A Novel Secure Electronic Voting Protocol Based On Bilinear Pairings

Jue-Sam Chou [1] , Yalin Chen [2] , Jin-Cheng Huang [3]

[1]Department of Information Management, Nanhua University Chiayi 622 Taiwan, R.O.C

jschou@mail.nhu.edu.tw

Tel: 886+ (0)5+272-1001 ext.56226

[2] Institute of Information systems and applications, National Tsing Hua University

d949702@oz.nthu.edu.tw

Tel: 886+(0)3-5738997

[3] Department of Information Management, Nanhua University Chiayi, 622, Taiwan

heartenhuang@gmail.com

Tel: 886+(0)5-2721001 ext.2017

Abstract

In 1997, Cranor and Cytron proposed an electronic voting protocol, Sensus protocol, intended to be applied in a real election. However, in 2005 Fabrizio et.al. pointed out there is a vulnerability exists in their protocol that the validator can impersonate anyone of those abstained voters to cast vote. They proposed a scheme, Seas protocol, to solve this weakness. But in this paper, we will show that Seas protocol is not only inefficient but also impractical. Moreover, we also propose a sound electronic voting protocol based on Sensus protocol from bilinear pairings, which can really satisfy the security requirements of an e-voting system.

***Keywords**: electronic voting, bilinear pairings, ID-based cryptographic system*

1. Introduction

For traditional voting has some problems such as its inconvenience and high social cost, many measures have been proposed intended to overcome these problems. Among the proposed schemes, electronic voting could be the most effective way. Besides, due to the fast technology promotion in computer science, several benefits can be obtained by applying an electronic voting system (EVS) such as, cost of building polling stations can be reduce greatly, people can participate in an election easily at home with only clicking their mouse or touching the monitor, and the lower cost and time spent when counting the voted ballots.

Because Internet is a public and insecure environment, many secure electronic voting schemes have been proposed [2,4,6,7,8,9,10,14,15,16,17] attempting to securely fulfill a real electronic voting. In 1992, Fujioka et al.[2] proposed a practical secret voting scheme for a large scale election. In their scheme, they list some properties which an EVS should have, such as soundness, unfeasibility, completeness, privacy, eligibility, fairness, verifiability and uncoercibity. We briefly delineate them as follows.

(1)Soundness: No one can interfere with or disrupt an election.

(2)Unreusability: An eligible voter can vote only once and decide his intention himself.

(3)Completeness: An eligible voted ballot can always be counted in the tally.

(4)Privacy: No one can know who casts the ballots.

(5)Eligibility: An eligible and registered voter can be accepted by the system to vote without failure.

(6)Fairness: Before the system publishes the final tally, anyone can't know the result.

(7)Verifiability: An eligible voter can check to see if his ballot has been counted in the tally.

Later, in 1997, Cranor et al.[5] proposed the Sensus protocol based on Fujioka et al.'s scheme [2]. Unfortunately in 2005, Fabrizio et al.[1] pointed out there exist the common vulnerabilities in both [2] and [5] that anyone can maliciously replace the abstained voter to cast his ballot. In other words, while the voter abstains from voting his ballot, the validator can impersonate this abstaining voter to vote the ballot. Meanwhile, Fabrizio et al.[1] also proposed the Seas e-voting protocol to overcome this drawback. However, after our analysis, we find that their protocol is not only complex but also impractical. For instance, the pollsters must have two IDs in their protocol, which incurs the increment in the computational cost. In addition, the pollsters have to exchange information to the tallier before voting. This action is absurd in a real election. To realize a secure EVS, we propose a new simple realistic scheme from pairings based on Sensus protocol. Our protocol not only can overcome the vulnerability in Sensus, but also is efficient and practical.

The structure of this paper is as follows. In Section 2, we first review the concept and properties of bilinear pairings, then introduce Sensus and Seas protocols respectively, and finally demonstrate the vulnerabilities in Sensus and Seas protocols. In Section 3, we present a secure electronic voting protocol based on bilinear pairings, and in Section 4, we analyze the security of our protocol. Finally, a conclusion is given in Section 5.

2. The background

In this section, we first briefly elucidate the basic concept and some properties of bilinear pairings, then review the Sensus protocol and Seas protocol, and finally, explain the weakness existing in these protocols.

2.1 bilinear pairings

Let G_1 be a cyclic group generated by O, whose order is a prime q and G_2 be a cyclic multiplicative group of the same order q. We assume that the discrete logarithm problem (DLP) in both G_1 and G_2 are hard. Let e: $G_1 \times G_1 \rightarrow G_2$ be a pairing which satisfies the following conditions：

(1)Bilinear: $e(aO,bQ) = e(O,Q)^{ab}$, for any $a,b \in Z$ and $O,Q \in G_1$

(2)Computability: there is an efficient algorithm to compute e(O,Q) for all $O,Q \in G_1$

(3)Non-degenerate: there exists $O \in G_1$ and $Q \in G_1$ such that $e(O,Q) \neq 1$

2.2 Review of Sensus protocol

There are three parties in Sensus protocol：

(1)P: The pollster. The pollster is a set of hardware and software through which a voter can cast its ballot.

(2)V: The validator. A server that first checks the eligibility of the pollster P, then the uniqueness of its submission, and finally it validates the submitted vote.

(3)T: The tallier. A server devotes to counting all the valid votes.

We illustrate the process of Sensus protocol in fig.2 and explain it using the following steps. The definitions of used notations are given in table1.

Table.1 The notations and it's definitions in Sensus protocol

Notation	Description
pk_V, pk_V^{-1}	V's public/private key pair
pk_P, pk_P^{-1}	P's public/private key pair
pk_T, pk_T^{-1}	T's public/private key pair
ek,dk	P's asymmetric key pair of encryption/decryption keys
RVL	a list of people who are both eligible and registered for voting
RL	a list of valid ballots
B	a voted ballot
I	identifier
rec#	receipt number

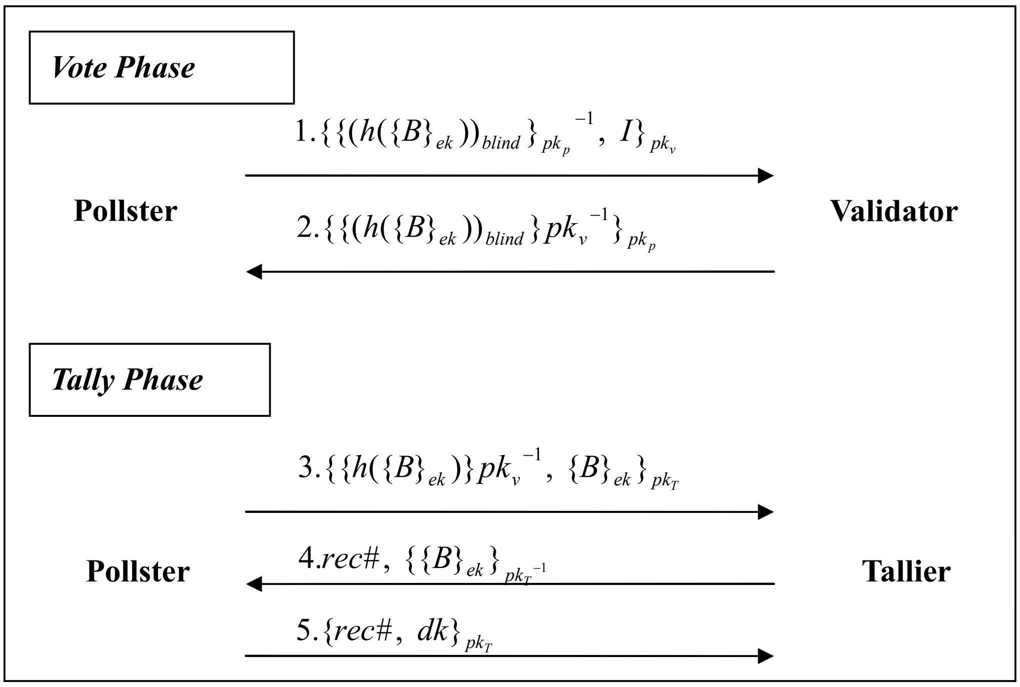
Step.1 When pollster P wants to vote a ballot B, he generates a secret encryption key ek, encrypts B with ek, and then blinds the digest of the encrypted message h({B}ek) to obtain $(h(\{B\}_{ek})_{blind})$ (bD: blind digest). Then, P signs it using his private key $pk_P{}^{-1}$ to obtain the signature $\{\{(h(\{B\}_{ek}))_{blind}\}_{pk_P{}^{-1}}\}$ (PS). After that, P concatenates PS with the election identifier I to obtain R and encrypts R using the validator's public key to obtain $\{\{(h(\{B\}_{ek}))_{blind}\}_{pk_P{}^{-1}}, I\}_{pk_V}$ (ER). Finally, P sends it to the validator V.

Step.2 After receiving ER, the validator first decrypts ER using his private key $pk_V{}^{-1}$ to obtain PS and I. V then verifies the signature PS by applying P's public key pk_P and checks to see if I belongs to the list, RVL, of the eligible registered voters. Then he also checks to see if P is the first time that casts his ballot in the election I. If all of these checks succeed, V then signs on the blind digest BD to obtain, $\{\{(h(\{B\}_{ek}))_{blind}\}_{pk_v{}^{-1}}\}$ (VSbD) and encrypts it using P's public key, obtaining EVSbD then sends EVSbD to P.

Step.3 After receiving EVSBD, P decrypts it using his private key, obtaining VSbD and then unblinds it, getting $\{(h(\{B\}_{ek}))\}_{pk_v{}^{-1}}$ (VSD). Then, P uses T's public key to encrypt the concatenation of VSD and $\{B\}_{ek}$ and then sends the result, E-VSD-EB, to T.

Step4. The tallier first decrypts the receipt E-VSD-EB from P in step3, obtaining $\{h(\{B\}_{ek})\}_{pk_V{}^{-1}}$ and $\{B\}_{ek}$, then T checks the honesty of the pollster by verifying whether the encrypted ballot $\{B\}_{ek}$ has been validated by V. To this aim, verifies the signature $\{\{(h(\{B\}_{ek}))_{pk_v{}^{-1}}\}$ of V by applying V's public key pk_V to obtain $\{h(\{B\}_{ek})\}$, then he computes the digest on $\{B\}_{ek}$ and compares this newly calculated result with the one verified. If these two values are equal, T adds $\{B\}_{ek}$ into the list RL. Finally, T signs the encrypted ballot $\{B\}_{ek}$ to obtain $\{\{B\}_{ek}\}_{pk_T{}^{-1}}$ and sends it back to P together with a received number rec#. T also updates RL by inserting the receipt number.

Step5. After verifying the received message, Pollster sends the ballot decryption key dk and the rec# which both is first concatenated and then encrypted using T's public key, to the tallier. The tallier uses dk to decrypt the encrypted ballot, obtaining B. The tallier adds the ballot to the final tally and adds the decryption key *dk* corresponding to *ek* to RL.



Vote Phase
$1.\{\{(h(\{B\}_{ek}))_{blind}\}_{pk_p^{-1}},\ I\}_{pk_v}$
Pollster
$2.\{\{(h(\{B\}_{ek}))_{blind}\}pk_v^{-1}\}_{pk_p}$
Validator
Tally Phase
$3.\{\{h(\{B\}_{ek})\}pk_v^{-1},\ \{B\}_{ek}\}_{pk_T}$
Pollster
$4.rec\#,\ \{\{B\}_{ek}\}_{pk_T^{-1}}$
Tallier
$5.\{rec\#,\ dk\}_{pk_T}$

Fig.1 The Sensus protocol

2.3 Review of Seas protocol

There are three phases in Seas protocol. We describe them as follows. The process is also demonstrated in figure 2a through figure 2c respectively.

2.3.1 Registration phase

In this phase, the pollsters register to the tallier. We describe it using the following steps.

Step.1 P blinds the digest of the concatenation of both P's second public key, $pk_P{}^2$, and his second ID, ID^2, obtaining $(h(pk_P{}^2, ID^2))_{blind}$. Then, he adds the identifier ID^1, signs them with his first private key pk_P^{-1} and sends the result $\{\{(h(pk_P{}^2, ID^2))_{blind}, ID^1\}_{pk_P{}^{1^{-1}}}\}_{pk_T}$ to T

Step.2 Upon receiving $\{\{(h(pk_P{}^2, ID^2))_{blind}, ID^1\}_{pk_P{}^{1^{-1}}}\}_{pk_T}$, T decrypts it using his private key and then retrieves $pk_P{}^1$ and ID^1 from the list RVL1 to verify the signature $\{(h(pk_P{}^2, ID^2))_{blind}, ID^1\}_{pk_P{}^{1^{-1}}}$. If the signature is valid, he signs on $(h(pk_P{}^2, ID^2))_{blind}$, obtaining $\{h(pk_P{}^2, ID^2))_{blind}\}_{pk_T{}^{-1}}$. Then, he encrypts this signature with P's first public key and sends the result to P. T then updates RVL1.

Step.3 After decrypting the result transmitted in step2, P removes the blinding, obtaining $\{\ \{h(pk_P{}^2, ID^2))\}_{pk_T{}^{-1}}\ \}$ and concatenates it with $(pk_P{}^2, ID^2)$ then encrypts the result using T's public key. He then sends this encryption to T.

Step.4 After decrypting the receipt in step3, the tallier verifies his own signature. If the verification succeeds, T computes the digest of $(pk_P{}^2, ID^2)$, getting $h(pk_P{}^2, ID^2)$. Then, he compares this newly computed digest with the verified result. If both are equal, T records the pair $(pk_P{}^2, ID^2)$ in RVL2.

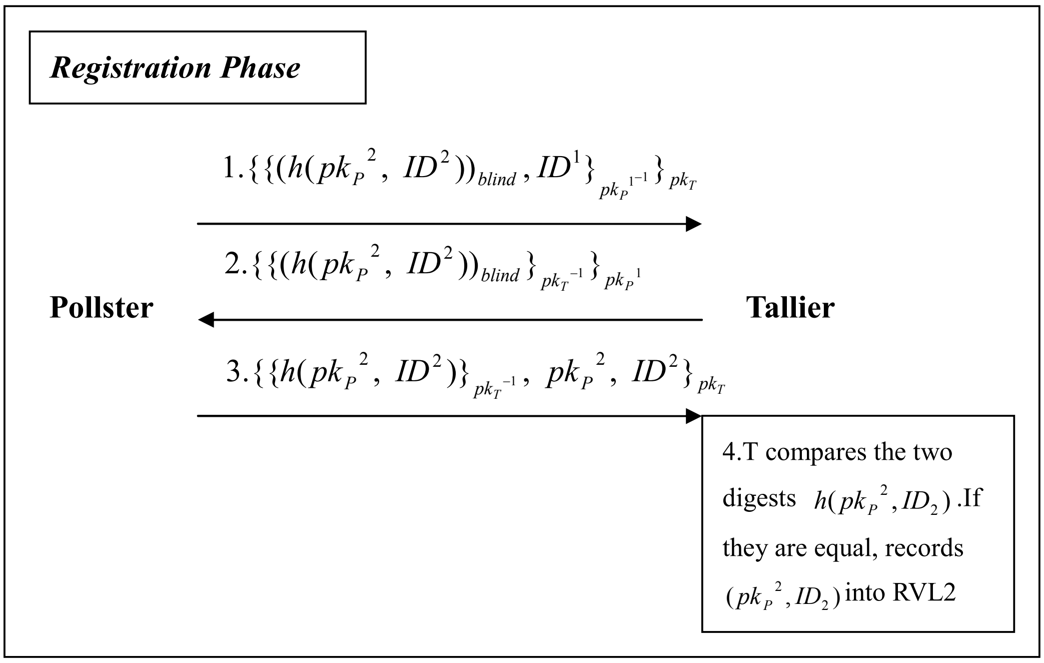

Registration Phase
$1.\{\{(h(pk_P{}^2,\ ID^2))_{blind}, ID^1\}_{pk_P{}^{1^{-1}}}\}_{pk_T}$
$2.\{\{(h(pk_P{}^2,\ ID^2))_{blind}\}_{pk_T{}^{-1}}\}_{pk_P{}^1}$
Pollster
Tallier
$3.\{\{h(pk_P{}^2,\ ID^2)\}_{pk_T{}^{-1}},\ pk_P{}^2,\ ID^2\}_{pk_T}$
4.T compares the two digests $h(pk_P{}^2, ID_2)$ .If they are equal, records $(pk_P{}^2, ID_2)$ into RVL2

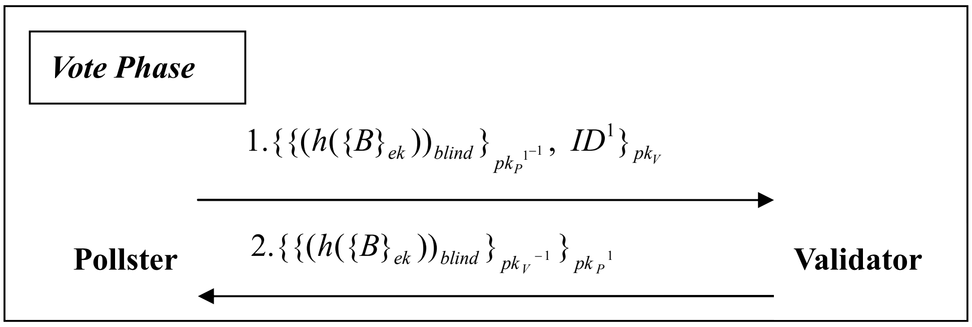
Fig.2a Registration Phase

2.3.2 Voting phase

This phase is to let the pollster can validly vote his ballot to the validator. We describe it using the following steps.

Step.1 P encrypts the ballot B with a randomly generated secret encryption key *ek* and then blinds the digest of this result, obtaining $(h(\{B\}_{ek}))_{blind}$. Then, he signs on $(h(\{B\}_{ek}))_{blind}$ with his private key ${pk_P^{1}}^{-1}$, concatenates this signature with ID^1, and finally encrypts the concatenation using V's public key and sends this encryption result to V.

Step.2 After decrypting the receipt from step 1, V verifies to see if the signature $\{(h(\{B\}_{ek}))_{blind}\}_{{pk_P^{1}}^{-1}}$ is valid. If it so, V updates RVL1 and signs on $(h(\{B\}_{ek}))_{blind}$ with pk_V^{-1}. Then, he encrypts the result with P's first public key pk_P^1 and sends the result to P.



Vote Phase
$1.\{\{(h(\{B\}_{ek}))_{blind}\}_{{pk_P^{1}}^{-1}},\ ID^1\}_{pk_V}$
Pollster
$2.\{\{(h(\{B\}_{ek}))_{blind}\}_{pk_V^{-1}}\}_{pk_P^1}$
Validator

Fig.2b Voting Phase

2.3.3 Tally phase

This phase is for pollster to add his vote to the final tally. We describe it using the following steps.

Step.1 P decrypts the receipt in Step 2 of the voting phase and then removes the blinding, obtaining $\{h(\{\{B\}_{ek}))\}_{pk_V^{-1}}$ and verifies the validator's signature using pk_V. If it is valid, then P uses his second private key to sign on $\{h(\{\{B\}_{ek}))\}_{pk_V^{-1}}$, obtaining $\{\{h(\{\{B\}_{ek}))\}_{pk_V^{-1}}\}_{{pk_P^{2}}^{-1}}$ and sends it together with

$\{B\}_{ek}$ and ID^2, which are all encrypted using T's public key, then he sends the encryption to T.

Step.2 T decrypts the receipt in Step1 and verifies the signature $\{h(\{\{B\}_{ek})\}_{pk_V^{-1}}\}_{pk_P^{2^{-1}}}$ with $pk_P{}^2$ and pk_V, then he compares the two digests, the newly calculated from the transmitted *{B}ek* and the one in V's signature, to see whether they are equal. If all verifications succeed, he signs on $\{B\}_{ek}$ and updates RL, then sends $rec\#$ and $\{\{B\}\}_{ek}\}_{pk_T^{-1}}$ to P.

Step.3 P first verifies the signature using pk_T, then he concatenates both the *rec#* and the decrypt key *dk* corresponding to *ek*, encrypts this concatenation T's public key and finally sends the result to T.

Step.4 T opens $\{B\}_{ek}$ with *dk*, and then updates RL and tally.

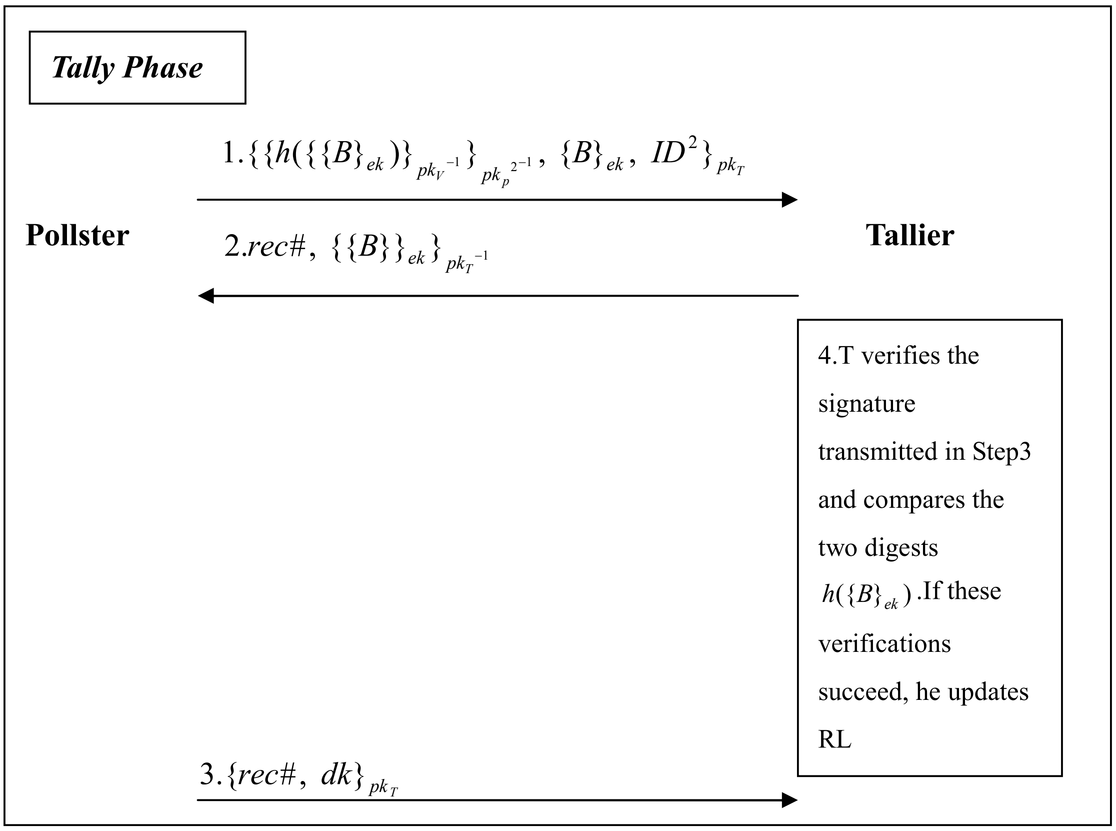

Tally Phase
1. $\{\{h(\{\{B\}_{ek})\}_{pk_V^{-1}}\}_{pk_P^{2^{-1}}}, \{B\}_{ek}, ID^2\}_{pk_T}$
Pollster
Tallier
2. $rec\#, \{\{B\}\}_{ek}\}_{pk_T^{-1}}$
4.T verifies the signature transmitted in Step3 and compares the two digests $h(\{B\}_{ek})$.If these verifications succeed, he updates RL
3. $\{rec\#, dk\}_{pk_T}$

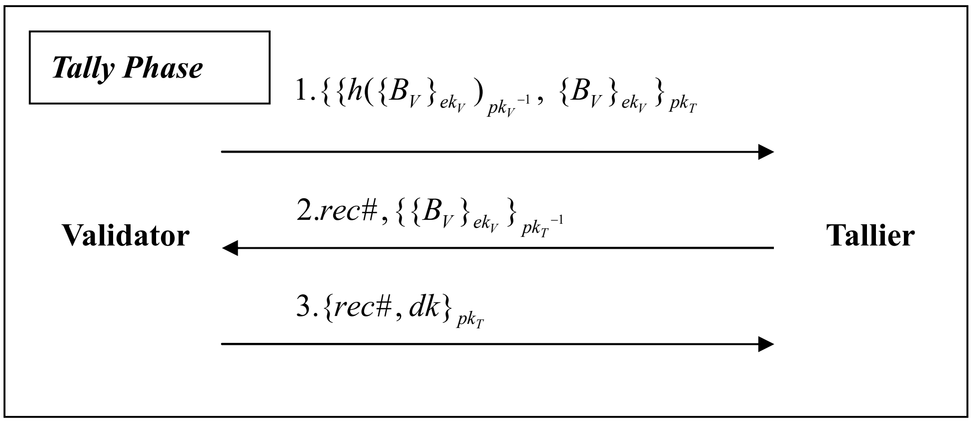
Fig.2c Tally Phase

2.4 Vulnerabilities in Sensus and Seas protocols

In this section, we explain the vulnerabilities in Sensus protocol pointed by [1] and Seas protocol, respectively.

2.4.1 The Sensus vulnerability

Assume that someone, say Bob, among the eligible and registered voters abstains from voting. The validator V can then maliciously replace Bob to cast the vote, as though the vote is casted by Bob himself. More precisely, the validator V can maliciously replace someone's abstained vote B by his own vote B_V, where B_V represents a forgery ballot prepared by the validtor to send to the tallier in the tally phase. However, the tallier T can't distinguish B from B_V. Hence, the illegal ballot replaced by the validator would also be counted in the tally successfully. A pictorial view of the above mentioned attack is shown in Fig.3.



Tally Phase
1. $\{\{h(\{B_V\}_{ek_V})\}_{pk_V^{-1}}, \{B_V\}_{ek_V}\}_{pk_T}$
2. $rec\#, \{\{B_V\}_{ek_V}\}_{pk_T^{-1}}$
3. $\{rec\#, dk\}_{pk_T}$
Validator
Tallier

Fig.3 The Sensus vulnerability

2.4.2 The Seas vulnerability

The Seas protocol is both complexity and inefficient, because the pollster has to use two identifiers ID^1 and ID^2. Besides, he also needs to generate two pairs of public/private key $pk_P^{1}, pk_P^{1^{-1}}, pk_P^{2}, pk_P^{2^{-1}}$. Furthermore, there are totally ten interactive passes in the protocol. Thus, if the election is large scale this would incur an intolerant increment in the computation load and the time spent in traveling the network when compared with Sensus protocol. Other than its complexity and inefficiency, the Seas protocol is also unrealistic. Since in the registration phase, the pollster must exchange information with the tallier before he votes the ballot to the validator. In a real election, this is absurd.

3.Our protocol

For the complexity and unreality of Seas protocol, in this section, we propose a new scheme based on Sensus protocol from pairings. Our protocol not only can remove the weakness in Sensus protocol but also is more efficient. We explain our protocol using the following steps and also illustrate the process in fig.4.

3.1 System Setup

Assume that there exists a key generation center and someone has registered and become a legal user with his identity ID. KGC computes the user's public/private key pair as $Q_{ID} = H_1(ID)$ and $S_{ID} = sQ_{ID}$, where s is KGC's secret and H_1:{0,1}*→G_1. Then KGC sends Q_{ID} and S_{ID} to the user via a secure channel. Then pollster P pre-computes $h_{PV} = H(e(S_P, Q_V))$ and $h_{PT} = H(e(S_P, Q_T))$, validator V pre-computes $h_{VP} = H(e(Q_P, S_V))(= h_{PV})$ and $h_{VT} = H(e(Q_T, S_V))$, and tallier T pre-computes $h_{TP} = H(e(Q_P, S_T)(= h_{PT})$ and $h_{TV} = (H(e(S_T, Q_V))(= h_{VT})$ in advance before the protocol running with each keeping his computed values secret.

3.2 Proposed protocol

Step.1 To vote a ballot, pollster P computes $h_{PCC} = H(e(c_1O, c_2O))$, where H: G_1→ {0,1}* , c_1,c_2 are two random numbers chosen by P and O is G_1's generator. Then P sends $\{B_{PVT}(= B \oplus h_{PCC} \oplus h_{PT} \oplus h_{PV}),\ I'(= I \oplus h_{PV}), H(B_{PVT}, I), H(h_{PCC})\}$ to V, where B is the voted ballot and I is the identifier of an election.

Step.2 For V to validate the voted ballot, as V has received $\{B_{PVT}, I'\}$ from P in step1, he can obtain $B_{PT}(= B \oplus h_{PCC} \oplus h_{PT})$ by computing $B_{PVT} \oplus h_{VP}$, and I which V can use to verify the election identifier by computing $I' \oplus h_{VP}$ Then V randomly choose a number r_V and sends $\{B_{VPT}(= B_{PT} \oplus H(r_V) \oplus h_{VP}), H(B_{VPT}), R_V(= r_V \oplus h_{VT}), H(h_{PCC}) \oplus h_{VT}\}$ to P.

Step.3 After receiving $\{B_{VPT}, H(B_{VPT})\}$ from step2, P can obtain $B_{PTV}(= B \oplus h_{PCC} \oplus H(r_V))$ by computing $B_{VPT} \oplus h_{PV} \oplus h_{PT}$. Then he calculates $B_{PT}'(= B \oplus h_{PCC} \oplus h_{PT})$ and sends $\{B_{PTV}, B_{PT}', H(B_{PTV}, B_{PT}'), R_V, H(h_{PCC}) \oplus h_{VT}\}$ to tallier T.

Step.4 For T to count the ballot to the tally correctly, after receiving $\{B_{PTV}, B_{PT}', H(B_{PTV}, B_{PT}), R_V, H(B_{PT} \oplus h_{VT})\}$ from step3, he can obtain r_V by computing $R_V \oplus h_{VT}$ and compute to check whether the two values of $B_{PTV} \oplus h_{TP} \oplus H(r_V)(= B_{PT})$ and $B_{PT}' \oplus h_{TP} \oplus h_{TP}(= B_{PT}')$ are the same. If they are equal, T sends the receipt number rec#, $B_{PT}'(= B \oplus h_{pcc} \oplus h_{TP})$ and $H(B_{PT}')$ to P.

Step.5 After receiving {Rec#, B_{PT}', $H(B_{PT}')$} from T, P checks to see if the calculated value of $B_{PT} \oplus h_{PT}$ equals to $B \oplus h_{PCC}$. If it so, P sends Rec#, $h_{PCC}'(= h_{PCC} \oplus h_{PT})$ and $H(h_{PCC}')$ to T.

Finally, tallier T first using the received h_{PCC}' in step5 and $H(h_{PCC}) \oplus h_{VT}$ in step3 to compute $h_{PCC}' \oplus h_{PT}$, obtaining h_{PCC} and compute $H(h_{PCC}) \oplus h_{VT} \oplus h_{VT}$, obtaining $H(h_{PCC})$, respectively. Then he checks to see whether the digest of h_{PCC} is equal to $H(h_{PCC})$. If they are equal, T uses this obtained h_{PCC} to compute $B_{PT}'(= B \oplus h_{pcc} \oplus h_{TP}) \oplus h_{PCC} \oplus h_{TP}$, obtaining B. Then, T adds this voted ballot B to the final tally. If pollster P wants to prove his vote, he can reveal c_1 and c_2 because c_1 and c_2 are P's secret.

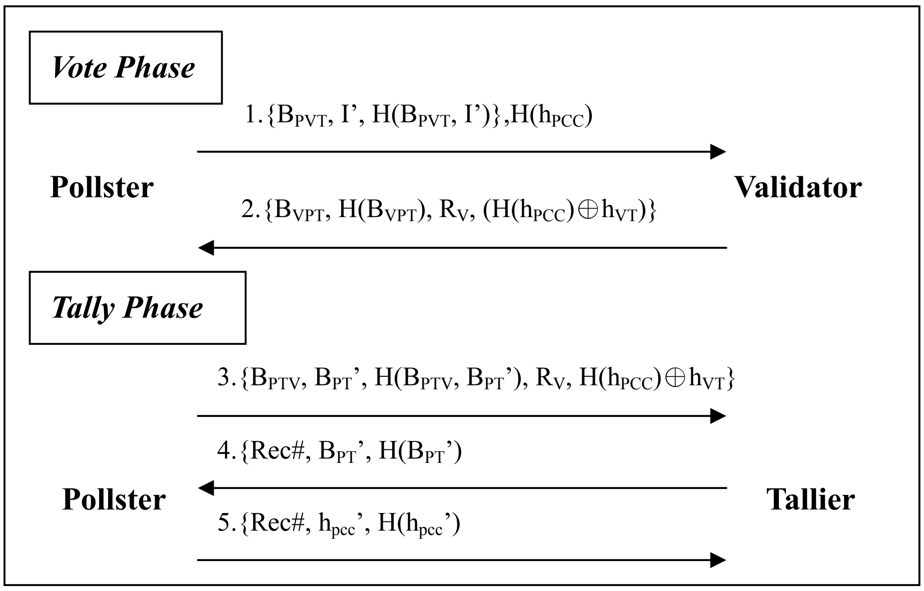

Vote Phase
1. {B_PVT, I', H(B_PVT, I')},H(h_PCC)
Pollster
Validator
2. {B_VPT, H(B_VPT), R_V, (H(h_PCC)⊕h_VT)}
Tally Phase
3. {B_PTV, B_PT', H(B_PTV, B_PT'), R_V, H(h_PCC)⊕h_VT}
4. {Rec#, B_PT', H(B_PT')}
Pollster
Tallier
5. {Rec#, h_pcc', H(h_pcc')}

Fig.4 Our protocol

4. Security Analysis

In the following, we will propose some attacks which often occurs in an EVS on our scheme. After analyzing, we find that any attack on our protocol can't work successfully.

Attack 1: If an eligible and registered voter abstains from voting, can the validator replace him to vote successfully?

Although V can compute $h_{VP} = h_{PV}$, (for $e(Q_P, S_V) = e(Q_P, sQ_V) = e(sQ_P, Q_V) = e(S_P, Q_V)$), he can't forge h_{PT} . Since he doesn't have P's private key S_P. Therefore, V can't generate B_{PTV} and B_{PT}' successfully. For this reason, we can conclude that the validator's attack is deemed to failure.

Attack 2: Can the validator forge a valid ballot to be counted in the final tally?

According to ID-based cryptographic system [18], $Q_{ID}=H_1(ID)$ and $S_{ID}=sQ_{ID}$, if the validator V wants to forge a valid ballot, he must forge S_{ID} first. However, he can't know s because s is KGC's secret. Therefore, we can declare that the validator's attack fails as well.

Attack 3: Can someone impersonate the validator V successfully?

If someone wants to impersonate the validator to verify the validity of voted ballots, he must first get the validator's private key S_V; otherwise, he can't compute h_{VP} and h_{VT} . However, it is impossible for him to get the validator's private key. Thus, this attack is also unsuccessful.

Attack 4: Can someone impersonate the tallier T successfully?

The attacker has to face the problem of getting the tallier's private key S_T to compute $h_{TP} = H(e(Q_P, S_T))$ and $h_{TV} = H(e(S_T, Q_V))$. However, it is also infeasible for him to get the taller's private key via a public channel. Thus, we can conclude that this attack is deemed to failure as well.

Attack 5: If P is dishonest, he wants to claim that the voted ballot is a counterfeit ballot (C-B) instead of B which he really voted. Can he achieve this malicious purpose?

He can't. Since, if P wants to replace B to C-B, he must simultaneously generates C-B and C-h_{PCC} to satisfy $C\text{-}B \oplus C\text{-}h_{PCC} = B \oplus h_{PCC}$. However, because tallier T has checked whether the calculated $B_{PT} (= B_{PTV} \oplus h_{TP} \oplus H(r_V)$, where r_V can be obtained by T in step4.) and B_{PT}' obtained in step3 are the same, and he also has checked whether the two values, the calculated digest of h_{PCC} computed from h_{PCC} ' received in step5 by X-or-ing out the

h_{PT}, and the $H(h_{PCC})$ which P had committd in $H(h_{PCC}) \oplus h_{VT}$ received in step3 by X-or-ing out the h_{VT}, are the same. This implies that under the fixed h_{PCC}, the value of B is fixed as well. Therefore, the voted ballot can not be changed and thus this attack can not hold.

Attack 6: Can tallier T be dishonesty? In other words, can he replace the voted ballot B received from P, to C-B to the final tally?

He can't. Because when tallier T adds the voted ballot B to the final tally, according to the protocol in step4, he must add the receipt number rec# and the corresponding B_{PT}' to the RVL at the same time which implies that the specific B had been bound to a specific rec#. Hence, pollster P can get B_{PT}' from RVL to determine if tallier T had replaced B to C-B by comparing whether the computed value of B from $B_{PT}' \oplus h_{PT} \oplus h_{PCC}$ and his voted ballot B are the same.

5. Conclusion

In this paper, we have pointed out the weakness in Seas protocol. Besides, based on Sensus protocol which has a flaw pointed by [1], we propose an efficient and secure electronic voting protocol based on bilinear pairings. Our scheme can let the participants; P, V and T precompute their needed values before the protocol running. Moreover, the remaining operations in our protocol are the exclusive-or operations which can save large amount of computation time when compared with the two public-key operation based protocols, Seas and Sensus. Besides, the number of interactive passes needed in our method only five, which is much less than Seas protocols. In addition, after our analysis in Section 4, we conclude that our protocol is really secure. Thus, our protocol is not only quite efficient but also more practical than all the protocols proposed so far.

Reference

[1] F. Baiardi, A. Falleni, R. Granchi, F. Martinelli, M. Petrocchi, A. Vaccarelli. " SEAS, a secure e-voting protocol: Design and implementation, " Computers & Security (2005) 24, 642-652.

[2] Fujioka A, Okamoto T, Ohta K. " A practical secret voting scheme for large scale election. " In: Proceedings of Auscrypt'92, vol.LNCS 718; 1992. p. 244e60

[3] Horng-Twu Liaw, " A secure electronic voting protocol for general elections," Computers & Security (2004) 23, 107-119.

[4] Benaloh J, Tuinstra D. " Receipt-free secret-ballot election. " In: Proceedings of ACM STOC'94. ACM; 1994. p. 544e53.

[5] Cranor L, Cytron RK. “ Sensor: a security-conscious electronic polling system for the internet. “ In: Proceedings of HICSS’97. IEEE; 1997. p. 561e70.

[6] Karro J, Wang J. “ Towards a practical, secure, and very largescale online election.“ In: Proceedings of ACSAC’99. IEEE;1999. p. 161e9.

[7] Magkos E, Burmester M, Chrissikopoulos V. “ Receipt-freeness in large-scale elections without untappable channels. “ In: 13E. Kluwer; 2001. p. 683e94.

[8] Ray I, Ray I, Narasimhamurthi N. “ An anonymous electronic voting protocol for voting over the internet. “ In: Proceedings of WECWIS’01. IEEE; 2001. p. 188e91.

[9] Rubin AD. “ Security considerations for remote electronic voting.“ Communications of the ACM 2002;45(12):39e44.

[10] Ryan P, Bryans J. “ Security and trust in digital voting systems.“ In: Proceedings of FAST’03. Tech Rep. IIT TR-10/2003;2003. p. 113e20.

[11] Chaum D. “ Elections with unconditionally-secret ballots and disruption equivalent to breaking RSA. “ Advances in CryptologydEurocrypt’ 88; 1988. p. 177e82.

[12] Coleman S. “Elections in the 21st century: from paper ballot to e-voting.“ Report by the Independent Commission on alternative voting methods, London, Electoral Reform Society;February 2002.

[13] Cramer R, Gennaro R, Schoenmakers B. “ A secure and optimally efficient multi-authority election scheme. “ Advances in Cryptology: Proceedings of EuroCrypt’97, LNCS 1233. Springer-Verlag; 1997. p. 103e18.

[14] DTLR News Release. “ May elections to trial online voting; “ 2002. Available from: http://www.press.dtlr.gov.uk/pns/DisplayPN. cgi?pn_id=2002_0033.

[15] Hoffman LJ. “ Internet voting: will it spur or corrupt democracy? “ Proceedings of the 10th Conference on Computers, Freedom and Privacy: Challenging the Assumptions; 2000. p.219e23.

[16] Hwang JJ. “A conventional approach to secret balloting in computer networks.“ Comput Secur 1996;15(3):249e62.

[17] Jan JK, Tai CC. “ A secure electronic voting protocol with IC cards. “J Syst Software 1997;39:93e101.

[18] Boneh, M. Franklin,” Identity-based encrtption from the weil pairing ”, Advances in Cryptology-CRYPTO 2001, LNCS 2139, 2001, pp. 213-229.