FACTA UNIVERSITATIS
Series: **Electronics and Energetics** Vol. 29, N^o 2, June 2016, pp. 205 - 218
DOI: 10.2298/FUEE1602205P

A SECURE E-VOTING FOR THE STUDENT PARLIAMENT

Dragoljub Pilipovic[1], Djordje Babic[2]

[1]Slobomir P University, Bijeljina, Bosnia and Herzegovina
[2]School of Computing, Union University, Belgrade, Serbia

Abstract. *E-voting is a service or system which serves to get individual human inputs and to summarize them to a certain group decision. Usually, e-voting is a take for e-government part, but in this paper we consider e-voting for particular and specific population. The proposed e-voting system is intended for student population and student parliament election. In this paper, we describe concept of P.U.T. (personal unique token) and ways to distribute P.U.T.s to students. At the end, we present a software designed for the student parliament use case.*

Key words: *e-voting, EVS (electronic voting system), P.U.T., randomizer, secure distribution, student parliament, UML (unified modeling language).*

1. INTRODUCTION

In today's society, digital services have become an important part in everyone's lives. Electronic voting is deployed in many countries worldwide [1] [2]. Every year in the last two decades, the number of theoretical and practical solutions and research papers in this field is increasing. We have noticed the existence of more than two hundred papers from the year 2000 onwards. However, there is a scientific paper, which is considered as the pioneer in this area: it is an overview of mix-net scheme for e-voting by David Chaum [3].

The main focus in the e-voting field is most often on technical dimensions: cryptographic algorithms, e-voting protocols and scheme, trusted hardware, software implementation, security, etc. Recently, the focus has moved to the organizational, social, and political aspects of e-voting. Despite of a large amount of research in e-voting field, e-voting has no big real-world success. Direct Recording Electronic (DRE) machines for e-voting have been criticized many times [4-6]. The US Department of Defense proposed a remote and Internet based voting system for elections: Secure Electronic Registration and Voting Experiment (SERVE), but in report [7] there are strong recommendations against deploying this e-voting system. E-voting schemes are the core of e-voting protocols and e-voting systems. We will mention three most often represented in the literature.

Received January 29, 2015; received in revised form November 1, 2015

Corresponding author: Dragoljub Pilipovic
Slobomir P University, Bijeljina, PF 70 Pavlovića put 76, 76300 Slobomir, Bosnia and Herzegovina
(e-mail: dragoljub.pilipovic@gmail.com)

Mix networks (mix-nets) are a cryptographic primitive generally used to obfuscate a path through a network [3]. Mix-nets usually consist of a set of servers, named mixes, which ensure that the output of a mix-net cannot be correlated with its input. In e-voting, mix-nets simulate an anonymous channel between a voter and ballot box [8-10]. The second e-voting scheme mentioned here is the one based on blind signature primitive. In this primitive, messages are signed by a signer, but the message content is kept hidden from the signer. In e-voting, a voter sends encrypted and blinded ballot to the ballot box. After validating the ballot, the voter unblinds the ballot and, therefore, gets a validated ballot which cannot any longer be linked to the original content of it [11-12]. The last e-voting schemes are homomorphic encryption schemes. The scheme is based on the algebraic homomorphic properties of few public-key cryptosystems which permit tallying of an election without the decryption of any single vote [13-14]. In addition to the three aforementioned schemes, there is certainly an interesting paper-based scheme Prêt à Voter. Here, a voter retains a part of the ballot as their encrypted receipt [15]. Technological measures like e-voting approach may increase voter turnout, but some researchers found that e-voting gets the turnout to the initial level at a later stage [16].

Electronic voting system (EVS) described in this paper is specific because it is designed for the student population, consisting mostly of young people, who are characterized by a particular state of mind and practical attitude. They are open-minded, mobile, independent, not included in politics (at least this is valid for majority). They use all the latest technological advances, thus all devices used in e-voting are familiar to them. However, students are apolitical in the sense that they have low voting turnout [17-19].

A problem of distrust in voting can be significantly reduced; perhaps they can even disappear, if e-elections are conducted with educated voters, in addition to the increased simplicity of e-voting. The system proposed in this paper is intended only for student population. The students' 'job' is exactly composed of adoption (and construction) of something new. We believe that students of computer science will have no obstacles to understand and adopt this way of voting. Furthermore, we hope that one day the whole population will have such an attitude.

The EVS for student parliament has to be safe like others EVSs, but it does not need to have an extremely high level of safety features, as it affects a smaller part of the society. In addition, due to the characteristics of the student population, especially with their open mind and independence, we assume that it will be unusual for students to sell their e-vote to the other. The presented EVS will prevent misuses, for example the coercion to vote for someone else will be responded by appropriate mechanisms proposed in our EVS.

The main contribution of our paper is a concept of voting vectors, called P.U.T., which consists of readable string of numbers and characters. Printed P.U.T. list is anonymous as long as the voter keeps it in a secret and secure place. In addition, we show design and implementation of electronic voting system (EVS) based on P.U.T. concept. The EVS provides a very high usability because of its simplicity. The system requires only limited capabilities on the side of the voter: ordinary computer and practically any web browser.

The paper is organized as follows. In the second section, we present the concept of P.U.T. vectors for e-voting. The third section explains two ways to secure the distribution of the generated vectors for e-voting. In the next section, the entire protocol i.e. system for e-voting in student elections, is described. Finally, at the end, we show the design and layout of software for e-voting based on the principles given in the previous sections.

2. P.U.T. CONCEPT

The most important issue in voting is voter identity problem. Here, this problem is solved by using a method called personal unique tokens (we use abbreviation P.U.T. from this point further). In the classical, paper based model of e-voting, the voter identity and his/her choice are sent separately to the information system for e-voting. The voters' identities are sent most often through a digital signature, and the voters' choice has to use a secure connection to the database. The main idea of the P.U.T. concept is that we merge into one entity these two things which are separated physically and in-time, although one follows immediately after the other. This entity is sent to the information system of e-voting. In this way, P.U.T. represents identity of the voter and his choice at the same time (Fig. 1).

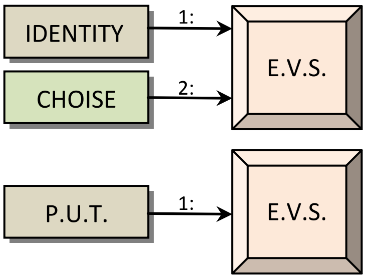


Fig. 1 Personal unique tokens

Data sent as a vote of specific user is given as a unique answer (a vector), which is related to the uniqueness of each voter for each option he can vote. P.U.T. is formed in such way that it is easy to read it and easy to enter it to the computer. In Table 1, an example of P.U.T. list for a specific voter is shown. Each row in Table 1 represents P.U.T data that should be sent to EVS for a specific voting option. In this case, the voting option is represented by name and surname of an election candidate. One voter has only one P.U.T. list for specific election race.

Table 1 Example of the P.U.T. list for e-voting

Option	P.U.T. – First Voter			...	P.U.T. – Last Voter		
Slobodan Milutinović	ABC	123	CBA		GHW	111	KKK
Milan Milošević	FGH	907	USO		JAP	231	GAP
Boris Nikolić	UUU	000	III		HAG	790	GRW
Tomislav Tadić	EAE	888	NHF		IAE	834	YIY

Here, we propose the following format AAA XXX AAA for generating P.U.T.s, where A is a letter of the Latin alphabet (such alphabet exists on every keyboard, every operating system, every type of device and at least everyone knows Latin letters) and X is a decimal digit. In order to calculate the number of available P.U.T.s, for each group of three positions we use a formula to calculate the variations of *k* elements over a set of *n* elements with repetition:

$$\overline{V}(n,k) = n^k \tag{1}$$

After that, we multiply the resulting numbers to each other:

$$26^3 \times 10^3 \times 26^3 = 17.576 \times 1.000 \times 17.576 \tag{2}$$

In this way, we calculate the number of available personal unique answers, which is equal to 308.915.776.000 (a little over 300 billion). This number should satisfy any existing elections for the student parliament.

The alternative is to additionally use lower-case letters of the Latin alphabet, but reducing the number of the text position at the same time. In this way, there are two groups of two letters instead two groups of three letters (current AA XXX AA). This case sensitive alternative offers 7.311.616.00 unique P.U.T. values.

$$(26+26)^2 \times 10^3 \times (26+26)^2 = 2.704 \times 1.000 \times 2.704 \quad (3)$$

The next alternative is to use alphabet with a huge number of different letters. It is believed that a Chinese must know 4.000 characters for conventional literacy [20]. If the positions of the characters from Chinese alphabet are still reduced over to the 1 + 1 position, we will get 16.000.000.000 variations of P.U.T. values. The current format looks like A XXX A.

$$4.000^1 \times 10^3 \times 4.000^1 = 4.000 \times 1.000 \times 4.000 \quad (4)$$

3. Distribution of Answers

Distribution of potential e-votes in the form of P.U.T. list to the student voters is a critical and very sensitive step. The reason is that the P.U.T. list contains the identity of the voter and the virtual ballots, and therefore should be protected from misuse. In fact, this list must be kept in the strictest confidentiality. Only a voter can have an insight into the content. There are two possibilities that meet this requirement: paper option and software option. In both distribution options, it is necessary that the voter himself (i.e. a student in our case) appears at the site of student election organizers.

The paper option is triggered when the operator chooses this option by pressing the corresponding button in the election software (details of election software are provided in the following sections). The P.U.T. list is printed for a present student, whose identity has previously been established. The default printer prints without a preview. The printer is designed in such a way that the text is printed at the bottom of the paper, and therefore printed text is invisible to the operator or anyone else. In addition, the operator can not see P.U.T. list on the screen because this option is not implemented in election software. The sealed envelope with the printed values is given to the voter after loading the paper in an opaque envelope without turning the printed page (envelope and paper are of the same size), or the paper is folded without turning on the printed page and then it is put in an usual size envelope. The voters, who watch out for irregularities in the operator's work, monitor the registration process all the time.

The software option of P.U.T. distribution encrypts the list with a strong symmetric algorithm. EVS generates the Windows executable file which is transferred to the voter's USB flash drive, or possibly to optical media (for example CD or/and DVD), and then handed to the voter. When executed, this applet has only one text box for entering password that unlocks and displays the bitmap image with P.U.T. list. The design and features of this applet are similar to the digital wallet. For more portability, Java applets can be generated with a similar purpose, or even native applications for all popular operating systems (Windows, Linux, Mac OS X, Android, iOS).

4. ARCHITECTURE AND PROTOCOL

The protocol of the proposed EVS has three phases. Each phase consists of several actions. In a sequel, we give description of these three phases and corresponding actions Fig. 2).

A. ***Phase I***. The phase I is ***Pre-election phase***, which comprises all the necessary preparatory work for the second phase. It consists of the following two actions:

1. *EVS initialization* consists of: launching software for e-voting in the appropriate mode; allocation of responsibilities/duties to administrators, as well as the establishing list of students eligible to vote, which is obtained externally or the list is made in this sub-phase.
2. *Voter registration* of students who want to vote through the e-voting system, which must be done personally (face to face) when a voter gets the P.U.T. list. The voters will subsequently receive additional instructions that enable successful e-voting, e.g. address for voting, authentication data, etc.

B. ***Phase II***. This is ***Voting phase***, in which voters make selection electronically, and it consists of the following sub-phases:

1. *Voter authentication* confirms that the person is exactly a voter eligible to vote, i.e. he/she is on the list for e-elections. The voters use the instructions and data obtained in the sub-phase I-2.
2. *Ballot* is available to voters. It should be blank and anonymous.
3. *Act of voting* in which the voter fills ballot and submits it to the EVS.

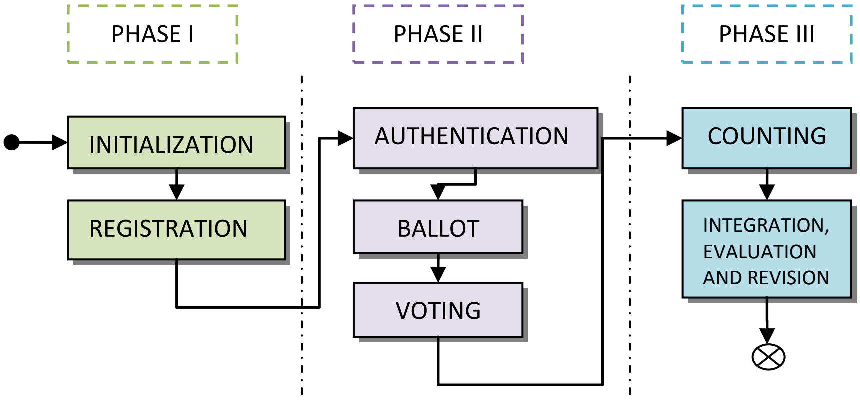


Fig. 2 Phases in proposed EVS

C. ***Phase III***. This is ***Election phase*** in which are all those activities that take place after the closing moment of elections occur.

1. *Vote counting* consists of a ballot collection (e.g. from individual polling stations, the ballots are forwarded to the central location), preparation for processing (e.g. if ballots come encrypted, then their decryption starts), and finally there is counting of e-votes according to the rules defined in the sub-phase I-1 in order to obtain accurate results. The results can be made public, for example through the news channel on the official website of election commission.

2. *Integration, evaluation and revision*. If e-voting is only part of the overall election, then voting results will be summed up in order to obtain the final results. Evaluation assesses the process of e-voting, and then generates statistics that will help to improve the whole process and make it more transparent. Revisions are performed if there is suspicion about the results or the implementation of e-voting procedures, but most often it comes down to a recount of e-votes. [21]

Architecture and protocol for e-voting proposed here, are intended for imaginary elections for a student parliament. Its properties should be easily scaled to the general purpose elections. It is assumed that there is a database with several thousand students – eligible voters. Next assumption is that there will also be candidates, a few dozen, for example. The entire process of e-voting is divided into three stages, which are not time-overlapping. These stages are registration process, voting day and post-election period. In the first stage, students are registered for e-voting with identification documents like ID card or student card. Registered students are recorded in the database and at that point, they receive their P.U.T. lists. P.U.T.s are unique, randomly generated series of decimal digits and Latin alphabet letters, as explained above. Each personal token is located at the intersection of voters' row and candidates' column of P.U.T.s matrix. The P.U.T list is given to student voter using one of the two options described in the previous section.

Voting day is realized for students to vote on the web site with the corresponding simple form which any web browser is able to read and render because it consists mainly of basic HTML elements. Web site for the elections represents the equivalent of polling stations. The main e-voting page is removed from the web server after the voting closes.

5. Unlinkability, Anonymity and Verifiability

In the proposed EVS, there are two databases. The first database (DB) is completely offline (e.g. separated from the Internet) and the second DB is used just for e-voting on the Internet. The second DB contains anonymous P.U.T. lists and the results of voting, while the first DB contains everything else. The second DB is made from the first DB (see details in the next section). The database on the web server (the second one), which is used to check whether a certain P.U.T. value is valid or not, does not contain personal data of students, but only a list of P.U.T. values and their associated candidate. The following is a description of the properties of the proposed EVS (graphically depicted on Fig. 3).

Unlinkability: Since P.U.T.s are independent from personal data of voters, because they are not derived from them, but they are uniformly distributed in the domain of values, there is no direct link between voters and e-votes.

Anonymity: If P.U.T. list is kept as top secret, then the voter and his e-vote will be anonymous.

Verifiability: Upon closing e-voting, all valid P.U.T.s are gathered from second DB and published on the website of the election commission. At this point all students can check whether their e-votes are recorded. Whole P.U.T. list is also published on the web site, thus anyone can verify whether the final results are correct because each P.U.T. value can be connected to an election candidate.

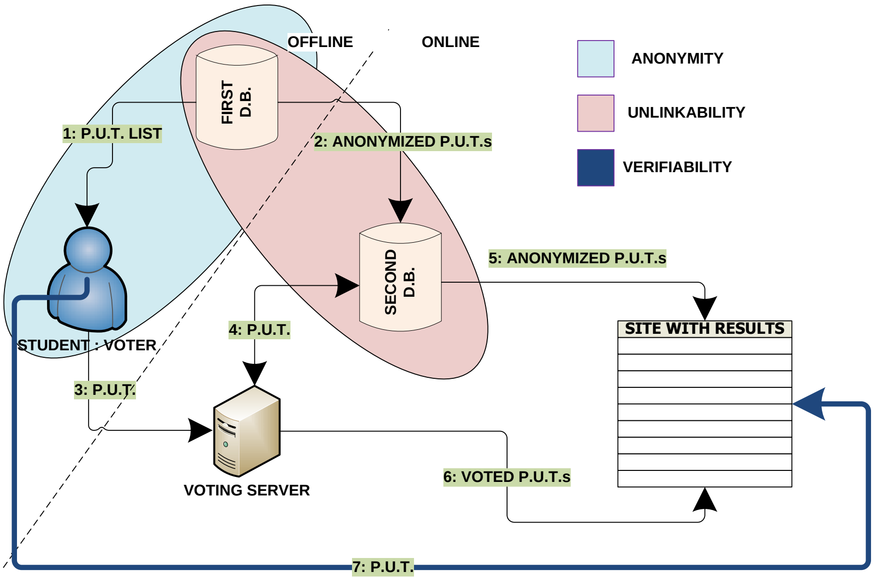


Fig. 3 The flow of P.U.T.s, P.U.T. lists, voted P.U.T.s and anonimized P.U.T.s

6. Design and Implementation of EVS

All previous sections can be seen as a programming task, or user requirements for the process of developing software. For the software development, we use well known and common Larman method [22]. Here, we show only the most significant parts of the project documentation.

Figure 4 provides a global use case diagram divided by domain areas. It can be seen that there are two actors: an operator (synonym to a voting official), and a student (synonym to an eligible voter). In total, there are sixteen use cases (not shown here) in four domain areas. Based on these use cases, we design EVS software.

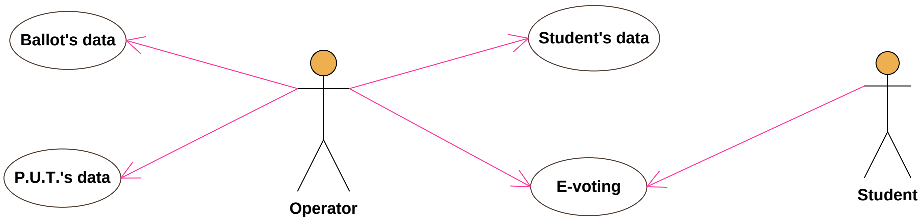


Fig. 4 Global use case diagram

The following three figures shows the GUI (graphical user interface) of the proposed EVS software. The software is developed in Visual Studio IDE, the C# programming language, and ADO.NET and ASP.NET technologies. Figure 5 shows the server part of the software that is targeted for Windows 8 implemention platform, and the remaining Figures 6-8 show the client part.

At Student's data tab are implemented operations to work with eligible voters - students with voting rights. The students can be added, modified or deleted, and the data about them can be imported from .xml file with the appropriate scheme. This is designed and implemented based on Student's data use cases. At Ballot's data tab are placed the maintenance of elections data for the student parliament and in particular setting the options to vote (e.g. candidates) for the virtual ballot. This is based on Ballot's data use cases. At P.U.T.s data tab are implemented the features for P.U.T. generation, printing and creating digital wallet application within P.U.T. list. There is an issue (we call it 'randomizer efficiency problem') that is reflected in the slow rate of generating P.U.T. values, which requires further optimization.

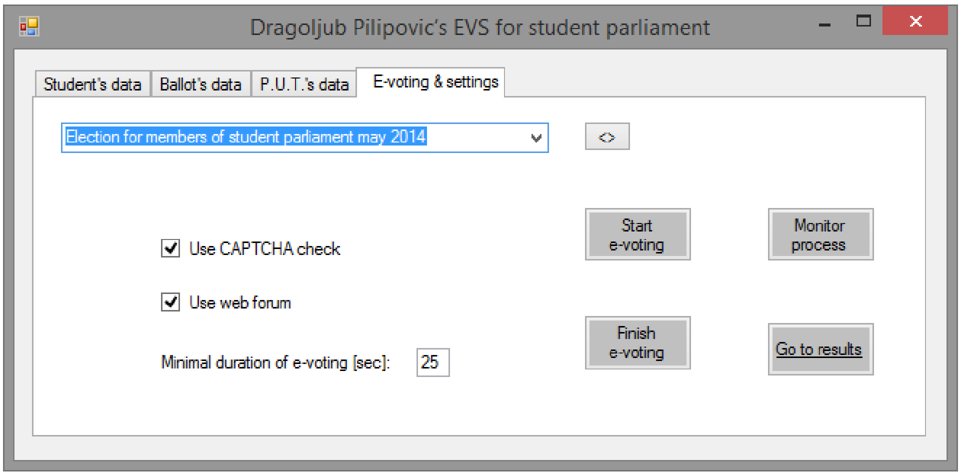
Algorithm 1 Pseudo-code for generation of P.U.T. list for every eligible voters

```
main_program();
initialize put_list as matrix[MaxVoters,MaxVoteOptions];
put_list(1,1)=generate_candidate_for_put();
for i:=1 to (MaxVoters*MaxVoteOptions) do
    found:=true;
    while found do
        temp:=generate_candidate_for_put();
        for j:=1 to i-1 do
            y:=j-(round(j/MaxVoters)-1)*MaxVoters;
            if put_list(x:=round(j/MaxVoters),y)==temp then
                found:=false;
                break;
            end if
        end for
    end while
    y:=i-(round(i/MaxVoters)-1)*MaxVoters;
    if found then put_list(x:=round(i/MaxVoters),y):=temp else i--;
end for

generate_candidate_for_put();
initialize candidate as array[9];
for i:=1 to 6 do
    temp:=rand_between('A','Z');
    if i<3 then place:=i else place:=i+6;
    candidate[place]:=temp;
end for
for i:=1 to 3 do
    temp:=rand_between('0','9');
    candidate[i+3]:=temp;
end for
return candidate;

round(x);
if integer(x)==x then y:=x else y:=integer(x)+1;
return y;
```

Figure 5 displays a part of the EVS software that is used to adjust settings of the e-voting. The check box 'Use CAPTCHA check' is used to determine whether or not the voter is human (CAPTCHA - Completely Automated Public Turing test to tell Computers and Humans Apart). The check box 'Use web forum' determines whether the output of e-voting goes to public web site. The text box 'Minimal duration of e-voting' determines shortest time in seconds to finish cast an e-vote.



Dragoljub Pilipovic's EVS for student parliament

Election for members of student parliament may 2014

☑ Use CAPTCHA check

☑ Use web forum

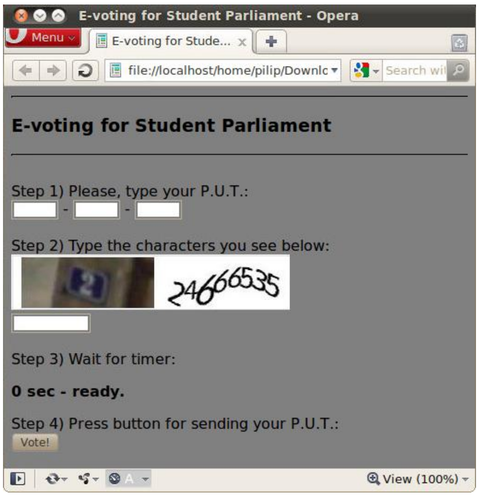
Minimal duration of e-voting [sec]: 25

Fig. 5 GUI for some *E-voting* use cases

Figures 6 and 7 show two different client parts of the proposed EVS software, which are designed and implemented by two use cases of E-voting domain area.

Figure 6 displays the client part of EVS in Ubuntu Linux on the web browser Opera. It is a virtual polling place and a virtual ballot. A voter fills in the first row of text boxes form with his/her own P.U.T. list, and then types CAPTCHA series, in the second row. When the timer ends up, a student can press Vote button, and send completed virtual ballot to the server.

Figure 7 depicts applet with P.U.T. list. This is partly covered by E-voting use cases. Physically, the applet consists of three parts. The first part of the applet is executable file (extension is .exe), which is always the same for all voters. At the top of the applet, there is a password field (in fact it is a decryption key), besides that there is a View button and below there is a decrypted image with P.U.T. values. The second part of the applet is a Readme.txt file with instructions for using the applet. The third part, the most important component of the applet, is Picture.jpg. This file represents the bitmap image with high compression rate, which is encrypted with AES algorithm. Encryption key length is 128 bits (16 bytes in the form of password). The symmetric algorithm AES is selected because there is a good support for it in the Windows operating system with .NET Framework 3.5 with whom whole software ecosystem is built.


E-voting for Student Parliament - Opera
Menu
E-voting for Stude...
file://localhost/home/pilip/Downlc
Search wi
View (100%)

E-voting for Student Parliament

Step 1) Please, type your P.U.T.:

Step 2) Type the characters you see below:

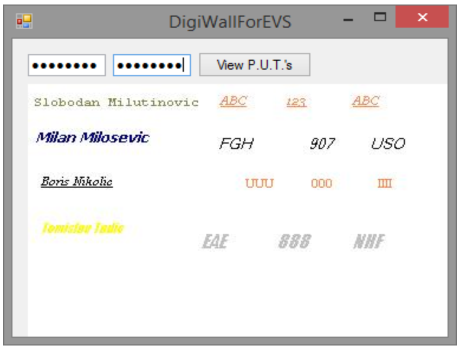
24666535

Step 3) Wait for timer:

0 sec - ready.

Step 4) Press button for sending your P.U.T.:

Vote!



Slobodan Milutinovic	ABC	123	ABC
Milan Milosevic	FGH	907	USO
Boris Nikolic	UUU	000	III
Tomislav Tadic	EAE	888	NHF

Fig. 7 Applet DigiWallForEVS with decrypted P.U.T.s

Fig. 6 Virtual ballot in the client part of EVS

All three components are packed in .zip archive with the ‘extract-to-folder-pseudorandom_number’ name. Here, the last number is obtained from the sequence whose pseudorandom generator seed is set at the start of the EVS software. The image with P.U.T.s is rendered with random font, size, color and style in order to decrease possibility of success of

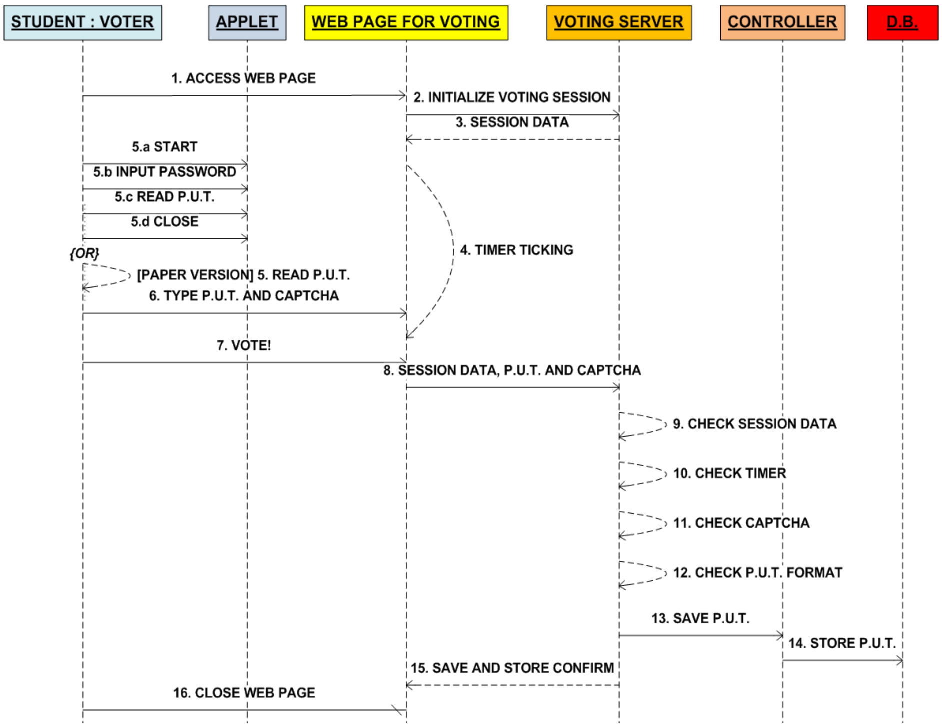


Fig. 8 Sequence diagram for Phase II of proposed EVS

the OCR (optical character recognition) process. The password for decryption is not stored anywhere. However, only the voter knows the password because the voter is a legitimate user. In order to improve protection of program code by adding prevention for readability free obfuscator software Obfuscar is used.

A sequence diagram in Figure 8 shows objects in the proposed EVS and their interactions in the sequential order that the interactions occur. The sequence diagram is a form of interaction diagram in UML and it models the collaboration of objects based on a time sequence.

Databases are stored in RDBS SQL Server. Figures 9 and 10 display schemes of both relational databases.

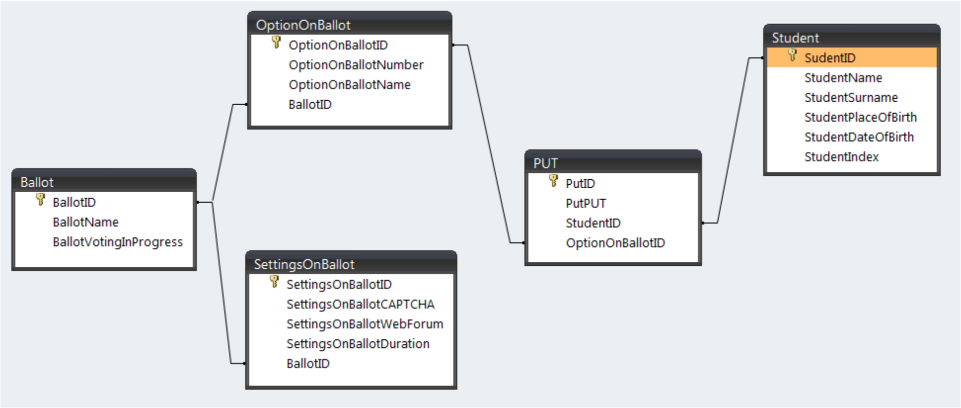


Fig. 9 Relational scheme for offline DB

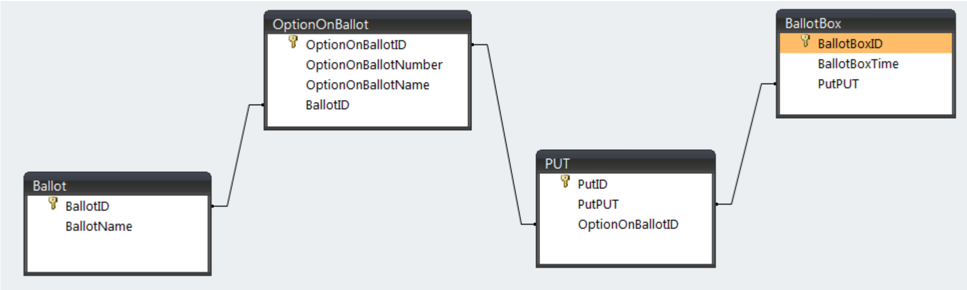


Fig. 10 Relational scheme for online DB

In order to better perceive components and layout of hardware and software of our EVS, Figure 11 shows the UML deployment diagram.

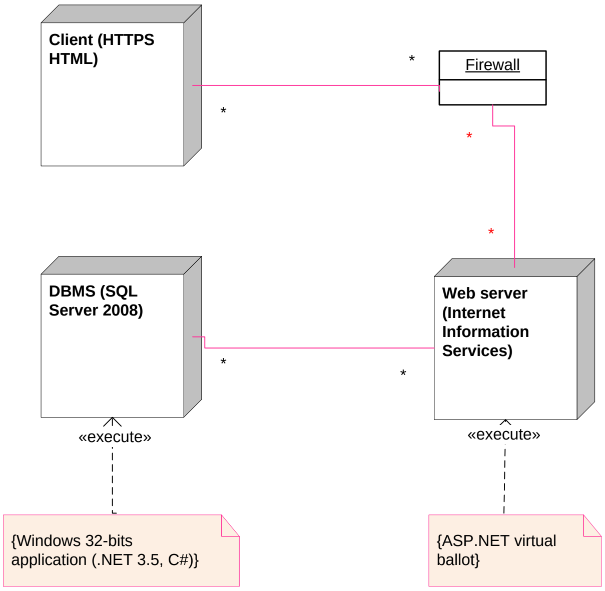


Fig. 11 Deployment diagram of EVS components

7. Discussion and Conclusion

In June 2015, we conducted a pilot e-voting with software presented in the article. The aims were to obtain practical experience in conducting process of e-voting and to collect feedback from the participants. The pilot was locally coordinated by the pilot election team. There were two separated e-voting groups. First of them consisted of second year students and second group were fourth year students. Both groups belonged to department of information technology, the total of 47 students. Students from the first group voted for types of project in database course they needed to finish as a pre-exam engagement, while students from the second group voted for a leader of software project they needed to make. First group's voting imitated elections with political parties and/or political options, while the other one resembled voting for candidates in the elections.

After the election period, we organized mini surveys among e-voting participants. The survey had five questions: one yes/no type, one question with open answers and the rest of them were with Likert type with 1-6 scale. Students' evaluation was generally positive to e-voting and the EVS. Nevertheless, usability of the EVS was rated at the middle of scale. The reason for this can be seen from the answers to the open-type question. Students were asked for mobile application for e-voting, particularly application for Android operating system.

Prêt à Voter was used in student elections in both Luxembourg and Surrey [23] (the EVS is mentioned in Introduction). In [24] is documented use of Punchscan in the 2007 student elections at the University of Ottawa. Punchscan is paper-based EVS with optical-scan counting of votes. Bingo Voting was applied in the election of the student parliament in Karlsruhe Institute of Technology in 2008 [25]. Security of the EVS relies on trusted random number generating devices like Bingo machines. It has property of E2E (end-2-end) verifiability.

These EVS, used in academic and student environment, are not considered comparable to our EVS, since they have not fully eliminated the need for paper. Therefore, for the sake of comparison, we use EVS described in [26]. It is based on personal smart cards with digital signatures protocol and mixing of votes. It is more technically complex than our EVS, since it requires an additional device, i.e. a smart card reader. Our system does not seek personal smart card as a prerequisite, which is still not widespread.

Otherwise, any general election EVSs or e-voting schemes can be used for the purposes of academic and student voting, e.g. most frequently mentioned schemes from the Introduction. Still, it is not happening because it would be irrational and the waste of resources. Our EVS has a relatively simple structure and organization, which leads to overall practicality of implementation. Beside its simplicity, the EVS has three important properties that make it suitable for the intended purpose. These are properties of unlinkability, anonymity and verifiability.

The proposed EVS can be applied with additional effort at general-purpose elections. In addition, with some minor changes or even in identical form given here, it might be applied to any kind of e-voting, such as corporate voting, syndicate voting, etc.

However, the e-voting pilot showed that the usability of e-voting software must be improved. Web site for e-voting must have a professional appearance. Next, trusted mobile applications for e-voting should be developed, which will be a sort of gateway or shortcut for an e-voting web site. Security can be improved easily and significantly with qualified digital certificate at e-voting server for to give the HTTPS connection.

For the application in the general-purpose elections, it is necessary to analyze the EVS with appropriate methodology or protection profiles. Currently, there are three protection profiles for EVS: BSI-PP-0031 in Germany, PP-CIVIS in France, and IEEE P1583 in USA [27]. Further analysis could be modeling of potential threats based on the components in a scheme, in conjunction with attack trees offering possible ways to handle such threats and/or errors [28]. Another option is using applied π-calculus to symbolically describe the system and their operations and attributes [29].

Finally, the pilot project should be implemented on a large electorate in order to identified errors and received feedback from a larger number of participants.

REFERENCES

[1] *Project World Map of E-Voting.CC GmbH*, [Online]. Available at: http://www.e-voting.cc/en/it-elections/world-map/ (current 2015).

[2] D. Pilipović, “Razvoj, trenutno stanje i perspektive e-glasanja“ (in Serbian), In Proc. Infoteh ’14, Jahorina, pp. 1225-1228, 2014.

[3] D. Chaum, “Untraceable Electronic Mail, Return addresses, and Digital Pseudonyms”, *Communications of the ACM,* vol 24, no 2, pp. 84-90, 1981.

[4] T. Kohno, A. Stubblefield, A. Rubin, and D. Wallach, “Analysis of an Electronic Voting System”, In Proc. of IEEE Symposium on Security and Privacy, pp. 27–40, 2004.

[5] Electronic Voting Machine Information Sheet, Sequoia Voting Systems AVC Edge, Version 1.1., EFF (Electronic Frontier Foundation), 2006.

[6] J. Bannet, D. W. Price, A. Rudys, J. Singer, and D. S. Wallach, “Hack-a-vote: Security issues with electronic voting systems”, IEEE Security and Privacy, vol. 2, no. 1), pp. 32–7, 2004.

[7] D. Jefferson, A. D. Rubin, B. Simons, and D. Wagner, “A Security Analysis of the Secure Electronic Registration and Voting Experiment (SERVE)”, 2004, [Online]. Available at: http://www.servesecurityreport.org (current January 2004).

[8] C. A. Neff, “A verifiable secret shuffle and its application to e-voting”, *ACM Conference on Computer and Communications Security*, pp. 116–125, 2001.

[9] M. Jakobsson, A. Juels, and R. L. Rivest, “Making Mix Nets Robust for Electronic Voting by Randomised Partial Checking”, In Proc. of the 11th USENIX Security Symposium, Berkeley, pp. 339–53, 2002.

[10] D. Wikström, “A Sender Verifiable Mix-Net and a New Proof of a Shuffle”, In Proc. of the Advances in Cryptology - ASIACRYPT 2005, pp. 273–92, 2005.

[11] G. Dini, “A secure and available electronic voting service for a large-scale distributed system”, Future Generation Computer Systems, vol. 19, no. 1 , pp. 69-85, 2003.

[12] Yu-Yi Chen, Jinn-ke Jan, and Chin-Ling Chen, “The design of a secure anonymous Internet voting system”, Computers & Security, vol. 23, no. 4, pp. 330-337, 2004.

[13] R. Cramer, R. Gennaro, and B. Schoenmakers, “A Secure and Optimally Efficient Multi-Authority Election Scheme”, European Transactions On Telecommunications 8, pp. 481–490, 1997.

[14] M. Hirt and K. Sako, “Efficient Receipt-Free Voting Based on Homomorphic Encryption”, In Proc. of the Advances in Cryptology - EUROCRYPT 2000, Bruges Belgium, pp. 539–56, 2000.

[15] P. Ryan, “A Variant of the Chaum Voter-verifiable Scheme”, In Proceedings of the workshop on Issues in the theory of security, WITS ’05, New York, pp. 81–88, 2005.

[16] V. D. Besselaar, et. al. “Experiments with E-Voting Technology: Experiences and Lessons”, Building the Knowledge Economy: Issues, Applications, Case Studies, IOS Press, 2003.

[17] E. Dahlstrom, J. D. Walker, and C. Dziuban, “ECAR study of undergraduate students and information technology”, Boulder, CO: EDUCAUSE Center for Applied Research, 2012.

[18] M. R. Jeffreys, Nursing student retention: Understanding the process and making a difference, Springer Publishing Company, 2012.

[19] T. Nyundu, K. Naidoo, and T. Chagonda, “Getting Involved on Campus: Student Identities, Student Politics, and Perceptions of the Student Representative Council (SRC)”, *JSSA Vol 6*, pp. 149-161, 2015

[20] J. Norman, *Chinese*, Cambridge University Press. 1988, p. 73.

[21] D. Pilipović, “Međunarodni standardi i preporuke kod elektronskog glasanja” (in Serbian), *Infoteh* 14, Jahorina, pp. 1233-1236, 2014.

[22] C. Larman, “Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development”, *Prentice Hall PTR*, 2004

[23] C. Z. Acemyan , P. Kortum , M. D. Byrne, and D. S. Wallach, Usability of Voter Verifiable, End-to-end Voting Systems: Baseline Data for Helios, Prêt à Voter, and Scantegrity II, USENIX Journal of Election Technology and Systems (JETS), vol. 2, no. 3, July 2014.

[24] A. Essex, J. Clark, R. T. Carback, and S. Popoveniuc, "Punchscan in practice: An E2E election case study", In Proc. 2007 IAVoSS Workshop on Trustworthy Elections (WOTE), Ottawa, Canada, 2007.

[25] M. Bär, C. Henrich, J. Müller-Quade, S. Röhrich, and C. Stüber, Real World Experiences with Bingo Voting and a Comparison of Usability, Workshop On Trustworthy Elections, WOTE 2008, 2008.

[26] R. Krimmer, A. Ehringfeld, M. Traxl, The Use of E-Voting in the Austrian Federation of Students Elections 2009., In Proc. of the 4th International Conference EVOTE 2010, 2010.

[27] K. Lee, Y. Lee, D. Won, and S. Kim, Protection Profile for Secure E-Voting Systems, ISPEC 2010, LNCS 6047, pp. 386–397, 2010.

[28] J.H. Espedahlen, Attack trees describing security in distributed internet-enabled metrology. Master’s thesis, Department of Computer Science and Media Technology, Gjøvik University College, 2007.

[29] S. Kremer, M. Ryan, and B. Smyth, Election Verifiability in Electronic Voting Protocols, ESORICS 2010, LNCS 6345, pp. 389–404, 2010.