AN E-VOTING PROTOCOL BASED ON PAIRING BLIND SIGNATURES

L. López-García, F. Rodríguez-Henríquez
CINVESTAV-IPN, Department of Computing, Av. Instituto Politécnico Nacional No. 2508
Col. San Pedro Zacatenco, D.F. 07300, Mexico

M. A. León-Chávez
Benemérita Universidad Autónoma de Puebla, Facultad de Ciencias de la Computación
14 Sur y Av. San Claudio, CP 72570, Puebla, Mexico

Keywords: e-voting protocols, pairing-based signatures, digital signatures.

Abstract: In this paper we present a fair e-voting protocol able to guarantee voter's anonymity and double vote detection. The main cryptographic building blocks used by our system are two, namely, pairing-based blind signatures and elliptic curve digital signatures. We give both, a security and a cryptographic cost analysis of our proposed protocol, showing that it has a computational cost similar to other e-voting schemes previously reported, and the same time, it provides a good robustness against the potential attacks analyzed in this paper.

1 INTRODUCTION

In an electronic election system, privacy and security are mandatory features. However, it is not always obvious how to achieve these two characteristics at a reasonable price, due to the fact that when an election process takes place, mechanisms that assure both, security and privacy may be too expensive for system administrators on one side, and inconvenient for users on the other. In general terms, the basic properties that an electronic voting system must fulfill, are Authentication, Fairness, Accuracy, Integrity, Anonymity, Transparency, Verification and Accountability (Qadah and Taha, 2006). In this paper we present an e-voting scheme that uses pairing-based blind signatures as they were proposed in (Boldyreva, 2003). In order to protect votes' data integrity we sign it using an elliptic curve digital signature scheme (ECDSA) (NIST, 1994). To the best of our knowledge, e-voting schemes using pairing-based blind signatures have not been proposed before.

2 MATHEMATICAL BACKGROUND

Let us consider an elliptic curve E defined over a finite field $\mathbb{F}_{p^m}$, where p is a prime number and m is a positive integer and let $E(\mathbb{F}_{p^m})$ represent the additive group of points in E. Let $P \in E(\mathbb{F}_{p^m})$ be a point of order n, i.e., P generates a subgroup $\mathbb{G}_1$ of order n. Then, finding an integer $d \in [2, n-1]$ such that $Q = dP$ holds, is known as the *Elliptic Curve Discrete Logarithm Problem* (ECDLP). For carefully chosen elliptic curves E, the ECDLP is considered a hard difficult problem.

The Elliptic Curve Digital Signature Algorithm (ECDSA), is the elliptic curve analogue of the Digital Signature Algorithm (DSA) (NIST, 1994). No subexponential-time algorithm is known for the elliptic curve discrete logarithm problem. For this reason, the strength-per-key-bit is substantially greater in an algorithm that uses elliptic curves. In the protocol presented in this paper, we use a slight modification of the ECDSA scheme as discussed next.

The ECDSA key generation requires three domain parameters, that are considered public domain information, those parameters are the elliptic curve E, a base point P of prime order n. With this information the ECDSA key generation procedure generates the public key Q and the private key d by computing $Q = dP$. The digital signature of a given message m consists of a pair (r, s).

Let k (the integer k is called the *embedding degree* of the elliptic curve E) be the smallest positive integer such that $n|(p^{km} - 1)$. Then, there exists a unique subgroup $\mathbb{G}_2$ of order n defined in the multiplicative

López-García L., Rodríguez-Henríquez F. and A. León-Chávez M. (2008).
AN E-VOTING PROTOCOL BASED ON PAIRING BLIND SIGNATURES.
In *Proceedings of the International Conference on Security and Cryptography*, pages 214-217
DOI: 10.5220/0001925802140217

group $\mathbb{F}^*_{p^{km}}$. We can now define a bilinear pairing on $(\mathbb{G}_1, \mathbb{G}_2)$ as the mapping $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$ satisfying the following properties:

1. Bilinearity: $\forall R,S,T \in \mathbb{G}_1$
   $\hat{e}(S+R,T) = \hat{e}(S,T) \cdot \hat{e}(R,T)$
   $\hat{e}(S,R+T) = \hat{e}(S,R) \cdot \hat{e}(S,T)$
2. Non-degeneracy: $\hat{e}(P,P) \neq 1$
3. Computability: $\hat{e}$ can be efficiently computed.

2.1 Blind Signatures

Blind signatures are digital signatures with the property that the message to be signed is hidden from the Signer authority by means of a blind factor. This property can be exploited to offer anonymity in the electoral process to all the participants. The RSA blind signature scheme introduced by Chaum in (Chaum, 1983), allows to obtain a signed document but at the same time it guarantees that the signer entity will not be able to know the document that has just signed. Across the years, several blind signature schemes have been proposed. Some examples include a DSA-based blind signature scheme proposed in (Camenisch et al., 1994), an Elliptic Curve Cryptography (ECC) based blind signature presented in (Jena et al., 2007), among others.

In 2003, Boldyreva (Boldyreva, 2003) proposed a blind signature scheme based on pairings. That signature is computed into three steps, two of them must be performed by the requester and one by the signer entity. The required operations are summarized in Algorithm 1. The algorithm starts when the requester hash the message m to be signed to a nonzero group element $M \in \mathbb{G}_1$. Then, the requester blinds M, by computing a scalar multiplication as, $\hat{M} = bM$, where b is a random blind factor generated by her. Then, the requester sends $\hat{M}$ to the signer entity. The signer entity signs the message by performing a scalar multiplication on $\hat{M}$ using its private key d. Thereafter, it sends the blind signed document to the requester. After receiving the blind signature, the requester removes the blind factor, using, $b^{-1}\hat{\gamma} = b^{-1} \cdot b \cdot \gamma = \gamma$. Taking advantage of the pairing bilinear properties, that signature can be verified by checking whether,

$$\hat{e}(Q, H_1(m)) = \hat{e}(dP, M) = \hat{e}(P, M)^d = \hat{e}(P, \gamma),$$

holds.

3 PROTOCOL DATAFLOW

Our scheme requires three entities: one of them responsible for authenticating, another responsible of receiving and validating the votes and the last one specifically responsible of performing the vote tally. In the rest of this paper, these three authorities will be called Authentication Server (*AS*), Voting Server (*VS*) and Counting Server (*CS*), respectively.

Algorithm 1: *BlindSignature* (Boldyreva, 2003)

Blind factor: $b \in \mathbb{Z}_n$
$M = H_1(m)$ where $H_1 : \{0,1\}^* \longrightarrow \mathbb{G}_1 \setminus O$.
Blindness: $\hat{M} = bM$
Signing: $\hat{\gamma} = d\hat{M}$
Unblindness: $\gamma = b^{-1}\hat{\gamma}$
Verification: $\hat{e}(Q, H_1(m)) = \hat{e}(P, \gamma)$

In Figure 1, the interaction among the voter and the three servers is depicted. The most important exchanged object, is the electoral ballot that contains the vote. In the initial authentication phase, the voter requests a blank electoral ballot B from the *AS*. This request will be granted only in the case that the *AS* authenticates the voter. If everything works fine, the voter will receive a blank ballot blind signed by the *AS*. Once that this action is accomplished the voter fills out the ballot and then sends it to the *VS*. This is done in a second phase called Voting phase. It is worth remarking that the ballot must be signed by the voter but using a public/private key pair especially generated for signing the ballot. Finally, in the third phase called the Counting phase, all the received votes are counted by the *CS* so that the election result can be obtained.

4 THE PROPOSED PROTOCOL

Notation

- o $\{d_{AS}, Q_{AS}\}$ *AS'* public/private key pair.
- o $\{d_V, Q_V\}$,$Cert_V$ *V'* public/private key pair and digital certificate.
- o $\{d, Q\}$ public/private key of the vote.
- o $k_1, k_2 \in \mathbb{Z}_n$, where k_1 is a unique identifier chosen by V, whereas k_2 denotes a unique identifier chosen by the *AS*.
- o t a time stamp.

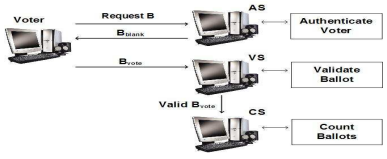

Figure 1: Structure and functionality of the proposed scheme.

- o $x\|y$ denotes the values x and y concatenated.
- o $b \in \mathbb{Z}_n$ denotes a random blind factor.
- o $\{m\}_{ENC_x}$ denotes the encryption function that uses the secret key x applied to the message m.
- o $\{m\}_X$ denotes the ECC signature of the message m generated by the entity X

4.1 Authentication Phase

The voter randomly generates his/her secret key d and session key k_1 and it must compute the public key Q, that he/she will use to sign the electoral ballot in the next phase. In order to guarantee full anonymity, these values should be different than the voter's public/private key.

It is customary to use the coordinate x of the point R for obtaining the first value of the ECDSA signature in (r,s). For convenience, in our scheme we use the pair (R,s) as the vote's signature. This simplifies the introduction of the voter's unique identifier in the vote signature. This way, R and Q are public values that will be sent to the VS. These parameters are blindly sent to the AS for its signature, so that there is no possibility of associating the vote with the voter in the following phases.

In order to request a valid blank ballot, the voter sends to the AS, his/her digital certificate and blinded messages, all of them signed with his/her private key (as shown in the authentication phase of Table 1).

$$\{Cert_V, \hat{Q}, \hat{R}, \{\hat{Q}\|\hat{R}\|t\}_{d_V}\}$$

After receiving this message, the AS attempts to authenticate the voter. In case of positive identification, the AS assigns a unique identifier k_2 to that voter. This value will be kept in the registers of the AS.

Before proceeding to sign the blind messages, the AS mix up to k_2 with $\hat{R}$ doing the scalar multiplication $k_2\hat{R}$ to avoid duplicity of the signature. In order to avoid forgery, the AS requires a link value to relate both voter's signatures.

This value should not be altered by the voter and at the same time it cannot be generated by the AS since, otherwise, the anonymity of the voter will not be preserved. These two restrictions suggest that a good candidate for the link value could be a parameter that can be taken from the vote's signatures. $k_2\hat{R}$ satisfies both restrictions, owed that it can not be modified by the voter and neither the AS can link it with the voter. Then, the AS adds $k_2\hat{R}$ to $\hat{Q}$.

Finally, the AS signs the blind messages and replies sending the parameter k_2 encrypted with voter's public key Q_V along with the two blind signatures. In order to guarantee its authenticity, the whole message is signed by the AS using its private key d_{AS}.

$$\{\{k_2\|t\}_{ENC_{Q_V}}\hat{\gamma}_Q, \hat{\gamma}_R, \{\hat{\gamma}_Q\|\hat{\gamma}_R\|t\}_{d_{AS}}\}$$

The voter receives the message, verifies that the signature comes from the AS and removes the blind factors. After recovering his/her unique identifier k_2, the voter generates the signature for his/her vote with ECDSA as shown in table 1 in the voting phase. The voting ballot filled in by the voter is then sent to the VS as

$$\{Q, R, s, vote, \gamma_Q, \gamma_R\}$$

where (R,s) is the digital signature of $vote$, Q is the public key of the signature (R,s), γ_Q is a blind signature of Q and γ_R is a blind signature of R.

The VS receives the ballot and verifies the blind signatures. Let us recall that, the messages of blind signatures γ_Q and γ_R are linked with R, then, VS verifies the signature γ_R for R and γ_Q for the message $Q+R$.

If the two signatures verify correctly, then the VS proceeds to check the vote's signature with the ECDSA verification.

If the three signatures were all valid, the VS accepts the ballot, produces a hash value of it and it stores the

Table 1: The scheme proposed.

Authentication phase

Voter		AS
$k_1, d, b \in [1, n-1]$		
$Q = dP$, $R = k_1P$		
$\hat{Q} = bQ$, $\hat{R} = bR$		
	$\{Cert_V, \hat{Q}, \hat{R}, \{\hat{Q}\|\hat{R}\|t\}_{d_V}\}$ $\longrightarrow$	
		$k_2 \in [1, n-1]$
		$\hat{\gamma}_Q = d_{AS}(\hat{Q} + k_2\hat{R})$
		$\hat{\gamma}_R = d_{AS}(k_2\hat{R})$
	$\{\{k_2\|t\}_{ENC_{Q_V}}\hat{\gamma}_Q, \hat{\gamma}_R, \{\hat{\gamma}_Q\|\hat{\gamma}_R\|t\}_{d_{AS}}\}$ $\longleftarrow$	
$\gamma_Q = b^{-1}\hat{\gamma}_1$		
$\gamma_R = b^{-1}\hat{\gamma}_2$		

Voting phase

Voter		VS
$R = k_2k_1P$		
$s = (k_2k_1)^{-1}(vote + R_x d) \bmod n$		
	$\{Q, R, s, vote, \gamma_Q, \gamma_R\}$ $\longrightarrow$	
		$\hat{e}(Q_{AS}, Q+R)? = \hat{e}(P, \gamma_Q)$
		$\hat{e}(Q_{AS}, R)? = \hat{e}(P, \gamma_R)$
		$ECDSA\_VER(Q, R, s, vote)$

Counting phase

CS		VS
Compare and counting	$\longleftarrow$ Valid Ballots	

ballot. Finally, it sends to the voter the hash value of its ballot as a receipt.

In the counting phase, the *CS* receives all the ballots that were successfully validated by the *VS*. Since each ballot contains three signatures, the only possible value that can be repeated is the one of the vote.

Therefore, in order to avoid duplicity, the *CS* compares each received ballot with all the others already stored. If two or more ballots have repeated signatures, then they are candidates for possible fraud and the *CS* will take as valid the first ballot.

5 ANALYSIS OF PROPOSED SCHEME

Solution of possible Attacks to the System

- *Associating the pair (vote, voter)*: Only the *AS* has information that links the voter with the ballot that was given to him/her. Nevertheless, if the *AS* wants to obtain the voter information, it needs to remove the blind factor b from the equation $\hat{R} = bR$. This equation is protected by the *ECDLP* and therefore, it should be computationally unfeasible to obtain it.

  In order to identify any voter, it suffices that the *VS* can manage to extract k_2 from k_2P, since this information leads to identify the voter from the *AS* registers. However, three of the signatures contained in the ballot, namely, γ_Q, γ_R and R are protected by the ECDLP.

- *Modifying vote's digital signatures*: This attack is prevented by the signature of the *AS*. The voter must prove that the first part of the signatures were signed by the *AS* during the authentication phase. Otherwise, the ballot will not pass the checks performed by the *VS*.

- *Combining ballots by several voters*: This attack is prevented by the *AS* by including the link value R to the blind message $\hat{Q}$ before signing them (during the authentication phase). If a given voter decides to combine his/her signatures with another authorized voter, the link value will not be the same for all the signatures. This will produce that at least one of them will not verify.

Efficiency. The number of cryptographic algorithms performed by our scheme in the authentication and voting phases are summarized in Table 2.

Table 2: Computational Cost of our Protocol by Phase.

Phase	**Voter**	**AS**	**VS**
Auth.	6 scalar mult. 1 inversion	2 scalar mult.	- -
Voting	1 scalar mult. 1 inversion		4 pairing 2 scalar mult.

6 CONCLUSIONS

In this paper we presented an e-voting security protocol that utilizes pairing-based blind signatures and ECDSA digital signatures as its main building blocks. The pairing-based blind signatures help to bring better security to the system guaranteing the anonymity of the voters, whereas the ECDSA digital signatures are used to assure votes' integrity.

REFERENCES

Boldyreva, A. (2003). Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme. In *PKC '03: Proceedings of the 6th International Workshop on Theory and Practice in Public Key Cryptography*, pages 31–46. Springer-Verlag.

Camenisch, J., Piveteau, J. M., and Stadler, M. (1994). Blind Signatures Based on the Discrete Logarithm Problem. *In Advances in Crypology - EUROCRYPT '94, LNCS*, 950:428–432.

Chaum, D. (1983). Blind Signatures for Untraceable Payments. *Advances in Cryptology Proceedings of Crypto '82*, pages 199–203.

Jena, D., Kumar, S., and Majhi, B. (2007). A Novel Untraceable Blind Signature Based on Elliptic Curve Discrete Logarithm Problem. *International Journal of Computer Science and Network Security, IJCSNS*, 7(6):269–275.

NIST (1994). Digital Signature Standard (DSS). Federal Information Processing Standards Publications (FIPS PUBS). http://www.itl.nist.gov/fipspubs/fip186.htm.

Qadah, G. and Taha, R. (2006). Electronic Voting Systems: Requirements, Design and Implementation. *Computer & Interfaces*, 29(3):376–386.