Double Voter Perceptible Blind Signature Based Electronic Voting Protocol

Yaser Baseri[1], Mohsen Pourpouneh[2], Javad Mohajeri[3]

[1,3] *Electronics Research Center,Sharif University of Technology,Tehran, Iran.*
[2] *Department of Mathematics, Shahid Beheshty University, Tehran, Iran.*

Abstract

Mu et al. proposed an electronic voting protocol and claimed that it protects voter's anonymity, detects double voting and authenticates eligible voters. It has been shown that it does not protect voter's privacy and prevent double voting. After that several improvement have been presented on it which suffer from the same weaknesses. One of the latest improvement on it, relates to Rodriguez-Henriquez et al. In this paper first we present some weaknesses of Rodriguez-Henriquez et al.'s protocol and then present a new efficient e-voting protocol based on Okamoto's blind signature to overcome the proposed attacks.

Key words:
Electronic voting, Anonymity of voter, Unforgeability of ticket,
Perceptibility of double voting, Security of voting, Blind signature.

1. Introduction

Nowadays computers are almost everywhere and they are used for many purposes. One of these purposes is electronic voting. Using network computer and internet, traditional voting can be substituted by electronic voting, which speeds up election process decreases costs and facilitate voting process.

Electronic voting schemes can be classified into three types: blind signature based electronic voting schemes [4], [10], [21], [13], homomorphic encryption based electronic voting schemes [3], [15], and the schemes which

Email address: yaser_baseri@alum.sharif.ir, m.pourpouneh@mail.sbu.ac.ir, mohajer@sharif.edu (Yaser Baseri[1], Mohsen Pourpouneh[2], Javad Mohajeri[3])

use randomization such as the schemes that employ mixnets [6], [7]. In the schemes which use blind signature, the voter first gets a token which is a blindly signed message unknown to any one except him, and then sends his token together with his vote anonymously.

One of the first schemes which is based on blind signature and used to claim that it can detect double voters relates to Mu and Varadharajan [16]. They also claimed that their scheme is suitable for large scale elections. They have proposed two versions of an electronic voting scheme to be applied over network. Both versions are based on the ElGamal digital signature [9] without any anonymous channel. One of these schemes assumes that the authentication server is trusted, and therefore it does not generate any voting ticket without the voter's consent. In this version the authentication server does not leak out any information to the voting server or ticket counting server. The other version assumes that authentication server is not trusted, which is closer to truth. In 2003 Chien et al. [5] showed that Mu-Varadharajan's schemes suffers from weaknesses include: 1) the authentication server can easily identify the owner of a cast ballot, 2) a valid voter can vote more than once without being detected, 3) any one can forge ballot without being authenticated. In 2003 Lin et al. [14] proposed an improvement on Mu and Varadharajan's scheme. They improved the weakness that voters could successfully vote more than once without being detected. The proposed scheme did not require any special voting channel and detect double voting effectively. Yang et al. in 2004 [20] proposed another improvement on Mu-Varadharajan's scheme. Although their scheme is resistance against the attacks which has proposed in [5], it can not determine the identity of double voters. In 2005 Hwang et al. [11] represented an attack on Lin et al. protocol. They showed that the Lin et al.'s modification allows the authentication server to identify the voters of published tickets so that voters will lose their privacy. They also proposed a new scheme to solve this problem and enhance the security. They used two generators so that after publishing all cast tickets by ticket counting server, authentication server could not trace the owner of the tickets. By these changes they tried to improve the privacy of voters in Lin et al. protocol. However Hwang et al. scheme had the weakness that an eligible voter with a valid ticket could vote more than once without being detected [2]. Furthermore in another paper, Asaar et al. [1] proposed one more scheme based on Lin et al. scheme. Their scheme resists against the attacks which have been proposed in [2], [14]. In 2007 F. Rodriguez-Henriquez et al. [19] proposed another improvement over the Lin

et al. scheme. They presented a fully functional RSA/DSA-based e-voting protocol for online elections. They presented a weakness of Lin et al. scheme arising from the structure of ElGamal digital signature. For preventing the proposed weakness, they substituted the ElGamal digital signature employed by other protocols with DSA signature [8]. These changes guarantee that independently choosing values by the voter and authentication server would not have undesirable effect on the ticket obtaining procedure. But we can employ some attacks on their scheme. First attack allows the authentication server to identify voters of published tickets at will, thereby losing voters privacy. Second and third attacks allow malicious voters to forge a ballot without being detected, which vulnerable the unforgeability of tickets.

In this paper first we describe the weaknesses of Rodriguez-Henriquez et al.'s protocol as one of the last improvement on Mu et al.'s protocol which has not been already investigated. All the electronic voting schemes that mentioned above have the problem due to their employed structure that cause vulnerability of them against the attacks which is presented until now. In our new scheme by using another structure for authentication and detecting double voters, we prevent the attacks which have presented until now on this family of protocols. Section 2 reviews Rodriguez-Henriquez et al.'s protocol and it's weaknesses. Section 3 mentions digital signature due to Okamoto. Section 4 proposes the new e-voting protocol. Finally the conclusion is given by section 6.

2. Rodriguez-Henriquez et al.'s protocol and it's failures

In this section first we review the protocol due to Rodriguez-Henriquez et al. Then we present some weaknesses of it.

2.1. Rodriguez-Henriquez et al.'s scheme

The scheme consists of three phases: authentication, voting and counting. In order to describe the scheme, we will use the following notations:

- V: voter's name; AS: authentication server; VS: voting server; TCS: ticket counting server
- t: timestamp
- q: DSA parameter, $2^{159} < q < 2^{160}$

- p: for a given l such that $0 \leq l \leq 8$, let p be a prime such that $2^{511+64l} < p < 2^{512+64l}$, with the property that $q|(p-1)$
- g: a generator for $\mathbb{Z}^*_{p-1}$
- a: DSA private key $1 \leq a \leq q-1$
- $\alpha = g^{(p-1)/q} \bmod p$
- $y = \alpha^a \bmod p$
- $Cert$: digital certificate issued by an authority
- $\|$: bit concatenation
- $(e_X,\ n_X)$, d_X: a pair of RSA keys for entity X, where $n_X = p_1 \times p_2$, p_1 and p_2 are two large primes and $e_X \times d_X = 1 \bmod \phi(n_X)$
- $ENC_x(m)$ and $VER_x(m)$ denote the RSA public operation applied over m using entity's X public key e_X.
- $SIG_x(m)$ and $DEC_x(m)$ denote the RSA private operation applied over m using entity's X private key d_X.

2.1.1. Authentication

1. The voter chooses two blind factors $b_1, b_2 < n_{AS}$ and random numbers $k_1 < q/3$ and a in $\mathbb{Z}^*_{q-1}$. Then he computes the following equations:

$$\begin{aligned} y &= \alpha^a \ mod\ p \\ z_1 &= (y\ .\ ENC_{AS}(b_1))\, mod\ n_{AS} \\ z_2 &= ((\alpha^{k_1})\ mod\ p\ .\ ENC_{AS}(b_2))\, mod\ n_{AS} \end{aligned} \tag{1}$$

   where a is the voter's private DSA key and p and α are DSA public domain parameters. Then the voter sends $\{V,\ AS,\ Cert_V,\ t,\ z_1,\ z_2,\ SIG_v(z_1\|z_2\|t)\}$ to the AS.
2. After validating V's identity, AS chooses a unique random number $k_2 < q/3$ for the voter and stores it in its own database as an identification

of V. Then AS generates z_3, z_4, z_5, z_6 as follow:

$$\begin{aligned} z_3 &= ENC_V(k_2 \| t) \\ z_4 &= SIG_{AS}(z_1 \times AS) = b_1.SIG_{AS}(y \times AS) \\ z_5 &= SIG_{AS}(z_2 \times (\alpha^{k_2} \bmod p) \times AS) \\ &= b_2.SIG_{AS}(\alpha^{k_1+k_2} \bmod p \times AS) \\ z_6 &= SIG_{AS}(z_2^2 \times (\alpha^{k_2} \bmod p) \times AS) \\ &= b_2^2.SIG_{AS}(\alpha^{2k_1+k_2} \bmod p \times AS) \end{aligned} \tag{2}$$

After that, AS sends the following reply message to V:

$$\{AS,\ V,\ z_3,\ ENC_V(z_4 \| z_5 \| z_6 \| t)\}$$

3. The voter decrypts z_3 to get k_2. Furthermore, he unblinds the signatures z_4, z_5 and z_6 to obtain the values of s_1, s_2 and s_3 as follows:

$$\begin{aligned} s_1 &= z_4 \times b_1^{-1} = SIG_{AS}(y \times AS) \\ s_2 &= z_5 \times b_2^{-1} = SIG_{AS}(\alpha^{k_1+k_2} \bmod p \times AS) \\ s_3 &= z_6 \times b_2^{-2} = SIG_{AS}(\alpha^{2k_1+k_2} \bmod p \times AS) \end{aligned} \tag{3}$$

2.1.2. Voting phase

1. In the voting phase the voter proceeds to sign the ballot m using the DSA scheme and a, x_1 and x_2 as private keys. The voter is able to generate x_1 and x_2 because he has already decrypted k_2. Notice that the two DSA signatures consists of the pairs (r_1, s_4) and (r_2, s_5) which are computed as follows:

$$\begin{aligned} x_1 &= k_1 + k_2 \\ x_2 &= 2k_1 + k_2 \\ r_1 &= (\alpha^{x_1} \bmod p) \bmod q \\ r_2 &= (\alpha^{x_2} \bmod p) \bmod q \\ s_4 &= x_1^{-1}(m + ar_1) \bmod q \\ s_5 &= x_2^{-1}(m + ar_2) \bmod q \end{aligned} \tag{4}$$

Furthermore the voter computes the values l_1 and l_2 in the following way:

$$\begin{aligned} l_1 &= [((\alpha^{k_1} \bmod p) \bmod n_{AS}) \times ((\alpha^{k_2} \bmod p) \bmod n_{AS})] \bmod n_{AS} \\ l_2 &= [((\alpha^{k_1} \bmod p)^2 \bmod n_{AS}) \times ((\alpha^{k_2} \bmod p) \bmod n_{AS})] \bmod n_{AS} \end{aligned} \tag{5}$$

These last two values together with r_1 and r_2 are encapsulated by using the *Chinese Residue Theorem*. That is done with the goal of allowing

VS to perform the corresponding verifications in the proper arithmetic (either modules n_{AS} or modules q) and also in order to keep the size of the vote as small as possible.

$$\begin{aligned} pr_1 &= [(r_1 \times n_{AS}) + (l_1 \times q)] \, mod \, (n_{AS}.q) \\ pr_2 &= [(r_2 \times n_{AS}) + (l_2 \times q)] \, mod \, (n_{AS}.q) \end{aligned} \tag{6}$$

Finally, the voting ticket is generated as follow:

$$Ticket = \{s_1,\ s_2,\ s_3,\ s_4,\ s_5,\ y,\ pr_1,\ pr_2,\ m\} \tag{7}$$

2. The voter sends the ticket to *VS*. *VS* verifies the signatures to validate the ticket. These verifications are done in the following way:

$$\begin{aligned} &(AS \times y) mod \, n_{AS} \stackrel{?}{=} VER_{AS}(s_1) \\ &(AS \times pr_1.q^{-1}) mod \, n_{AS} \stackrel{?}{=} VER_{AS}(s_2) \\ &(AS \times pr_2.q^{-1}) mod \, n_{AS} \stackrel{?}{=} VER_{AS}(s_3) \\ &r_1 \stackrel{?}{=} DSAverifier(r_1, s_4) \\ &r_2 \stackrel{?}{=} DSAverifier(r_2, s_5) \end{aligned} \tag{8}$$

Note that the DSA signatures are verified by the standard procedure shown below:
$DSAverifier(r, s)$ {

1. *Check whether* $0 < r < q$ *and* $0 < s < q$
2. $w = s^{-1} \, mod \, q$
3. $u_1 = w \, . \, m \, mod \, q$
4. $u_2 = r \, . \, w \, mod \, q$
5. $v = (\alpha^{u_1} y^{u_2} \, mod \, p) \, mod \, q$
6. *return* v}

3. If all five signatures are correctly verified, *VS* will accept and store the ticket sent by the voter as a valid one. Once the voting election process has been completed, *VS* sends all valid votes that were received to *TCS* over the communication network.

2.1.3. Counting phase

In this phase *TCS* detect malicious voters that may have sent two or more tickets with different votes. In order to detect double voters, we consider the scenario where a given voter uses the same key to sign different votes. Therefore, *TCS* will receive at least two tickets with the following form:

$$B_1 = \{s_1, s_2, s_3, s_4, s_5, y, pr_1, pr_2, m\}$$
$$B_1 = \{s_1, s_2, s_3, s'_4, s'_5, y, pr_1, pr_2, m'\}$$

With the information contained in these two tickets, *TCS* can calculate x_1, x_2, k_1 and k_2 by the following equations:

$$\begin{aligned} x_1 &= \tfrac{m'-m}{s'_4-s_4} \bmod q \\ x_2 &= \tfrac{m'-m}{s'_5-s_5} \bmod q \\ k_1 &= x_2 - x_1 \\ k_2 &= x_1 - k_1 \end{aligned}$$

Finally, *TCS* can identify the malicious voter by searching *AS*'s database to find out which voter is associated with the unique random number k_2.

2.2. Weaknesses of this scheme

2.2.1. First attack (the attack on anonymity of voters):

Suppose that *TCS* has published all cast tickets. *AS* wants to trace the owner of the *Ticket* $\{s_1, s_2, s_3, s_4, s_5, y, pr_1, pr_2, m\}$. *AS* can perform the following procedure to identify the owner of this ticket.

1. *AS* uses his own public key to extract the message which is signed:

$$\begin{aligned} f_2 &= s_2^{e_{AS}} \bmod n_{AS} = \alpha^{k_1+k_2} \bmod p \times AS \bmod n_{AS} \\ f_3 &= s_3^{e_{AS}} \bmod n_{AS} = \alpha^{2k_1+k_2} \bmod p \times AS \bmod n_{AS} \\ x' &= \tfrac{f_3}{f_2} \bmod n_{AS} = \tfrac{\alpha^{2k_1+k_2}}{\alpha^{k_1+k_2}} \bmod n_{AS} = (\alpha^{k_1} \bmod p) \bmod n_{AS} \end{aligned} \tag{9}$$

   In which e_{AS} is the RSA public key of *AS*.
2. *AS* selects a record (V', k'_2) from his own database and checks the following equations:

$$\begin{aligned} &((x' \times ((\alpha^{k'_2} \bmod p) \bmod n_{AS})) \times AS) \bmod n_{AS} \stackrel{?}{=} f_2 \\ &[(((x'^2 \bmod p) \bmod n_{AS}) \times ((\alpha^{k'_2} \bmod p) \bmod n_{AS})) \times AS] \bmod n_{AS} \stackrel{?}{=} f_3 \end{aligned} \tag{10}$$

If these equations hold, V' would be the owner of this ticket, else AS chooses another record form his database and checks equations (10), until the owner of this vote is determined.

3. These steps are done until the vote of all voter is determined.

2.2.2. Second attack(the attack on unforgeability of tickets):

When the voter gets a valid ticket from AS, he can forge a new ticket by the following procedure:
The malicious voter chooses three nonzero integer θ_1, θ_2, $\theta_3 \in \mathbb{Z}_q$ randomly and computes a' and y' as follow:

$$\begin{aligned} a' &= a + (\theta_1 e_{AS} \bmod q) \bmod q \\ y' &= \alpha^{a'} \bmod p \end{aligned} \tag{11}$$

Then he computes the values $\{s'_1, s'_2, s'_3, s'_4, s'_5, r'_1, r'_2, l'_1, l'_2, pr'_1, pr'_2\}$ as forged values of $\{s_1, s_2, s_3, s_4, s_5, r_1, r_2, l_1, l_2, pr_1, pr_2\}$ by the following procedure:

$$\begin{aligned} s'_1 &= s_1 \times ((\alpha^{\theta_1} \bmod p) \bmod n_{AS}) \bmod n_{AS} \\ s'_2 &= s_2 \times ((\alpha^{\theta_2} \bmod p) \bmod n_{AS}) \bmod n_{AS} \\ s'_3 &= s_3 \times ((\alpha^{\theta_3} \bmod p) \bmod n_{AS}) \bmod n_{AS} \\ x'_1 &= k_1 + k_2 + \theta_2 e_{AS} \bmod q \\ x'_2 &= 2k_1 + k_2 + \theta_3 e_{AS} \bmod q \\ r'_1 &= (\alpha^{x'_1} \bmod p) \bmod q \\ r'_2 &= (\alpha^{x'_2} \bmod p) \bmod q \\ s'_4 &= {x'_1}^{-1}(m' + a' r'_1) \bmod q \\ s'_5 &= {x'_2}^{-1}(m' + a' r'_2) \bmod q \\ l'_1 &= [((\alpha^{k_1} \bmod p) \bmod n_{AS}) \times ((\alpha^{k_2 + \theta_2 e_{AS}} \bmod p) \bmod n_{AS})] \bmod n_{AS} \\ l'_2 &= [((\alpha^{k_1} \bmod p)^2 \bmod n_{AS}) \times ((\alpha^{k_2 + \theta_3 e_{AS}} \bmod p) \bmod n_{AS})] \bmod n_{AS} \\ pr'_1 &= [(r'_1 \times n_{AS}) + (l'_1 \times q)] \bmod (n_{AS}.q) \\ pr'_2 &= [(r'_2 \times n_{AS}) + (l'_2 \times q)] \bmod (n_{AS}.q) \end{aligned} \tag{12}$$

Finally the forged ticket would be

$$Ticket' = \{s'_1, s'_2, s'_3, s'_4, s'_5, y', pr'_1, pr'_2, m'\}$$

in which m' is the forged vote. Then the malicious voter sends his forged ticket to VS. VS checks the signature verification needed for ticket validation

as follow:

$$\begin{aligned}
&(AS \times y')mod\, n_{AS} \stackrel{?}{=} VER_{AS}(s'_1)\\
&(AS \times pr'_1.q^{-1})mod\, n_{AS} \stackrel{?}{=} VER_{AS}(s'_2)\\
&(AS \times pr'_2.q^{-1})mod\, n_{AS} \stackrel{?}{=} VER_{AS}(s'_3)\\
&r'_1 \stackrel{?}{=} DSAverifier(r'_1, s'_4)\\
&r'_2 \stackrel{?}{=} DSAverifier(r'_2, s'_5)
\end{aligned} \tag{13}$$

2.2.3. Third attack(the attack on unforgeability of tickets):

When a voter gets a valid ticket from AS, he can produce a forged ticket by exchanging the values of s_2, s_4 and pr_1 with s_3, s_5 and pr_2 respectively. So the forged ticket will be:

$$Ticket' = \{s_1, s_3, s_2, s_5, s_4, y, pr_2, pr_1, m\}$$

Since this ticket is different from the original one, it passes the following verifications:

$$\begin{aligned}
&(AS \times y)mod\, n_{AS} \stackrel{?}{=} VER_{AS}(s_1)\\
&(AS \times pr_2.q^{-1})mod\, n_{AS} \stackrel{?}{=} VER_{AS}(s_3)\\
&(AS \times pr_1.q^{-1})mod\, n_{AS} \stackrel{?}{=} VER_{AS}(s_2)\\
&r_2 \stackrel{?}{=} DSAverifier(r_2, s_5)\\
&r_1 \stackrel{?}{=} DSAverifier(r_1, s_4)
\end{aligned} \tag{14}$$

3. Okamoto's blind signature

In this signature [17] it suppose that the user wants to sign the message msg. The general parameters used in this signature are: two large prime numbers p and q such that $q|p-1$, two elements h_1 and h_2 in $\mathbb{Z}^*{}_p$ of order q such that solving discreet logarithm problem in group $G = \langle h_1 \rangle \approx \langle h_2 \rangle$ with respect to bases h_1 and h_2 be hard. It also suppose that two private keys and one public key of the signer are x_1, x_2 and $y = h_1^{x_1} h_2^{x_2}\, mod\, p$ respectively. We assume that $\mathcal{H}$ is a public one way hash function. The different stages of the signature run are illustrated in Figure 1.

The resulting signature for message msg is the triple (ε, ρ_2, ρ_1). A signature is valid if it satisfies

$$\varepsilon = \mathcal{H}(h_1^{\rho_1} h_2^{\rho_2} / y^{\varepsilon} \| msg).$$

signer		**user**
$(p, q, h_1, h_2, x_1, x_2)$		$(y = h_1^{x_1} h_2^{x_2}, msg)$
$u_1, u_2 \in_R \mathbb{Z}_q$ $a = h_1^{u_1} h_2^{u_2} \, mod \, p$		
	$\xrightarrow{a}$	
		$t_1, t_2, t_3 \in_R \mathbb{Z}_q$ $\alpha = a h_1^{t_1} h_2^{t_2} y^{t_3} \, mod \, p$ $\varepsilon = \mathcal{H}(\alpha \| msg)$ $c = \varepsilon - t_3 \, mod \, q$
	$\xleftarrow{c}$	
$re_1 = u_1 + c x_1 \, mod \, q$ $re_2 = u_2 + c x_2 \, mod \, q$		
	$\xrightarrow{re_1, re_2}$	
		$h_1^{re_1} h_2^{re_2} \stackrel{?}{=} a y^c \, mod \, p$ $\rho_1 = re_1 + t_1 \, mod \, q$ $\rho_2 = re_2 + t_2 \, mod \, q$
		$\varepsilon \stackrel{?}{=} \mathcal{H}(h_1^{\rho_1} h_2^{\rho_2} / y^{\varepsilon} \| msg)$ $\downarrow$ $(\varepsilon, \rho_1, \rho_2)$

Figure 1: Okamoto's Blind Signature

In [18] Pointcheval and Stern proved that Okamoto's digital signature is provably secure in random oracle model. Also Juels et al. in [12] proved the anonymity of this blind signature in a random oracle model. They also proved the unforgeablity of the signature.

4. The new electronic voting scheme

Our electronic voting environment involves at least the following parties: voters (V's), an authentication sever (AS), voting servers(VS's), a ballot counting server (BCS) and a trusted certificate authority (CA). The scheme consists of three phases: 1) voting preparation, in which the voter authenticates himself and gets a valid ticket from the authentication server, 2) voting and collecting ballot, in which the voter sends the ballot to a voting server,

then the voting server verifies the eligibility of the voter by checking signature of the authentication server which is in the ticket and then sends the ballot to the ballot counting server, 3) counting ballots in which the ballots are counted and double voters are detected.

Beside the allocation of certificates, the certificate authority publishes h_1, h_2 as the generators of $\mathbb{Z}_p$, g_0 as the generators of $\mathbb{Z}_{n_{CA}}$ and g_1, g_2 as the generators of $\mathbb{Z}_q$. Note that p and q are two large prime numbers such that $q|p-1$. It also publishes the RSA public keys of CA, AS and VS. Voters should get their own certificate from the certificate authority. Obtaining such certificate can be done via such a physical access to CA that could act as a local election server. In our scheme the certification of the voters also contains $g_1^{ID_v} mod\, n_{CA}$ in which ID_v is the identifier of the voter which is chosen by CA in $\mathbb{Z}_q$. Then CA chooses $u_1, u_2 \in_R \mathbb{Z}_q$ and computes $a = h_1^{u_1} h_2^{u_2} mod\, p$. Certificate authority sends $Cert_V$, a, $I = (g_0^{ID_v})^{d_{CA}} mod\, n_{CA}$ and $(u_1, u_2)^{e_{AS}}$ to the voter. Note that (e_{CA}, n_{CA}), d_{CA} are RSA public\private keys of the certificate authority and (e_{AS}, n_{AS}), d_{AS} are RSA public\private keys of the authentication server. Since all the certifications of voters are hold for a long time, the RSA public\private keys of the authorities must remain unchanged until these certifications are valid.

4.1. First phase: obtaining a voting ticket

Before starting this phase, AS chooses x_1 and x_2 as its own private keys and gets a certificate on corresponding public key y from certificate authority, while CA publishes y. In order to participate in election, voters should get a ticket from AS. In order to get a ticket, the voter should get the AS's signature on A, B_1, B_2, C. Getting ticket is done in three stages as follow:

1. Voter selects β_1, $\beta_2 \in \mathbb{Z}_q$ and computes $B_1 = {g_1}^{\beta_1} mod\, p$ and $B_2 = {g_2}^{\beta_2} mod\, p$. Then he chooses random value $\theta \in_R \mathbb{Z}_q$ and computes $A = ({g_1}^{ID_v} g_2)^{\theta} mod\, p$. He also chooses random value $\gamma \in_R \mathbb{Z}_{n_{CA}}$ to compute $C = I {g_0}^{\gamma} mod\, n_{CA}$. After that he selects three random values t_1, t_2, $t_3 \in_R \mathbb{Z}_q$ and uses a to compute $\alpha = a h_1^{t_1} h_2^{t_2} y^{t_3} mod\, p$ and computes $\varepsilon = \mathcal{H}(\alpha \| A \| B_1 \| B_2 \| C)$ and $c = \varepsilon - t_3$, Where $\mathcal{H}$ is a hash function. Finally the voter sends $(c, Cert_V, I, (u_1, u_2)^{e_{AS}})^{e_{AS}}$ to the AS.
2. When AS receives $(c, Cert_V, I, (u_1, u_2)^{e_{AS}})^{e_{AS}}$, first decrypts it and checks the validity of I by using the $Cert_V$. If I is valid, it decrypts $(u_1, u_2)^{e_{AS}}$ and gets u_1 and u_2. Then it computes the value of re_1 and

re_2 by the following equations and sends them to the voter.

$$\begin{aligned} re_1 &= u_1 + cx_1 \, mod \, q \\ re_2 &= u_2 + cx_2 \, mod \, q \end{aligned} \tag{15}$$

3. After receiving re_1 and re_2, the voter first verifies the following equation:

$$h_1^{re_1} h_2^{re_2} \stackrel{?}{=} ay^c \, mod \, p \tag{16}$$

Then computes the values of ρ_1 and ρ_2 as follow:

$$\begin{aligned} \rho_1 &= re_1 + t_1 \, mod \, q \\ \rho_2 &= re_2 + t_2 \, mod \, q \end{aligned} \tag{17}$$

After that it verifies the signature by checking the following equation:

$$\varepsilon \stackrel{?}{=} \mathcal{H}(h_1^{\rho_1} h_2^{\rho_2} / y^{\varepsilon} \| A \| B_1 \| B_2 \| C) \tag{18}$$

If the equation (18) is hold, the ticket of the voter would be

$$Ticket = \{\rho_1, \, \rho_2, \, \varepsilon, \, A, \, B_1, \, B_2, \, C\}. \tag{19}$$

4.2. Second phase: voting and collecting tickets

1. When the voter gets a ticket from the authentication server, he can cast his vote. He computes the following equations:

$$\begin{aligned} d &= \mathcal{H}_0(Ticket \| t) \\ r_0 &= (ID_v + \gamma e_{CA}) \, mod \, n_{CA} \\ r_1 &= (ID_v \, d\theta + \beta_1) \, mod \, q \\ r_2 &= (d\theta + \beta_2) \, mod \, q \end{aligned} \tag{20}$$

Where $\mathcal{H}_0$ is a public one way hash function and t is a timestamp. The voter computes the values $a_1 = {g_1}^{ID_v \theta} \, mod \, p$ and $a_2 = {g_2}^{\theta} \, mod \, p$ and sends the message

$$(Ticket, \, r_0, \, r_1, \, r_2, \, a_1, \, a_2, \, ID_{VS}, \, t, \, vote)^{e_{VS}}$$

to a voting server *VS*. Note that ID_{VS} is the identity of the voting server and (e_{VS}, n_{VS}), d_{VS} are RSA public\private keys of *VS*.

2. When a voting server reserves a message, decrypts it, computes the value of d, checks that t has not expired and validates the legality of the voter by checking the signature of the $Ticket$. For ensuring the correct use of ID_v, the signature of CA is checked by VS using the equation (21). Also for ensuring the correctness of r_1, r_2 and A, it checks the equations (22) to (24).

$$g_0^{r_0} \stackrel{?}{=} C^{e_{CA}} \, mod \, n_{CA} \tag{21}$$

$$g_1^{r_1} \stackrel{?}{=} a_1^d B_1 \, mod \, p \tag{22}$$

$$g_2^{r_2} \stackrel{?}{=} a_2^d B_2 \, mod \, p \tag{23}$$

$$g_1^{r_1} g_2^{r_2} \stackrel{?}{=} A^d B_1 B_2 \, mod \, p \tag{24}$$

After checking the correctness of equations (21) to (24), VS will ensure the validation of ballot $\{Ticket, d, r_0, r_1, r_2, vote\}$ and then sends it to the ballot counting server.

4.3. Third phase: counting ballots

All voting servers stores ballots in voting boxes. For instance, each voting box might contain 100 such ballots. Once a box is full, it is send to the ballot counting server (BCS) over a network. BCS checks for double voting where a voter may vote more than once with different voting server. If the BCS finds two ballots that contain the same ticket (i.e. $\{Ticket, d, r_0, r_1, r_2\}$ and $\{Ticket, d', r'_0, r'_1, r'_2\}$) by using the relation between r_1, r_2, d and consequently between r'_1, r'_2, d', it computes the identity of the voter as follow:

$$ID_v = \frac{r_1 - r'_1}{r_2 - r'_2} = \frac{d \; ID_v \theta - d' \; ID_v \theta}{d\theta - d'\theta} \tag{25}$$

Finally the BCS counts the valid votes and publishes the couple $\{C, vote\}$ in the bulletin board for giving assurance to the voter that his ballot is counted.

4.4. Security analysis of our electronic voting scheme

4.4.1. The proposed scheme achieves the requirement of anonymity of voters

In this protocol, since there is no relation between each cast ballot and the information which is given to AS by the voters, it is impossible to find the owners of the cast ballots even by cooperation between AS and VS. So the proposed scheme satisfies the anonymity of the voters.

4.4.2. The proposed scheme achieves the requirement of unforgeability of tickets

Assume that an attacker wants to generate a forged ballot

$$\{Ticket,\, d,\, r_0,\, r_1,\, r_2,\}$$

to cheat *VS*. Then the forged ballot must satisfy the following equations:

$$\begin{aligned} g_0^{r_0} &\stackrel{?}{=} C^{e_{CA}} \bmod n_{CA} \\ g_1^{r_1} &\stackrel{?}{=} a_1^d B_1 \bmod p \\ g_2^{r_2} &\stackrel{?}{=} a_2^d B_2 \bmod p \\ g_1^{r_1} g_2^{r_2} &\stackrel{?}{=} A^d B_1 B_2 \bmod p \end{aligned} \tag{26}$$

Also he must forge the $Ticket$ and the Okamoto's signature in the $Ticket$. But as we mentioned before, it is proved that the Okamoto's signature is unforgeable [12]. If the attacker uses the same ticket and forges the values of r_0, r_1, r_2 and d such that the above equations are hold, *BCS* would be able to detects him as a double voter in ballot counting phase.

4.4.3. The proposed scheme achieves the requirement of perceptibility of double voters

In the proposed scheme, *BCS* is responsible for detecting double voters in the last phase. When a voter votes more than once, he uses the same ticket in two different ballots. *BCS* checks ballots and finds different ballots with the same ticket. Therefore *BCS* can detect the malicious voter as described in 4.3.

4.5. Efficiency of the scheme

Table 1 shows the comparison of the number of multiplications and exponentiations in our scheme and other schemes of the family of Mu-Varadharajan's protocol. Also, since in our voting scheme most of the multiplications and exponentiations are done by voter, which are local, servers do a few multiplications and exponentiation. Table 1 expresses the comparison among our scheme and other schemes.

5. Conclusion

In this paper we showed that Rodriguez-Henriquez et al. protocols has some weaknesses on anonymity of voter and unforgeability of ticket. As

Schemes	Multiplication			Exponentiation		
	Phase1	Phase2	Phase3	Phase1	Phase2	Phase3
Line et al. [14]	13	7	6	19	6	0
Asaar et al. [1]	13	7	6	19	6	0
Hwang et al. [11]	18	8	6	22	8	0
Asaar et al. [2]	18	8	6	22	8	0
Rodriguez-Henriquez et al.[19]	10	21	2	14	12	0
Our Scheme	11	6	1	17	9	0

Table 1: Comparing efficiency of our scheme with the other schemes

we mentioned, all previous works in this context are vulnerable to one or more attacks which were presented in this paper. It seems that unsuitable application of the signatures in this family of protocols is the main reason that prevents achieving the expected properties. So we changed this structure. By using new structure we overcame the weaknesses of the previous works.

References

[1] M. Asaar, J. Mohajeri and M. Salmasizadeh. Another security improvment over the Lin et al.'s electronic voting scheme. *International Journal of Electronic Security and Digital Forensics*, 1(4):413–422, 2008.

[2] M. Asaar, J. Mohajeri and M. Salmasizadeh. Security Modification for the Hwang-Wen-Hwang 's E-voting Scheme. *Proceedings of International Conference on Security and Management(SAM08)*, 486–490, 2008.

[3] J. Benaloh. Verifiable Secret Ballot Elections. ,Ph.D. Thesis, Yale University, 1987.

[4] D. Chum. Elections with Unconditionally-Secret Ballots and Disruption Equvalnt to Breaking RSA. *In Advances in Cryptology-Eurocrypt'88*, 330:177–182, LNCS, Springer, 1988.

[5] H.Chien, J.Jan and Y.,Tseng. Cryptanalysis on MuVaradharajan's E-voting Schemes. *Appl. Math. Comput* 139 (23):525-530, 2003.

[6] D. Chaum. Secret-Ballot Receipts: True Voter-Verifiable Elections. *IEEE Securityand Privacy* , 2004.

[7] D. Chaum, P. Rayan and S. Schneider. A Practical, Voter-Verifiable Election Scheme. *inProc. 10th European Symposium on Research in Computer Security (ESORICS'05)*, 3679:118–139, LNCS, Springer, 2005.

[8] FIPS 186-2, Digital Signature Standard, National Institute of Standards and Technology(NIST), http://csrc.nist.gov/publications/fips/October 2001.

[9] T. ELGamal. A public-key cryptosystem and a signature scheme based on discrete logarithm. *IEEE Transactions on Information Theory*, 31:469–472, July 1985.

[10] A. Fujioka, T. Okamoto and K. Ohta. A Practical Secret Voting Scheme for Large Scale Elections. *Advances in Cryptology-AUSCRYPT '92*, 718:244–251, LNCS, Springer, 1988.

[11] S. Hwang, H. Wen and T. Hwang. On the security enhancement for anonymous secure e-voting over computer network. *Computer Standards and Interfaces*, 27(2):163–168, 2005.

[12] D.Juels, M.Luby and R.Ostrovsky. Security of blind digital signatures. *Proceedings of international conference on the theory and applications of cryptography and information security:Advanced in cryptology*, 1294:150–164, LNCS, Springer, 1997.

[13] B. Lee, C. Boyd, E. Dawson, K. Kim, J. Yang and S. Yoo. Providing Receipt-Freeness in Mixnet-Based Voting Protocols. *Proceedings of Information Security and Cryptology (ICISC'03)*, 2971:245–258, LNCS, Springer, 2004.

[14] I. Lin, M. Hwang, and C. Chang. Security enhancement for anonymous secure e-voting over a network. *Computer Standards and Interfaces*, 25:131–139, 2003.

[15] M. Hirt and K. Sako. Efficient Receipt-Free Voting Based on Homomorphic Encryption. *Advances in Cryptography-Eurocrypt 2000*, 1807:539–556, LNCS, Springer 2000.

[16] Y. Mu and V. Varadharajan. Anonymous secure e-voting over a network. *Proceedings of the 14th Annual Computer Security Application Conference. IEEE Computer Society*, pages 293–299, 1998.

[17] T. Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. *Proceedings of crypto92*, 740:31-53, LNCS, Springer verlog, 1993.

[18] D. Poincheval and J. stern. Provably secure blind signature schemes. *Proceedings of international conference on the theory and applications of cryptography and information security:Advanced in cryptology, Asiacrypt96*, 1163:255-263, LNCS, Springer verlog, 1997.

[19] F. Rodrguez-Henrquez, D. Ortiz-Arroyo and C. Garca-Zamora. Yet another improvement over the Mu-Varadharajan e-voting protocol. *Computer Standards and Interfaces*, 29:471–480, 2007.

[20] C. Yang , C. Lin and H.Yang Improved anonymous secure e-voting over a network. *INFORMATIM & SECURITY*, 15(2):185–191, 2004.

[21] W. Juang and C. Lei. A Secure and Practical Electronic Voting Scheme for Real World Environments. *IEICE Tranaction on Fundamentals of Electromics, Communications and Computer Science*, E80-A(1):64–71, January, 1997.