Verifying Privacy-Type Properties of an Electronic Voting Protocol that uses Identity Based Cryptography

Audrey C. Garais[1], Gabrielle Ira M. Tan[2]
Danny F. Wuysang[3], Henry N. Adorna[4]
Algorithms and Complexity Laboratory
Department of Computer Science
College of Engineering
University of the Philippines - Diliman
[1]garaisaudreyc@gmail.com, [2]gabrielleira.tan@gmail.com,
[3]dfwuysang@up.edu.ph, [4]ha@dcs.upd.edu.ph

ABSTRACT

The Philippines carried out their national elections last 2010 using electronic voting. One of the concerns for the election was the security of computerized elections. It is important that not only the technology is taken into consideration but also the protocol by which the voting is done. Securing the protocol secures that the votes are properly counted without any anomaly, just as it would be done manually in an ideal setting. In this paper, we verify the properties of an electronic voting protocol in applied pi calculus, as this field is an active research with its growing significance in society.

1. INTRODUCTION

Electronic voting has been introduced in numerous countries because it promises fast, convenient and secure voting. Many protocols have been developed specifically for electronic voting. A few examples are the FOO 92 scheme[6], Lee et al. protocol [11] and Juels et al. protocol[8]. Three main kinds of protocols can be found in the literature, these are classified according to the mechanism they employ. In blind signature schemes, an administrator blindly signs a message sent to the voter, the token. This token proves the voter's eligibility to vote. In schemes that use homomorphic encryption [4], the voter cooperates with the administrator in order to construct an encryption of her vote. The administrator then exploits homomorphic properties of the encryption algorithm to compute the encrypted tally directly from the encrypted votes. The third kind of scheme uses randomisation to remove the link between the voter and her vote. All of which have their own set of satisfied properties significant to secure the voting process. Some properties that security protocols may satisfy are: eligibility, fairness, individual verifiability, universal verifiability, vote-privacy, receipt-freeness and coercion - resistance. Of these properties, the last three [4] are broadly privacy-type properties since they guarantee that the link between the voter and her vote is not revealed by the protocol.

The research focuses on electronic voting using identity based cryptography [7] devised by Gallegos et al. The protocol can be summarized into four phases: Set-Up, Authentication, Voting and Counting. The Set-Up phase is generally for generating the private and public keys for the election. The Authentication phase ensures that a voter is a registered voter and is therefore allowed to participate in the election. Voting includes choosing a vote, blinding the vote and signing of the vote by the authorized election Entity and the voter, and verifying the said signatures. Lastly, the votes are counted during the Counting phase by a combining entity, and the tally is published. The authors of the protocol have claimed that the following properties are satisfied by the protocol: Privacy, Eligibility and Uniqueness, Uncoercibility, Transparency, Accuracy and Robustness. Among those, Privacy and Uncoercibility are the privacy-type properties that the researchers will verify using the applied pi-calculus.

The applied pi calculus and the formalisation of the privacy type properties only deal with the activities of the voter process and its interactions with other entities and the coercer. The method of proving involves the protocol and not any of its possible implementations. The paper will be focusing on the formal specification of the IBC protocol and verification of the privacy - type properties the authors claim.

The applied pi calculus is defined as a language for describing and analysing security protocols [12]. It provides an intuitive process syntax for detailing the actions of the participants in a protocol, emphasising their communication. It has been used in formalising and analyzing various security protocols in various areas. It is based on pi calculus but is easier to use. The applied pi calculus allows one to define less usual primitives (often used in electronic voting protocols) by means of an equational theory [4]. Using this, together with the semantics and equivalences of the calculus, protocols can be modeled and analysed formally.

The properties mentioned above have been formalised and verified for some protocols in the work of Delaune et al [4] using applied pi calculus. The literature gives a formalisation of the privacy-type properties that a cryptographic

protocol may possess and in effect, it provides the method of deriving the proof for the said properties. Vote-privacy, the weakest of the three, states that the way the voter voted is not revealed to anyone else. Receipt-freeness states that the voter does not receive a receipt to prove to an attacker that she voted in a certain way. Coercion-resistance states that an attacker cannot establish the connection between a voter and her vote even if the voter willingly cooperates with the attacker. Receipt-freeness implies vote-privacy and coercion-resistance implies receipt-freeness as shown in [3]. We now use the same method as in the literature to verify the properties of the protocol due to Gallegos-Garcia et al.

We first discuss the identity-based cryptography (IBC) protocol due to Gallegos-Garcia et al in Section 2, followed by a discussion of the applied pi-calculus in Section 3. We present to you our formalization of the IBC protocol in Section 4. Section 5 summarizes our verification and analysis. Lastly, we compare our results with the properties claimed by the authors of the IBC and layout further plans for our research in Section 6.

2. IDENTITY BASED CRYPTOGRAPHY PROTOCOL

The protocol involves voters, an authentication authority who verifies if the voters are eligible to vote, a combining entity who counts and publishes the votes, a storage device, and entities who may or may not be one of the aforementioned election officials. Entities sign the votes to encrypt them and decrypt portions of all votes to retrieve the plaintext after the voting phase. It relies on bilinear mappings and hash functions for security. Voters do not need public and private keys for encryption as this protocol involves identity based cryptography.

In the first phase the keys are generated and shared between all entities.

- A PKG sets up g_1, g_2, the private key of the protocol s_1, and the public key of the protocol P_{enc}.
- Each Entity E sends the PKG their ID_1. The PKG computes their public key and a share of the private key s_1 and sends these to the entities.
- PKG sets up a signing private key s_2 and a signing public key P_{sig}. Signing public and private keys are computed using the entities' ID_2 and are sent to the respective entities.

The second phase is the authentication phase.

- Voter V shows an identity card to the authentication authority to prove that he is a legitimate voter.
- V then selects a random ID_2 of any entity, this entity will be the one to sign her vote.

The third phase is the actual voting phase.

- V selects a vote v and a random number r_1 to compute $< U,\ W > = < mul(r_1,\ pointgen),\ xor(v,\ H_2(exp(\hat{e}(encpubkey,\ Epubkey),\ r_1))) >$ where mul is multiplication, pointgen is a point generator available to the public, $\hat{e}$ is a bilinear mapping, encpubkey and Epubkey are the public key of the entity and the public signing key respectively. This is the encrypted vote.
- Entity E chooses a random number r_2, and computes $X = mul(r_2,\ p)$ where p is the point generator. He sends this to V as a commitment.
- V chooses numbers a and b and computes $c = add(H_3(ev, add(add(mul(b, Esignkey), mul(a, p)), x), esigpkey), b)$. This c is then sent to E. $Esignkey$ is the public signing key of the entity and $esigpkey$ is the public signing key of the protocol.
- E computes $S = add(mul(c, aprivsignkey)$ where $aprivsignkey$ is his private signing key. He sends this S to V.
- V computes $(S', c') = (add(S, mul(a, esigpkey)), sub(c, b))$. This is now the signature.
- V sends $< U, W >$, (S', c') and the id of the entity who signed the vote to the storage device.
- V receives a hash with a timestamp from the storage device as a receipt.

The fourth phase is the counting phase.

- Combining Entity C verifies the signatures from the votes obtained from the storage device. If they are correct, U from $< U, W >$ are sent to all entities E.
- Every E computes a decryption share for each vote $\hat{e}(U, encskeyshare)$ where encskeyshare is a share of s_2. Shares are sent back to C.
- C recovers the plaintext v from the shares and publishes these along with the receipts.

The first and second phase must be completed before the voting phase. A synchronization between all election parties is needed. This ensures that all entities receive their respective keys before voters ask for their commitment to the votes. Another synchronization is needed after the voting phase and before the counting phase. This is to make sure that no partial results are released.

3. APPLIED PI CALCULUS

3.1 Syntax

$L, M, N, T, U, V ::=$	terms
$a, b, c, \ldots, k, m, n, \ldots, s$	name
x, y, z	variable
$g(M_1, \ldots, M_l)$	function application

To create a process in the calculus, a set of names, a set of variables and a signature Σ must be defined. Names may be identifiers of communication channels or other atomic

data. Σ consists of all function symbols which will be used to define terms. Typical function symbols deal with cryptographic primitives such as encrypt, decrypt and hash. Terms may be names, variables and function symbols applied to other terms. An equational theory (E) defines the relations between the functions. Equivalence is denoted by $=_E$. A typical example of an equational theory E used in cryptographic protocols is $dec(enc(x,k),k) = x$. $T_1 = enc(n,k)$ and $T_2 = dec(enc(enc(n,k),l),l)$ are equal, $T_1 =_E T_2$.

Plain processes and extended processes are used in the calculus. An extended process may be a plain process or several plain processes running in parallel.

The grammar for a plain process is as follows:

$P, Q, R ::=$	plain process
0	null process
$P \mid Q$	parallel composition
$!P$	replication
$\nu n.P$	name restriction
if $M = N$ then P else Q	conditional
$u(x).P$	message input
$\overline{u}(x).P$	message output

The null process does nothing. $P \mid Q$ is process P and Q running in parallel. This is used to represent participants in a protocol running in parallel. $!P$ is an infinite number of process P's running in parallel, used to simulate an unbounded number of sessions. $\nu n.P$ binds n to P, this is used for random numbers, keys and private channels. If $N = M$ then P else Q is standard, but the equivalence used is equivalence under an equational theory rather than strict algebraic equivalence.

The grammar for an extended process is:

$A, B, C ::=$	extended processes
P	plain process
$A \mid B$	parallel composition
$\nu n.A$	name restriction
$\nu x.A$	variable restriction
$\{^M/_x\}$	active substitution

Active substitutions generalise "let" statements. $\{^M/_x\}$ is a substitution that replaces the variable x with the term M whenever it is found in the process. Variables and names have scopes, $fv(A)$, $bv(A)$, $fn(A)$ and $bn(A)$ are the free variables, bound variables, free names and bound names of the process A, respectively. An extended process is closed if all its variables are either bound or in an active substitution.

The frame of process A (ϕ(A)) is obtained from A by replacing every process in A by a null process. It is the set of all active substitutions in A. The domain of a frame ψ, denoted by dom(ψ), is the set of variables for which ψ defines a substitution (those variables x for which ψ contains a substitution $\{^M/_x\}$ not under a restriction on x).

3.2 Semantics

Comm	$\overline{a}\langle x\rangle.P \mid a(x).Q$	$\rightarrow$	$P \mid Q$
Then	if $M = N$ then P else Q	$\rightarrow$	P
Else	if $M = N$ then P else Q	$\rightarrow$	Q

for ground terms M, N where $M \neq_E N$

Processes in the applied pi calculus have operational semantics that follow the structural rules that define two relations. One is structural equivalence which was explained in the Syntax discussion, and the other is the internal reduction, noted $\rightarrow$. Internal reduction $\rightarrow$ allows the transformation of a process such that the functions and primitives that were executed during the reduction may now be removed from the process, following the reduction rules above.

In $$a(x).P \xrightarrow{a(M)} P\{M/x\}$$

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

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

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

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

Struct $$\frac{A \equiv B \qquad B \xrightarrow{\alpha} B' \qquad B' \equiv A'}{A \xrightarrow{\alpha} A'}$$

Labelled reduction extends the operational semantics where the relation $\xrightarrow{\alpha}$ reduces the process where α is either an input or the output of a channel name or a variable of base type. This shows the interaction of the process with its environment, justifying the reductions done.

Static equivalence $\approx_s$ and labelled bisimilarity $\approx_\ell$ are important equivalence relations to be established between processes. Two extended processes A and B are said to be statically equivalent, denoted by $A \approx_s B$ if $\phi(A) \approx_s \phi(B)$.The largest symmetric relation on closed extended processes is labelled bisimilarity, which also implies static equivalence. Its definition is similar to the definition of bisimilarity, only that it requires static equivalence of the processes at each step of the reduction. Being the largest relation, it is crucial and helpful in formalising security properties.

3.3 Privacy-type Properties in Applied Pi Calculus

Voting protocols are modeled as closed plain processes using the applied pi calculus. The privacy-type properties are defined as well in terms of applied pi calculus as follows:

3.3.1 Vote-privacy

Definition 1:

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

As defined in [4], a voting protocol respects vote-privacy if the relation above holds for all possible votes a and b. This

means that if the voters swap their votes and the attacker is unable to identify that the swap occurred, then generally the attacker cannot possibly find out how each of the voters voted. If a protocol is said to be vote-private, then this relation should still hold even if the number of voters are increased and the votes are permuted.

3.3.2 Receipt-freeness

Definition 2:

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

Other definitions needed to arrive at the above relations can be found in [4]. Consider the case where the voter willingly cooperates with the coercer (attacker), who would want the voter to vote in a certain way. Receipt-freeness means that whether or not a voter cooperates with the coercer, the voter will not be able to prove to the coercer that she voted in a certain way through a receipt.

3.3.3 Coercion-resistance

Definition 3:

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

A protocol is coercion - resistant if a voter cannot cooperate with the coercer (attacker) to vote a certain way. It is very similar to the definition of receipt - freeness. The difference is, even if the attacker can send messages to the voter and receive messages from the voter, the voter still cannot prove that she voted in a certain way. This will be further explained in the verification section.

4. FORMAL SPECIFICATION

We model the protocol formally in applied pi so that we may be able to perfom formal verification on its properties. Using the syntax and semantics described in the previous section, we include in the model the functions and computations that are performed in the protocol. After modelling the protocol, a proof of correctness is done to show that the protocol was indeed modeled in applied pi calculus and still performs accordingly.

4.1 The Protocol in Applied Pi Calculus

$PKG = \nu g_1.\ \nu g_2.\ \nu pg.\ \nu s_1.$
$\{^{mul(s_1,pg)}/_{P_{enc}}\} \mid \{^{computeshare(pg,1)}/_{P^1_{enc}}\} \mid$
$out(ch,\ (g_1,\ g_2,\ pg,\ P_{enc},\ P^1_{enc})).\ in(ch1,\ Eid_1).$
$\{^{H_1(\ Eid_1\ )}/_{Q_{ide}}\} \mid \{^{computeshare(Q_{ide},1)}/_{D_{id1}}\} \mid$
$out(ch2, (Q_{ide}, D_{id1})).\ in(ch3, E_{id2}).\ \nu s_2.$
$\{^{mul(s_2,\ pg)}/_{P_{sig}}\} \mid out(ch, P_{sig}).\ \{^{H_1(E_{id2})}/_{Q_{ids}}\} \mid$
$\{^{mul(Q_{ids},\ S_2)}/_{S_{ids}}\} \mid out(ch4,\ (Q_{ids},\ S_{ids})).\ synch1$

$E_i = \nu id_1.\ \nu id_2.\ in(ch, msgE).\ out(ch1, id_1).\ in(ch2, m1).$
$\{^{msgE}/_{(a,b,p,encpkey,enckeyshare)}\} \mid$
$\{^{m1}/_{(pubkey,encskeyshare)}\} \mid \{^{\hat{e}(enckeyshare,pubkey)}/_{l}\} \mid$
$\{^{\hat{e}(p,encskeyshare)}/_{r}\} \mid if(l\ =\ r)\ then\ out(ch3,\ id_2).$
$in(ch,\ sigpkey).\ in(ch4,\ m2).\ synch1$
$in(che,\ id).\ \nu r_2.\ \{^{mul(r_2,p)}/_{commitment}\} \mid$
$out(ch5,\ commitment).\ in(ch6,\ cmsg).$
$\{^{m2}/_{(apubsignkey,aprivsignkey)}\} \mid$
$\{^{add(mul(cmsg,\ aprivsignkey),\ mul(r_2,\ sigpkey))}/_{smsg}\} \mid$
$out(ch7,\ smsg).\ synch2.\ in(che,\ uvote).$
$\{^{\hat{e}(uvote,\ encskeyshare)}/_{share}\} \mid out(ch, share).$
$synch3$

$V = \nu V_{id}.\ in(ch,\ msgV).\ in(ch,\ esigpkey).\ synch1.$
$out(cha,\ V_{id}).\ in(cha,\ entityid).\ \nu r_1.$
$\{^{msgV}/_{(d,\ f,\ pointgen,\ encpubkey,\ enckeyshare1)}\} \mid$
$\{^{mul(r_1,\ pointgen)}/_{u}\} \mid \{^{H_2(exp(\hat{e}(encpubkey,\ Epubkey),\ r_1))}/_{h}\} \mid$
$\{^{xor(v,h)}/_{w}\} \mid \{^{<u,\ w>}/_{ev}\}.\ out(che,\ entityid).$
$in(ch5,\ entitym).\ \nu a.\ \nu b.\ \{^{entitym}/_{(x,\ Esignkey)}\} \mid$
$\{^{add(mul(b,\ Esignkey),\ mul(a,\ p))}/_{p_1}\} \mid$
$\{^{add(H_3(ev,\ add(p_1,\ x),\ esigpkey),\ b)}/_{c}\} \mid$
$out(ch6,\ c).\ in(ch7,\ s).\ \{^{add(s,\ mul(a,\ esigpkey))}/_{sprime}\} \mid$
$\{^{sub(c,\ b)}/_{cprime}\} \mid \{^{(sprime,\ cprime)}/_{signature}\} \mid$
$\{^{(ev,\ signature,\ entityid)}/_{storagemsg}\} \mid$
$out(chs,\ storagemsg).\ in(chs,\ receipt).\ synch2$

$S = synch1.\ in(chs,\ m3).\ \{^{m3}/_{(encvote,sig,Evid_2)}\} \mid$
$\{^{stamp(\ m3\ )}/_{timestamp}\} \mid \{^{H_4(encvote,\ sig,\ timestamp)}/_{rcpt}\} \mid$
$out(chs,\ rcpt).\ synch2.\ \{^{(sp,\ cp)}/_{sig}\} \mid$
$\{^{(encvote,\ sp,\ cp,\ Evid_2,\ rcpt)}/_{msgtoc}\} \mid$
$out(chs,\ msgtoc).\ synch3$

$C = in(ch,\ pubparameters).\ in(ch,\ spk)\ .\ synch1.$
$synch2.\ in(chs,\ m).$
$\{^{pubparameters}/_{(cgrp_1,\ cgrp_2,\ pgen,\ epk,\ eks_1)}\} \mid$
$\{^{m}/_{(encryptedvote,\ sigs,\ sigc,\ ide,\ token)}\} \mid \{^{H_1(ide)}/_{pubskey}\} \mid$
$\{^{exp(H_3(encryptedvote,\ mul(\hat{e}(sigs,\ pgen),\ \hat{e}(pubskey,\ spk))),\ sigc)}/_{k}\} \mid$
$if(sigc = k)\ then\ \{^{ecryptedvote}/_{<u,\ w>}\} \mid$
$out(che,\ u).\ in(ch,\ shares).$
$\{^{combine(shares)}/_{g}\} \mid \{^{xor(w,\ H_2(g))}/_{votetxt}\} \mid$
$synch3.\ out(ch,\ (votetxt,\ token)).$

$A = \nu\ voterList\ .\ in(ch,\ msgA).\ synch1.$
$let(lookup(voterid,\ voterList) = true)\ in\ in(cha,\ voterid)$
$if\ (lookup(voterid,\ voterList) = true)$
$then\ out(cha,\ rdmeid_2).$
$let(lookup(voterid,\ voterList) = false)\ in\ synch2$

$Main = \nu\ ch1.\ \nu\ ch2\ .\ \nu\ ch3\ .\ \nu\ ch4\ .\ \nu\ chs\ .\ \nu\ che\ .\ \nu\ cha\ .$
$(\ processK \mid !processE \mid !processC$
$\mid !processA \mid !processS \mid$
$(\ \{Vida\ /\ Vid\ \} \mid \{^{ch5a}\ /_{\ ch5}\ \} \mid \{^{ch6a}\ /_{\ ch6}\ \} \mid$
$\{^{ch7a}\ /_{\ ch7}\ \} \mid \{^{a}\ /_{\ v}\ \} \mid\ processV\ ) \mid$
$(\ \{Vidb\ /\ Vid\ \} \mid \{^{ch5b}\ /_{\ ch5}\ \} \mid \{^{ch6b}\ /_{\ ch6}\ \} \mid$
$\{^{ch7b}\ /_{\ ch7}\ \} \mid \{^{b}\ /_{\ v}\ \} \mid\ processV\ )$

In the model, equations and functions used are simplified. The correspondence between entities is highlighted and given the most importance. Another crucial factor that affects the privacy of the protocol is the information that is known by every entity. All information broadcasted to the public channel ch may be used by everyone. Everything else is only known by the entity which uses it or to which it is bound and the entity to whom this information is sent to.

Equational Theory

$$xor(xor(x,y),y) = x$$
$$add(sub(x,y),y) = x$$
$$mul(x,y)$$
$$exp(x,exp(x,y)) = exp(exp(x,y),x)$$
$$computeshare(p,i)$$
$$\hat{e}(aP,bQ) = exp(\hat{e}(P,Q),mul(a,b))$$
$$\hat{e}(add(P,Q),R) = mul(\hat{e}(P,R),\hat{e}(Q,R))$$
$$\hat{e}(P,add(Q,R)) = mul(\hat{e}(P,Q),\hat{e}(Q,R))$$
$$H_1(x)$$
$$H_2(x)$$
$$H_3(x,y)$$
$$H_4(x,y,z)$$
$$combine(shares)$$

The equational theory is the set of all functions used in the formalisation and their relationships with one another. These functions are abstracted for simplicity. Being a protocol based on identity based cryptography, almost all functions used are asymmetric and cannot be easily deconstructed. H_1,...,H_4 are hash functions. The bilinear mapping is represented by $\hat{e}$. The shares of the keys are computed by means of the function computeshare. Other functions used, like *add*, *mul* and *sub*, are arithmetic in nature. As mentioned in the description, there is a synchronization between entities after the set - up phase, after the voting phase and lastly, after the counting phase before the release of results.

4.2 Proof of Correctness

For the proof of correctness, it is enough to use one instantiation of each process to run in parallel as described in the main process. We perform labelled reductions on the main process (substituting the respective processes in the terms) and arrive at the frame below. No function or process remain, thus, the protocol model in applied pi calculus is correct and will still function as it was described.

$\phi R'' = \nu ch1.\nu ch2.\nu ch3.\nu ch4.\nu chs.\nu che.\nu cha.$
$\nu g_{1m}.\nu g_{2m}.\nu pg_m.\nu s_{1m}.\nu id_{1E}.\nu id_{2E}.\nu voterListA.$
$\nu Vid_V.\nu s_{2K}.\nu r_{1V}.\nu r_{2E}.\nu a_V.\nu b_V.$
$\{^{(g_{1m},g_{2m},pg_m,mul(s1_m,pg_m),}$
$_{computeshare(pg_m,1))}/_{publicparam}\} \mid$
$\{^{id_{1E}}/_{x_1}\} \mid$
$\{^{(H_1(Eid_{1m}),computeshare(H_1(Eid_{1m}),1))}/_{x_2}\} \mid$
$\{^{id_{2E}}/_{x_3}\} \mid$
$\{^{mul(s_{2K},pg_m)}/_{x_4}\} \mid$
$\{^{(H_1(Eid_{2m}),mul(H_1(Eid_{2m}),s_{2K}))}/_{x_5}\} \mid$
$\{^{Vid_V}/_{x_6}\} \mid$
$\{^{rdmeid_2}/_{x_7}\} \mid$
$\{^{entityid_V}/_{x_8}\} \mid$
$\{^{mul(r_{2E},p)}/_{x_9}\} \mid$
$\{^{add(H_3(<mul(r_{1V},pointgen),xor(v,H_2(exp(\hat{e}(encpubkey,}$
$Epubkey),r_{1V})))>,add(add(mul(b_V,Esignkey),$
$mul(a_V,pointgen)),x),esigpkey),b_V)}/_{x_{10}}\} \mid$
$\{^{add(mul(cmsg_E,aprivsignkey),mul(r_{2E},sigpkey_E))}/_{x_{11}}\} \mid$
$\{^{(<mul(r_{1V},\ pointgen),}$
$xor(v,\ H_2(exp(\hat{e}(encpubkey,\ Epubkey),\ r_{1V})))>,$
$(add(s_v,mul(a_v,esigpkey)),sub(add(H_3(<mul(r_{1V},pointgen),$
$xor(v,\ H_2(exp(\hat{e}(encpubkey,Epubkey),\ r_{1V})))>,$
$add(add(mul(b_V,\ Esignkey),\ mul(a_V,\ pointgen)),\ x),\ esigpkey),$
$b_V),\ b_V)),entityid)}/_{x_{12}}\} \mid$
$\{^{H_4(encvote,sig,stamp(m3_S))}/_{x_{13}}\} \mid$
$\{^{msgtoc}/_{x_{14}}\} \mid$
$\{^{u}/_{x_{15}}\} \mid$
$\{^{\hat{e}(uvote_E,encskeyshare)}/_{x_{16}}\} \mid$
$\{^{out(ch,xor(w,H_2(combine(shares_E))),token)}/_{x_{17}}\} \mid$
$\{^{sigpkey_V}/_{esigpkey}\} \mid$
$\{^{receipt_V}/_{receipt}\}$

5. VERIFICATION

Now that we have a formalisation of the protocol in applied pi calculus, we can proceed with the verification of its privacy-type properties: vote-privacy, receipt-freeness and coercion-resistance.

5.1 Vote-Privacy

We use Definition 1 (subsection 3.3.1) in proving vote - privacy. We make the necessary substitutions for each

processV and S is the parallel composition of the processes. We pay attention to the voter processes as no election entity needs to be honest for we will look into that case with the attacker context. To derive this equivalence, we show that

$$\nu ch.(V_A\{^{alpha}/_{v}\}|V_B\{^{beta}/_{v}\}|processK)$$
$$\approx_{\ell}$$
$$\nu ch.(V_A\{^{beta}/_{v}\}|V_B\{^{alpha}/_{v}\}|processK)$$

We denote the left-hand process as P and the right-hand process as Q for readability. The protocol starts out with the setup phase of the PKG where we see the communication starts once the PKG broadcasts on the public channel the public parameters to be used for the whole election.

$$P \xrightarrow{in(ch,msgV_A)} P_1 \xrightarrow{in(ch,msgV_B)} P_2 \rightarrow^*$$
$$\xrightarrow{\nu x_1.out(cha,x_1)} \nu Vid_A.(P_3|\{^{Vid_A}/_{x_1}\})$$
$$\xrightarrow{\nu x_2.out(cha,x_2)} \nu Vid_A.\nu Vid_B.(P_4|\{^{Vid_A}/_{x_1}\}|\{^{Vid_B}/_{x_2}\})$$

Similarly,

$$Q \xrightarrow{in(ch,msgV_A)} Q_1 \xrightarrow{in(ch,msgV_B)} Q_2 \rightarrow^*$$
$$\xrightarrow{\nu x_1.out(cha,x_1)} \nu Vid_A.(Q_3|\{^{Vid_A}/_{x_1}\})$$
$$\xrightarrow{\nu x_2.out(cha,x_2)} \nu Vid_A.\nu Vid_B.(Q_4|\{^{Vid_A}/_{x_1}\}|\{^{Vid_B}/_{x_2}\})$$

In this context, we have to show that the scenario where Voter A votes for a and Voter B votes for b is observationally equivalent to Voter A voting for b and Voter B voting for a. This shows privacy because a third party will be unable to tell what each voter really voted because they vote at the same time, within the same phase of the election. Essentially, all the computations, sending out and receiving of messages of one voter should be matched and balanced by the other voter.

$\phi P'' = \nu ch.\nu Vid_A.\nu Vid_B.\nu r_{1A}.\nu r_{1B}.\nu a_A.\nu b_A.\nu a_B.\nu b_B.$
$\{^{Vid_A}/_{x_1}\}|$
$\{^{Vidb_B}/_{x_2}\}|$
$\{^{entityid}/_{x_3}\}|$
$\{^{entityid}/_{x_4}\}|$
$\{^{add(H_3(<mul(r_{1A},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>,add(add(mul(b_A,Esignkey),mul(a_A,p)),x),esigpkey),b_A)}/_{x_5}\}|$
$\{^{add(H_3(<mul(r_{1B},pointgen),xor(beta,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1B})))>,add(add(mul(b_B,Esignkey),mul(a_B,pointgen)),x),esigpkey),b_B)}/_{x_6}\}|$
$\{^{(<mul(r_{1A},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>,(add(s,mul(a_A,esigpkey)),sub(add(H_3(<mul(r_{1A},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>,add(add(mul(b_A,Esignkey),mul(a_A,p)),x),esigpkey),b_A),b_A)),entityid)}/_{x_7}\}|$
$\{^{(<mul(r1_B,pointgen),xor(beta,H_2(exp(\hat{e}(encpubkey,Epubkey),r1_B)))>,(add(s,mul(a_B,esigpkey)),sub(add(H_3(<mul(r1_B,pointgen),xor(beta,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1B})))>,add(add(mul(b_B,Esignkey),mul(a_B,p)),x),esigpkey),b_B),b_B)),entityid)}/_{x_8}\}$

$\phi Q'' = \nu ch.\nu Vid_A.\nu Vid_B.\nu r_{1A}.\nu r_{1B}.\nu a_A.\nu b_A.\nu a_B.\nu b_B.$
$\{^{Vid_A}/_{x_1}\}|$
$\{^{Vidb_B}/_{x_2}\}|$
$\{^{entityid}/_{x_3}\}|$
$\{^{entityid}/_{x_4}\}|$
$\{^{add(H_3(<mul(r_{1A},pointgen),xor(beta,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>,add(add(mul(b_A,Esignkey),mul(a_A,p)),x),esigpkey),b_A)}/_{x_5}\}|$
$\{^{add(H_3(<mul(r_{1B},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1B})))>,add(add(mul(b_B,Esignkey),mul(a_B,pointgen)),x),esigpkey),b_B)}/_{x_6}\}|$
$\{^{(<mul(r_{1A},pointgen),xor(beta,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>,(add(s,mul(a_A,esigpkey)),sub(add(H_3(<mul(r_{1A},pointgen),xor(beta,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>,add(add(mul(b_A,Esignkey),mul(a_A,p)),x),esigpkey),b_A),b_A)),entityid)}/_{x_7}\}|$
$\{^{(<mul(r_{1B},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1B})))>,(add(s,mul(a_B,esigpkey)),sub(add(H_3(<mul(r_{1B},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1B})))>,add(add(mul(b_B,Esignkey),mul(a_B,p)),x),esigpkey),b_B),b_B)),entityid)}/_{x_8}\}$

The corresponding frames shown above are statically equivalent, and even if we take a closer look at the values, an attacker would not be able to distinguish the votes because the functions are one-way only. To ensure vote privacy, the keys does not need to be secret, but the decryption shares play a big part. The synchronization is crucial because it allows for all the moves of Voter A to be balance by the moved of Voter B. In any case, no partial result will be revealed before any of the voters have finished voting because they will have to synchronize first before the entities can compute their shares and send to the Combining Entity. Also, the tally of the votes will only be published after all the votes have been combined and decrypted. Storage of the votes still does not violate vote-privacy because each of the votes are hashed with a time-stamp while in their decrypted form.

Definition 1 requires a stronger equivalence, labelled bisimilarity. The two processes, P and Q, are closed under labelled bisimilarity as proven by the static equivalence of the frames. Same reductions are used for both sides, fulfilling the second requirement of the equivalence. Lastly, the free names and bound names of the frames are the same, fulfilling the third requirement of the equivalence.

5.2 Receipt-Freeness

In proving receipt - freeness, we use Definition 2 (subsection 3.3.2). We construct a V' that will satisfy the above

conditions. V' is the same as process V with some modifications. V' is made to communicate with the coercer by sending messages to him, but not receiving messages. Also, it is important to note that V' fakes cooperation with the coercer by using a faking function on the values that she uses during the election before sending out these values that the coercer wants to know. Described below is the process V' to be used in this context.

$$
\begin{aligned}
processV' = {} & \nu chc.\ \nu Vid.\ out(chc, Vid).\ in(ch, msgV).\\
& in(ch, esigpkey).\ synch1.\ out(cha, Vid).\\
& in(cha, entityid).\ \nu r_1.\\
& \{^{msgV}/_{(d,f,pointgen,encpubkey,enckeyshare1)}\}\\
& \mid \{^{mul(r_1,pointgen)}/_{u}\}\\
& \mid \{^{H_2(exp(\hat{e}(encpubkey,Epubkey),r_1))}/_{h}\}\\
& \mid \{^{xor(alpha,h)}/_{w}\} \mid \{^{<u,w>}/_{ev}\}.\\
& \{^{xor(gamma,h)}/_{fw}\} \mid \{^{<u,fw>}/_{fvote}\}\\
& \mid out(chc, fvote).\ out(che, entityid).\\
& in(ch5, entitym).\ \nu a.\ \nu b.\\
& \{^{entitym}/_{(x,Esignkey)}\}\\
& \mid \{^{add(mul(b,Esignkey),mul(a,pointgen))}/_{p1}\}\\
& \mid \{^{add(H_3(ev,add(p_1,x),esigpkey),b)}/_{c}\}\\
& \mid out(ch6, c).\ in(ch7, s).\ \{^{add(s,mul(a,esigpkey))}/_{sprime}\}\\
& \mid \{^{sub(c,b)}/_{cprime}\} \mid \{^{(sprime,cprime)}/_{signature}\}\\
& \mid \{^{(ev,signature,entityid)}/_{storagemsg}\}\\
& \mid out(chs, storagemsg).\\
& \{^{add(H_3(fvote,\hat{e}(add(p_1,x),esigpkey)),b)}/_{fc}\}\\
& \mid \{^{sub(fc,b)}/_{fcprime}\} \mid \{^{(fcpime,sprime)}/_{fsig}\}\\
& \mid out(chc, fsig).\ in(chs, receipt).\ out(chc, receipt).\\
& synch2
\end{aligned}
$$

We establish the first bullet point of Definition 2 using $processV'$, still denoting the left-hand process as P and the right-hand process as Q.

$$
\begin{aligned}
P \xrightarrow{in(ch,msgV_A)} {} & P_1 \rightarrow^*\\
& \xrightarrow{\nu x_1.out(cha,x_1)} \nu chx.\nu Vid_x.(P_2 | \{^{Vid_x}/_{x_1}\})\\
& \xrightarrow{\nu x_2.out(che,x_2)} \nu chx.\nu Vid_x.(P_3 | \{^{Vid_A}/_{x_1}\} | \{^{entityid_V}/_{x_2}\})\\
& \rightarrow^* \xrightarrow{\nu x_2.out(ch6,x_3)} \nu chx.\nu Vid_x.\nu a_V.\nu b_V.(P_4 | \{^{Vid_A}/_{x_1}\}\\
& | \{^{entityid_V}/_{x_2}\} | \{^{add(H_3(<mul(r_{1V},pointgen),xor(alpha,\\
& H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1V})))>,add(add(mul(b_V,\\
& Esignkey),mul(a_V,p)),x),esigpkey_V),b_V)}/_{x_3}\})
\end{aligned}
$$

Similarly,

$$
\begin{aligned}
Q \xrightarrow{in(ch,msgV_A)} {} & Q_1 \rightarrow^*\\
& \xrightarrow{\nu x_1.out(cha,x_1)} \nu chx.\nu Vid_x.(Q_2 | \{^{Vid_x}/_{x_1}\})\\
& \xrightarrow{\nu x_2.out(che,x_2)} \nu chx.\nu Vid_x.(Q_3 | \{^{Vid_A}/_{x_1}\} | \{^{entityid_V}/_{x_2}\})\\
& \rightarrow^* \xrightarrow{\nu x_2.out(ch6,x_3)} \nu chx.\nu Vid_x.\nu a_V.\nu b_V.(Q_4 | \{^{Vid_A}/_{x_1}\}\\
& | \{^{entityid_V}/_{x_2}\} | \{^{add(H_3(<mul(r_{1V},pointgen),xor(alpha,}
\end{aligned}
$$

$$
\begin{aligned}
& H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1V})))>,add(add(mul(b_V,\\
& Esignkey),mul(a_V,p)),x),esigpkey_V),b_V)}/_{x_3}\})
\end{aligned}
$$

The first bullet point states that if all outputs to the coercer was ignored in process V' then it would be exactly the same as process V. This was proved by getting the frames of process V and process V'. In getting the frame of V', whenever an active substitution running parallel with an output to the coercer or simply an output to the coercer is found, it is ignored and removed from the frame. This results in exactly the same frame for process V and V'. This can be informally verified by visually comparing V and V'. They are statically equivalent, and as before, the values will not be distinguishable since the functions are one-way.

Proving receipt-freeness is slightly different from proving vote-privacy because at each step we require that the terms substituted are also equivalent, which is what we achieved. The frames described are shown below.

$$
\begin{aligned}
\phi P'' = {} & \nu Vidx.\ \nu r_{1V}.\ \nu a_V.\ \nu b_V.\\
& \{^{Vid_x}/_{x_1}\} \mid\\
& \{^{entityid_V}/_{x_2}\} \mid\\
& \{^{add(H_3(<mul(r1_V,pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,}\\
& Epubkey),r1_V)))>,add(add(mul(b_V,Esignkey),mul(a_V,p)),x),\\
& esigpkey_V),b_V)}/_{x_3}\} \mid\\
& \{^{(<mul(r_{1V},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,Epubkey),}\\
& r_{1V})))>,(add(s_V,mul(a_V,esigpkey_V)),sub(add(H_3(<mul(r_{1V},}\\
& pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1V})))>,}\\
& add(add(mul(b_V,Esignkey),mul(a_V,p)),x),esigpkey_V),b_V),}\\
& b_V)),entityidV)}/_{x_4}\}
\end{aligned}
$$

$$
\begin{aligned}
\phi Q'' = {} & \nu Vid_x.\ \nu r_{1V}.\ \nu a_V.\ \nu b_V.\\
& \{^{Vid_x}/_{x_1}\} \mid\\
& \{^{entityid_V}/_{x_2}\} \mid\\
& \{^{add(H_3(<mul(r_{1V},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,}\\
& Epubkey),r_{1V})))>,add(add(mul(b_V,Esignkey),mul(a_V,p)),x),\\
& esigpkey_V),b_V)}/_{x_3}\} \mid\\
& \{^{(<mul(r_{1V},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,Epubkey),}\\
& r_{1V})))>,(add(s_V,mul(a_V,esigpkey_V)),sub(add(H_3(<mul(r_{1V},}\\
& pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1V})))>,}\\
& add(add(mul(b_V,Esignkey),mul(a_V,p)),x),esigpkey_V),b_V),}\\
& b_V)),entityid_V)}/_{x_4}\})
\end{aligned}
$$

To prove the second bullet point, we use the same context as in vote-privacy where the processes run in parallel. With the left-hand side as P and the right-hand side as Q, we show that

$$
\begin{aligned}
& \nu ch.(V_A\{^{gamma}/_{v}\}^{chc} | V_B\{^{alpha}/_{v}\} | processK)\\
& =\\
& \nu ch.(V' | V_B\{^{alpha}/_{v}\} | processK)
\end{aligned}
$$

Same as the first one, we require that at each step the terms substituted are equivalent. We perform the labelled reductions on both the left and right side as follows.

$$P \xrightarrow{in(ch,msgV_A)} P_1 \xrightarrow{in(ch,msgV_B)} P_2 \rightarrow^* \xrightarrow{\nu x_1.out(chx,x_1)} \nu ch.\nu chx.\nu Vid_A.\nu Vid_B.(P_3|\{^{Vid_x}/_{x1}\}) \xrightarrow{\nu x_2.out(cha,x_2)} \nu ch.\nu chx.\nu Vid_A.\nu Vid_B.(P_4|\{^{Vid_A}/_{x_1}\} |\{^{x_1}/_{x_2}\}) \rightarrow^* \xrightarrow{\nu x_3.out(cha,x_3)} \nu ch.\nu chx.\nu Vid_A.\nu Vid_B.(P_5|\{^{Vid_A}/_{x_1}\} |\{^{x_1}/_{x_2}\}|\{^{Vid_B}/_{x_3}\})$$

Similarly,

$$Q \xrightarrow{in(ch,msgV_A)} Q_1 \xrightarrow{in(ch,msgV_B)} Q_2 \rightarrow^* \xrightarrow{\nu x_1.out(chx,x_1)} \nu ch.\nu chx.\nu Vid_A.\nu Vid_B.(Q_3|\{^{Vid_x}/_{x_1}\}) \xrightarrow{\nu x_2.out(cha,x_2)} \nu ch.\nu chx.\nu Vid_A.\nu Vid_B.(Q_4|\{^{Vid_A}/_{x_1}\} |\{^{x_1}/_{x_2}\}) \rightarrow^* \xrightarrow{\nu x_3.out(cha,x_3)} \nu ch.\nu chx.\nu Vid_A.\nu Vid_B.(Q_4|\{^{Vid_A}/_{x_1}\} |\{^{x_1}/_{x_2}\}|\{^{Vid_B}/_{x_3}\})$$

The second bullet point states that if Voter A votes for c while outputting secrets to the coercer, she would go through the same steps if she would vote for a while fooling the coercer to think that she voted for c. This should be true as long as there is another voter who would counterbalance the votes. This point was proved by obtaining the frames of both sides of the equation. The first requirement of labelled bisimilarity was readily observed from the frames. All corresponding terms are structurally equivalent. The second and third requirements of labelled bisimilarity are proved by obtaining the frames. All reductions done on the left side were also done on the right side. The frames obtained are shown below.

$\phi P'' = \nu ch.\nu chx.\nu Vid_A.\nu Vid_B.\nu r_{1A}.\nu r_{1B}.\nu a_A.\nu b_A.\nu a_B.\nu b_B.$

$\{^{Vid_A}/_{x_1}\}|$

$\{^{x_1}/_{x_2}\}|$

$\{^{Vid_B}/_{x_3}\}|$

$\{^{<mul(r_{1A},pointgen),xor(gamma,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>}/_{x_4}\}|$

$\{^{entityid}/_{x_5}\}|$

$\{^{entityid}/_{x_6}\}|$

$\{^{add(H_3(<mul(r_{1A},pointgen),xor(gamma,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>,add(add(mul(b_A,Esignkey),mul(a_A,p)),x),esigpkey),b_A)}/_{x_7}\}|$

$\{^{add(H_3(<mul(r_{1B},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1B})))>,add(add(mul(b_B,Esignkey),mul(a_B,p)),x),esigpkey),b_B)}/_{x_8}\}|$

$\{^{(<mul(r_{1A},pointgen),xor(gamma,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>,(add(s_A,mul(a_A,esigpkey)),sub(add(H_3(<mul(r_{1A},pointgen),xor(gamma,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>,add(p_1,x),esigpkey),b_A),b_A)),entityid)}/_{x_9}\}|$

$\{^{(<mul(r_{1A},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1B})))>,(add(s_B,mul(a_B,esigpkey)),sub(add(H_3(<mul(r_{1B},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1B})))>,add(p_1,x),esigpkey),b_B),b_B)),entityid)}/_{x_{10}}\}|$

$\{^{(sub(add(H_3(<mul(r_{1A},pointgen),xor(gamma,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>,\hat{e}(add(p_1,x),esigpkey)),b_A),b_A),add(s_A,mul(a_A,esigpkey)))}/_{x_{11}}\}|$

$\{^{receipt_A}/_{x_{12}}\})$

$\phi Q'' = \nu ch.\nu chx.\nu Vid_A.\nu Vid_B.\nu r_{1A}.\nu r_{1B}.\nu a_A.\nu b_A.\nu a_B.\nu b_b.$

$\{^{Vid_A}/_{x_1}\}|$

$\{^{x_1}/_{x_2}\}|$

$\{^{Vid_B}/_{x_3}\}|$

$\{^{<mul(r_{1A},pointgen),xor(gamma,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>}/_{x_4}\}|$

$\{^{entityid}/_{x_5}\}|$

$\{^{entityid}/_{x_6}\}|$

$\{^{add(H_3(<mul(r_{1A},pointgen),xor(gamma,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>,add(add(mul(b_A,Esignkey),mul(a_A,p)),x),esigpkey),b_A)}/_{x_7}\}|$

$\{^{add(H_3(<mul(r_{1B},pointgen),xor(gamma,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1B})))>,add(add(mul(b_B,Esignkey),mul(a_B,p)),x),esigpkey),b_B)}/_{x_8}\}|$

$\{^{(<mul(r_{1A},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>,(add(s_A,mul(a_A,esigpkey)),sub(add(H_3(<mul(r_{1A},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>,add(p_1,x),esigpkey),b_A),b_A)),entityid)}/_{x_9}\}|$

$\{^{(<mul(r_{1B},pointgen),xor(gamma,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1B})))>,(add(s_B,mul(a_B,esigpkey)),sub(add(H_3(<mul(r_{1B},pointgen),xor(gamma,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1B})))>,add(p_1,x),esigpkey),b_B),b_B)),entityid)}/_{x_{10}}\}|$

$\{^{(sub(add(H_3(<mul(r_{1A},pointgen),xor(gamma,H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1A})))>,\hat{e}(add(p_1,x),esigpkey)),b_A),b_A),add(s_A,mul(a_A,esigpkey)))}/_{x_{11}}\}|$

$\{^{receipt_A}/_{x_{12}}\}$

As with vote - privacy, static equivalence of the frames fulfill the first requirement of labelled bisimilarity. The same reductions are used for P and Q in both bullet points. This fulfills the second requirement. By visually inspecting the frames on both bullet points, the same names are bound on P and Q. This fulfills the third requirement of the equivalence.

5.3 Coercion-Resistance

The third and strongest property to be proved is coercion - resistance. We already modeled a plain process V' in proving receipt - freeness. To prove coercion - resistance, we use Definition 3 (subsection 3.3.3).

Using the derivations by Delaune et al on the second bullet point of Definition 3, we arrive at

$$S[V_A\{^{c}/_{v}\}^{chc} \mid V_B\{^{a}/_{v}\}] \approx_\ell S[C[V'] \mid V_B\{^{c}/_{v}\}]$$

where C is the context with a hole where we model the coercer. In context C, the coercer performs all the computations to get the values that he wishes the coerced voter to vote for. Then, he sends these to the voter. The model of context C is shown below.

$$\begin{aligned}
contextC[_] = \nu chc1.\ \nu chc2.\ (\ _\ \mid\ & (in(chc1, id).\ in(ch, pp).\\
& in(ch, spk).\ synch1.\ in(chc1, eid).\\
& in(chc1, em).\ \nu rn1.\ \{^{pp}/_{(j,k,gp,epk,epks1)}\}\ \mid\\
& \{^{mul(rn_1,gp)}/_{cu}\}\ \mid\\
& \{^{H_2(exp(\hat{e}(epk,Epubkey),rn1))}/_{ch}\}\ \mid\\
& \{^{xor(cv,ch)}/_{cw}\}\ \mid\ \{^{<cu,cw>}/_{cev}\}\ \mid\\
& out(chc2, cev).\ \nu ca.\ \nu cb.\ \{^{em}/_{(cx,keyes)}\}\ \mid\\
& \{^{add(mul(cb,keyes),mul(ca,gp))}/_{part1}\}\ \mid\\
& \{^{add(H_3(cev,add(part1,cx),spk),cb)}/_{cc}\}\ \mid\\
& out(chc2, cc).\ in(chc1, vs).\\
& \{^{add(vs,mul(ca,spk))}/_{csp}\}\ \mid\ \{^{sub(cc,cb)}/_{cp}\}\ \mid\\
& out(chc2, (csp, cp)).\ in(chc1, rct).\ synch2))
\end{aligned}$$

Now that we have our context C, we must model our process V' similar to that of receipt-freeness, wherein process V' would fake cooperation with the coercer. This time, V' is slightly different because the coercer can send messages to her.

$$\begin{aligned}
V' = \nu\ & V_{id}.\ out(chc_1,\ V_{id}).\ in(ch,\ msgV).\\
& in(ch,\ esigpkey).\ synch1\ .out(cha,\ V_{id}).\\
& in(cha,\ entityid).\ out(chc_1,\ entityid).\ \nu\ r_1.\\
& \{^{msgV}/_{(d,\ f,\ pointgen,\ encpubkey,\ enckeyshare^1)}\}\ \mid\\
& \{^{mul(r_1,\ pointgen)}/_{u}\}\ \mid\\
& \{^{H_2(exp(\hat{e}(encpubkey,\ Epubkey),\ r_1))}/_{h}\}\ \mid\ \{^{xor(v,\ h)}/_{w}\}\ \mid\\
& \{^{<u,w>}/_{ev}\}.\ out(che,\ entityid).\ in(ch5,\ entitym).\\
& out(chc_1, entitym).\ in(chc_2,\ coerev).\ in(chc_2,\ coerc).\\
& \nu\ a.\ \nu\ b.\ \{^{entitym}/_{(x,Esignkey)}\}\ \mid\\
& \{^{add(mul(b,Esignkey),mul(a,pointgen))}/_{p_1}\}\ \mid\\
& \{^{add(H3(ev,add(p_1,x),\ esigpkey),\ b)}/_{c}\}\ \mid\\
& out(ch6,\ c).\ in(ch7,\ s).\ out(chc_1,\ s).\\
& in(chc_2,\ coersig).\ \{^{add(s,\ mul(a,\ esigpkey))}/_{sprime}\}\ \mid\\
& \{^{sub(c,\ b)}/_{cprime}\}\ \mid\ \{^{(sprime,\ cprime)}/_{signature}\}\ \mid\\
& \{^{(ev,\ signature,\ entityid)}/_{storagemsg}\}\ \mid\\
& out(chs,\ storagemsg).\ in(chs,\ receipt).\\
& out(chc_1,\ receipt).\ synch2
\end{aligned}$$

Next, we must model V'' that cooperates with the coercer in context C. In a simpler sense, this would be a certain voter, say Voter A, who would receive messages from the coercer and use those messages in the election proper. This behavior of V_A is as follows:

$$\begin{aligned}
processV_A = \nu Vid.\ & out(chc1, Vid).\ in(ch, msgV).\\
& in(ch, esigpkey).\ synch1.\ out(cha, Vid).\\
& in(cha, entityid).\ out(chc1, entityid).\\
& out(che, entityid).\ in(ch5, entitym).\\
& out(chc1, entitym).\ in(chc2, coerev).\\
& in(chc2, coerc).\ out(ch6, coerc).in(ch7, s).\\
& out(chc1, s).\ in(chc2, coersig).\\
& out(chs, (coerev, coersig, entityid).\\
& in(chs, receipt).\ out(chc1, receipt).\ synch2
\end{aligned}$$

Taking a closer look, it seems as if the voter is just a bridge between the coercer and the election process, which is exactly the behavior we need for V''. We place this in the hole in context C and proceed with establishing the equivalence for the first bullet point.

$$\begin{aligned}
P \xrightarrow{in(ch,msgV_A)} & P_1 \rightarrow^*\\
& \xrightarrow{\nu x_1.out(cha,x_1)} \nu chc1.\nu chc2.\nu Vid.(P_2 \mid \{^{Vid}/_{x_1}\})\\
& \rightarrow^* \xrightarrow{\nu x_2.out(che,x_2)} \nu chc1.\nu chc2.\nu Vid.\nu r_1.(P_2 \mid\\
& \qquad \{^{Vid}/_{x_1}\} \mid \{^{entityid_A}/_{x_2}\})\\
& \rightarrow^* \xrightarrow{\nu x_3.out(ch6,x_3)} \nu chc1.\nu chc2.\nu Vid.\nu r_1.\nu a.\nu b.\\
& \qquad (P_2 \mid \{^{Vid}/_{x_1}\} \mid \{^{entityid_A}/_{x_2}\} \mid\\
& \qquad \{^{add(H_3(<mul(r_1,pointgen),xor(v,H_2(}\\
& \qquad exp(\hat{e}(encpubkey,Epubkey),r_1)))>,\\
& \qquad add(add(mul(b,Esignkey),mul(a,pointgen)),\\
& \qquad x),esigpkey_A),b)}/_{x_3}\})
\end{aligned}$$

Similarly,

$$\begin{aligned}
Q \xrightarrow{in(ch,msg)} & Q_1 \rightarrow^*\\
& \xrightarrow{\nu x_1.out(cha,x_1)} \nu Vid_x.(Q_2 \mid \{^{Vid_x}/_{x_1}\})\\
& \rightarrow^* \xrightarrow{\nu x_2.out(che,x_2)} \nu Vid_x.\nu r_{1v}.(Q_2 \mid \{^{Vid_x}/_{x_1}\} \mid\\
& \qquad \{^{entityid_v}/_{x_2}\})\\
& \rightarrow^* \xrightarrow{\nu x_3.out(ch6,x_3)} \nu Vid_x.\nu r_{1v}.\nu a_v.\nu b_v.(Q_2 \mid \{^{Vid_x}/_{x_1}\} \mid\\
& \qquad \{^{entityid_v}/_{x_2}\} \mid\\
& \qquad \{^{add(H_3(<mul(r_{1v},pointgen),xor(alpha,H_2(}\\
& \qquad exp(\hat{e}(encpubkey,Epubkey),r_{1v})))>,\\
& \qquad add(add(mul(b_v,Esignkey),mul(a_v,p)),x),\\
& \qquad esigpkey_v),b_v)}/_{x_3}\})
\end{aligned}$$

Ignoring all communication with the coercer (including the channels they communicate on), we will see that the frames (shown below) are statically equivalent. In getting the frame of the left side, normal reductions are made. The only difference is whenever an out to the coercer or an active substitution parallel to an out to the coercer was encountered, these steps where just removed without applying the normal reductions. This results in a frame that is

exactly the same as the frame of the right side (previously obtained) except for the channels used to communicate with the coercer, $chc1$ and $chc2$, being bound. These were easily removed using the reduction NEW - 0. The resulting frames are now exactly the same.

$$\phi P'' = \nu chc1.\nu chc2.\nu Vid.\nu r_1.\nu a.\nu b.$$
$$\{^{Vid}/_{x_1}\} \mid$$
$$\{^{entityid_A}/_{x_2}\} \mid$$
$$\{^{add(H_3(<mul(r_1,pointgen),xor(v,H_2(}$$
$$exp(\hat{e}(encpubkey,Epubkey),r1))),add(add(}$$
$$mul(b,Esignkey),mul(a,pointgen)),x),esigpkeyA),b)}/_{x_3}\}$$
$$\{^{(<mul(r_1,pointgen),xor(v,H_2(exp(}$$
$$\hat{e}(encpubkey,Epubkey),r_1))),(add(s_A,mul(a,}$$
$$esigpkey_A)),sub(add(H_3(<mul(r_1,pointgen),xor(v,}$$
$$H_2(exp(\hat{e}(encpubkey,Epubkey),r_1))),}$$
$$add(add(mul(b,Esignkey),mul(a,pointgen)),x),}$$
$$esigpkey_A),b),b)),entityid_A)}/_{x_4}\}$$
$$\phi Q'' = \nu Vid_x.\nu r_{1v}.\nu a_v.\nu b_v.$$
$$\{^{Vid_x}/_{x_1}\} \mid$$
$$\{^{entityid_v}/_{x_2}\} \mid$$
$$\{^{add(H_3(<mul(r_{1v},pointgen),xor(alpha,H_2(}$$
$$exp(\hat{e}(encpubkey,Epubkey),r_{1v}))),add(add(}$$
$$mul(b_v,Esignkey),mul(a_v,p)),x),esigpkeyv),b_v)}/_{x_3}\} \mid$$
$$\{^{(<mul(r_{1v},pointgen),xor(alpha,H_2(exp(}$$
$$\hat{e}(encpubkey,Epubkey),r_{1v}))),(add(s_v,mul(a_v,}$$
$$esigpkey_v)),sub(add(H_3(<mul(r_{1v},pointgen),xor(alpha,}$$
$$H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1v}))),}$$
$$add(add(mul(b_v,Esignkey),mul(a_v,p)),x),}$$
$$esigpkey_v),b_v),b_v)),entityid_v)}/_{x_4}\}$$

The first bullet point simplistically states that if all the messages to and from the coercer is ignored, the voter V' goes through exactly the same steps as a voter V who votes for a certain candidate a. This would mean that V' would make the coercer believe that she is voting what the coercer wants her to vote, while actually voting for her own choice.

Now we are left with one more bullet point to establish. The second requirement states that an observer doesn't see the difference in the steps taken by a voter who fakes cooperation with a coercer and a voter that sincerely cooperates with a coercer provided there are other voters who counterbalance the votes they cast. Again, to prove this, the usual reductions are used to obtain the frames for the left and right side.

$$P \rightarrow^* \xrightarrow{in(ch,msgV_B)} P_1 \xrightarrow{in(ch,pp_c)} P_2 \xrightarrow{in(ch,msgV_B)} P_3$$
$$\rightarrow^* \xrightarrow{\nu x_1.out(cha,x_1)} \nu ch.\nu chc1.\nu chc2.\nu Vid_A.\nu Vid_B.$$
$$(P_4 \mid \{^{Vid_B}/_{x_1}\})$$
$$\rightarrow^* \xrightarrow{\nu x_2.out(cha,x_2)} \nu ch.\nu chc1.\nu chc2.\nu Vid_A.\nu Vid_B.$$
$$(P_5 \mid \{^{Vid_B}/_{x_1}\} \mid \{^{Vid_B}/_{x_2}\})$$

$$Q \rightarrow^* \xrightarrow{in(ch,msgV_B)} Q_1 \xrightarrow{in(ch,pp_c)} Q_2 \xrightarrow{in(ch,msgV_B)} Q_3$$
$$\rightarrow^* \xrightarrow{\nu x_1.out(cha,x_1)} \nu ch.\nu chc1.\nu chc2.\nu Vid_A.\nu Vid_B.$$
$$(P_4 \mid \{^{Vid_B}/_{x_1}\})$$
$$\rightarrow^* \xrightarrow{\nu x_2.out(cha,x_2)} \nu ch.\nu chc1.\nu chc2.\nu Vid_A.\nu Vid_B.$$
$$(P_5 \mid \{^{Vid_B}/_{x_1}\} \mid \{^{Vid_B}/_{x_2}\})$$

Doing the same reductions on both sides we arrive at the frames shown below. The resulting frames obtained at the end of the reductions are the same. All functions and values used on one side has a corresponding function or value on the other. One issue that was resolved to obtain this frame was the removing of the random numbers, a, b and r_1, used by the voter in faking cooperation with the coercer. These were bound at first then removed by the reduction NEW - 0. They need not be bound in the first place but we chose to bind, use and then remove them to make it clear that the voter used a set of random numbers different from the ones used by the coercer. The random numbers are secret anyway, and they need not be bound to the process, just like the actual vote is not bound to the voter process.

$$\phi P'' = \nu ch.\nu chc1.\nu chc2.\nu Vid_A.\nu Vid_B.$$
$$\nu r_{1B}.\nu vr_{1C}.\nu a_B.\nu b_B.\nu a_C.\nu b_C.$$
$$\{^{Vid_A}/_{x_1}\} \mid$$
$$\{^{Vid_B}/_{x_2}\} \mid$$
$$\{^{entityid_A}/_{x_3}\} \mid$$
$$\{^{entityid_B}/_{x_4}\} \mid$$
$$\{^{coerc_A}/_{x_5}\} \mid$$
$$\{^{add(H_3(<mul(r_{1B},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,}$$
$$Epubkey),r_{1B})))>,add(add(mul(b_B,Esignkey),mul(a_B,}$$
$$pointgen)),x),esigpkey_B),b_B)}/_{x_6}\} \mid$$
$$\{^{(coerev_A,coersig_A,entityid_A)}/_{x_7}\} \mid$$
$$\{^{(<mul(r_{1B},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,}$$
$$Epubkey),r_{1B})))>,add(s_B,mul(a_B esigpkey_B)),}$$
$$sub(add(H_3(<mul(r_{1B},pointgen),xor(alpha,}$$
$$H_2(exp(\hat{e}(encpubkey,Epubkey),r_{1B})))>,}$$
$$add(add(mul(b_B,Esignkey),mul(a_B,}$$
$$pointgen)),x),esigpkey_B),b_B),b_B)),entityid_B)}/_{x_8}\} \mid$$
$$\{^{Vid_A}/_{y_1}\} \mid$$
$$\{^{entityid_A}/_{y_2}\} \mid$$
$$\{^{entitym_A}/_{y_3}\} \mid$$
$$\{^{<mul(r_{1C},gp),xor(cv,H_2(exp(\hat{e}(epk,Epubkey),r_{1C})))>}/_{y_4}\} \mid$$
$$\{^{add(H_3(<mul(r_{1C},gp),xor(cv,H_2(exp(\hat{e}(epk,Epubkey),r_{1C})))>,}$$
$$add(add(mul(b_C,keyes),mul(a_C,gp)),cx),spkc),b_C)}/_{y_5}\} \mid$$
$$\{^{s_A}/_{y_6}\} \mid$$

$\{^{(add(vs_C,mul(a_C,spkc)),sub(add(H_3(<mul(r_{1C},gp),}$
$xor(cv,H_2(exp(\hat{e}(epk,Epubkey),r_{1C})))>,$
$add(add(mul(b_C,keyes),mul(a_C,gp)),cx),spkc),$
$b_C),b_C))}/_{y_7}\} \mid$
$\{^{receipt_A}/_{y_8}\}$

$\phi Q'' = \nu ch.\nu chc1.\nu chc2.\nu Vid_A.\nu Vid_B.\nu r_{1B}.\nu r_{1C}.$
$\nu a_B.\nu b_B.\nu a_C.\nu b_C.$
$\{^{Vid_A}/_{x_1}\} \mid$
$\{^{Vid_B}/_{x_2}\} \mid$
$\{^{entityid_B}/_{x_4}\} \mid$
$\{^{entityid_A}/_{x_3}\} \mid$
$\{^{add(H_3(<mul(r_{1A},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,}$
$Epubkey),r_{1A})))>,add(add(mul(b_A,Esignkey),mul(a_A,pointgen)),$
$x),esigpkey_A),b_A)}/_{x_5}\} \mid$
$\{^{add(H_3(<mul(r_{1B},pointgen),xor(gamma,H_2(exp(\hat{e}(encpubkey,}$
$Epubkey),r_{1B})))>,add(add(mul(b_B,Esignkey),mul(a_B,pointgen)),$
$x)esigpkey_B),b_B)}/_{x_6}\} \mid$
$\{^{(<mul(r_{1A},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,}$
$Epubkey),r_{1A})))>,(add(s_A,mul(a_A,esigpkey_A)),$
$sub(add(H_3(<mul(r_{1A},pointgen),xor(alpha,H_2(exp(\hat{e}(encpubkey,}$
$Epubkey),r_{1A})))>,add(p_1,x),esigpkey_A),b_A),b_A)),entityid_A)}/_{x_7}\} \mid$
$\{^{(<mul(r_{1B},pointgen),xor(gamma,H_2(exp(\hat{e}(encpubkey,}$
$Epubkey),r_{1B})))>,(add(s_B,mul(a_B,esigpkey_B)),sub(add(H_3($
$<mul(r_{1B},pointgen),xor(gamma,H_2(exp(\hat{e}(encpubkey,Epubkey),$
$r_{1B})))>,add(add(mul(b_B,Esignkey),mul(a_B,pointgen)),x),$
$esigpkey_B),b_B),b_B)),entityid_B)}/_{x_8}\} \mid$
$\{^{Vid_A}/_{y_1}\} \mid$
$\{^{entityid_A}/_{y_2}\} \mid$
$\{^{entitym_A}/_{y_3}\} \mid$
$\{^{<mul(r_{1C},gp),xor(cv,H_2(exp(\hat{e}(epk,Epubkey),r_{1C})))>}/_{y_4}\} \mid$
$\{^{add(H_3(<mul(r_{1C},gp),xor(cv,H_2(exp(\hat{e}(epk,Epubkey),r_{1C})))>,}$
$add(add(mul(b_C,keyes),mul(a_C,gp)),cx),spkc),b_C)}/_{y_5}\} \mid$
$\{^{s_A}/_{y_6}\} \mid$
$\{^{(add(vs_C,mul(a_C,spkc)),sub(add(H_3(<mul(r_{1C},gp),xor(cv,}$
$H_2(exp(\hat{e}(epk,Epubkey),r_{1C})))>,add(add(mul(b_C,keyes),mul(a_C,$
$gp)),cx),spkc),b_C),b_C))}/_{y_7}\} \mid$
$\{^{receipt_A}/_{y_8}\}$

Just like the first two properties, the frames, the reductions and the bound names and free names in the frames fulfill all three requirements of labelled bisimilarity.

6. CONCLUSION

In [7], the authors claimed:

- *Privacy* - The votes must not be decrypted by any other entity other than the Combining Entity during the Counting phase. This was assured in [7] by relying on the hardness of the bilinear Diffie-Helman problem and the hash value of the stamp on the receipt.
- *Uncoercibility* - The voter must be able to vote freely, thus she must not be forced to vote in a certain way. Again, the bilinear Diffie-Helman problem cannot be solved in a polynomial time making the encryption of the votes hard to break, thus hard for the coercer to find out the value of the vote.

Our results show that these are true, having privacy modeled to be vote-privacy while uncoercibility covers receipt - freeness and coercion-resistance. They also claimed Eligibility and Uniqueness, Transparency, Accuracy and Robustness. As mentioned before, among the properties claimed in [7], we formalized and verified Privacy and Uncoercibility because these are the privacy-type properties of the protocol. Thus, all three privacy-type properties, vote-privacy, receipt-freeness and coercion-resistance are satisfied by the IBC protocol.

At first, it seems that receipt - freeness will not hold true because of the physical receipt that the voter receives at the end of the voting phase. But as we examine the protocol closer, we see that a lot of factors contribute for it to be receipt - free. Aside from the hardness of the computations, the synchronizations make the votes undistinguishable. The frames are statically equivalent; an observer cannot tell the difference between a normal voter, a voter to cooperates with the coercer and a voter who fakes votes to the coercer. Thus, it is crucial for the Combining Entity to only reveal the list of the votes after all the shares have been combined and decrypted.

After obtaining the frames, we noticed that it might be possible to remove the counterbalancing vote and the properties will still hold. This will be the basis for our future work, as we might delve further into this to see if the properties may hold without the voter who balances the moves of the coerced voter.

Since most concerns about the security of ID-based encryption schemes are beyond the scope of this paper because they are implementation related, we now suggest to the authors of the IBC protocol to carefully design and review the design before deployment. This is to address issues regarding injected ballots and other vulnerability issues with the entities being online.

7. REFERENCES

[1] J. Baek and Y. Zhen, *Identity-Based Threshold Decryption*, In Proc. of the 7th International Workshop on Theory and Practice in Public Key Cryptography, LNCS 2947, Springer-Verlag, pp.262-276, 2004.

[2] B. Blanchet, *ProVerif: Automatic Cryptographic Protocol Verifier User Manual*, <www.preverif.ens.fr/proverif-manual.ps.gz>, 2005.

[3] S. Delaune, S. Kremer, and M. Ryan, *Coercion-Resistance and Receipt-Freeness in Electronic Voting*, Computer Security Foundations Workshop, IEEE, pp. 28-42, 19th IEEE Computer Security Foundations Workshop (CSFW'06), 2006.

[4] S. Delaune, S. Kremer, and M. Ryan, *Verifying privacy-type properties of electronic voting protocols* Journal of Computer Security 17(4), pp. 435-487, 2009.

[5] Y. G. Desmedt, and Y. Frankel, *Threshold cryptosystems*. In Proceedings on Advances in Cryptology (Santa Barbara, California, United States). G. Brassard, Ed. Springer-Verlag New York, New York, pp. 307-315, 1989.

[6] A. Fujioka, T.Okamoto, K. Ohta, *A practical secret voting scheme for large scale elections*, In J. Seberry and Y. Zheng, editors, Advances in Cryptology (AUSCRYPT '92), volume 718 of Lecture Notes in Computer Science, pp. 244-251, Springer, 1992.

[7] G. Gallegos-Garcia, R. Gomez-Cardenas, and G. I. Duchen-Sanchez, *Electronic Voting Using Identity Based Cryptography*, International Conference on the Digital Society, pp. 31-36, 2010 Fourth International Conference on Digital Society, 2010.

[8] A. Juels, D. Catalano, and M. Jakobsson, *Coercion-resistant electronic elections*, In WPES '05: Proc. workshop on Privacy in the electronic society, pp. 61-70, ACM, 2005.

[9] S. Kremer and M. Ryan, *Analysis of an Electronic Voting Protocol in the Applied Pi Calculus*, In Proceedings of the European Symposium on Programming (ESOP'05), Lecture Notes in Computer Science series, volume 3444, pp. 186-200, Springer Verlag, 2005.

[10] S. K. Langford, *Weakness in Some Threshold Cryptosystems*, In Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology 1996, N. Koblitz, Ed. Lecture Notes In Computer Science. Springer-Verlag, London, pp. 74-82, 1996.

[11] B. Lee, C. Boyd, E. Dawson, K. Kim, J.Yang, and S.Yoo. *Providing receipt-freeness in mixnet-based voting protocols*. In Proceedings of Information Security and Cryptology (ICISC'03), volume 2971 of LNCS, pp. 245-258, Springer, 2004.

[12] M. Ryan and B. Smyth, *Applied pi calculus*, In V. Cortier and S. Kremer, Formal Models and Techniques for Analyzing Security Protocols, Chapter 6, IOS Press, 2010.