A LAN Voting Protocol

Vesna Hassler *
Reinhard Posch

Abstract. This paper presents a set of network protocols which provide for a voting service. The service is currently being implemented in the client-server architecture. The cryptographic technique combined with the basic voting protocol from Salomaa [19] is the method based on the multiple key ciphers introduced by Boyd [2]. Special protocols for registration of voters and issuing voting slips are developed. This ensures that the anonymity of a voting strategy is cryptographically secure and, on the other hand, that only legitimate voters may obtain a valid voting slip.

Keywords: cryptography, secure applications, voting protocols

INTRODUCTION

Cryptographic mechanisms combined with networking technology introduce a tremendous range of possibilities into the field of whiteboarding and collaborative work. Internetworking spans distances and provides for a user-friendly and supportive environment, while cryptography imposes rules for the game of communication and prevents breaking of them.

An ideal voting protocol has at least the following characteristics [18]:

1. Only legitimate voters can vote.
2. Legitimate voters can vote only once.
3. No one can determine for whom anyone voted.
4. No one can change anyone else's vote without being discovered.
5. All voters can make sure that their vote has been counted.
6. Everyone knows who voted and who did not.

Until now many voting schemes have been proposed which satisfy different subsets of the ideal voting protocol's properties. Chaum [7] introduces a concept of untraceable mail addresses and proposes a voting protocol based on it. All voting protocols developed later, as well as the protocol described in this paper, assume the existence of an

*Institut für Angewandte Informationsverabeitung und Kommunikationstechnologie, TU-Graz, Klosterwiesgasse 32/I, A-8010 Graz, Austria, vhassler@iaik.tu-graz.ac.at anonymous channel such as the one in [7] which provides unconditional security against tracing the senders of messages. Park et al. [15] also propose an anonymous channel which circumvents the problem of ciphertext length expansion encountered in [7]. They also propose a new election scheme which, unlike the scheme in [7], satisfies the fairness criterion.

The voting protocol developed by Benaloh [1] allows only "yes"/"no" votes and is an extension using a composite (k, n) threshold scheme of the voting protocol in [9]. The identity of the voter is hidden as long as there are fewer than k conspiring sub-governments. In [2] Boyd proposes a voting protocol applying multiple key ciphers, with the trusted voting authority. The improved version of this protocol is given by Boyd in [3], requiring no trusted authority. Fujioka et al. proposes in [11] an excellent voting scheme for large scale elections in which the bit-commitment scheme, the digital signature scheme and the blind signature scheme are combined.

The protocol in [19, 14] by Salomaa et al. does not only satisfy the last of the ideal voting protocol's requirements. It has two additional properties:

1. Each legitimate voter can change his/her mind and recast a vote within a given period of time.

2. If a voter finds out that his/her vote has been miscounted, he/she can identify and correct the problem without jeopardizing the ballot secrecy.

For an anonymous issuing of voting slips and thus the preservation of the voter's anonymity, the protocol in [19] requires a method for the *secret selling of secrets.* The voting agency sells a valid voting slip to a legitimate voter in an oblivious way, so that the agency does not know which slip the voter actually received. This can be achieved by applying an *All-or-Nothing Disclosure of Secrets* protocol [4, 20, 16]. This type of protocol belongs to the group of *oblivious transfer* protocols.

Recently a new class of digital signatures called *oblivious signatures* have been proposed by Chen in [8]. Oblivious signature of n messages provides a way for a recipient (e.g. a voter) to get only one of n messages signed (e.g. voting slips), while the signer (e.g. a voting agency) cannot find out which message he/she signed.

In this work we propose a computationally simpler, and faster solution which is not perfect, but works well under the following two assumptions:

- All (or a high percentage of) registered voters apply for a voting slip.

- No authenticated (registered) voter tries to prevent voting.

If both assumptions are satisfied, our protocol has the same properties as the voting protocol in [19].

We combine the protocol of Salomaa [19] with the Boyd's multiple key ciphers concept [2] which is the generalization of the RSA algorithm [17], i.e. of its multiplicative property. Our security infrastructure assumes widely accepted and standardized security mechanisms from the area of symmetric and public-key cryptography. We also develop a protocol for oblivious transfer of voting slips to legitimate voters, as well as a voters registration protocol. The combination of the cryptographic techniques mentioned and the developed protocols enable us to satisfy the requirements of democratic voting, i.e. that only legitimate voters may vote and that nobody else but the voter him/herself knows his/her voting strategy.

1 VOTING

There are many and various types of voting systems. Let us describe the ones we selected for implementation.

With a voting system, the first question to be answered is: Who are the voters? The voters may all be citizens of a country as well as a small group of authorized persons. This means, depending on the circumstances and voting topic, that either everybody may access the service and receive a valid voting slip, or that only the person being able to prove his/her identity and having the proper voting rights may access the service. For example, authorization information can be citizenship or membership of some committee.

What are the rules to be observed during the voting phase? Firstly, only valid voting slips, one per voter, returned by some predetermined deadline may be taken into consideration. Secondly, based on information in the voting slip, it must be impossible for the ballot to draw any conclusions about the identity of the voter, but possible to ascertain his/her legitimacy and the number of times he/she has cast a vote. Thirdly, the voter must be able to check if his/her vote has been properly counted and to object to the ballot system if not. Fourthly, the voter can change his/her voting strategy before the ballot deadline, although the vote has already been cast.

Our voting system is to be implemented in the client-server technology. Clients are voters who establish a (network) connection with the voting server (i.e. voting agency). If a group of voters is so large that the generated traffic is too heavy or the number of requests too big, it is possible to establish a hierarchy of servers. A slave server may be responsible for a LAN and, at the higher level, a master server may collect results from all slaves. We do not address this problem here and assume that there is only one voting server and that the voters contact it directly.

2 THE BALLOTING PROTOCOL

The voting protocol we employ has been proposed by Salomaa [19]. Salomaa considers two cases, one with two voting agencies and the second with only one voting agency. In the former case with two agencies, the first one issues voting slips to legitimate voters, and the second one conducts the ballot casting procedure. It is assumed that they do not cooperate, otherwise the secrecy of the voting strategy cannot be guaranteed. The case with one agency requires the implementation of a special procedure called *secret selling of secrets* [20] which enables the issuing of voting slips to legitimate users only, without revealing the connection between the voter identity and the slip identification number.

We choose the model with one voting agency and employ a special *slips issuing protocol* which requires the slips issuing phase and the balloting phase to be performed over an anonymous communication channel, in order to provide for ballot secrecy. The protocol is described in one of the next sections.

Now we remind the reader of Salomaa's voting protocol [19]. Let V be a voter, VA a voting agency, $h_V(p_i, v_V)$ a cryptographic hash function known only to V, p_i the V's voting slip and v_V a voting strategy chosen by V. Let us also assume that y can always be computed from x, $h_V(x,y)$ and h_V^{-1}. The basic balloting protocol is then the following [19]:

Step 1: V sends the pair $(p_i, h_V(p_i, v_V))$ to VA.

Step 2: VA acknowledges the receipt by publishing the value $h_V(p_i, v_V)$.

Step 3: V sends the pair (p_i, h_V^{-1}) to VA. VA can now compute v_V and knows the interconnection between p_i and v_V, but does not know the interconnection between V and v_V.

Step 4: After the deadline for casting ballots, VA publishes the outcome of the election, i.e. the list of all numbers $h_V(p_i, v_V)$ such that $v_V = v$, for each voting strategy v.

Step 5: If V observers that his/her vote is allocated wrongly, he/she protests by sending VA the triple $(p_i, h_V(p_i, v_V), h_V^{-1})$

For discussion of the given voting protocol see [14, 19]. The published hash values in *Step 2* must also be signed by the VA in order to ensure proof of origin.

In our implementation of the protocol we use the multiple key cipher concept [2], i.e. the multiplicative property of the RSA public key scheme [17]. This means, we do not use the hash function h_V like Salomaa, but replace it with another function (modulo exponentiation). It enables us to take advantage of the difficulty of the factoring problem for the satisfaction of the voting condition about the anonymity of voting strategies, which is explained later.

The multiplicative property of RSA is the following: Let modulus m be chosen to be the product of two large RSA primes. A number of keys $k_1, k_2, \cdots, k_n$ are then chosen to satisfy the property

$$k_1 k_2 \cdots k_n \equiv 1 \bmod \phi(m), \tag{1}$$

so that in this case the following also holds (see [2])

$$E(E(\cdots E(M, k_1), k_2) \cdots), k_n) \equiv M^{k_1 k_2 \cdots k_n} \bmod m \equiv M. \tag{2}$$

After receiving the voting slip p_i in the slips issuing protocol (described later in the paper), every voter is supposed to have chosen an RSA modulus n_V. Only legitimate voters may choose a modulus, i.e. obtain a voting slip. Before the start of the actual voting protocol all the moduli are known to the VA, but not the identity of the voter who has chosen the particular modulus.

The voter then chooses two keys k_V and t_V so that (3) holds.

$$p_i k_V t_V \equiv 1 \bmod \phi(n_V) \tag{3}$$

The voter then tells (p_i, n_V) the voting agency VA, but keeps the factoring of n_V, k_V and t_V for him/herself. Prior to balloting phase, the VA publishes an assignment list in which each voting strategy is assigned a (different) prime v. The voting protocol is then as follows:

Step 1: V sends the pair $(p_i, v_V^{t_V} \bmod n_V))$ to VA.

Step 2: VA acknowledges the receipt by publishing the value $v_V^{t_V} \bmod n_V$.

Step 3: V sends the pair (p_i, k_V) to VA. VA can now compute $v_V = (v_V^{t_V} \bmod n_V)^{p_i k_V} \bmod n_V$ (see (3)). Now VA knows the interconnection between (p_i, n_V) and v_V. (We shall explain why VA does not know the interconnection between V and v_V, i.e. between V and (p_i, n_V) at a later stage in this paper.)

Step 4: After the deadline for casting ballots, VA publishes the outcome of the election, i.e. the list of all numbers $v_V^{t_V} \bmod n_V$ such that $v_V = v$, for each voting strategy v.

Step 5: If V observes that his/her vote is allocated wrongly, he/she protests by sending VA the triple $(p_i, v_V^{t_V} \bmod n_V, k_V)$.

Voting strategies v are chosen to be large primes. The RSA modulus n_V must satisfy the additional property $n_V > v$ in order to reconstruct the voting strategy v. In the issuing slips phase, which we describe later, the VA must check if any voting strategy v is a factor of any modulus n_V and reject such a n_V.

Only the voter who knows the factoring of n_V can send a valid vote. Let us assume that two voters, V_1 and V_2 work together against the voting agency, i.e. they try to use the special factoring to prove that the VA is cheating. They choose the same modulus n_V, and k_1, k_2, t_1, t_2 so that with p_1 and p_2 (3) holds. If they succeed in finding such k_2 so that $(v_1^{t_1} \bmod n_V)^{p_1 k_2} \bmod n_V = v'_V$, then V_1 can protest to VA by sending the triple $(p_1, v_1^{t_1} \bmod n_V, k_2)$, i.e. V_1 can claim to have chosen the voting strategy v'_V. But, the problem of finding such k_2 is as hard as finding the discrete logarithm.

Salomaa [19] also explains how the voter can recast the ballot. Under our terms, this additional step is:

Step 6: If V wants to recast the ballot, he/she chooses another pair of integers k'_V, t'_V to satisfy (3) and sends VA the pair $(p_i, (v'_V)^{t'_V} \bmod n_V)$. (Continue with *Step 2-5*.)

The modulus multiplication in *Step 3* and *Step 6* is a very time-consuming task, so that the special hardware is needed to achieve the acceptable speed (see e.g. [13]).

3 VOTING SERVICE

Our voting service consists of five phases:

1. Setting-Up the Voting Session
2. Announcement
3. Registration
4. Issuing Voting Slips
5. Balloting

$$\boxed{\begin{array}{lll} VI & \longrightarrow VA: & VI, VA, \textit{Session\_Request}, \text{Announcement}, \\ & & D_{VI}(\text{h(Announcement)} \| \text{Time\_Stamp}) \\ VA & \longrightarrow VI: & VA, VI, \textit{Session\_Acknowl}, \\ & & D_{VA}(\text{h(Announcement)} \| \text{Time\_Stamp+1}) \end{array}}$$

Figure 1: The voting initiator VI sets up a voting session at the voting agency VA

In the following sections we describe each of them. The notation is the following: $h(.)$ is a cryptographic one-way function, $\|$ denotes concatenation of two strings, $K_X(.)$ denotes encryption with the X's secret (symmetric) key, $E_X(.)$ encryption with the X's public key and $D_X(.)$ decryption with the X's private key. $\#items$ denotes the number of *items*.

3.1 Setting-Up the Voting Session

Prior to any other activities, the voting session must be registered at the chosen voting server (i.e. voting agency). The voting committee has to contact the server and demostrate the appropriate credentials in order to prove its right to organize the voting session (Fig.1). For instance, a voting session may be held for a company, so that the group which may organize the voting session is determined by certain regulations/standing orders of the company. The server can consult the company's data base to check whether the voting initiator is authorized to organize the voting session on the requested topic. The request consists of an announcement and "environment" variables for the voting session. The environment variables are in fact contained in the announcement message and presented later in this section.

3.2 Announcement

The voting session is announced by sending a message to all legitimate voters (Fig.2). At the very least, the announcement contains the following information:

- voting topic
- network address of the responsible voting server (and, eventually, a port number)
- nature of required authentication information for a legitimate voter
- registration deadline
- deadline/term for requesting a valid voting slip
- term of balloting

$VA \longrightarrow V$:	VA,V,*Session_Announc*,Announcement,
	D_{VI}(h(Announcement)||Time_Stamp),
	D_{VA}(h(Announcement)||Time_Stamp+1)

Figure 2: The voting agency VA informs the voter V about the voting session

$V \longrightarrow VA$:	V,VA,*Register_Request*,Announcement,
	D_V(h(Announcement)||Time_Stamp)
$VA \longrightarrow V$:	VA,V,*Register_Acknowl*,
	E_V(Voting_Key||Time_Stamp+1),
	D_{VA}(h(Voting_Key)||Time_Stamp+1)

Figure 3: The voter V registers for participation in voting session at the voting agency VA

3.3 Registration

Because only legitimate voters are allowed to vote, there must be some way to check their identity and determine the total number of voters intending to ballot. Therefore the first phase of the voting service includes authentication.

Registration includes the mutual authentication between the voter and the voting agency. Both are supposed to have a key pair consisting of a private key and a public key (e.g. RSA key pair [17]). Public keys must be valid, i.e. certified by the mutually trusted Certification Authority [5]. The certificates are publicly available, either in some Directory or on request, from the Certification Authority. Private keys are stored in an encrypted form (e.g. encrypted by the DES key mapped from the voter's password, see for example [21]) in the voter's personal directory or on his/her smart card.

After the first step (Fig.3), the VA can check the V's identity by decrypting the received string with the V's certified public key. Adding the time stamp prevents replay. In a similar way, the V obtains the secret (symmetric) voting key K_V, unique for this voting session. *Time_Stamp+1* assures the V that the message sent in the second step is really an answer to his/her previous request.

In order to prevent the voting agency from assigning "dummy" votes the list of all registered voters and signed registrations D_V(h(Announcement)||Time_Stamp) must be published before continuing with the next phase.

3.4 Issuing Voting Slips

This phase of the voting session is the most sensitive one. The reason for this is that the two conditions, firstly that only legitimate voters can vote, and secondly that each of them can vote only once, require strict protocols. Thus this part of the voting protocol must be performed separately.

The purpose of the protocol is the following. The voting agency VA has a list of voting slips, $p_i, i \geq N$, and each voter V has a modulus n_V (a product of two secret large primes). The VA and the V want to "exchange" the slip and the modulus in an oblivious way, so that the VA knows neither which p_i the V received nor the "identity" of the

VA	$\longrightarrow$	V:	VA,V,*Slip_List*,$K_V(p_1 \| \cdots \| p_{N+\delta})$,
			$D_{VA}(\mathrm{h}(p_1 \| \cdots \| p_{N+\delta}) \| \mathrm{Time\_Stamp} \| \mathrm{List\_Number})$
V	$\longrightarrow$	VA:	V,VA,*Slip_Request*, $K_V(p_i \| n_V \| \mathrm{Time\_Stamp{+}1} \| \mathrm{List\_Number})$
VA	$\longrightarrow$	V:	VA,V,*Slip_Acknowl*,$K_V((p_{i_1}, n_{i_1}) \| \cdots \| (p_{i_k}, n_{i_k}))$,
			$D_{VA}(\mathrm{h}((p_{i_1}, n_{i_1}) \| \cdots \| (p_{i_k}, n_{i_k})) \| \mathrm{Time\_Stamp{+}2} \| \mathrm{List\_Number})$
V	$\longrightarrow$	VA:	V,VA,*Slip_Protest*, $K_V(p_i \| n_V \| \mathrm{Time\_Stamp{+}3} \| \mathrm{List\_Number})$
VA	$\longrightarrow$	V:	If #protests $\neq (N - k)$ then
			VA,V,*Slip_Reject*,$D_{VA}(\mathrm{Time\_Stamp{+}4} \| \mathrm{List\_Number} \| \mathit{Rejected})$
			else if #protests $== (N - k)$ then
			VA,V,*Slip_Assign*,$D_{VA}(\mathrm{Time\_Stamp{+}4} \| \mathrm{List\_Number} \| \mathit{Assigned})$

Figure 4: A basic protocol for issuing voting slips

received modulus n_V, except that it comes from a registered voter.

All participants in the slips issuing protocol are supposed to have either an anonymous (e.g. public voting terminal) or an untraceable (see [7]) network address.

After the deadline of the registration phase, the number of all voters intending to vote N is publicly known. The legitimate voters who have not registered cannot proceed with the voting. Every registered (i.e. authenticated) voter is in possession of the secret (symmetric) voting key K_V, unique for this voting session. It must, however, be noted that every voter received information regarding the period in which voting slips were to be issued.

The voting server VA chooses $N + \delta$ large primes p_i, i.e. voting slips. The VA then forms the message which contains the voting slips list and the hash value of the list encrypted with its secret key (Fig.4). The list is encrypted with the unique voting key K_V and sent to all registered voters. In this way the VA is sure that only registered voters can read the list, and the voters are sure that this particular VA has created the list.

After receiving and decrypting the list, the voter V randomly picks *only one* voting slip p_i. V then chooses an RSA modulus, n_V (and later in the balloting phase two integers k_V and t_V so that (3) holds). The voter then tells (p_i, n_V) the voting agency VA, but keeps the factoring of n_V for him/herself without revealing his/her identity.

After the first deadline has expired, the VA uses the received pairs (p_i, n_V) to form a new list in a similar manner to the first one, and sends it to every voter/public voting terminal. It is possible that some p_i's occur more than once, and some p_i's do not occur at all. The p_i's which appear only once are declared ready for assignment (see Fig.4, *Slip_Acknowl*). But it can happen that a voter has picked more than one slip and that all these slips appear only once. Hence this voter could obtain many slips, and other voters no slips at all. That is why the voter with no "acknowledged" slip must send the *protest* (*Slip_Protest* in Fig.4), i.e. let the VA know he/she has no potential slip. On the other side, the VA knows that the sum of the number of acknowledged slips and the number of protests must be at most N. If this is not the case, this means that either there is a voter who has more than one acknowledged slip or that somebody is trying to disturb the slips issuing phase. Therefore, if the mentioned sum (after the previously specified deadline for protests) is greater than N, the current list of acknowledged slips does not become valid (*Assigned* in Fig.4), so that the protocol must be repeated. If the sum is at most N, the

k acknowledged slips become valid and are not considered in the next protocol run, i.e. $N := N - k$.

Another criterion which must be satisfied is that all n_V's must be different, because the knowledge of the factoring of the modulus must be strictly exclusive. If n_V's are large enough, the probability of choosing two equal ones is low enough for practical application.

3.5 Balloting

Until now we have fulfilled all prerequisites necessary for the voting protocol based on multiple key ciphers from the section *The Balloting Protocol.* The balloting protocol is shown in Fig.5.

The voter's messages may be sent in cleartext. It is nevertheless recommended to provide for Data Integrity [12], possibly on a lower communications layer, because an enemy may intercept and change the voter's messages, so that in *Step 3* of the balloting protocol the VA cannot recover a meaningful voting strategy. An enemy cannot send meaningful messages because only the owner of the pair (p_i, n_V) knows the factorization of n_V. In other words, the security is equivalent to that of RSA.

All data published by the VA during the protocol must be authenticated (Data Origin Authentication [12]) in order to prevent an outsider attack. The Non-Repudiation with the Proof of Origin [12] service is also necessary to prevent forgery by the VA.

In *Step 6* of the balloting protocol a voter V may recast the ballot. This may lead to some problems, because the voters have already learned the election outcome in *Step 4* of the balloting protocol which may influence their next decisions. Therefore it would be better to enable recasting *before Step 4* occurs. In any case it must be possible to declare the previous vote invalid. A possible solution is to use a time-stamped voting strategy, e.g. $v_V \|$Time_Stamp, so that only the "latest" vote is accepted as valid.

4 DISCUSSION

4.1 Necessary Assumptions

It is obvious that if any voter tries to pick more than one voting slip p_i and if all registered voters participate in the slips issuing phase, then there must be voters who send the protest. This means, if all registered voters participate in this phase, it is always possible to reveal a fraudulent trial to get more than one voting slip. It can also happen that all registered voters do not try to get a voting slip, but only $N - x < N$ of them. In this case, x slips are not valid. Our first assumption is that the percentage of such slips is small and has no significant impact on the voting outcome.

Any voter (i.e. anybody who knows the secret voting key K_V) can prevent the successful termination of the slips issuing phase. Our second assumption is that the main problem with voting is not its prevention, but rather some form of cheating in favour of some voting strategy (e.g. candidate or decision).

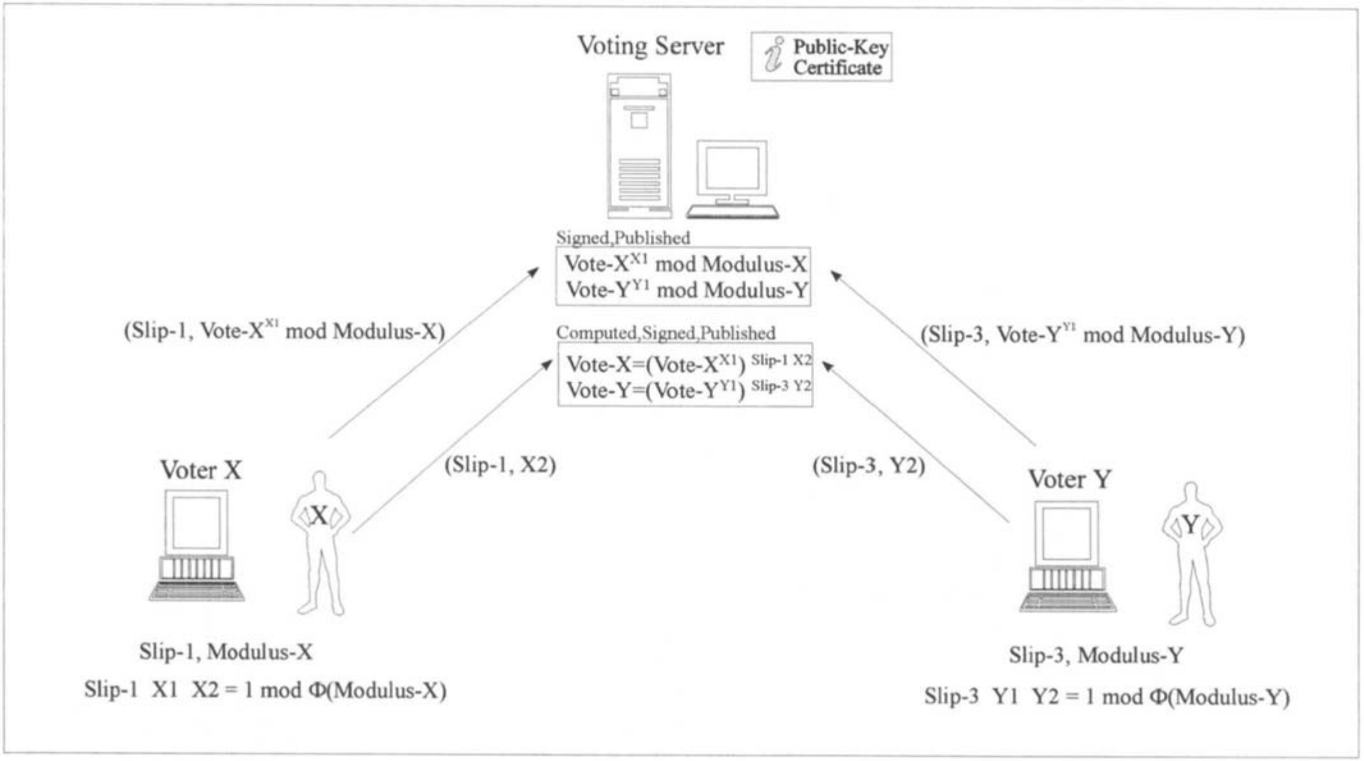
Voting Server
Public-Key Certificate
Signed,Published
Vote-X^X1 mod Modulus-X
Vote-Y^Y1 mod Modulus-Y
Computed,Signed,Published
Vote-X=(Vote-X^X1) ^Slip-1 X2
Vote-Y=(Vote-Y^Y1) ^Slip-3 Y2
(Slip-1, Vote-X^X1 mod Modulus-X)
(Slip-3, Vote-Y^Y1 mod Modulus-Y)
(Slip-1, X2)
(Slip-3, Y2)
Voter X
X
Voter Y
Y
Slip-1, Modulus-X
Slip-1 X1 X2 = 1 mod Φ(Modulus-X)
Slip-3, Modulus-Y
Slip-3 Y1 Y2 = 1 mod Φ(Modulus-Y)

Figure 5: The balloting protocol

4.2 Security

All voters have to be authenticated in the registration phase, prior to balloting, so that only the legitimate ones may obtain the unique secret voting key K_V necessary for obtaining a valid voting slip. This satisfies the condition that only legitimate voters can vote.

If all registered voters participate in the slips issuing phase using an untraceable network address [7], every legitimate voter may obtain only one voting slip and therefore may vote only once.

If the first assumption is not satisfied, the VA may misuse the remaining slips or some voter(s) may obtain more than one slip and vote more than once. In this case there is no way to detect the fraud unless all registered voters reveal their voting strategies by sending an authenticated message with the factorization of the modulus n_V. If our second assumption is not satisfied, any voter may prevent successful termination of the slips issuing protocol by repeatedly choosing more than one slip.

After the pair (p_i, n_V) has been properly assigned, only the voter V knows how to factor n_V. Because the interconnection between the V and the modulus n_V is not known to anybody, no one can determine for whom anyone voted. Only the V knows the factorization of n_V and therefore compute the valid ballot. If somebody has changed the value of the ballot, no meaningful voting strategy may be determined. The list of ballots is published together with the corresponding voting strategies, so that every voter may check if his/her vote has been properly assigned

4.3 Slips Issuing Protocol Analysis

In the slips issuing protocol the registered voters are supposed to choose a voting slip p_i randomly from the list of $N + \delta = \alpha N$ offered slips and return an RSA modulus together with the chosen slip. The question now is, how many steps it takes to finish the protocol, assuming that no voter tries to choose more than one slip, i.e. to prevent the issuing of the slips.

The table in Fig.6 shows the expected number of slips out of αN which have been chosen by only one voter in one run of the slips issuing protocol. The values are obtained by applying the following formula

$$Expect(Number\_of\_slips\_chosen\_once) = N(1 - \frac{1}{\alpha N})^{N-1} \quad (4)$$

The formula (4) can be interpreted in the following way [10]: after having chosen a slip, each voter compares his/her slip number p_i with each of the remaining $N - 1$ voters. The probability that the two slips being compared (out of αN slips) are *not* equal is $(1 - \frac{1}{\alpha N})$.

We observe that for $\alpha = 50$ the value of (4) is almost N (see also the last row of the table in Fig.6). This means that for a big α practically all slips can be assigned in only one protocol run. On the other hand, the percentage of slips chosen once in one run is approximately $100e^{-\frac{1}{\alpha}}$, because $\lim_{N\to\infty}(1 - \frac{1}{\alpha N})^{N-1} = e^{-\frac{1}{\alpha}}$.

$\alpha \downarrow \quad N \rightarrow$	10	50	100	200
2	6.30	30.55	60.88	121.53
5	8.33	41.08	82.02	163.89
10	9.13	45.32	90.56	181.05
50	9.82	49.02	98.03	196.05

Figure 6: The average number of slips which appear only once

Run	Slips to be assigned	Offered slips	Assigned slips
First	100	200	60
Second	40	160	31
Third	9	129	8
Fourth	1	121	1

Figure 7: An example of an ideal protocol development with 100 voters

In Fig.7 an example of an undisturbed protocol development is given. "Undisturbed" means that the *#protests* is always equal to $N - k$ (Fig.4), i.e. no voter tries to obtain more than one slip. The number of assigned slips is computed by expression (4).

To reduce the network traffic, only the first list with slips offered may be sent complete by the VA. The second list may then be sent only with the reference numbers to the first list.

4.4 Implementation

Because of the nature of the voting service, i.e. occasional need and usage, our implementation choice is the client-server architecture. The most suitable is the connection-oriented, concurrent model of server. We need a reliable protocol to be sure that our messages have reached the receiver. The server must also be "stateful", because the resulting information from each voting phase is to be used as a necessary prerequisite in the next one (the actions are not idempotent).

A hierarchical structure of servers is also possible. There can be a slave server for each voting unit (e.g. on the same LAN), and a master server which collects voting results from slaves. Each slave must publish its voting results during the balloting phase, so that the master server need not be trusted and is incapable of forgery.

The anonymity of the network address can be achieved by using public voting terminals with anonymous accounts. One drawback of this approach is that the voter must use a public account twice, the first time to get a slip and the second time to cast a vote.

SUMMARY

In this work a LAN voting service is described. The service is based on the voting protocol in [19, 14]. The mathematical background is the generalization of RSA, i.e. multiple key ciphers [2]. The necessary conditions for democratic elections are satisfied, i.e. that only legitimate voters can vote and that each of them votes only once. Every voter can check that his/her vote has been properly counted. Recasting a ballot is also possible. For the protocol to work, two assumptions must be fulfilled. Firstly, all registered voters participate in the slips issuing protocol and secondly, no voter wants to prevent the distribution of voting slips.

Acknowledgement

We wish to express our gratitude to Hannes Hassler for useful discussions and help in the slips issuing protocol analysis. We would also like to thank all referees for their valuable comments, especially the referee who proposed a *birthday paradox* analysis [10] of the slips issuing protocol.

References

[1] Benaloh, J.C., *Secret Sharing Homomorphisms: Keeping Shares of a Secret Secret (Extended Abstract)*, Proceedings of EUROCRYPT 86, Springer Verlag, 251–260, 1986

[2] Boyd, C.A., *Some Applications of Multiple Key Ciphers*, Proceedings of EUROCRYPT 88, Springer Verlag, 455–467, 1988

[3] Boyd, C.A., *A New Multiple Key Cipher and an Improved Voting Scheme*, Proceedings of EUROCRYPT 89, Springer Verlag, 617–625, 1989

[4] Brassard, G., Crepeau, C., Robert, J.-M., *All-or-Nothing Disclosure of Secrets*, Proceedings of CRYPTO 86, Springer Verlag, 234-238, 1987

[5] CCITT, Recommendation X.509: *The Directory: Authentication Framework*

[6] Chaum, D.L., *Elections with Unconditionally-Secret Ballots and Disruption Equivalent to Breaking RSA*, Proceedings of EUROCRYPT 88, Springer Verlag, 177–182, 1988

[7] Chaum, D.L., *Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms*, Communications of the ACM, Vol.24, 84–88, 1981

[8] Chen, L., *Oblivious Signatures*, Proceedings of ESORICS 94, Springer Verlag, 161–172, 1994

[9] Cohen, J., Fischer, M., *A Robust and Verifiable Cryptographically Secure Election Scheme*, Proc. 26th IEEE Symp. on Foundations of Computer Science, 372–382, Portland (OR), 1985

[10] Davies, D.W., Price, W.L., *Security for Computer Networks. An Introduction to Data Security in Teleprocessing and Electronic Funds Transfer*, John Wiley&Sons, 1989

[11] Fujioka, A., Okamoto, T., Ohta, K., *A Practical Secret Voting Scheme for Large Scale Elections*, Proceedings of AUSCRYPT 92, Springer Verlag, 244–251, 1992

[12] Information technology - Open Systems Interconnection - Basic Reference Model - Part 2: Security Architecture, ISO IS 7498-2, 1989

[13] Lippitsch,P., Posch, K.C.,Posch R., Schindler, V. *A scalable RSA design with encryption rates from 200 Kbit/s to 1.5 Mbit/s*, 32nd International Science Week, Damascus, Dec.1992

[14] Nurmi, H., Salomaa, A, Santean, L., *Secret Ballot Elections in Computer Networks*, Computers&Security 10, 553–560, 1991

[15] Park, C., Itoh, K., Kurosawa, K., *Efficient Anonymous Channel and All/Nothing Election Scheme*, Proceedings of AUSCRYPT 91(?), Springer Verlag, 248–259, 1991

[16] Renvall, A., *ANDOS: A Simple Protocol for Secret Selling of Secrets*, EATCS Bulletin ??, 200–205, 199?

[17] Rivest, R.L., Shamir, A. and Adleman L., *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*, Comm. ACM 21, 120–126, 1978

[18] Schneier, B., *Applied Cryptography*, John Wiley&Sons, 1994

[19] Salomaa, .A., *Verifying and Recasting Secret Ballots in Computer Networks*, EATCS Bulletin 44, 1991

[20] Salomaa, A, Santean, L., *Secret Selling of Secrets with Several Buyers*, EATCS Bulletin, 42, 178–186, 1990

[21] Schneider, W., *SecuDE: Overview (Version 3.0)*", Institut fuer TeleKooperationsTechnik, Darmstadt, March 1992