, while specification requires that two voters should be independent, the system mandates that two arbitrary voters should be different, which means that the two instances of voters are not entirely independent.

The second question was whether the problem is the specification or the system. Normally specifications are correct and systems need to be changed, but in this case, it is the specification, which does not suit to voting systems. However, in our model check, specification allows some sort of events that system does not allow, which is to give two different voters the same serial number. As a result, we had to tweak our specification, which requires changing anonymity definition. Therefore, we used weak anonymity as specification, which states that a coercer should not be able to notice the difference between the normal system run and the system in which the voters swapped their activities or interactions with the system.

Further anonymity analysis will be conducted for trustworthy voting systems, which use cryptography as the notion for privacy and which are claimed to be anonymous. We will also investigate other trustworthy voting system requirements such as receipt-freeness, coercion-resistance and verifiability using CSP.

Acknowledgements

The authors would like to thank to David M. Williams for his technical discussion.

This work was carried out under the project Trustworthy Voting Systems (TVS) with support from EPSRC.

References

[1] Michael Backes, Catalin Hritcu, and Matteo Maffei. Automated verification of remote electronic voting protocols in the applied pi-calculus. In *CSF*, pages 195–209, 2008.

[2] Stéphanie Delaune, Steve Kremer, and Mark Ryan. Coercion-resistance and receipt-freeness in electronic voting. In *CSFW*, pages 28–42, 2006.

[3] Stéphanie Delaune, Steve Kremer, and Mark Ryan. Verifying privacy-type properties of electronic voting protocols: A taster. In *Towards Trustworthy Elections*, pages 289–309, 2010.

[4] Cédric Fournet and Martín Abadi. Hiding names: Private authentication in the applied pi calculus. In *ISSS*, pages 317–338, 2002.

[5] C. A. R. Hoare. Communicating sequential processes. *Commun. ACM*, 21:666–677, August 1978.

[6] Dominic Hughes and Vitaly Shmatikov. Information hiding, anonymity and privacy: a modular approach. *Journal of Computer Security*, 12(1):3–36, 2004.

[7] Ari Juels, Dario Catalano, and Markus Jakobsson. Coercion-resistant electronic elections. In *Proceedings of the 2005 ACM workshop on Privacy in the electronic society*, WPES '05, pages 61–70, New York, NY, USA, 2005. ACM.

[8] Steve Kremer and Mark Ryan. Analysis of an electronic voting protocol in the applied pi calculus. In *ESOP*, pages 186–200, 2005.

[9] Sjouke Mauw, Jan Verschuren, and Erik P. de Vink. A formalization of anonymity and onion routing. In *ESORICS*, pages 109–124, 2004.

[10] Andreas Pfitzmann and Marit Köhntopp. Anonymity, unobservability, and pseudonymity - a proposal for terminology. In *Workshop on Design Issues in Anonymity and Unobservability*, pages 1–9, 2000.

[11] Steve Schneider and Abraham Sidiropoulos. CSP and anonymity. In *ESORICS*, pages 198–218, 1996.