Design and implementation of a novel secure internet voting protocol using Java Card 3 technology

Mostafa Mohammadpourfard and Mohammad Ali Doostari*

Department of IT and Computer Engineering, Shahed University, Tehran, Iran
E-mail: m.mohamadpour@shahed.ac.ir
E-mail: doostari@shahed.ac.ir
*Corresponding author

Mohammad Bagher Ghaznavi-Ghoushchi

School of Engineering, Shahed University, Tehran, Iran
E-mail: ghaznavi@shahed.ac.ir

Hadi Mikaili

Faculty of Electrical and Computer Engineering, Qazvin University, Qazvin, Iran
E-mail: h.mikaili@qiau.ac.ir

Abstract: Internet voting is highly regarded for its speed, automatic counting, cost reduction and lower possibility of error. However, in order to replace traditional voting system with internet-voting, i-voting must meet many requirements and overcome some challenges like collusion, uncoercibility, unfairness and in particular voter insecure platforms. In this paper, we propose a very practical and secure internet voting protocol based on FOO with use of Java Card 3 which can help prevent voting fraud and addresses its major challenges. Java Card 3 technology is a new concept which provides the expected client-side security in a much higher level than an ordinary PC, which is highly vulnerable to infection via the network. Since with Java Card 3 technology the card acts as a secure web client (server) can address the voter side insecure platform problem which is one of main reasons for the delay in implementing I-voting in real world.

Keywords: elections; internet-voting; Java Card 3; coercion; cryptographic protocols; JEE technologies.


Copyright © 2014 Inderscience Enterprises Ltd.
Biographical notes: Mostafa Mohammadpourfard is an MSc student in Information Technology (majoring in information security trend) at Department of IT and Computer Engineering, Shahed University. His main research interests are information and network security, internet voting (e-services), knowledge management and Java cards.

Mohammad Ali Doostari received his BS degree from Shiraz. From 1987 to 1992 he did his Master and Doctoral course in Kyoto University of Technology in the field of Information and Electronics Engineering. He is Guidance Professor of many students in BS, MS and doctoral course doing research in the different aspects of IT security such as e-voting, e-payment, trusted computing and smart cards related topics as their graduation thesis. Furthermore, he has been a manager and founder of a research team implementing e-passport and smart national ID card in Iran.

Mohammad Bagher Ghaznavi-Ghoushchi received his BSc degree from the Shiraz University, Shiraz, Iran in 1993, his MSc and PhD degrees both from the Tarbiat Modares University, Tehran, Iran in 1997 and 2003 respectively. During 2003–2004, he was a Researcher in TMU Institute of Information Technology. He is currently an Assistant Professor with Shahed University, Tehran, Iran. His interests include VLSI design, low-power and energy-efficient circuit and systems, computer aided design automation for mixed-signal and UML-based designs for SOC and mixed-signal.

Hadi Mikaili is a PhD student in Software Engineering in Qazvin University. His main interests are cloud computing and distributed systems; wireless sensor and actor network, routing and energy conservation; software engineering and analysis/design/developing systems.

1 Introduction

The first electronic election scheme was proposed by Chaum (1981). The increasing popularity of e-voting is mainly because of the interesting advantages that this technology offers like more participation, transparency, widening access for people with disability, delivering voting results more quickly and reducing costs (Buchsbaum, 2004; Cooke and Anane, 2012). Figure 1 shows the general process of e-voting (Lee et al., 2010). Internet-voting, a derivative of e-voting, enables voters to cast their votes on their PCs via the internet participate.

Because internet-voting is done over network and voters are not under physically observation or control, its implementation must be resistant against many kinds of attacks, to be perceived as secure by voters. We can divide security issues in two categories. The first one is the attacks that aims electoral officer server (also malicious voting servers). The second one which is more difficult to guarantee is voter side platform security (in this paper this one is considered). One of the most important open problems in building secure internet voting is insecure platforms (Haenni and Spycher, 2011). Voter computers which are infected with malicious software presents severe obstacle to its usage (Heiberg et al., 2010; Ryan and Teague, 2013). Attacker can exploit these vulnerabilities and break the system. Compromising internet voting system on client side is more feasible than server side. In Estehghari and Desmedt (2010) the
attacker easily hacked the client side machine and has broken the Helios (Adida, 2008) which was the first web-base, open-auditing voting system.

Proposals for internet voting systems are often assume that the voter side platform is trusted (Nestas and Hole, 2012). This assumption is to guarantee voter’s privacy and integrity of election (Joaquim et al., 2013). But as mentioned, since computers are susceptible to cyber-attacks and malwares, this assumption is not so applicable. Therefore, for addressing this issue we have used Java Card 3 which can provide desired security in client side.

The main contribution of this paper is proposing a novel i-voting protocol based on FOO (Fujioka et al., 1993) which can satisfy mentioned requirements in Section 1.1 and also implementing the proposed protocol through the use of Java Card 3 technology capabilities (which is the known first implementation of an i-voting protocol with Java Card 3).

The remainder of the paper is organised as follows. In Section 1.1 main requirements of e-voting (internet-voting) protocols are mentioned. An overview of the related work is given in Section 2. In Section 3, the internet-voting leading challenge: client side insecure platform is investigated. Section 4 introduces Java Card 3 technology. Our proposed scheme is introduced in Section 5. Section 6 briefly sketches our implementation. Security evaluation of the proposed protocol is presented in Section 7. Section 8 concludes the paper and we finish the paper by appreciating people who have shared their knowledge with us in doing this research and implementing our protocol.

Figure 1 General process of e-voting (see online version for colours)

Source: Lee et al. (2010)

1.1 Internet-voting properties

Overall, e-voting systems must avoid voting fraud possibilities and guarantee voter’s anonymity at the same time. In fact, for accepting and using e-voting system by voters, it
must be at least as secure as traditional voting method. In particular, researchers have proposed a set of criteria that must be addressed as requirements for a secure electronic voting system (Fan and Sun, 2008; Liaw, 2004; Mauw et al., 2007; Spycher et al., 2012; Based and Mjølsnes, 2013):

- **Fairness**: Ensures that no one can obtain knowledge about tally result before the end of election. In FOO (Fujioka et al. 1993), supporting fairness stands against breaching the vote-and-go property because the voter must waits till to see his vote on the public bulletin board and then sends his private key to the ‘taller’.

- **Anonymity**: No one can link a ballot to the voter who has casted it.

- **Verifiability**: Voters must be able to check that their votes have been casted and counted correctly.

- **Integrity and accuracy**: Any invalid vote should not be counted. Also altering, deleting and adding votes are not permitted.

- **Resistance to collusion**: Collusion in election is a critical matter and in this research, we have investigated it from two novel perspectives:
  1. Masquerade: electoral officers (especially those who are responsible for identification and registration) can collude to vote instead of eligible but absent voters.
  2. Breaching voter’s anonymity: an ideal internet-voting protocol accomplishes identification and registration processes in two completely separate phases and shares voter’s information between the voter and the other parties.

- **Vote-and-go**: Voter participation is not required in the counting phase.

- **Eligibility**: Only authorised voters can vote and only one vote per eligible voter is counted.

- **Mobility**: Voters can cast their votes from anywhere and are not restricted to a special location for casting vote.

- **Uncoercibility**: Voters cannot convince (prove to) the coercer about their selections. It is noteworthy that in this paper coercion does not mean physical presence of coercer beside a voter and voting instead of voter.

For preventing coercion and vote buying/selling, different solutions have been proposed. Such as: Benaloh and Tuinstra (1994), Rivest and Smith (2007), Clarkson et al. (2008), Chung and Wu (2009) and Popoveniuc and Hosp (2010).

1. Providing voter complete privacy so that monitoring his/her electoral behaviour will be impossible for any observer. In e-voting this trait is achieved by isolation of each voter in voting kiosks.
2. Deceiving briber or coercer: voter can pretend that have done the attacker expected electoral behaviour while in reality she/he has voted to her/his considered candidate.
3. Using receipt-freeness concept so that proving voter electoral behaviour is not possible.
4. Producing vote buying/selling resistant receipt.
Considering advantages and disadvantages of each approach, it seems that choosing a method which is among the second category will be better because only this approach has aimed the motivation of coercion and bribing and somehow reduces the likelihood of these two threats up to a large percentage. In this paper we have proposed a method based on this approach.

- **Robustness**: Is related to performance of hardware components of the voting system. Components should be functional during the voting period.

- **Voter side platform security**: The platform used by the voter to accomplish voting tasks must be completely secure in order to guarantee vote privacy and integrity and voter’s anonymity.

- **Efficiency**: The computations can be performed within a reasonable amount of time.

## 2 Related work

Since 1980 that Chaum (1981) proposed the first electronic voting, many e-voting protocols have been proposed and developed. However, in general they can be classified in three categories:

1. Schemes based on complex cryptography designs (Cohen and Fischer, 1985; Benaloh and Yung, 1986; Iversen, 1992; Cramer et al., 1997). They are based on homomorphic encryption which makes use of zero knowledge proof techniques to keep their ballots and anonymities secret. One of its advantages is that vote can be performed without decrypting any of the ballots (Cooke and Anane, 2012). However, they have the problem of high, complex and expensive computational overhead (Ray et al., 2001).

2. Schemes based on anonymous channel to cast a ballot. Anonymous channel conceals the correspondence between input and output items (Chaum, 1981; Boyd, 1990). In anonymous channel, the sender (voter) is unknown for the receiver (authority). In other words, the link between voters and ballots is difficult to find. For example, in Chaum (1981) anonymous channel is used for casting ballot. However, breaching anonymity of voters is not guaranteed (Chaum, 1981) because of assuming one mix node to be trusted in a mix-net (Ray et al., 2001). In Boyd (1990), multiple key encipherment with anonymous channel is used to support voter’s anonymity. But it suffers from this vulnerability that administrator can modify votes. Also it should be mentioned that so far there is no implemented election based on mix-nets (Nguyen and Dang, 2013).

3. Schemes based on blind signatures (Ibrahim et al., 2003; Cetinkaya and Doganaksoy, 2007; Song and Cui, 2012). This makes possible to have a signed message by an entity without knowing its content. FOO (Fujioka et al., 1993) is based on blind signatures and anonymous channel. Because of comparatively smaller communications and low computational complexity of blind signatures approaches, they are proper for large scale elections (Cooke and Anane, 2012).

Also proposed protocols can be classified differently in three groups:
1. Paper based voting: In this approach, the voter uses a booth to vote on a paper ballot. Then their ballots are scanned and counted electronically (Chaum et al., 2005).

2. Polling booth based: The voter enters the polling booth and makes her choice (Graaf, 2009; Chaum et al., 2012).

3. Internet based: In internet voting, voters use their PCs to cast their votes (Joaquim et al., 2013; Nguyen and Dang, 2013; Yi and Okamoto, 2013).

Our opposed schema is fully implementable over internet and also make use of blind signature. Our proposed addresses coercion, vote selling and voter side insecure platforms.

3. The internet-voting leading challenge: client side insecure platform

The vulnerabilities of internet voting technology are main security issue. Household and business PCs are very susceptible to malwares and complicated attacks which leads to voter’s privacy breach (Smith, 2008).

According to Panda company (security lab) annual security report of all the computers scanned by Panda Security’s security software in 2012, an astonishing 31.98% were infected by some form of malware (Security, 2012). Also in 2007, Vint Cerf, one of the founding fathers of the internet, made a shocking claim during a conference held in Davos by the World Economic Forum. Cerf claimed that a quarter of all personal computers around the world, roughly 100 to 150 million computers, connected to the internet are totally or partially controlled by computer criminals. Since these criminals control the PCs remotely, they can intercept and modify any activity done by the PC user, including a web vote. In this case, any form of security devised for the overall online voting system is in the end defeated by the user’s PC (Pasquinucci, 2007).

To provide client-side security, employing an anti-virus toolkit is not enough. Of course, there should be some methods to implement the system’s security in OS layer, consequently, in order to prevent the OS or hardware from installing any malicious codes or programs. Accordingly, some voting protocols including Adida et al. (2010), Adida (2008), Joaquim et al. (2010) and Heiberg et al. (2010) have proposed new solutions for better security in online voting:

1. In the first approach, which is unique to kiosk electronic voting (not practical for internet-voting), voters are obliged to cast a ballot from a private voting booth.

2. In the second approach like the proposal presented by Zúquete et al. (2007), in order to implement a trusted computing environment within the voter’s insecure system, a combination of public key cryptography, smart cards and FINREAD terminal readers are employed. According to FINREAD framework, security is ensured through three components: a secure display, a secure pin pad and a card reader authentication function. The FINREAD platform adopts and extends Java applet technology; in a FINREAD Java virtual machine, applets are called Finlet (Specification, 2011) and are used as PKI services for identification, authentication and digital signature (IAS). Although this solution protects all interactions between card and terminal but card and PC interactions remain unmanaged. The existence of malicious codes on a voter’s PC or the remote control of a voter’s system by hackers can cause vote
modification regardless of any effort we utilise to secure card and terminal interactions.

To guarantee client side security, we propose replacing the voter’s PC with a Java Card 3 (latest version of Java Card) smart card.

4 Java card technology

Java card technology essentially provides a platform in which allows programs (applets) written in the Java programming language to run on smart cards and other resource-constrained devices model. Because of the small memory footprint, the java card platform supports only a carefully chosen, customised subset of the features of the Java language. Portability and security are the main design goals of the java card technology (Patel et al., 2011).

4.1 Java Card 3 technology

Java Card 3 technology targets IP-based web applications and has been introduced in two separate but coherent editions: the classic edition and connected edition. The classic edition is based on evolution of Java Card 2.2.2 platform which targets APDU based applications. It includes several incremental changes like: supporting for latest cryptography algorithms including 4096-bit RSA and many other changes (SUN Microsystems Inc., 2008; Oracle, 2010).

Connected edition includes network orient ed features such as supporting for web application and for applets with extended and advanced capabilities. It allows the card to truly act as a node in an IP network and provide security services for other network nodes or uses network resources. Innovation of this edition is based on supporting for Java servlets – an application that runs on web server on top of a web container. The web container dispatches requests to servlets and servlets can process incoming requests. On-card web applications, inside the web containers, provide services to entities on an IP network. They are accessible using standard internet protocols such as HTTP/HTTPS, therefore they can be easily integrated into existing internet service infrastructure (SUN Microsystems Inc., 2008). High-level architecture of Java Card 3 Connected edition is shown in Figure 2(a). Java Card 3 technology on the client side is able to act as a secure personal computer or secure customised web server.

The new connectivity layers and protocol stack features in connected edition, enables the card to directly participate and connect to a network in order to provide services. These layers are shown in Figure 2(b).

With Java Card 3 technology, smart card application development has instantly become easier and has now taken a significant leap-ahead in capability; making the following scenarios possible:

- Enabling simultaneous deployment of web applications and traditional card applications (Oracle, 2010).
- Developers no longer need to create individual client applications to access the data and resources on the smart card. The only required client interface is an ordinary web browser [Hopkins (oracle), http://www.oracle.com/].
Design and implementation of a novel secure internet voting protocol

- Smart card applications are now fully functioning TCP-based servers. These server applications are Java servlets and they have a full HTTP(S) stack allowing them to process ‘get’ requests, ‘post’ requests, headers, cookies, sessions and so on [Hopkins (oracle), http://www.oracle.com/].

- Provides richer interaction with end user and brings web look-and-feel to card application. So user (voter) easily can use it (as shown in implementation snapshots in Section 6) (Violleau, 2011).

In addition to Java Card 3 attractive capabilities, providing end-to-end security is among its prominent achievements that highly enhance internet-voting security.

Figure 2 (a) High-level architecture of java card platform, Connected Edition technology (b) Connectivity layers and protocol stack (see online version for colours)

Source:  (a) SUN Microsystem Inc. (2008), (b) Allenbach (Oracle) (2012)
5 Our proposed protocol

The greatest challenge for securing the entire voting protocol (process) is insecurity of client side PCs that may be infected with many types of programs and malicious codes (or botnet network) (Heiberg et al., 2010; Joaquim et al., 2010; Lauer, 2004). In most of the offered protocols a completely false assumption that client systems are secure is considered (Joaquim et al., 2003; Mauw et al., 2007; Yi and Okamoto 2013).

In internet voting which allows voters to cast their votes from remote terminal, protocol and electoral authorities cannot control the voter side security. To secure voter side and counter the mentioned threats, we propose to use a Java Card 3 which include a voting servlet and other needed modules instead of malicious voter computers which is one of the crucial challenges in implementing internet voting.

In our proposed protocol the assumption is that every voter has a Java Smart Card 3 and by using it can easily participate in e-voting and internet-voting. Within each voter card, there is a voting servlet which can be used to interact with other units involved in the election in the form of web services and sends messages on the HTTP(S) protocol. It should be mentioned that all communications between parties that are involved in this protocol are secured via SSL protocol and therefore regardless of utilising trusted client devices, end-to-end security between card and other network nodes are accomplished. Figure 3 shows the structure of our schema. Our schema includes five parties and done in four phases.

Figure 3  Structure of proposed scheme (see online version for colours)

5.1 Notations

In the proposed protocol, we use the following notations:

- $RA$: Registrant Authority
• A: Administrator
• V: Validator
• T: Tallier
• Aid: voter alias id
• SK: a symmetric key
• \( E_x(m) \): encryption function for message m using key x
• \( S_x(m) \): signature functions for message m using key x
• \( D_x(m) \): decryption function for message m using key x
• \((PK-Voter, PRK-Voter)\): voter’s public and private key
• \((PK-RA, PRK-RA)\): registration authority’s public key and private key
• \((PK-A, PRK-A)\): administrator’s public key and private key
• \((PK-V, PRK-V)\): validator’s public key and private key
• \((PK-T, PRK-T)\): Tallier’s public key and private key
• \((PK-Election, PRK-Election)\): election’s public key and private key (ballots encrypting key)
• \( B \): blinded message.

In order to ensure fairness, the election’s private key is secretly shared between candidates and election authorities and obtained at the end of the vote reception period. Also, the key can be stored in an HSM which is placed in a safe location (according to ISMS-based physical security controls) and prevents disclosure before the election deadline.

5.2 Structure of our scheme

Our proposed internet-voting phases are as follow:

5.2.1 Registration phase

Registration phase begins a few days before the election. In this phase, each voter refers to the Registrant Authority’s (RA) website to be recognised as an eligible voter. For accessing voting servlet inside the card, card holder must be authenticated by PIN or fingerprint. Then voter sends his/her hashed identity information which is computed and loaded into card upon card issuance (it is noteworthy that voting servlet by itself can compute hash of needed fields but this is done, because of avoiding repetitious value of hashed data) to RA. In addition to this data, each voter must choose an alias ID (Aid) and sends it to RA after blinding it by voting servlet. This is done for avoiding the possibility of collusion between RA and other officers to violate voter anonymity.
After the RA has received message one, it decrypts the message by its private key and checks the voter eligibility and produces message two.

\[ A_1 = S_{PRK-RA}[B[Aid]] \]

Voter by receiving message decrypts it and checks the validity of RA signature. Now, voter can obtain signed Aid by removing blind factor. We call this value, RS-License. It is right that in our scheme, we send hash of identity information and upon collusion, units cannot breach voter identity, but consider a situation that they can crack hashed value. With RS-License, all mapping between the voter and his/her identity information are hided. Each voter can have only one RS-License (Now, voter uses this token to introduce himself to other electoral authorities as an eligible voter without revealing his real identity).

5.2.2 Administration phase

In this phase, voting servlet encrypts signed Aid and a symmetric key with administrator’s public key PK-A and sends it to the Administrator as follows.

\[ V \rightarrow \text{Admin}: \ E_{PK-A}\{RS-License,SK\} \]

By receiving above message, Administrator verifies signature of RS-License and then the Administrator prepares an empty ballot EB with a particular watermarking to show its originality. Next, the Administrator uses its private key to sign EB and RS-License and then encrypts the message with the symmetric key that has received in message 3.

\[ \text{Admin} \rightarrow V: \ M_1 = E_{SK}\{S_{PRK-A}[RS-License,EB]\} \]

After receiving the message, the servlet verifies the Administrator’s signature and ballot’s watermark. Then prepares the ballot to be filled by the voter (The signed RS-License by Administrator is so called voting license or V-Lic).

5.2.3 Voting phase

In this phase, each voter fills the ballot (FB) and then blinds the ballot known as BB. Then, he encrypts the message by Validator public key and sends it to the validator.

\[ BB = B[\text{Ballot}] \]

\[ V \rightarrow \text{Validator}: \ E_{PK-V}\{BB.V-Lic\} \]
Once the Validator receives the $E_c$, verifies and compares $RS\text{-License}_i$, signature with $V\text{-License}_i$, then uses its private key $PRK-V$ to sign $BB$ and sends it to the voter as follows.

6 Validator $\rightarrow$ V:  
$$V_i = S_{PRK-V}[BB, \text{Digest}(BB)]$$

After receiving message 6, voter unblinds the $BB$ and retrieves the signature of the filled ballot ($SFB$).

5.2.4 Collecting and counting phase

In this phase, in order to prevent coercion (as mentioned in Section 1.1, this approach is based on second category), voters in each time that cast their votes, in addition to data that send to the Tallier, send another information element (must be sent). That element is a unique pass phrase which voter must send with any vote to count. Now, if a voter is in coercion can send its previous pass phrase. Tallier with receiving a repeated pass phrase, detect that the voter has voted in coerced state and invalidate the vote. In continue the voter can replace forced vote with another valid vote.

7 V $\rightarrow$ Tallier:  
$$E_{PK-T} \{E_{PK\text{-Election}} \{SFB_i \}, V\text{-License}_i \text{ pass phrase} \}$$

For each received vote, the Tallier generates a vote unique identifier ($V\text{-uid}$) and sends it to voter for individual verification. Table 1 briefly shows this process. Each received vote alongside its $V\text{-uid}$ is inserted in public bulletin board to ensure voters.

Once the voting phase is completed, the election’s private key (PRK-Election) is revealed and the encrypted votes are counted. For each V-Lic, the Tallier checks all $V\text{-uids}$ and selects the vote with latest valid pass phrase to decrypt. After decryption, the vote can be a (n):

- valid vote with valid signature of Validator with a valid content
- rejected vote with valid signature of Validator, but an invalid content
- invalid vote without the valid signature of Validator.

After counting, decrypted content of each vote along with $V\text{-Lic}$, latest $V\text{-uid}$ and pass phrase is added as a new column to the bulletin board. Polling sequence diagram is shown in Figure 4.

Table 1 Vote status for each time of casting a ballot

<table>
<thead>
<tr>
<th>Vote status</th>
<th>Pass phrase</th>
<th>Vote</th>
<th>$V\text{-uid}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Valid but expired</td>
<td>123456</td>
<td>Vote$_1$</td>
<td>67894566</td>
</tr>
<tr>
<td>Coercion</td>
<td>123456</td>
<td>Vote$_2$</td>
<td>89745896</td>
</tr>
<tr>
<td>Valid</td>
<td>12345</td>
<td>Vote$_3$</td>
<td>94315278</td>
</tr>
</tbody>
</table>
6 Implementation issues

For implementing our proposed scheme using Java Card 3, we have used NeBeans IDE 7.0.1. 4 electoral servers are java web applications. For voter side, we have used java card web project (which contain Java Card 3 servlet). Java Card 3 uses jetty web server. Jetty provides an HTTP server, HTTP client and javax.servlet container. For communication between card and other web application, classes and interfaces like Connector (javax.microedition.io.Connector), httpConnection (javax.microedition.io.HttpConnection), InputStream (java.io.InputStream), bufferedReader (java.io.BufferedReader) and other classes and interfaces have been used in voter side card. For voter side, we have designed a servlet (voting servlet) which do all needed processes. Figure 5(a) shows the first page which voter sees it when he/she enters the address of voting servlet.

Figure 5(b) shows the step that card is interacting with RA. Hash value of identity information (here is: voter name, nationality, age) on the page has been read from the card automatically and it is not changeable by voter. As mentioned in this step the voter chooses an ‘aid’. The chosen ‘aid’ is then blinded and encrypted by the card and will be sent to RA by pressing the send data to registrant authority.
At the RA side (Java web application), we have designed a servlet which receives voters requests and after verifying voter’s eligibility, sends back the signed blinded ‘aid’ As shown in Figure 6(a). Also it is noteworthy that in our implementation, among electoral servers except Tallier which needs UI for showing encrypted/decrypted votes for public/individual verifiability, other dos not need UI. But based on needs, we can design UI for them too.

In order to doing cryptography functions, we have used APIs like (javacardx.crypto.Cipher), (javax.microedition.io.SecureConnection), (javacard.security.RSAPrivateKey), (javax.microedition.io.SecurityInfo), (javacard.security.RSAPublicKey), etc. are used.

In Figure 6(b), by pressing Send Data to Administration, signed Aid along with a symmetric key is sent to administration server. Administration server after verifying signature on Aid sends an empty ballot to the card at administration server side there is a servlet which does these operations. The structure of the ballot is defined by an XML document. The received ballot can be stored in the card by using the FileConnection interface (javacardx.io.FileConnection). For simplicity, we have used a very simple ballot in xml format which includes <Administrator></Administrator> tag as a watermark to show the ballots originality.

Figure 5 Screen shots of implementation (a) first page of voting servlet of proposed I-voting protocol (b) registration phase (see online version for colours)
In Figure 7(a), voters fill the ballot by entering his candidate code and name (our ballot is very simple). Then he/she sends blinded ballot along with $V$-$Lic$ [which is shown in Figure 7(a)]. At Validator server side, there is a servlet which is responsible for receiving filled blinded ballot, verifying the signature of $V$-$Lic$, and returning the blindly signed filled ballot to the voter. In Figure 7(b), voter by pressing Send Data to Tallier the ballot and sending the ballot to tallier server, protocol will be finished and the $V$-$uid$ will be shown, as showed in Figure 8(a). For $V$-$uid$ we have used eight digit random numbers.

For Tallier server side, we have designed two servlets, one for receiving ballot and inserting them into database (we have used MySQL database version 5.5.21) and generating id for each vote and sending it to voter’s card. The other servlet is responsible for individual/public verifiability.

Figure 8(b), shows the Tallier server’s bulletin board during election allowed time. In Figure 9(a) encrypted votes along with their IDs are shown. By searching (entering it in search area) your $V$-$uid$, your vote will be highlighted with red colour as shown in Figure 9(b). Figure 10(a), shows decrypted votes with their IDs and voting licenses (after election). If you want to see your vote you must search your vote id and upon finding, it will be highlighted with red colour, as shown in Figure 10(b).
Design and implementation of a novel secure internet voting protocol

Figure 7  Screen shots of implementation (a) voting phase (b) sending votes and needed data to Tallier (see online version for colours)

Figure 8  Screen shots of implementation (a) showing V-uid of vote (b) Tallier server bulletin board (see online version for colours)
Figure 9  Screen shots of implementation – Tallier bulletin board – collecting phase (a) encrypted voted with V-uid (b) highlighted vote after searching vote id (see online version for colours)
Design and implementation of a novel secure internet voting protocol

Figure 10  Screen shots of implementation –Tallier bulletin board-counting phase: (a) decrypted votes with V-uids and V-Lics (b) highlighted vote after searching vote id (see online version for colours)

7  Protocol evaluation

7.1  Integrity and accuracy

In our scheme because of using PKI, × .509 certification and bulletin board, there is no possibility to change the vote.

7.2  Anonymity

In this scheme, confidentiality is guaranteed through three mechanisms:

- Using blind signature approach.
• Use of Aid which hides all links between voter’s information and her/his vote. RA has only blinded ‘aid’ and other electoral servers have signed ‘aid’ (thus the link between this information is a blinding factor that only the voter knows).

• Intelligently isolating registration phase from administration and other voting phases. The RA deals with voter identity information (hashed value); while others know only the voter’s voting information. Even in case of cracking hashed information and collusion between units, voter’s presence is obligatory for breaching his/her anonymity.

For reducing the possibility of IP tracking, we recommend voters to use public proxy servers to sending their votes.

7.3 Verifiability

Our scheme achieves verifiability by using public bulletin board. As mentioned, in collecting phase, all received voted associated with their voting unique ids are shown in bulletin board. Voters with noticing their encrypted votes on public bulletin boards will make sure their vote has been gathered properly. In the counting phase decrypted votes along with their V-uids and V-Lics are shown. Voters can check the accuracy of an election by comparing the content of public bulletin board before and after election.

7.4 Robustness

In order to achieve robustness each server can use clustering and load balancing methods that can be served by a number of servers operating in an active state as a cluster. Furthermore, the communication channel can use different independent paths.

7.5 Resistance to collusion

In our scheme collusion is prevented within two levels:

1 Masquerade (impersonation) Since masquerade can occur in any of the protocol phases, so we will analyse it in each phase:]

• Masquerade in registration phase: Since each voter is identified with his/her own Java Smart Card 3 and personal identification data are fetched from the card after voter authentication (by PIN or preferably biometric templates), masquerade is not possible. But suppose the case which a briber collects voter’s cards to vote instead of them. Regarding to this point that card is personal; giving the card with its pin is somehow illogical and impractical. However, by authenticating the card holder through fingerprint, this scenario is highly prevented.

• In proposed protocol, justifiability of voters done through received information from the voter card (there is no database in RA keep voter information). So, due to a lack of information about the presence or absence of the voter (eligible voter), it cannot vote instead of them.
Design and implementation of a novel secure internet voting protocol

• Masquerade in administration phase: Is prevented because of RS-License. Since all interactions are encrypted, no one can eavesdrop the channel to gain RS-License. Without the RS-License the attacker cannot produce message three and so the administrator won’t produce a valid EB

• Masquerade in voting and collecting phase: since in administration phase, the V-Lici is protected from eavesdropping by encrypting it with a symmetric key that voter sends in message 3, no one except the voter can know about the V-Lici, therefore without a valid V-Lici, the attacker cannot produce message 5 to vote in place of the voter. Also for more security padding the V-Lici value by a random number shared only with administrator is possible.

2 Breaching voter anonymity: Based on proposed protocol upon cracking the hash value of identity information, only the RA can access voter identity information. In proposed scheme, this is prevented by distributing voter information among voter and electoral officers. Under collusion between units involved in the election they cannot determine who has been given a special vote. Because the voter is the only one who knows the link (blinding factor of Aid) between information delivered to RA and those delivered to other entities.

7.6 Fairness

In this protocol, all votes are encrypted by the election’s public key and the private key is secretly shared between the authorities and candidates. This makes it impossible to decrypt and count votes early before the end of an election. Also as mentioned, private key can be stored in HSM (which is kept in a safe place). It is noteworthy that providing security (controlled access) for physical environment and also HSM is more relevant to ISMS controls and there would be no discussion about it in this paper.

7.7 Bribery and uncoercibility

Since internet-voting is done over network and voters can vote from any location, there is no definite solution to prevent all kinds of bribery and coercion. In proposed protocol all votes are encrypted with election’s public key and decrypted votes will be presented on bulletin board after election. So during an election voters cannot prove their vote in order to buying/selling. This means, paying for a vote will be postponed to after the election and announcing election result. Since bribery is a social issue, once the briber’s favourite candidate cannot be election winner, she/he would have no motivation left to pay voters. Also in this situation, voter cannot trust to briber to pay for his vote after election even if briber’s candidate be the winner of the election.

For coercion, we have used a method from second category which is mentioned in Section 1.1. Bulletin board displays only votes and V-uids during an election. For each vote that voter casts a new V-uid will be assigned and previously casted votes will not be replaced. Because only the voter knows the V-uid, the coercer does have knowledge about number of times that a voter has voted. If a voter, votes in a coercion state, sends the previous pass phrase and the vote along with V-uid will be displayed in bulletin board but Tallier server will not involve it in counting phase. Bulletin board information is enough for convincing the voter but is not enough for to convince a coercer who strives to trace voter’s behaviour. This cause uncertainty of the coercer because she/he can no
longer predict the voter’s behaviour and voter can change her/his votes without knowledge of the coercer.

This solution dramatically decreases the probability of bribery and coercion, although it does not make it impossible. It is noteworthy that by bribery and coercion in a protocol, we mean meeting this property by the protocol itself, regardless of whether the protocol is implemented as an electronic or internet-voting protocol. So in protocols such as (Cetinkaya and Doganaksoy, 2006) in which using a private kiosk is suggested, bribery is not actually solved. Furthermore, we have to keep in mind that when we are talking about bribery and coercion, we assume that the briber and coercer do not physically have access to either the voters or their cards.

7.8 Vote and go

Proposed schema does not require any involvement of voter in the counting phase.

7.9 Mobility

Our scheme is implementable on a station based voting and or as an internet voting protocol.

7.10 Voter-side’s security

As mentioned earlier, one of the main obstacles of implementing internet-voting is voter insecure platforms. Voter’s PCs easily can be infected by malicious software and be controlled by an attacker who can vote instead of a voter. Regarding Java Card 3 properties, it seems to be a very good alternative for voter’s insecure PCs for voting and storing private keys. It can act as a secure web server that directly connects to the network to receive and process HTTP(S) requests and send HTTP(S) responses. Also by using the SSL stack implemented inside the card, communications between electoral servers and card is entirely secured.

7.11 Efficiency

In our scheme, the voter by spending a little time can cast her/his ballot and time complexity of voting is reasonable.

8 Conclusions

In this paper, a new secure i-voting protocol has been presented. Our proposed protocol solves coercion issue, by letting voter to recast with considering special policies about vote display in bulletin boards. Vote selling is postponed to after the election in the proposed protocol. In this way, briber and seller distrust each other. Therefore, possibility of this event is strongly reduced. Moreover, our proposed protocol addresses other requirements of i-voting.
Table 2: A comparison of voting schemes

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Anonymity</td>
<td>Yes</td>
<td>Yes</td>
<td>C</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Accuracy</td>
<td>Yes</td>
<td>C</td>
<td>C</td>
<td>Yes</td>
<td>C</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Resistance to collision</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>C</td>
<td>C</td>
<td>No</td>
<td>C</td>
</tr>
<tr>
<td>Verifiability</td>
<td>Yes</td>
<td></td>
<td>Individual verifiability</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td>Public: Yes individual: partial</td>
</tr>
<tr>
<td>Resistance to collusion</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Verifiability</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uncorecibility</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Vote buying: Yes</td>
<td>Coercion: No</td>
<td>No</td>
<td>An officer verify votes counting, not voters</td>
</tr>
<tr>
<td>Uncorecibility</td>
<td>(coercion not mean physically presence)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Uses voting booths with guards for Casting ballots</td>
<td></td>
</tr>
<tr>
<td>Fairness</td>
<td>Yes</td>
<td></td>
<td>No</td>
<td>C</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Vote and go</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Voter-side's platform security</td>
<td>Yes</td>
<td>Not considered</td>
<td>Not considered</td>
<td>Not considered</td>
<td>Not considered</td>
<td>Yes</td>
<td>Uses voting booths with guards</td>
</tr>
<tr>
<td>Mobility</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

Notes: Yes: satisfy, No: does not satisfy, C: conditionally satisfied
To address voter side insecure platforms, one of the biggest challenges of i-voting systems, we introduced Java Card 3 technology which can be utilised as an independent secure voting module to greatly guarantee client-side security. By providing security in voter side, in addition to securing the entire voting system, voter’s trust and interest to participate in i-voting is increased.

Also we have proposed an implementation of our scheme. Implementation is done by using Java Card 3 capabilities (its servlets and etc.), J2EE, J2ME and XML technologies and shows our proposed scheme feasibility. To the best of our knowledge, this is not only the first implementation of an internet-voting protocol using Java Card 3 but also is the first implementation in using Java Card 3 in e-services.

For future, we intend to produce a formal verification of our protocol correctness. We want to reduce the number of electoral authorities while preserving the advantage and strength of the system. Also, we want to use our proposed system in a small scale election like school council election. A comparison of our scheme and some other proposed schemas [FOO (Fujioka et al., 1993), SEAS (Baiardi et al., 2005 ), REVS (Joaquim et al., 2003; Cetinkaya and Doganaksoy, 2006; Karro and Wang, 1999; Chen et al., 2004)] is shown in Table 2.

Acknowledgements

We would like to give special thanks to Nicolas Bousquet – Senior Engineer from Oberthur Technologies, Samia Bouzefrane – Associate-Professor at CNAM (Conservatoire National des Arts et Metiers, An institution dedicated to life-long higher education in Paris) and Researcher at the CEDRIC Labo in embedded systems and smart card area in particular, Eric Vetillard – from Oracle, Thierry Violleau – from Oracle, Vincent Guerin – from Oberthur Technologies and Pierre Ricourt – Mobile Embedded Systems Engineer, for their valuable and sharp comments and guidelines and helps in implementing this protocol (very practical tips about interactions between Java Card 3 and other web applications, cryptography in Java Card 3, Java Card 3 servlet functionality and many other issues that we encountered in implementation) and also in understanding Java Card 3 technical concepts. We really are grateful to them for sharing their experiences and knowledge with us.

References


Design and implementation of a novel secure internet voting protocol


