Tatra Mt. Math. Publ. **41** (2008), 119–131

TATRA MOUNTAINS
Mathematical Publications

REMOTE VOTING USING SMART CARDS WITH DISPLAY

Łukasz Nitschke

ABSTRACT. Internet voting is one of the most desired elements of modern democracy. It would make the voting process more convenient, less expensive, and it could also be a democracy enhancing factor. Unfortunately, almost all contemporary protocols for Internet voting have an inherent flaw — they neglect susceptibility of the voter's PC to cyber attacks. Another issue is providing verifiability and disabling massive vote selling. This paper presents a protocol which does not suffer from any of these disadvantages and can be integrated with traditional elections. The protocol is based on blind signatures and mixnets and is designed to be run by secure devices with an integrated display. As a result implementation costs rise, however with the advent of e-paper displays such devices, in the form of smart cards, are expected to become commodity items in the near future [M. Torrieri: *On card displays become reality, making cards more secure,* 2006].

1. Introduction

If we take a critical look at the traditional voting method that we have been using for years we can observe many opportunities for fraud along with the inability of the citizens to verify the election results. This gives a strong motivation for computer scientists to design electronic mechanisms that could realize voting, and that would not only disable cheating and allow checking, but also lower the costs. Three approaches to the problem are proposed: poll-site voting, kiosk voting and Internet voting. The first one involves special voting machines placed in the voting booths. The votes are cast in interaction with the machines. In this case, contrary to kiosk voting, we can assume control of the voting environment and presence of election officials. The second approach realizes the idea of publicly available terminals (e.g., in shopping malls) that enable citizens to cast votes. Internet voting, by contrast, requires

2000 Mathematics Subject Classification: 94A60, 68P30.
Keywords: internet voting, remote voting, cyber attack, verification, protocol.

no dedicated machines. The voting process is performed by a client-server application, run by a voters PC, and on the server side, by trusted authority or authorities. Of course, the third approach is more convenient and economical. It also reflects the needs of the modern society. Nonetheless it is considerably more demanding from the security perspective, as we have to take into account less control of the voter's environment and various cyber attacks. The attacks may be launched from anywhere, and may target many voters causing major damage to the state and discrediting Internet voting in general. Possible abuses that we can expect are automated vote selling and malicious changing of votes [6, 16].

The proposed scheme and infrastructural model for Internet elections is an attempt to minimize chances of a successful outsider attack. The key element used in this solution is a tamper-resistant device with a built-in display. Since the device should be portable, it can have the form of a smart card or, ideally, a plastic ID card. It should also be able to perform basic public key cryptography.

The scheme is based on blind signatures. This type of protocols are characterized in comparison to other solutions in Section 2. Next section gives a technical description of the proposed protocol, which is followed by the discussion, in Section 4, on the design goals and achieved properties.

2. Voting schemes

2.1. Voting based on blind signatures

Blind signatures were introduced by C h a u m [4] as a means of authenticating a message without learning the content (see Section 3.1). This signing method was originally used for e-cash, but was adopted to e-voting as well. Such voting schemes require three phases:

- *registration* — using blind signatures a voter obtains a voting token (a blindly signed identifier) from the *Election Committee* (EC) (through authenticated channel),
- *voting* — voter sends his/her ballot and token to the *Counting Committee* (CC) (through an anonymous, private channel),
- *counting* — votes are counted and usually published by the CC.

During the registration phase the voter's identity is checked, and his/her voting rights are verified. It means that all of the protocols have the *eligibility* property. Voting phase employs a channel which guarantees privacy, and anonymity.

Moreover, the information published in the final phase allows *global verifiability* and, in most protocols (e.g., Okamoto, Fujioka and Ohta (FOO) [2], Radwin [15], Juang and Lei (JL) [9]), *individual verifiability* — ability to verify election's result and voter's individual vote, respectively. Nonetheless, disabling vote selling possibilities (*receipt-freeness*) requires abandoning individual verifiability (e.g., Radwin (NL) [15]). This trade-off is quite obvious, as on the one hand, the voter needs digital evidence (remote voting) for verification. On the other hand, the voter should be unable to prove to a third party for whom he/she voted.

The problem of providing receipt-freeness and verifiability was overcome in Okamoto (OKA) [14] by splitting the functionality of CC between two authorities. One authority receives and publishes trapdoor commitments to votes and tokens, whereas the other obtains the uncovering values. The values are used to publish the list of permuted votes along with a zero-knowledge proof of their correctness in respect to the commitments published by the first authority. This is a slightly different type of verifiability — the voter sees his/her commitment published, and is assured that the votes, as a group, were correctly uncovered. Nonetheless, the voter has to trust that the software he/she uses sent correct uncovering data, as it is in the case of Okamoto [14], or has to perform additional testing as in poll-site voting protocols described below. We will denote this type of verifiability as *indirect*.

2.2. Poll-site vs Internet voting

So far, the biggest effort was put into developing protocols for poll-site voting. The most advanced solutions are not based on blind signatures. They offer, however, indirect verifiability (and receipt-freeness) along with the interactive verification of the machine that produces ballots [1, 5, 13]. The verification takes place through printing of two ballots and letting the voter choose which to check. The chosen ballot is compromised by providing (usually printing) additional data which allows verification. The process requires a computer program and equipment (e.g., a scanner). In practice, the verification is meant to be carried out by watchdog organizations that collect the ballots at polling stations or somewhere else. As a consequence, the remaining ballot is believed to be properly constructed and is used to cast a vote. The ballot without additional data cannot serve as evidence for vote selling (receipt--freeness). It has to be emphasized that probabilistic testing is a strong enhancement to indirect verifiability in the context of untrusted hardware and software.

Although indirect verifiability was successfully adopted to poll-site voting, it cannot be easily applied to Internet voting by employing the same techniques (probabilistic testing). This is the consequence of the following observations:

- It is difficult and expensive to reduce the size of voting machines used in poll-site voting to make them portable (integrated printer and other implementation aspects).
- We cannot replace dedicated machines with voters' personal computers. This is caused by the fact that a PC is not tamper-proof, and may leak secret information to the voter, enabling vote selling. For instance, probabilistic checking of ballots relies on the assumption that the voter does not learn verification data for the proper ballot.
- Verification of ballots requires either a trusted testing device or a third party organization.

Some of the voting protocols for poll-site voting have the property that printing of the ballot does not require interaction with the voter [21, 17]. These protocols with pre-printed ballots, may be used to remotely cast a vote if we assume pre-elections distribution of paper ballots. Another example of a protocol based on paper ballots designed specifically for Internet voting is the SureVote system [19]. Apart from the solutions based on paper ballots, we can also distinguish protocols based on trusted hardware. Most of them do not deal with the problem of kleptography (see Section 4.2) or PC corruption [8, 7, 11], while other tend to be complicated [10].

3. New protocol

3.1. Building blocks

RSA blind signatures. Blind signature schemes are a form of digital signatures in which a message, before being signed, is blinded by the receiver. As a result the signer is unable to see what he/she is singing. The signature can be publicly verified. We will utilize blind signatures scheme based on the RSA algorithm that was introduced in [4]. The scheme requires the following steps to be performed by the participants. The receiver chooses a random value r, and blinds message m by computing $r^e m \mod n$, where (e, n) is the signer's public key. The blinded message is sent to the signer, who computes $s' = (r^e m)^d \mod n$. The value s' is returned to the other party. The signature $s = m^d \mod n$ is obtained by computing $s' \cdot r^{-1} \mod n$.

Mixnets. One of the most important branches in research of electronic voting are protocols based on mix networks. Mix network protocols allow to shuffle a list of encrypted messages in a distributed way by λ trusted parties (mix servers). The parties sequentially permute and transform elements on the list. The resulting list is passed on to the next mix server via an authenticated public channel (BB). Transformations carried out by a single server obfuscate relations

between input and output elements. Therefore, it is hard to determine the secret permutation of a single server, and in consequence the global permutation of the whole mixnet. We need two functions to perform distributed shuffling:

- $onion(m)$ — creates an initial encrypted form (called onion) of m that passes through the mix servers,
- $transform_k(c)$ — transforms ciphertext c into c' so that : c' encrypts the same message as c, and it is difficult to prove this fact without the knowledge of k.

There are different types of mixnets, depending on the transformation function. For the purpose of this paper we will employ partially decrypting mixnets [3]
— each server partially decrypts ("peals the onions") elements on its input list
— the last server yields messages.
This type of mixing uses the following functions :

- $(p, y_i, g;\ x)$ – ElGamal asymmetric keys of the mixing authorities,
- $onion_k(m) = \big(m(y_1 y_2 \dots y_\lambda)^k g^k\big)$ — onion function,
- $transform_{i,k}((a,b)) = \big(b^{-x_i} a(y_{i+1} y_{i+2} \dots y_\lambda)^k, b \cdot g^k\big)$ — transform function.

A mix network protocol can be employed as a component of electronic voting if it can be guaranteed that none of the list elements was replaced or maliciously altered. This property called *robustness* is provided by additional checking. Randomized Partial Checking is a fairly simple and effective verifying technique which was introduced in [12]. The mix servers are obliged to reveal a random half of their input-output relations, with the assurance that no path of length greater than 2 can be uncovered. To achieve this property the servers are paired, and forced to uncover complementary halves of their transformations.
In more detail, RPC consists of the following steps:

(1) *(Before shuffling)* The mix servers publish commitments to their permutations.

(2) *(After shuffling)* The servers establish a fairly chosen value

$$r = r_1 \oplus r_2 \oplus \dots \oplus r_\lambda$$

— each server contributes its r_i using commitments, so that no party is able to determine r. Then a value $q = hash\big(r, content(BB)\big)$ is computed, and $q_i = hash(q, i)$ are derived. Values q_i determine transitions to be revealed in pair i. To prove validity of a selected transition of jth input server A_i publishes a value $validator(i, j)$ that may consist of $decommit\big(\Pi_i(j)\big), k_{ij}$, where k_{ij} is the randomization value used in the jth transformation.

3.2. Protocol steps

Actors and setup. The following actors participate in the protocol. Note that the voter and the device are treated as separated parties.

- Voter (V_i):
  - owns a terminal V^{pc}, and a device V_i^{dev} which contains:
    - pk_i, sk_i (or pk_{V_i}, sk_{V_i}) — a pair of asymmetric keys used only in the protocol to assure authenticated channel in the registration phase (the keys cannot be used on user's demand for regular signing),
    - pk_{EC} — EC's public key, pk_A — public key of the mixnet.
- Election Committee (EC):
  - owns pk_{EC}, sk_{EC}, a pair of asymmetric keys;
  - supplies blindly signed voting tokens to legitimate voters.
- Bulletin Board (BB):
  - BB provides an authenticated publicly available broadcast channel;
  - owns pk_{BB}, sk_{BB}, a pair of asymmetric keys.
- Mix servers $(A_1, A_2, ..., A_\lambda)$:
  - provide an anonymous channel;
  - each server owns a pair of asymmetric ElGamal keys, the public keys form a global public key pk_A of the mixnet; the key is used to create onions.

Notation:

- $sig_A(m)$ — A's signing transformation (using sk_A);
- $ver_A(m, s)$ — verification function of signature s under m using A's public key pk;
- $h(m)$ — hashing function;
- $A \xrightarrow{auth} B : m$ — denotes communication through an authenticated channel, which is realized by sending m, with a signature $sig_A(m)$.

Preparation:

1. EC generates the election's identifier E and the election's asymmetric pair of RSA keys $(sk_E, pk_E) = \big(d_E, (e_E, n_E)\big)$.
2. EC publishes the election's public key
$$EC \xrightarrow{auth}, BB : E, pk_E.$$
3. The device is provided with the parameters
$$V_i^{dev} \longleftarrow BB : E, pk_E, sig_{EC}(E, pk_E).$$

TABLE 1. Preparation.

BB	$auth$	V_i^{dev}	$auth$	EC
	$\longrightarrow$ E, pk_E $sig_{EC}(E, pk_E)$			E, pk_E, sk_E publishes: E, pk_E

Registration:

1. V_i^{dev}:
   (a) Chooses a random value r, and a part of the voter's anonymous identifier ID_i' ($r, ID_i' \in \mathbb{Z}_{n_E}^*$).
   (b) Blinds $b' = r^{e_E} \cdot ID_i' \mod n_E$, and stores (ID_i', r)
   $$V_i^{dev} \longrightarrow V_i^{pc} : h(b').$$
2. V_i^{pc}:
   (a) Chooses the second part of the voter's anonymous identifier $ID_i'' \in \mathbb{Z}_{n_E}^*$
   $$V_i^{pc} \longrightarrow V_i^{dev} : ID_i''.$$
3. V_i^{dev}:
   (a) Computes $b = r^{e_E} \cdot ID_i' \cdot ID_i'' \mod n_E$, and stores $ID_i = ID_i' \cdot ID_i'' \mod n_E$
   $$V_i^{dev} \overset{auth}{\longrightarrow} V_i^{pc} : b.$$
4. V_i^{pc}:
   (a) Verifies $h(b/ID_i'' \mod n) \overset{?}{=} h(b')$. If the value is correct b is passed on to the EC.
   (b) $V_i^{dev} \overset{auth}{\longrightarrow} EC$: b.
5. EC:
   (a) Checks the V_i identity and voting rights.
   (b) $EC \longrightarrow BB : \big(b, V_i, sig_{V_i}(b)\big)$.
   (c) Signs blindly: $s = b^{d_E} \mod n_E$.
   (d) $EC \longrightarrow V^{dev} : s$.
6. V_i^{dev}: Recovers token $t_i = s \cdot r^{-1} \mod n_e$. t is verified.

Voting:

1. V_i: Chooses vote v_i and sends it to the device.
2. V_i^{dev}:
   (a) Displays v_i, and asks for confirmation.

(b) Computes, and displays $onion(ID_i, t_i, v_i)$.
(c) $V_i^{dev} \overset{auth}{\longrightarrow} BB : onion(ID_i, t_i, v_i)$.

TABLE 2. Registration.

V_i^{dev}	$auth$	V_i^{pc}	$auth$	EC
ID'_i, r				
$b' = r^{e_E} \cdot ID'_i$				
	$\longrightarrow$			
	$h(b')$			
		ID''_i		
	$\longleftarrow$		$\longrightarrow$	
	ID''_i			
$b = b' \cdot ID''_i$				
$ID_i = ID'_i \cdot ID''_i$				
	$\longrightarrow$		$\longrightarrow$	
	b		b	
				$s = b^{d_E}$
	$\longleftarrow$		$\longleftarrow$	
	s		s	
$t_i = s \cdot r^{-1}$				

TABLE 3. Voting and publication of the results.

V_i^{dev}	$auth$	BB	$auth$	EC
v_i, ID_i, t_i				
	$\longrightarrow$			
	$onion(ID_i, t_i, v_i)$			
		...mixing...		
		publishes results		
			$\longleftarrow$	
			$sig_{sk_E}(E)$	

Verification.

1. Global verification involves Randomized Partial Checking of the mixing process as well as counting of the published votes by some open source program, and comparing the number of registered voters to the number of votes.

2. Individual verification (stage 1) :
   (a) Voter checks if the onion that contains his vote reached the Bulletin Board. The onion, or a few of its initial bytes are displayed by the device.
3. Individual verification (stage 2) :
   (a) V_i installs $sig_{sk_E}(E)$ in the device.
   (b) V_i^{dev} verifies sk_E (checks whether $ver_{pk_E}\big(sig_E(ID_i)\big)$ returns "true"). If the key is valid, ID_i is displayed.
4. V_i finds his/her pair (ID_i, v_i) on the list and verifies v_i.

4. Characteristics of the protocol

The proposed protocol is designed to be implemented on a tamper-resistant device with a built-in display, thanks to which it provides many attractive features (apart from eligibility, global and individual verifiability).

Firstly, it is difficult to organize vote selling on a massive scale (receipt--freeness), as the verifying information is not available before publication of the result. The device hides it using its black-box property till the end of elections, when it is displayed by the device. As a result, the voter obtains an evidence that enables delayed verification. The evidence is useless for automated vote selling, as no digital data can convince the buyer of its authenticity — the voter can send the verifying information or a photo of his/her device, but they can be falsified at this stage. Note that image recognition algorithms may be mislead, so the only way to convince a potential buyer is sending the device for manual verification. Nonetheless, such malicious behavior would be difficult to arrange and conceal, especially if the device is integrated with the ID card.

Secondly, the registration phase does not require a voting decision (*independent registration property*). It means that this phase can take place some reasonable time before the voting. As a result, electronic voting can be easily integrated with traditional voting — the Voting Committee can mark the e--voters on traditional voters' lists (see Estonian case [7]). This was not the case in Okamoto, Fujioka and Ohta (FOO) [2] which required commitment to vote in the registration phase. Easy integration with traditional voting was indicated to be crucial in possible implementation investigations [6].

In case of our protocol, the vote is attached to the token (*token-vote binding* property) – it can be verified, whether the vote was sent by the token's owner. This property is also achieved by the scheme presented in Okamoto, Fujioka and Ohta (FOO) [2], but it requires voting decision in the registration phase. Token-vote binding is provided by the properties of a robust mixnet — — content of an onion cannot be maliciously altered during mixing. Employing

mixnets also has another advantage — no authority is privileged in the sense that it can compute partial results of the election. This important property of a voting protocol is called *fairness*. Although Okamoto, Fujioka and Ohta (FOO) [2] and Juang, Lei (JL) [9] offer fairness, they require either resigning from independent registration or performing complicated multi-party protocols.

Finally, the trust put in voter's PC is minimized. The trust is actually moved to the device, which is not prone to cyber attacks. The device has a special pair of asymmetric keys, apart form the pair which realizes ordinary signatures, in order to perform the election protocol. As a consequence, it is not possible to simulate the protocol outside the device using it only as a signing black-box. Moreover, the choice made by the voter can be displayed by the device assuring that the vote chosen is the same as the vote sent. It was observed by Schneier and Shostack [18] that adding display to a smart card, in general, eliminates a class of terminal-to-card attacks. These attacks take advantage of the fact that a smart card is a totally limited device when it comes down to its input-output abilities. Consequently, it needs to rely completely on the terminal (voter's PC with a card reader) to pass data (PINs, "whom I vote for") from the user to the card properly and not to secretly change data sent by the card. It has to be underlined that although the PC cannot change the vote it is still able to learn the voter's choice. This is, however, true for all hardware based solutions with a device that has no direct input abilities, and may be eliminated by employing additional measures, e.g., code sheets [19].

Table 4. Comparison of blind-signatures-based protocols (* indirect verifiability).

	FOO	OKA	Radwin	RadwinNL	JL	New
Eligibility	Yes	Yes	Yes	Yes	Yes	Yes
Fairness	Yes	Yes	No	No	Yes	Yes
Indep. registration	No	Yes	Yes	Yes	No	Yes
Token-vote binding	Yes	Yes	No	No	Yes	Yes
Receipt freeness	No	Yes	No	Yes	No	Yes
Indiv. verifiability	Yes	Yes*	Yes	No	Yes	Yes

4.1. Communicational model

The crucial security features are preserved based on the fact that there exist some direct interactions between the voter and the device — the device sends to its owner the description of the vote or the vote's ID by displaying them. On the other hand, the voter is expected to directly "send" a confirmation of the chosen vote. In the minimum version it can be achieved by not removing the device from the reader within a given time after the vote has been displayed. We can

also imagine a device equipped with "ok" and "cancel" buttons, small numeric keyboard or even a fingerprint reader. The last option is especially tempting, as it would bind the device to its owner and enhance the security of the whole system.

The communication between V^{dev} and the election servers is realized via a proxy program (potentially untrusted) running on the voter's PC (V^{pc}) — it provides a network interface to the device. Similarly, some indirect communication takes place between the device and the voter — the terminal supplies keyboard and mouse for entering the vote. The role of the terminal may be played by voter's PC or by a publicly available terminal, e.g., a sophisticated ATM.

4.2. Kleptographic countermeasures

Authors of [22, 23] observed in the 90's that utilizing black-box devices may be disastrous for their users, as the designer of the device may embed malicious code that leaks secret information without being noticed. In other words, the device may behave according to its specification, but still may be a source of precious information for its designer. Such kleptographic attacks, in general, exploit any form of randomness that might be substituted with asymmetrically encrypted secrets. The public key is built into the device, while the corresponding private key is kept by the malicious designer. In the proposed protocol a potential backdoor for kleptography might have been the registration phase, where the card obtains a blindly signed token. If the device was able to independently determine the ID, and in consequence, the blinded identifier b, then a subliminal channel would become wide open. We proposed an interactive key generation protocol performed by the device and the terminal. It allows the terminal to affect the identifier without learning it. Moreover, the terminal has a guarantee that its part of the ID was used, as it can check validity of the blinded value. Since a valid token may only be obtained through interaction with the terminal, the token is also a certificate of the ID subliminal-channel-free randomness. A similar method can be used to assure the randomness of the hiding value r, or randomizing value k involved in creation of the onion. In this case a kleptographic attack could still be launched if the malicious designer would gain control over the PC (by infecting it with a computer virus), but such misbehavior is far easier to detect than regular kleptography.

Another kleptographic threat, which is not specific to the voting protocol, is the signing algorithm implemented in the device. The RSA scheme, as a representative of deterministic signing, seems to be appropriate in the context of kleptography. Nonetheless, it was shown in [22] that RSA's weakness is the key generation algorithm. The algorithm may be maliciously modified so that it can generate a public modulus $n = pq$ that contains n's factorization — the most significant bits of n are a public key encryption of prime p. This problem was

countered in [24] by introducing a verifiable generation of the RSA modulus. The scheme works under the assumption that all users of the devices share an universally acknowledged public exponent e, usually $e = 2^{16} + 1$.

5. Conclusions

This paper is an attempt to apply contemporary smart card technology to the challenging problem of elections over the Internet. On the one hand, there is a growing political pressure to introduce such solutions, on the other hand — many concerns and threats arise. One of the biggest problems is how the government can protect Internet voting from potential outsider attacks. Smart cards with a display seem to have many attractive features that can be utilized in this area of research and development and in security sensitive systems, in general.

REFERENCES

[1] ADIDA, B.—RIVEST, R.: *Scratch and vote: Self-contained paper-based cryptographic voting,* in: Proc. of the 5th ACM Workshop on Privacy in Electronic Society—WPES '06 (R. Dingledine, T. Yu, eds.), ACM New York, NY, USA, 2006, pp. 29–40.

[2] FUJIOKA, A.—OKAMOTO, T.—OHTA, K.: *A practical secret voting scheme for large scale elections,* in: Advances in Cryptology—AUSCRYPT '92 (J. Seberry et al., eds.), Lecture Notes in Comput. Sci., Vol. 718, Springer-Verlag, Berlin, 1993, pp. 244–251.

[3] PARK, CH.—ITOH, K.—KUROSAWA, K.: *Efficient anonymous channel and all/nothing election scheme,* in: Advances in Cryptology — EUROCRYPT '93 (T. Helleseth, ed.), Lecture Notes in Comput. Sci., Vol. 765, Springer-Verlag, Berlin, 1994, pp. 248–259.

[4] CHAUM, D.: *Blind signatures for untraceable payments,* in: Advances in Cryptology — —Crypto '82 (D. Chaum, R. L. Rivest, A. T. Sherman, eds.), Plenum Press, New York, 1983, pp. 199–203.

[5] CHAUM, D.: *Secret-ballot: True voter verifiable elections,* IEEE Security and Privacy **2** (2004), 38–47.

[6] CIVTF: *A report on the feasibility of internet voting,* `http://www.ss.ca.gov/executive/ivote/final-report.htm`, 2000.

[7] Estonian National Election Committee: *E-voting system—overview,* `http://www.vvk.ee/elektr/docs/Yldkirjeldus-eng.pdf`, 2006.

[8] JUELS, A.—CATALANO, D.—JAKOBSSON, M.: *Coercion-resistant electronic elections,* in: Proc. of the ACM Workshop on Privacy in the Electronic Society—WPES '05, ACM, New York, NY, USA, 2005, pp. 61–70.

[9] JUANG, W.—LEI, CH.: *A secure and practical electronic voting scheme for real world environment.* IEICE Trans. on Fundamentals of Electronics, Communications and Computer Sciences **E80-A** (1997), 64–71.

[10] KUTYŁOWSKI, M.—ZAGÓRSKI, F.: *Verifiable internet voting solving secure platform problem,* in: 2nd International Workshop on Security—IWSEC '07 (A. Miyaji et al., eds.), Lecture Notes in Comput. Sci., Vol. 4752, Springer-Verlag, Berlin, 2007, pp. 199–213.

[11] LEE, B.—KIM, K.: *Receipt-free electronic voting scheme with a tamper-resistant randomizer,* in: Information Security and Cryptology—ICISC '02 (P. J. Lee et al., eds.), Lecture Notes in Comput. Sci., Vol. 2587, Springer-Verlag, Berlin, 2003, pp. 389–406.

[12] JAKOBSSON, M.—JUELS, A.—RIVEST, R.: *Making mix nets robust for electronic voting by randomized partial checking,* in: Proc. of the 11th USENIX Security Symposium (D. Boneh, ed.), USENIX Association, Berkeley, CA, USA, 2002, pp. 339–353.

[13] NEFF, C. A.: *A verifiable secret shuffle and its application to e-voting,* in: Proc. of the 8th ACM conference on Computer and Communications Security—ACM CCS '01 (P. Samarati, ed.), ACM Press, New York, NY, USA, 2001, pp. 116–125.

[14] OKAMOTO, T.: *Receipt-free electronic voting scheme for large scale elections,* in: 5th International Workshop on Security Protocols (B. Christianson, ed.), Lecture Notes in Comput. Sci., Vol. 1361, Springer-Verlag, Berlin, 1998, pp. 25–35.

[15] RADWIN, M. J.: *An untraceable, universally verifiable voting scheme,* (abstracted by Prof. P. Klein) Seminar in Cryptology, December 12, 1995. `http://www.radvin.org/michael/project/voting.pdf.`

[16] RUBIN, A.: *Security considerations for remote electronic voting over the internet,* `http://avirubin.com/e-voting.security.pdf`, 2001.

[17] RYAN, P.—SCHNEIDER, S.: *Prûet à voter with re-encryption mixes,* in: Computer Security—ESORICS '06 (D. Gollmann et al., eds.) Lecture Notes in Comput. Sci., Vol. 4189, Springer-Verlag, Berlin, 2006, pp. 313–326.

[18] SCHNEIER, B.—SHOSTACK, A.: *Breaking up is hard to do: Modeling security threats for smart cards,* in: Proc. of the USENIX Workshop on Smartcard Technology on USENIX Workshop on Smartcard Technology—WOST'99, USENIX Association, Berkeley, CA, USA, 1999, p. 19.

[19] SureVote. *E-voting system—overview,* `http://www.vote.caltech.edu/wote01/pdfs/surevote.pdf`, 2001.

[20] TORRIERI, M.: *On card displays become reality, making cards more secure,* 2006.

[21] VAN DE GRAAF, J.: *Merging prêt à voter and punchscan,* `http://eprint.iacr.org/2007/269.pdf`, 2007.

[22] YOUNG, A.—YUNG, M.: *The dark side of black-box cryptography, or: Should we trust Capstone*? in: Advances in Cryptology—CRYPTO '96. (N. Koblitz, ed.), Lecture Notes in Comput. Sci., Vol. 1109, Springer-Verlag, Berlin, 1996, pp. 89–103.

[23] YOUNG, A.—YUNG, M.: *Kleptography: Using cryptography against cryptography,* in: Advances in Cryptology—EUROCRYPT 97 (W. Fumy, ed.), Lecture Notes in Comput. Sci., Vol. 1233, Springer-Verlag, Berlin, 1997, pp. 62–74.

[24] YOUNG, A.—YUNG, M.: *Yygen: A backdoor-resistant rsa key generator,* `http://cryptovirology.com/cryptovfiles/newbook/Chapter12.pdf`, 2005.

Received September 30, 2007

Faculty of Mathematics
and Computer Science
Adam Mickiewicz University
Umultowska 87
PL-61–614 Poznań
POLAND
E-mail: lukaszn@amu.edu.pl