Secret Ballot Elections in Computer Networks

Hannu Nurmi
Department of Political Science
University of Turku
SF-20500 Turku
Finland

Arto Salomaa
Academy of Finland and Department of Mathematics
University of Turku
SF-20500 Turku
Finland

Lila Santean
Academy of Finland and Department of Mathematics
University of Turku
SF-20500 Turku
Finland

Abstract

The paper presents a secret balloting system for elections carried out in a computer network. The system has some features not possessed by customary secret balloting systems and does not rely on trusted persons and group work to the same extent as customary systems. Our system uses protocols based on public-key cryptography.

Keywords : ballot secrecy, cryptographic protocol, public key.

All correspondence should be sent to Arto Salomaa, Department of Mathematics, University of Turku, SF-20500 Turku, Finland.
The work reported here has been supported by the Academy of Finland grants 1071178 (Nurmi) and 1071041 (Salomaa and Santean).

1 Introduction

The institution of secret ballot is often mentioned as one of the hallmarks of democratic electoral systems. The reason is obvious. Without ballot secrecy, the voters could be deterred from revealing their true opinions about the issues to be voted upon. Thus, the very rationale of voting, viz. giving voters the possibility to express their opinions without fearing that they would be punished for having them, would be undermined.

Customary secret balloting systems rely to a large extent on trusted persons and group work. The counting of votes is done by specifically elected officials. The basic method of securing ballot secrecy is to make sure that the ballots of individual voters are not counted individually but in aggregates. Typically all the ballots cast in a given election locale are counted simultaneously. This means that the link between a voter and his/her (hereafter her) vote is broken.

The fact that voting takes place in specifically designated areas and that all voters are identified before entering the balloting booth makes many forms of electoral fraud difficult. Thus, deliberate errors in vote counting presuppose the cooperation – voluntary or forced – of all persons working at a given voting locale. The more persons supervise the electoral procedure and the more variegated political views they represent, the less likely one is to encounter electoral fraud of this kind, is a plausible conjecture. Yet fraudulent elections are known to have been conducted.

When elections are conducted in computer networks, the secrecy issues are somewhat different. On the other hand, present cryptographic techniques open entirely new vistas for secret communication, and seemingly impossible tasks become possible. For instance, a voter can check whether her vote has been correctly counted and also recast her vote within a certain period without jeopardizing ballot secrecy. Before beginning a discussion concerning the requirements for balloting systems in computer networks, we would like to mention an analogous situation: banking and cashless payment systems. Most existing payment systems are completely unacceptable, since banks and even computer manufacturers can easily observe who pays what amount to whom and when. Payment systems guaranteeing security against fraud, and also enabling unobservability of clients, are necessary [3]. Measures of jurisdiction alone are insufficient, since infringements can hardly be discovered. For instance, the following requirements are connected with the unobservability of clients. Each payment should be secret. Unless the client wishes otherwise, each of her actions should be unlinkable to actions that have taken place earlier. The client should be able to do business anonymously; the bank and the client's business partners should not be able to find out her identity. These and similar requirements should be fulfilled without a trusted referee.

The purpose of this paper is to present a protocol to satisfy the following seven requirements, discussed in more detail in [6] (see also [5]). The requirements 5–7 are usually not satisfied in contemporary balloting systems. There has been a vivid public discussion in Finland recently concerning the issue of to what extent these requirements increase the motivation of an average voter to

vote. We would like to emphasize that there are many approaches in cryptography dealing with related matters. They include an early contribution [4], as well as recent ideas of G.Simmons. The work of Benaloh, [1], about balloting systems has aims somewhat different from ours. Altogether a comprehensive bibliography would have to include numerous contributions to cryptographic protocols.

We now list our requirements.

By definition, any secret balloting system must satisfy the condition:

(i). *No one but the voter knows which voting strategy the voter adopts.*

But a satisfactory balloting system must also have at least two additional properties:

(ii). *Only legitimate voters may cast a valid vote.*

(iii). *Each legitimate voter may cast only one valid vote.*

The conditions 1–3 are obvious desiderata that, no matter which other plausible properties a system may have, cannot be dispensed with. Most contemporary secret balloting systems strive to satisfy these conditions. Thus, any system which fails to meet one of them is vulnerable to the criticism that none of the existing systems fails on these conditions.

In addition to the above three, the following conditions would obviously be desirable:

(iv). *The voting may take place in a computer network, i.e. each voter may use a terminal in casting her ballot.*

(v). *Each voter may check that her vote has been counted.*

(vi). *Each legitimate voter can change her mind (i.e. cancel her vote for a given candidate and cast it in favour of another candidate) within a given period of time.*

(vii). *Shoud the voter find out that her vote is misplaced, she can point this out to the ballot counting system without jeopardizing ballot secrecy.*

2 The protocol

We refer the reader to [7] for all unexplained notions in cryptography. In particular, we shall use without further explanations terminology customary in **public-key cryptography**: **one-way function, public encryption key, secret decryption key, RSA, cryptographic hashing**. As customary in public-key cryptography, the possession of the secret decryption key is equivalent to knowledge of a **secret trapdoor** to the one-way function.

In general, there are many security related questions in telecommunication. They include:

- Who is out there?

- Is she allowed to get this information?

- Will the information I send reach the right person?

- Has the information been seen by somebody else?

- Has the information been changed in transit?

- Can a sender deny having sent particular information?

- Can a receiver pretend not to have received particular information?

What kind of precautions and safety measures are called for depends largely on the communication environment. In our protocol below, no further safety measures are needed in most steps. Whenever needed, **encryption** by the receiver's public key and/or **signature** by the sender's secret decryption key may be applied. (For details, see [7]). To avoid unnecessary complications in the exposition, we shall not dwell on this issue further.

In what follows we denote by **A** the **agency** supervising the electoral procedure. We try to minimize **A**'s possibilities for fraudulent behaviour but cannot entirely exclude them. Our main technical tool is the **ANDOS** (**A**ll or **N**othing **D**isclosure **O**f **S**ecrets) protocol, explained in the Appendix. This is a protocol for "secret selling of secrets" in the following sense. **S**, a seller of secrets, has listed a number of questions and offers to sell the answers to any of them . A buyer **B** wants to buy a secret but does not want to disclose which one. The protocol guarantees that **B** gets the secret she wants and nothing else, whereas **S** does not know which secret **B** got.

We are now ready to describe our protocol.

Step 1. The agency **A** publishes a list of all legitimate voters.

Step 2. Within a specified deadline, everybody intending to vote reports her intention to **A**.

Step 3. **A** publishes a list of voters participating in the election.

Steps 1–3 are preliminary ones. The main purpose is to find out and publicize the total number n of active voters. Although some of them might actually not participate, the possibilities of **A** for adding fraudulent votes to the final count are considerably reduced. On the other hand, even in contemporary balloting systems, at least in smaller circles, it is known who exercized her right to vote. Also, it is easy to point out any possible error in Step 3.

Step 4. **A** chooses n (= number of voters) identification tags and conducts the ANDOS protocol for the voters. The identification tags are large random primes. They are listed using the numbers $1, \ldots, n$.

More specifically, if a voter **B** has chosen the number i, then she gets the ith prime p_i from **A**'s list but **A** does not know the interconnection

between i and **B**. It is possible in the ANDOS protocol that two voters choose the same number i and, thus, get the same identification tag p_i. The protocol is also quite complicated if n is large. Both of these difficulties will be discussed in the next section.

Step 5. A voter **B** with the identification tag p_i chooses a cryptographic hash function $h_B(x, y)$ of two variables and sends **A** the pair $(p_i, h_B(p_i, v_B))$, where v_B is her vote (name of candidate or, more generally, her voting strategy) expressed numerically. (We assume that the hash functions map pairs of integers into integers.)

Step 6. **A** acknowledges receiving the information by publishing the value $h_B(p_i, v_B)$.

Step 7. **B** sends **A** the pair (p_i, h_B^{-1}). Assuming that y can always be computed, given $h_B(x, y)$, x and h_B^{-1}, **A** now knows the interconnection between p_i and v_B (but **not** between **B** and v_B).

A simplified version of Steps 5–7 would be that **B** sends **A** directly the pair (p_i, v_B). However, it would then be impossible for **B** both to check that her vote has been properly counted and to recast her vote at a later stage. This follows because if **A** publishes the tag p_i in the list of those who adopted the strategy v_B, then **B** surely knows that her vote has been properly counted but anybody can later masquerade herself as the one having the tag p_i and, thus, recast **B**'s vote. On the other hand, if **A** only publishes the number of the voters who adopted a specific strategy, then the voters cannot check anything and **A** can publish any election result whatsoever.

Step 6 gives **B** the possibility to check that **A** received her vote before **A** publishes the result of the election. Moreover, if **A** does not publish at all **B**'s vote in Step 8, or else publishes it in a wrong list, **B** can immediately prove that **A**'s behaviour is fraudulent.

The use of the hash function adds to the security. Even if they know h_B^{-1}, the other voters cannot necessarily compute v_B from the published $h_B(p_i, v_B)$. The knowledge of many v_B's before the end of the voting period would, of course, open new vistas for strategic voting. Typically, the hash function $h_B(x, y)$ is a one-way function applied to the product xy. Then, one has to be able to factorize, even if one knows the inverse of the one-way function, provided the numerical encoding of v_B is chosen in a suitable way. One possible way to do this is the following. Assume that every voting strategy (name of the candidate, etc.) can be expressed in five bits. Recall that p_i is a large prime. Choose also v_B to be a large prime q_i whose 10th, 20th, 30th, 40th and 50th bits constitute the desired voting strategy. This method of choice has been agreed upon by all participants. It does not slow down the generation of q_i too much: about 3% of all primes satisfy the condition. If one is able to factor $p_i q_i$, one is able to break RSA.

Step 8. When the deadline for casting ballots is over, **A** announces the outcome of the election by publishing, for each voting strategy v, the list of all numbers $h_B(p_i, v_B)$ such that $v_B = v$.

Step 9. If a voter **B** observes that her vote is not properly allocated, she protests by sending **A** the triple $(p_i, h_B(p_i, v_B), h_B^{-1})$.

In view of Step 6, a protest also constitutes proof of the fact that voter **B** is right and, therefore, **A** has to correct the result accordingly.

After observing the actions of the elected people for some time, some of the voters might want to change their minds and recast their ballots. Such a procedure is described in the final step. A simpler variant (Step 10) can be used if the recasting of the ballot is done only once. A more involved variant (Step 10') is called for if several possibilities of recasting are allowed.

Step 10. The voters **B** wanting to recast their ballot send **A** the triple $(p_i, h_B(p_i, v_B), v'_B)$, where v'_B is the new voting strategy. (In fact, $h_B(p_i, v_B)$ is sent only to make **A**'s job easier.) When the deadline for recasting is over, **A** publishes the modified election result, where the numbers $h_B(p_i, v_B)$ have been reallocated in the list. The voters can also now check that their new votes have been properly counted.

Step 10'. As Step 10, but now **B** sends **A** the pair $(p_i, h'_B(p_i, v'_B))$, where h'_B is a new hash function chosen by **B**. **A** acknowledges receiving the message by publishing the value $h'_B(p_i, v'_B)$, after which **B** sends **A** the pair $(p_i, (h_B{}')^{-1})$. **A** now knows the interconnection between p_i and v'_B. In the new election result, $h_B(p_i, v_B)$ is removed from the list for $v = v_B$, and $h'_B(p_i, v'_B)$ is added to the list for $v' = v'_B$. Voters **B** can protest as before.

In comparison with Step 10, Step 10' has the additional advantage that the voters other than **B** only observe that something vanished from the list for v but do not know that it went to the list for v'.

3 Further discussion

Our protocol was already discussed from many points of view in Section 2. Some additional remarks are in order.

The ANDOS protocol may assign the same identification tag to different voters. There are two ways to overcome this difficulty.

(i) **A** ensures that the number of identification tags is so much larger than the number of voters that the probability for such a coincidence is negligible. In spite of the birthday paradox, the number of tags still remains within reasonable bounds.

(ii) When receiving an identification tag p_i for the second time, **A** publishes the pair $(p'_i, h_B(p_i, v_B))$, where p'_i is a tag that was not "for sale" in the

ANDOS protocol. (**A** has a few such surplus tags available.) **B** sees from the second component that she is in question and sends **A** the pair $(p'_i, h_B(p'_i, v_B))$. Other voters cannot do this, since they do not know the hash function h_B. The process continues as before: **A** publishes $h_B(p'_i, v_B)$, etc. After getting h_B^{-1}, **A** is assured that the correct voter reacted.

Thus, we may assume that different voters have been assigned different identification tags. We may also assume that the important numbers $h_B(p_i, v_B)$ are all different, i.e., that the equation $h_B(p_i, v_B) = h_C(p_j, v_C)$ does not hold for any other voter C with the identification tag p_j. For instance, if the primes involved have 100 digits (as recommended in RSA), then the probability of such a coincidence is negligible for any realistic population of voters.

We have shown that our protocol satisfies the requirements 1–7. Moreover, an observed fraud or error leads to the correction of the result, at least in normal cases. Cheating by the agency **A** may remain undiscovered if some of the voters who reported in Step 2 do not actually vote. Then **A** can allocate their "votes" arbitrarily.

There are two obvious drawbacks. The incentives for selling and buying of votes become considerably stronger as the buyer can be sure that the seller also delivers the goods, i.e., votes as promised. However, this is a necessary characteristic of any system satisfying our requirements.

Another drawback is the complexity of the ANDOS protocol. The ANDOS protocol can be avoided, as done in [6], if the agency gives the voters only a common password and the voters choose the identification tags themselves. Then, however, only the total count of votes can be used to prevent the voters from voting several times.

On the other hand, excessively large populations can be avoided by dividing them into smaller parts. Then the agency will know the distribution of votes in each part. This is not a very serious drawback: the distribution of votes within voting districts is also known in contemporary elections. It might also be possible to "nest" ANDOS protocols. However, we have not been able to find any simple way of doing that.

4 Conclusions

We have presented a secret balloting system which satisfies, in addition to the customary requirements, also some new requirements without jeopardizing secrecy. One of the drawbacks is inherent in the requirements. The other, the computational complexity, can be removed by making some of the new requirements less demanding. We hope to return to these issues in our forthcoming work.

5 Appendix

Some rather complicated ANDOS protocols, based on zero knowledge proofs, are hinted at in [2]. The following protocol, first discussed in [8], is based on the fact that there are several buyers.

Assume that $s_1, \ldots, s_k$ are secrets possessed by **S**, each of them containing n bits. For each s_j, **S** has publicized what the secret is about.

We consider first the case of two buyers **B** and **C** who want to buy secrets s_j and $s_{j'}$, respectively. The idea is that the buyers have individual one-way functions and each of them operates on numbers provided by the other.

Step 1. **S** tells **B** and **C** individually the one-way functions f and g but keeps the inverses to herself.

Step 2. **B** tells **C** (respectively **C** tells **B**) k random n-bit numbers $x_1, \ldots, x_k$ (respectively $x'_1, \ldots, x'_k$).

For an injection f mapping n-bit numbers into n-bit numbers and an n-bit number x, we say that an index $i, 1 \leq i \leq n$, is a *fixed bit index* (FBI) with respect to the pair (x, f) if the ith bit in x equals the ith bit in $f(x)$. Clearly, i is FBI with respect to (x, f) iff i is FBI with respect to $(f(x), f^{-1})$. If f has reasonably random behaviour (like the customarily considered encryption functions) then, for a random x, roughly $n/2$ indices are FBI's with respect to (x, f).

Step 3. **B** tells **C** (respectively **C** tells **B**) the set FBI_B of FBI's with respect to (x'_j, f) (respectively the set FBI_C of FBI's with respect to $(x_{j'}, g)$).

Step 4. **B** (respectively **C**) tells **S** the numbers $y_1, \ldots, y_k$ (respectively $y'_1, \ldots, y'_k$), where y_i results from x_i by replacing every bit whose index is not in FBI_C with its complement (respectively y'_i results from x'_i by replacing every bit whose index is not in FBI_B with its complement).

Step 5. **S** tells to **B** (respectively **C**) the numbers

$$s_i \oplus f^{-1}(y'_i) (\text{respectively } s_i \oplus g^{-1}(y_i)), i = 1, \ldots, k.$$

Step 6. **B** (respectively **C**) is able to compute s_j (respectively s'_j) since she knows $x'_j = f^{-1}(y'_j)$ (respectively $x_{j'} = g^{-1}(y_{j'})$).

B and **C** learn the secret they want. **S** does not learn anything about the choices, and neither do **B** and **C** learn more than one secret or the choice of the other. A coalition between **B** and **C** amounts to **B** and **C** learning all secrets. A coalition between **S** and one of the buyers reveals which secret the other buyer wants.

Let us consider a simple example. **RSA** is used to construct the one-way functions needed.

Example. Choose $k = 8, n = 12$. Assume that **S** has the following eight 12-bit secrets for sale: $s_1 = 1990$, $s_2 = 471$, $s_3 = 3860$, $s_4 = 1487$, $s_5 = 2235$, $s_6 = 3751$, $s_7 = 2546$, $s_8 = 4043$.

Step 1. **S** tells **B** (respectively **C**) the function f (respectively g) based on $n_1 = 7387$ (respectively $n_2 = 2747$) which is the product of the primes $p_1 = 83, q_1 = 89$ (respectively $p_2 = 67, q_2 = 41$). The encryption and decryption moduli are $d_1 = 777, e_1 = 5145$ (respectively $d_2 = 2261, e_2 = 1421$).

Step 2. **B** tells **C** eight 12-bit numbers $x_i, 1 \leq i \leq 8 : x_1 = 743$, $x_2 = 1988$, $x_3 = 4001$, $x_4 = 2942$, $x_5 = 3421$, $x_6 = 2210$, $x_7 = 2306$, $x_8 = 912$.

C tells **B** eight 12-bit numbers $x'_i, 1 \leq i \leq 8 : x'_1 = 1708$, $x'_2 = 711$, $x'_3 = 1969$, $x'_4 = 3112$, $x'_5 = 4014$, $x'_6 = 2308$, $x'_7 = 2212$, $x'_8 = 222$.

Step 3. **B** wants to buy the secret s_7. Therefore she computes

$$f(x'_7) = {x'_7}^{e_1} (\text{mod } n_1) = 2212^{5145} (\text{mod } 7387) = 5928.$$

Comparing the binary representations of x'_7 and $f(x'_7)$,

$$\begin{aligned} 2212 &= 0100010100100 \\ 5928 &= 1011100101000 \end{aligned}$$

B tells **C** the set $\text{FBI}_B = \{0, 1, 4, 5, 6\}$ of FBI's with respect to (x'_7, f).

C wants to buy the secret s_2. After the computations, **C** tells **B** the set $\text{FBI}_C = \{0, 1, 2, 6, 9, 10\}$ of FBI's with respect to (x_2, g).

Step 4. **B** tells **S** the numbers $y_i, 1 \leq i \leq 8$, where y_i results from x_i by replacing every bit whose index is not in the set $\{0, 1, 2, 6, 9, 10\}$ with its complement, for instance:

$$y_2 = 011001111100 = 1660.$$

C tells **S** the numbers $y'_i, 1 \leq i \leq 8$, where y'_i results from x'_i by replacing every bit whose index is not in the set $\{0, 1, 4, 5, 6\}$ with its complement, for instance:

$$y'_7 = 1011100101000 = 5928.$$

Step 5. **S** tells **B** the numbers $s_i \oplus f^{-1}(y'_i), 1 \leq i \leq 8$ (recall that $f^{-1}(y') = y'^{d_1} (\text{mod } n_1) = y'^{777} (\text{mod } 7387)$), for example:

$$\begin{array}{rcccl} s_7 & = & 2546 & = & 0100111110010 \\ f^{-1}(y'_7) & = & 2212 & = & 0100010100100 \\ \hline s_7 \oplus f^{-1}(y'_7) & = & & & 0000101010110 \quad = \quad \mathbf{342} \end{array}$$

S tells **C** the numbers $s_i \oplus g^{-1}(y_i), 1 \leq i \leq 8$ ($g^{-1}(y) = y^{d_2} (\text{mod } n_2) = y^{2261} (\text{mod } 2747)$), for example:

$$\begin{array}{rcccl} s_2 & = & 471 & = & 000111010111 \\ g^{-1}(y_2) & = & 1988 & = & 011111000100 \\ \hline s_2 \oplus g^{-1}(y_2) & = & & & 011000010011 \quad = \quad \mathbf{1555} \end{array}$$

Step 6. **B** learns the secret s_7 by computing the bitwise addition of x'_7 and the 7th number received from **S**, that is:

$$\begin{array}{rcrcl} x'_7 & = & 2212 & = & 100010100100 \\ & & 342 & = & 000101010110 \\ & & & & \overline{100111110010} \quad = \quad \mathbf{2546}. \end{array}$$

As **C** wants to buy the secret s_2 she computes the bitwise addition between x_2 and the 2nd number received from **S**, that is:

$$\begin{array}{rcrcl} x_2 & = & 1988 & = & 11111000100 \\ & & 1555 & = & 11000010011 \\ & & & & \overline{00111010111} \quad = \quad \mathbf{471}. \end{array}$$

We have observed that in case of many buyers, the main difficulty is due to coalitions. However, if there are at least three buyers, it seems that one honest buyer is enough to make the cheating of the other buyers impossible. So no honest majority is needed. Let us see how this works.

We assume that there are three buyers **A**, **B**, **C** and describe the protocol from **A**'s point of view. **A** wants the secret s_j.

Step 1. **S** tells **A** two one-way functions f_A^B and f_A^C.

Step 2. **B** (respectively **C**) tells **A** k random n-bit numbers $x_1^{BA}, \ldots, x_k^{BA}$ (respectively $x_1^{CA}, \ldots, x_k^{CA}$).

Step 3. **A** tells **B** (respectively **C**) the set FBI_A^B (respectively FBI_A^C) of FBI's with respect to the pair (x_j^{BA}, f_A^B) (respectively the pair (x_j^{CA}, f_A^C)).

Step 4. **B** (respectively **C**) tells **S** the numbers y_i^{BA} obtained from x_i^{BA} (respectively y_i^{CA} obtained from x_i^{CA}), $i = 1, \ldots, k$, by replacing every bit whose index is not in FBI_A^B (respectively FBI_A^C) with its complement.

Step 5. **S** tells **A** the numbers

$$s_i \oplus (f_A^B)^{-1}(y_i^{BA}) \oplus (f_A^C)^{-1}(y_i^{CA}), i = 1, \ldots, k.$$

Step 6. **A** is able to compute s_j since she knows $x_j^{BA} = (f_A^B)^{-1}(y_i^{BA})$ and $x_j^{CA} = (f_A^C)^{-1}(y_i^{CA})$.

Analogous parts should be stated for **B** and **C** to complete the protocol. Thus, **S** gives both of them two one-way functions, both of them receive numbers from the other two buyers, etc. The protocol works in exactly the same way for $t > 3$ buyers. Each of the buyers gets $t - 1$ one-way functions from the seller, as well as sets of numbers from all of her fellow buyers.

It is clear that each of the buyers gets the secret she wants. It is also clear that if all buyers are in coalition, they learn all the secrets. However, no coalition of $t - 1$ (or less) dishonest buyers can gain much because every bit in the sequences sent to them by **S** depends on a bit provided by the honest buyer.

References

[1] J.D.C.Benaloh. Verifiable secret-ballot elections. Yale University, Computer Science Department, Technical Report 561(1987).

[2] G.Brassard, C.Crepeau and J.-M. Robert. All-or-nothing disclosure of secrets. *Springer Lecture Notes in Computer Science* 263(1987), 234–38.

[3] H.Burk and A.Pfitzmann. Digital payment systems enabling security and unobservability. *Computers and Security* 9(1989), 399–416.

[4] D.Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Comm. ACM* 24(1981), 84-88.

[5] H.Nurmi and A.Salomaa. A cryptographic approach to the secret ballot. *Behavioral Science* 36(1991), 34-40.

[6] H.Nurmi and A.Salomaa. Secret ballot elections and public-key cryptosystems. Submitted for publication.

[7] A.Salomaa. *Public-Key Cryptography*. Springer-Verlag Berlin-Heidelberg-New York-Tokyo (1990).

[8] A.Salomaa and L.Santean. Secret selling of secrets with many buyers. *EATCS Bulletin* 42(1990), 178–186.