1998 International Computer Symposium
Workshop on Cryptology and Information Security
December 17-19, 1998, N.C.K.U., Tainan, Taiwan, R.O.C.

AN ELECTRONIC VOTING SYSTEM WITH ENHANCED SECURITY *

Shyh-In Hwang

Department of Computer Engineering and Science
Yuan Ze University
135 Yuan-Tung Road, Chung-Li 320, Taiwan
Email: shyhin@cs.yzu.edu.tw

Abstract

In this paper, we present a voting protocol, which satisfies a variety of desiring requirements of voting protocols. In particular, we propose to incorporate the concept of the session public-key/private-key pair, which serves as a permit to vote, into the voting scheme such that the privacy of the voters can be preserved and that the votes can not be forged by malicious parties. The computation and communication overhead of the scheme is reasonable for conducting a large scale election on the Internet.

1 Introduction to Electronic Voting Protocols

As modern society more and more relies on computers and networks, the application of conducting electronic voting over the Internet receives more and more attention. You can cast an unofficial vote for Presidential election, express your opinion in regards to a governmental strategy, or just vote for your favorite video games. That is fun, and reflects more or less the opinion of some people. However, no one may seriously consider to conduct serious elections through such a tool. Many of the implementations for electronic voting lack the characteristics required for a serious election. The characteristics of a secure electronic voting system depend on the purposes for which the system is to be used, and may vary from case to case. In the following a list of requirements of a secure electronic voting system is specified [7, 16, 10].

*This work is supported in part by NSC under NSC-87-2213-E-155-004.

1.1 Requirements of Voting Protocols

On the following list, the items numbered with *a* and *b* represent alternatives to its original corresponding item.

- Eligibility
  - R1 Only eligible voters can vote.
  - R2 Eligible voters can vote only once.
- Accuracy
  - R3 It is not possible for a vote to be altered.
  - R4 It is not possible for a legitimate vote to be eliminated from the final tally.
  - R5 An invalid vote can not be counted in the final tally.
- Privacy
  - R6 No one can determine for whom any other voter voted.
  - R7 No voter can prove to other parties for whom he voted.
- Verifiability
  - R8 It can be verified by anyone independently whether all votes have been counted correctly.
  - R8a The verification can be done by trusted representatives/programs.
  - R8b Any voter can verify that his own vote has been taken into account in the final tally.
- Correctability
  - R9 Mistakes can be corrected without revealing for whom a voter voted.
  - R9a Mistakes can only be corrected by revealing for whom a voter voted.
  - R9b Mistakes can only be detected, but can not be corrected.

- Robust
  - R10 The voting can only be disrupted when some specified crucial conditions are reached.
- Flexibility
  - R11 It is allowed that the vote consists of a variety of question formats, rather than just yes or no (single bit) answer.
- Convenience
  - R12 The protocol does not incur expensive computations and communications.
  - R13 Minimal equipment is required.
- Mobility
  - R14 There are no restrictions on the location in which a voter can cast a vote.
  - R14a A voter can only vote in some designated voting booths.
- Fairness
  - R15 The intermediate result of the election does not leak out. Otherwise, the voting can be affected by taking advantage of this information.
- Other
  - R16 Everyone knows who voted and who did not.

Secure electronic voting protocols have been proposed by many researchers. However, few of them in the literature can satisfy all of the requirements above. The difficulty lies partially in that some of the requirements are divergent and are not straightforward to meet at the same time. For example, the legitimacy of the voters must be examined to satisfy the eligibility requirement, while the votes must be kept secret for privacy. In other words, it must be ensured that a secret vote is from a legitimate voter. As another example, while from whom a vote was cast is kept secret, it is desirable for a voter to be able to verify that his vote has been correctly counted in the final tally, i.e., to achieve verifiability. Yet another example, a voter desires to keep some evidence, or some form of receipt from the voting authority, to correct any faulty behavior observed in the final tally. However, it is not desirable for a voter to be able to prove to a third party for whom he has voted. This is to prevent ticket buying or extortion. Also, while it is convenient for a voter to cast a vote sitting home through the Internet, it is difficult to prevent others standing around from watching how he votes. There are more to be explained in the context.

Many of the problems have been solved to some extent in the literature, based on some assumption. Several schemes have been proposed, which can be generally divided into two classes. The first class consists of the schemes without administrators [8, 9, 13, 2, 5, 11]. All of the procedures are performed by the participating voters. The communication and computation taken to verify the correctness of the actions are usually high, which makes this approach interesting only theoretically [10]. The other class consists of schemes with administrators [10]. There are two possibilities to submit votes. First, the individual vote is divided into multiple shares which are then sent to different authorities [6, 3, 12]. If the different authorities conspire, the privacy of the voters can not be preserved [10, 16].The other possibility is to send votes through an anonymous communication channel [5, 4, 10]. It provides unconditionally security against tracing the votes [5].

In this paper, we propose a voting scheme using the anonymous communication channel, which attempts to meet a variety of requirements described above. Our work is an improvement based on the work in [10, 14]. Section 2 describes the key idea in [10]. Our approach is presented in Section 3, which is followed by the discussion of extended work. The last section is the conclusion.

1.2 Notations

The following notations are used throughout this paper.

A :	the administrator (the voting authority)
V_i :	a voter V_i
v_i :	the vote cast by V_i
u_i :	V_i's (well − known) public key
r_i :	V_i's private key
$< u_i', r_i' >$:	V_i's session public − key/ private − key pair
k_i :	V_i's (ordinary) session key
β_{f_i} :	blinding function using factor f_i
$\beta_{f_i}^{-1}$:	unblinding function using the factor f_i

$H(msg)$: hash value of a message msg
σ_{r_i} : signing using the private key r_i
$\sigma_{r_i}(msg)$: $E_{r_i}(H(msg))$
E_{u_i} : encryption using public key u_i
D_{r_i} : decryption using private key r_i
$V_i \rightarrow A$: V_i sends A a message
$V_i \rightsquigarrow A$: V_i sends A a message anonymously

2 Related Work

Fujioka *et al.* proposed a practical secure voting scheme for large scale elections [10]. The scheme ensures the privacy of the voters and fairness of the elections. The computation and communication overhead is reasonable even if the number of voters is large. The scheme satisfies a great number of requirements for a secure voting system. In short, there are three participants in their scheme, i.e., an administrator, a counter, and voters. The administrator is responsible for checking the legitimacy of voters, while the counter is responsible for collecting, opening, and counting the votes. A simplified description of their scheme is given below. Note that k_i is the ordinary session key.

(F1) $V_i \rightarrow$ A : b_i
where $b_i = \beta_{f_i} E_{k_i}(v_i)$
(F2) A $\rightarrow V_i$: a_i
where $a_i = \sigma_{r_A}(b_i)$
(F3) $V_i \rightsquigarrow$ C : d_i
where $d_i = \beta_{f_i}^{-1}(a_i)$
(F4) C : C publishes the voting result
(F5) V_i : After checking that $E_{k_i}(v_i)$ is on the list, V_i sends k_i to C anonymously.
(F6) C : C retrieves v_i and publishes the votes

Their assumption that no voters abstain from voting is too restrictive and may not apply to the situation in a practical election. The most serious problem may be that a malicious administrator may cast forged votes for abstaining voters [7], or for those voters who register but do not vote [1]. The whole election may be disrupted when it is found by the voters. Our work to be described later will solve this problem.

3 The Voting Protocol

3.1 The Structure of the Protocol

In the work of Fujioka *et al.* [10], if the administrator's fraud is found, the social and political cost is extremely high to conduct a second election. We propose to add a preparation stage to generate a registration list, on which the legitimate voters register to express their willingness to vote. The idea is to incorporate a *session public-key/private-key pair* into the scheme so that a registered voter may choose to abstain in the voting stage without the risk of disrupting the election. Even a dishonest administrator can not vote for abstaining voters. Our scheme consists of three stages which are summarized in brief as follows.

Preparation stage: V_i sends A a blinded session public key. After checking the legitimacy of the voter, A signs and returns it to V_i. V_i unblinds the signed session public key and obtains a session public key signed by A. V_i anonymously sends the signed session public key to A for publication. The session public key will be used as a permit with the vote. Any problems with the legitimacy of the registered voters and the published session public keys can be cleared in this stage. Once every party is convinced of the correctness of the registration list, no one may falsify the results in the following stages.

Voting Stage: V_i encrypts his vote with a randomly chosen ordinary *session key*, and signs the encrypted vote with his session private key. The signed encrypted vote with the corresponding session public key are sent to the administrator anonymously. Since the session public key represents a permit, anyone having the session private key has the right to vote. After the administrator verifying and publishing his vote, the voter sends the ordinary session key to the administrator to open the vote.

Counting Stage: The administrator retrieves the votes and publishes the results.

3.2 The Protocol

Preparation Stage:

(P1) A: A announces the list of legitimate voters.

(P2) $V_i \rightarrow$ A: $< ID_i, b_i, s_i >$
(P2.1) where $b_i = \beta_{f_i}(u_i')$,
(P2.2) $s_i = \sigma_{r_i}(ID_i, b_i)$

(P3) A: if (V_i is a legitimate voter,
(P3.1) V_i has not registered yet, and
(P3.2) s_i is V_i's signature)
(P3.3) A $\rightarrow V_i$: $< a_i >$
(P3.4) where $a_i = \sigma_{r_A}(b_i)$
(P3.5) else
(P3.6) A rejects V_i's request

(P4) V_i : if (a_i is A's signature of b_i)
(P4.1) $V_i \rightsquigarrow$ A: $< u_i', d_i >$
(P4.2) where $d_i = \beta_{f_i}^{-1}(a_i)$
(P4.3) else
(P4.4) V_i claims A's fault

(P5) A: A receives $< u_i', d_i >$
(P5.1) if (u_i' has not been used and d_i is A's signature)
(P5.2) A saves $< u_i', d_i >$
(P5.3) else
(P5.4) $< u_i', d_i >$ is rejected

(P6) A publishes the list of $< ID_i, b_i, s_i >$ for every legitimate V_i who have sent their requests for A's blind signature.
(P7) A publishes the list of $< u_i', d_i >$ for every legitimate V_i who have anonymously sent their session public keys, blindly signed by A.

Voting Stage:

(P8) V_i : V_i generates an ordinary session key k_i
(P8.1) $V_i \rightsquigarrow$ A: $< e_i, x_i, u_i' >$
(P8.2) where $e_i = E_{k_i}(v_i)$, and
(P8.3) $x_i = \sigma_{r_i'}(e_i)$

(P9) A: if (u_i' is on the list in step P7,
(P9.1) u_i' is not used yet, and
(P9.2) $H(e_i) == D_{u_i'}(x_i)$)
(P9.3) A publishes $< l_i, e_i, x_i, u_i' >$
(P9.4) where l_i is a serial number for the list
(P9.5) else
(P9.6) A rejects $< e_i, x_i, u_i' >$

(P10) V_i : if ($< l_i, e_i, x_i, u_i' >$ has been published)
(P10.1) $V_i \rightsquigarrow$ A: $< l_i, k_i, \sigma_{r_i'}(k_i) >$

Counting Stage:

(P11) A: A computes $v_i = D_{k_i}(e_i)$
(P11.1) A checks that v_i is a valid vote.
(P11.2) A publishes $< l_i, e_i, x_i, u_i', k_i, v_i >$

Initially, A publishes the list of legitimate voters. Seeing himself on the list, a voter V_i registers by sending his identification number ID_i and a blinded session public key. In order to prove that the session public key is sent by the legitimate voter and to ensure that the message is not altered maliciously, the message is sent with V_i's signature s_i. In step P3, A ensures that only legitimate voters can vote and can vote exactly once. Then A blindly signs V_i's session public key, which authorizes V_i the right to cast a vote later in the voting stage. Note that the session public key represent a permit. Whoever has the corresponding session private key has the right to cast his vote. Since the session public-key/private-key pair is generated by V_i, only V_i has the session private key. Hence, it ensures that only V_i can cast his own vote. In step P4, V_i unblinds A's signature and obtains a session public key signed by A. V_i sends the session public key with its signature to A for publication through an anonymous channel so that A does not have the knowledge of the sender's identification. In this way, the privacy of the voter can be preserved.

A publishes those legitimate voters who have registered in step P6. V_i's signature, s_i , can be used by any interested party to verify that it is ID_i who has registered. The verification is done by using V_i's well-known public key Ku_i. In step P7, A publishes $< u_i', d_i >$ so that the number of session public keys, or the number of permits to vote, can be checked against the number of registered voters in step P6. The number should be the same. Any interested party can compare the lists in steps P6 and P7 to make sure that A does not submit forged session public keys for voters who do not register. If any problem has been found, it can be corrected

before the election is conducted. After A publishes the list of $< u_i', d_i >$ for every registered V_i V_i can verify whether his is on the list. If V_i finds that his session public key is not on the list, V_i can protest by submitting his $< u_i', d_i >$ to some other authorities. The correction process may continue until every party is convinced. The preparation stage ensures that only legitimate voters have the right to vote, using their session public-key/private-key pairs.

In the voting stage, V_i generates an ordinary session key, with which the vote is encrypted. Then V_i signs the encrypted vote using its session private key. After verifying the legitimacy of the encrypted vote, A publishes $< l_i, e_i, x_i, u_i' >$, where $l_i = 1, 2, \ldots$. Seeing that his vote is published, V_i sends the session key to A anonymously.

In the counting stage, A retrieves the votes using the session keys, and checks if the votes are valid. A publishes the final results of the election.

4 Extension

The scheme presented above enhances the robustness of the voting system. However, it still suffers from some problems to be discussed in the following.

4.1 Fairness

First, the fairness of the voting system is not reached. A voter submits his session key immediately after his vote has been published, and ends the voting stage in one session. The administrator may open the votes by using the session keys. This reveals the intermediate results of the election, which may be used by some party to influence the strategy for the election. To prevent the fraud, another public-key/private-key pair can be secret-shared by some authorities. Instead of submitting his plain session key, the voters submits the session key encrypted by the secret-shared public key. Since no single authority may decrypt the encrypted session key, the intermediate result of the election will not leak out.

4.2 Prevention of Ticket Buying

Another problem with the scheme is more serious. The voters have receipts to prove for whom they vote. This may lead to ticket buying or extortion. There are some possible ways that a voter can prove how he votes. First, b_i published in step P6 can be linked to u_i' published in step P11.2 by revealing the blinding factor f_i. In this way, V_i can show how he votes. The solution to this problem is to hide b_i and s_i in step P6. Note that the purpose of publishing s_i is to show that the voter with ID_i does have registered in the preparation stage. Any signature of V_i serves the same function. Hence, we can add into the tuple sent by V_i to A in step P2 an additional signature of V_i

$$s_i^t = \sigma_{r_i}(t)$$

where t is a plain text or a serial number to identify this election. And by having the tuple $< ID_i, s_i^t >$ be published in step P6, one can not link the tuple to the vote in step P11.2.

Secondly, the session public-key/private-key pair generated in the preparation stage before the voting takes place also serves as a receipt. A voter needs to save the key pair in order to cast his vote in the voting stage. A malicious candidate may force voters to reveal their session public keys or session private keys so that the candidate can monitor the result to see if the voters have cast the votes for him. As a partial solution to the problem of extortion, the session public-key/private-key pair can be generated on the fly in the preparation stage. At the end of the stage, the key pair is submitted to the administrator for storage such that the voter does not have the key pair with him. At the beginning of the voting stage, the voter requests his key pair from the administrator and proceed with the normal procedure. To accomplish this, V_i submits to A

$$< ID_i, E_{u_i}(g_i), \sigma_{r_i}(E_{u_i}(g_i)) >, \text{where}$$

$$g_i =< b_i, f_i, a_i, d_i, u_i', r_i' >$$

so that A does not have the knowledge of these personal information and the integrity and authenticity of the information are ensured.

Finally, the session private key of the voter can be used to show how he votes after the counting stage. As a partial solution to this problem, V_i submits his r_i' as well to A in step P10.1. Note that since A has published the encrypted vote signed with r_i' in step P9.3, no one can change the vote

thereafter. Also note that V_i has verified in step P10 that his vote has been published, he does not need r_i' any more. Hence, V_i does not have the session private key with him after he has cast his vote. It is also feasible that V_i sends his r_i' to some more authorities to ensure that his keys will not be ignored by malicious administrator.

5 Security Properties

Theorem 1 (Eligibility) Only Eligible voters can vote. And eligible voters can vote exactly once.

Proof. The eligibility of a voter is checked by the administrator in steps P3 and P3.2, while in step P3.1, multiple registration from the same voter is prohibited. The *permit*, the signed session public key, is issued only once to the eligible voter. With the permit, the voter can cast one vote. Double voting is checked in step P9.1. □

On the other hand, the administrator may act maliciously. In the scheme described in [10], it may be detected that a administrator may have cast forged votes for abstaining voters. However, there is no way to trace which votes are invalid. In our scheme, it is shown in the theorem of Robustness later that the administrator can not cast forged votes.

Theorem 2 (Accuracy) It is not possible to alter a vote or to eliminate a vote from the final tally. An invalid vote can not be counted in the final tally.

Proof. It is not possible to alter a vote, since the vote is signed with the session private key only known to the voter himself. A legitimate vote can not be eliminated by a malicious administrator. If the administrator does not accept a legitimate vote in step P9, the voter can protest by sending his vote to the candidates or to some other authorities. Once his vote is published in step P9.3, the vote can not be eliminated. It is also not possible to count an invalid vote, by the same reason described in the theorems of Eligibility and Robustness. □

Theorem 3 (Privacy) No one can determine for whom any other voter voted.

Proof. The legitimacy of a vote comes from the permit, or the session public key. Since the session public key has been blindly signed by the administrator even the administrator does not know to whom a session public key belongs. The session public keys and the votes thereafter have been sent to the administrator through an anonymous channel. There is no way to trace them back to the originating voters. Hence, the privacy of the voters is preserved. □

Note that the requirement R7 states that no voter can prove for whom he voted. An extended work to fulfill this purpose has been described in section 4.2

Theorem 4 (Verifiability) Any voter can verify that his vote has been taken into account in the final tally.

Proof. The verification can be performed by voters against the list published by the administrator in step P11.2. □

Theorem 5 (Correctability) Mistakes can be corrected without revealing for whom a voter voted.

Proof. In the preparation stage, any mistake can be claimed or corrected before the voting takes place. If there is mistake with the list in step P7, the voter can claim the fault by anonymously sending the correct $< u_i', d_i >$ to some other authorities. Since d_i is a valid signature from the administrator, the fault can be identified and corrected. In the voting stage, if one's vote is not correctly published by the administrator in step P9.3, the voter can protest by anonymously submitting $< e_i, x_i, u_i' >$ to some other authorities. The same procedure can be performed if there is any problem with the final tally in step P11.2. □

Theorem 6 (Robustness) The voting can not be disrupted by a malicious administrator.

Proof. The administrator can not make false registration for the voters who do not register in the preparation stage, since the administrator does not have the private keys of the voters to sign for the registration. Alternatively, the administrator may forge a false session public key and sign it to

make it look like legitimate. However, this can be detected when the number of entries on the list in step P7 is checked against that in step P6. Also any voters aware of the missing of their $< u_i', d_i >$ can protest by publishing $< u_i', d_i >$. Such problems can be solved before the election takes place. Once agreement of the list of session public keys in step P7 is nailed down, only the legitimate voters having the corresponding session private keys can cast valid votes. Even a malicious administrator can not cast votes for the voters who have registered but do not bother to vote. □

Theorem 7 (Flexibility) It is allowed that the vote consists of a variety of question formats, rather than just yes or no (single bit) answer.

Explanation. While some voting schemes [15] allow the voter only to make two choices, i.e., yes or no, it is clear that there is no such restriction in our scheme. □

Theorem 8 (Convenience) The protocol does not incur expensive computations and communications.

Explanation. While some voting schemes [6, 3, 12] may incur expensive computations and communications, the overhead in our scheme is reasonable. □

Theorem 9 (Mobility) There are no restrictions on the location from which a voter can cast a vote; or a voter can vote from some designated voting booths.

Explanation. It is clear that a voter can register and vote from home or elsewhere in our scheme. However, in our extension, to prevent ticket buying and extortion, one should vote in some designated voting booths. Any one casting his vote in home can have other people watch how he votes. □

Theorem 10 (Fairness) The intermediate result of the election will not leak out.

Proof. In our extension, the session key used to open the vote is encrypted by a public key secret-shared by several authorities. Only when more than a specified number of the authorities work together can the corresponding private key be recovered. □

As for the requirement R16, we do not know who has voted and who has not. We only know who has registered and who has not.

6 Conclusion

In this paper, we propose a voting protocol suitable for large scale elections. The protocol is divided into three stages. In the preparation stage, the voter has his session public key signed by the administrator, which serves as a permit to vote. Only the voter, which owns the corresponding session private key, can cast a legitimate vote in the voting stage. Hence, even a malicious administrator can not cast forged votes. Since the ownership of the session public key is not revealed by utilizing the technique of the blind signature and the anonymous channel, the privacy of the voter is preserved. We show that the voting protocol satisfies a variety of requirements for practical voting systems. An extended work to further improve the security of the protocol is also discussed.

References

[1] Ahmad Baraani-Dastjerdi, Josef Pieprzyk, and Reihaneh Safavi-Naini. A review study on electronic election. Technical report, University of Wollongong Department of Computer Science, Jun. 1994.

[2] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computations. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, pages 1–10, May 1988.

[3] J. C. Benaloh and M. Yung. Distributing the power of a government to enhance the privacy of voters. In *Proceedings of the 5th ACM Symposium on the Principles in Distributed Computing*, pages 52–62, 1986.

[4] C. Boyd. A new multiple key cipher and an improved voting scheme. In *Advances in*

Cryptology-EUROCRYPT '89 Proceedings, pages 617–625. Springer-Verlag, 1990.

[5] D. Chaum. Elections with unconditionally secret ballots and disruption equivalent to breaking rsa. In *Advances in Cryptology-EUROCRYPT '88 Proceedings*, pages 177–181. Springer-Verlag, 1988.

[6] J. D. Cohen and M. H. Fischer. A robust and verifiable cryptographically secure election scheme. In *Proceedings of the 26th Annual IEEE Symposium on the Foundations of Computer Science*, pages 372–382, 1985.

[7] L. F. Cranor and R. K. Cytron. Design and implementation of a security-conscious electronic polling system. Technical Report WUCS-96-02, Washington University Computer Science Technical Report, Feb. 1996.

[8] R. DeMillo, N. Lynch, and M. Merritt. Cryptographic protocols. In *Proceedings of the 14th Annual Symposium on the Theory of Computing*, pages 383–400, 1982.

[9] R. DeMillo and M. Merritt. Protocols for data security. *Computer*, 16(2):39–50, Feb. 1983.

[10] A. Fujioka, T. Okamoto, and K. Ohta. A practical secret voting scheme for large scale elections. In *Advances in Cryptology-AUSCRYPT '92 Proceedings*, pages 244–251. Springer-Verlag, 1993.

[11] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 218–229, May 1987.

[12] K. R. Iversen. A cryptographic scheme for computerized general elections. In *Advances in Cryptology-CRYPTO '91 Proceedings*, pages 405–419. Springer-Verlag, 1992.

[13] M. Merritt. Cryptographic protocols. In *Ph.D. dissertation, Georgia Institute of Technology, GIT-ICS-83/6*, Feb. 1983.

[14] H. Nurmi, A. Salomma, and L. Santean. Secret ballot elections in computer networks. *Computers & Security*, 10:553–560, 1991.

[15] K. Sako and J. Kilian. Secure voting using partially compatible homomorphisms. In *Advances in Cryptology-CRYPTO '94 Proceedings*, pages 411–424. Springer-Verlag, 1994.

[16] Bruce Schneier. *Applied Cryptography*. John Wiley & Sons, second edition, 1996.