The Theory and Implementation of an Electronic Voting System

Ivan Damgård, Jens Groth and Gorm Salomonsen

July 31, 2002

Abstract

We describe the theory behind a practical voting scheme based on homomorphic encryption. We give an example of an ElGamal-style encryption scheme, which can be used as the underlying cryptosystem. Then, we present efficient honest verifier zero-knowledge proofs that make the messages in the voting scheme shorter and easier to compute and verify, for voters as well as authorities, than in currently known schemes. Finally, we discuss various issues connected with the security of a practical implementation of the scheme for on-line voting. Notably, this includes minimizing risks that are beyond what can be handled with cryptography, such as attacks that try to substitute the software running on client machines.

1 Introduction

Voting schemes are one of the most important examples of advanced cryptographic protocols with immediate potential for practical applications. Such protocols should of course have security properties similar to those of ordinary paper based elections, but the fact that digital communication is used may also open up new possibilities. Informally, the most important goals for electronic voting schemes are:

- Privacy: only the final result is made public, no additional information about votes will leak.

- Robustness: the result reflects all submitted and well-formed ballots correctly, even if some voters and/or possibly some of the entities running the election cheat.

- Universal verifiability: after the election, the result can be verified by anyone.

Other properties may be considered as well, such as receipt-freeness. In a receipt-free election, voters are not able to prove that they voted for a particular candidate after the election, thereby discouraging vote-buying or coercing.

Various fundamentally different approaches to electronic voting are known in the literature: one may use *blind signatures and anonymous channels*[13], where the channels can be implemented using *MIX nets* (see [20, 1] for instance) or be based on some physical assumption. The idea in such a scheme is that a voter prepares a ballot in cleartext, i.e., a message stating for whom he votes. He then interacts with an authority that can verify that he is eligible to vote and has not already voted. If this is the case, the authority issues a blind signature on the ballot. Informally, this means that the voter obtains the authority's digital signature on the ballot, without the authority learning any information about the contents of the ballot. On the other hand, a voter cannot obtain such a signature without interacting with the authority, and is therrefore prevented from voting several times. Finally, all voters send their ballots to another authority that is responsible for counting votes. In order to preserve the privacy of voters, this must be done through an anonymous channel. Such a channel can be implemented based on cryptography, using a so-called MIX network or it may be based on physical assumptions. After all ballots have been received, votes can be counted directly. Ballots without the relevant authority's signature are, of course, ignored.

Another approach is to use several servers to count the votes and have voters *verifiably secret share* votes among the servers [8, 6]. In such a scheme, the voter interacts with all servers. Each server gets a *share* of each voter's ballot. These shares are constructed with respect to a threshold t in such a way that the servers together have complete information on each ballot, but any set of at most t servers has no information at all. The voter must convince all servers that the shares were correctly constructed, and so he is prevented from voting twice or voting incorrectly. Once the votes have been cast, the set of all servers can interact and compute the result of the election without any side information becoming public.

A final approach is to use *homomorphic encryption*[9, 11]. In such a system, a voter simply publishes an encryption of his vote, represented as a number. This encryption is done using a public-key cryptosystem, i.e., there is a public key known by everyone that can be used for encrypting each vote. When submitting his encrypted vote, the voter must identify himself to prove that he is eligible to vote and has not voted before. Furthermore, he must prove knowledge of the fact that his encryption contains a valid vote. Because all individual votes will remain encrypted and the proof is zero-knowledge, this does not violate privacy. On the other hand, because we use homomorphic encryption, the election result can be computed efficiently. This is because the cryptosystem comes with a method by which two encryptions of, say, numbers a and b can be combined to produce a new encryption that is guaranteed to contain $a + b$. By repeated use of this method, all votes can be "implicitly added" together without decrypting anything. This will produce an encryption of the result and so finally all that is needed is to decrypt this. This can be done securely assuming that the private key needed for this has been *secret-shared* among a set of authorities, each running a server responsible for helping computing the result. Each server holds a share of the private key. The shares have to be constructed w.r.t. a threshold value t so that no information about the private key leaks as long as at most t severs are corrupt, or are broken into by a hacker. On the other hand, if at least $t + 1$ servers behave correctly, then a decryption operation can be executed. This is also known as *threshold decryption.*

If the total number of servers participating is n, then we can set t to just below $n/2$, i.e. $t = \lfloor (n-1)/2 \rfloor$. Then, we are guaranteed that if a majority of the servers are in operation and are not corrupted, the election result, and only that will be decrypted. In practice, one may imagine that some public institutions and political parties could be running these servers in order to create broad trust in the process.

The last approach seems the most practical out of the three we have discussed: anonymous channels are quite difficult to implement. Even the best implementations (based on MIX nets) require that all votes have been cast before any processing can be done, and so they may introduce a significant delay in getting the final result. The second approach requires each voter to interact with every authority, and is therefore hardly practical either. Hence this paper deals only with variants of the approach based on homomorphic encryption.

2 Which Cryptosystems Can We Use?

the introduction above shows that the approach on which we concentrate here requires a homomorphic public-key cryptosystem with threshold decryption. In addition, some other technical properties come in handy; we discuss those in more detail below.

In [9], the use of ElGamal encryption is suggested. This is possible, but leads to efficiency problems if the number of candidates is large. Most of these problems can be solved by using Paillier's cryptosystem [21], or the generalization suggested by Damgård and Jurik in [11]. In that case the zero-knowledge protocols and threshold decryption presented in [4, 11] are also required.

In this paper, we suggest an alternative cryptosystem, which may be of interest for various reasons: it is based on a different intractability assumption (a general form of the Decision Diffie-Hellman assumption) and has other properties that neither Paillier nor ElGamal can satisfy at the same time.

We present the system from a general point of view: let R be a ring, fix some $g \in R$ and let $G =< g >$. We will assume that one can compute addition and multiplication efficiently in R and that a number T can be computed easily, so that $T \geq ord(g)^2$. This just requires that some upper bound on $ord(g)$ is publicly known. As for intractability assumption, we assume that a *generalized DDH assumption* holds w.r.t. R and g, i.e., given R, g, triples of form g^a, b^b, g^{ab} where a, b are random in $[0..T]$ are computationally indistinguishable from triples of form g^a, b^b, g^c where a, b, c are random in $[0..T]$. Note that the choice of T ensures that the distribution of elements such as g^a is statistically close to uniform in $< g >$ as long as $ord(g)$ is large.

It is now clear that we can define an ElGamal style cryptosystem where the public key is $R, g, h = g^x$ where x is random in $[0..T]$, and where the private key is x. The message space is $< g >$, and to encrypt a message m, choose $r \in [0..T]$ at random and output $E(m, r) = (g^r, mh^r)$. Decryption of a ciphertext (u, v) takes place by computing $v(u^x)^{-1}$. Clearly, this system is semantically secure under the generalized DDH assumption.

This system is not homomorphic as we required above. As a first step to solve this problem, we can redefine the system by fixing an element $w \in< g >$, and letting the message space be instead $\mathbf{Z}_{ord(w)}$. Now, we can define $E(m, r) = (g^r, w^m h^r)$. This does not affect the semantic security, but of course implies that we have the homomorphic property $E(m, r)E(m', r') = E(m+m' \bmod ord(w), r+r')$. But as the case was with the ElGamal variant used in [9], we now have the problem that to decrypt, we must find discrete logarithms to the base w, since the basic decryption from above only allows us to compute w^m.

The point is that in some rings, one can find elements for which computing the discrete logarithm is in fact easy. Suppose we have $w = \alpha + \beta$.

Then

$$w^i = (\alpha + \beta)^i = \sum_{j=0}^{i} \binom{i}{j} \alpha^j \beta^{i-j}$$

using the standard binomial expansion. Since i will typically be exponentially large, this is normally not going to be useful towards computing i. But if α is nilpotent, that is $\alpha^j = 0$ for some small j, then most of the terms in the expansion disappear, and it may be feasible to compute i.

As a concrete example of this, we can use let $R = \mathbf{Z}^*_{n^{s+1}}$, where $n = pq$ is an RSA modulus where $gcd((p-1),(q-1)) = 2$. We let g have Jacobi symbol 1 and maximal order, that is $ord(g) = n^s(p-1)(q-1)/2$. Now, $n \in R$ is nilpotent, since of course $n^{s+1} = 0$. So we set $w = n+1$. By classical algebra and number theoretic results, we have $ord(w) = n^s$ and that discrete logarithms base w are easy to compute, along the lines just sketched. A concrete algorithm can be seen in [11]. The threshold decryption only requires that we can compute securely $u^x \bmod n^{s+1}$ given u and a secret sharing of x. A protocol for this is given in [11].

Some comments on how our scheme differs from earlier systems: our scheme can be described as simply the ElGamal solution from [9], but transplanted to a ring where it happens to be easy to compute discrete logs base the fixed element w. The Paillier and Damgård-Jurik schemes also use the ring $\mathbf{Z}^*_{n^{s+1}}$ and (implicitly) the special properties of the element $n+1$, but as mentioned these are known results from algebra. The distinguishing feature of Paillier/Damgård-Jurik is that they propose a way to use the factorization of n as the trapdoor that makes decryption possible, while we use a secret discrete logarithm. Therefore, when keys are generated, a trusted party could choose n and g but then immediately delete the factorization. Then the private key x and the sharing of it can be generated independently of the factorization, perhaps in a distributed way. It also means that one can define several instances of the same system using the same n, i.e., several different public h-values. If one or more private keys are compromised, this does not affect the security of the other keys.

We note a couple of facts for later use: The cryptosystem satisfies a root opening assumption. If we are given the decryption of a ciphertext (u^e, v^e) for some $e < p, q$, then we can also find the message m contained in the ciphertext (u, v). The reason for this is that theplaintext corresponding to (u^e, v^e) must be $em \bmod n^s$, and so we can find m because e is always invertible modulo n^s.

Another observation is that when using standard techniques, the zero-knowledge protocols for proving various claims on encrypted values from [11] can all be transplanted to our cryptosystem quite easily - basically because the plaintext space is the same, and both systems are homomorphic.

3 Zero-Knowledge Proofs

In this section, we take a closer look at how the correctness of encrypted votes can be proved in zero-knowledge. We present an efficient zero-knowledge proof of knowledge for demonstrating the correctness of the vote in the case where each voter may select only one option or candidate. We then extend the proof system to cover the more complex elections where the voter on the same ballot may cast several votes with the restriction that they all are on different candidates.

We define two election parameters M and L. M is a strict upper bound on the number of voters participating in the election. L is the number of candidates or options each voter may choose from. Included in this number may be dummy candidates representing unused votes, blank votes or invalid votes.

In theory, only $\mathcal{O}(\log L)$ bits are needed to convey the choice of the voter. This possibility was investigated in [12] where the tally servers transform the encrypted votes into encrypted votes in a more usable format. In practice, their scheme places too large a workload on the tally servers though. Currently, the best choice seems to be to represent votes in a format that can use the homomorphic property of the cryptosystem directly.

We represent the candidates by numbers $j \in \{0, \ldots, L-1\}$. A vote on candidate j is represented as the number M^j. Notice that in this way the sum of several votes will be a number on the form $v_0 M^0 + \ldots + v_{L-1} M^{L-1}$ where v_j is the number of votes on candidate j. With this choice of vote representation the message space for the cryptosystem must be of size $\Omega(L \log M)$.

When the number of candidates is large, the ciphertexts are correspondingly large, and in the cryptosystems we know the computational complexity of the encryption process is large too. The encryption process is not the heaviest part in generating a vote though. Looking closer at the schemes proposed in the literature [9, 11, 4] it turns out that the zero-knowledge proofs used to prove the correctness of the encrypted votes involve several encryptions. The really heavy part of generating a vote and tallying a vote, both in terms of communication complexity and computational complexity, is producing and verifying the zero-knowledge proof associated with it. It is therefore highly interesting to find efficient zero-knowledge proofs for the correctness of encrypted votes.

In the zero-knowledge proof, the prover (the voter) wants to convince the verifier (the tally servers) of the correctness of the encrypted vote. For this purpose, we use Σ-protocols that are a type of 3-move honest verifier zero-knowledge proofs that work in the following way: The prover and verifier know a common input x and the prover knows a witness w such that $(x, w) \in R$ where R is some relation. The prover sends an initial message a to the verifier, is then given a randomly chosen challenge e, and responds with an answer z. On basis of (a, e, z), the verifier decides whether to accept the claim that $x \in L$ where L is the language specified by the relation R. We call such a proof system a Σ-protocol when it satisfies the following criteria:

- Completeness: Given w so that $(x, w) \in R$ the prover can make an honest verifier accept with overwhelming probability.

- Special soundness: Given x and two acceptable proofs (a, e, z) and (a, e', z') with the same initial message but different challenges it is possible to extract a witness w so that $(x, w) \in R$. Note that special soundness makes a Σ-protocol a system for proofs of knowledge.

- Special honest verifier zero-knowledge: Given $x \in L$ and any challenge e it is possible to simulate a proof (a, e, z) with the same probability distribution as the distribution of real proofs with any witness and conditioned on using the challenge e.

Using the Fiat-Shamir heuristic Σ-protocols can be made non-interactive by using a cryptographic hash function h and letting the challenge be created as $e = h(x, a)$. In the random oracle model, the resulting hash value $h(x, a)$ is completely random and we therefore have a non-interactive zero-knowledge proof of knowledge for $x \in L$.

Very efficient Σ-protocols exist for basic properties such as three ciphertexts being encryptions of plaintexts a, b, c so that $c = ab$, a ciphertext being an encryption of 0, two ciphertexts containing a, b so that $a = b$, etc. For more complex cases such as a ciphertext containing a vote on the form $M^j, 0 \leq j < L$, it is possible to build a zero-knowledge proof from the more basic Σ-protocols. However, the basic Σ-protocols, while being efficient, do need a few extra encryptions in the process. When several basic proofs are needed, it all adds up to the use of several encryptions, which, in the context of voting, as mentioned before, can be heavy to deal with both in terms of communication and computational complexity.

The ideas behind the basic Σ-protocols are quite general though and can be used not only in connection with homomorphic public key encryption schemes but also with homomorphic commitment schemes. To improve the efficiency of the needed zero-knowledge proof for correctness of the vote, Lipmaa suggests in [18] to create a commitment to the vote and prove knowledge of the commitment and the ciphertext holding the same content. Using a homomorphic integer commitment scheme this carries two advantages: The commitments do not need to be unconditionally binding as do the ciphertexts and so they can be much lighter to work with. By using an integer commitment scheme, we can potentially use special properties of this ring, in our case that of unique factorization.

Before proceeding, let us be more precise about the kind of commitment scheme we deal with. First, there is the key generation phase in which a public key is generated. In our case, the election authorities will be the ones generating the key. From now on we will just assume that some key K has been generated, and accordingly there is an associated message space $\mathcal{M}_K$, a randomizer space $\mathcal{R}_K$, an opening space $\mathcal{B}_K \supset \mathcal{R}_K$, a commitment space $\mathcal{C}_K$, a commitment function $com_K(\cdot,\cdot) : \mathcal{M}_K \times \mathcal{R}_K \to \mathcal{C}_K$ and a verification function $ver_K(\cdot,\cdot,\cdot) : \mathcal{M}_K \times \mathcal{B}_K \times \mathcal{C}_K \to \{0,1\}$.

Given the key, we can commit to an element $m \in \mathcal{M}_K$ by selecting at random according to some distribution specified by the commitment scheme $r \in \mathcal{R}_K$ and letting the commitment be $c = com_K(m;r) \in \mathcal{C}_K$. This (m,r,c) satisfies $ver_K(m,r,c) = 1$.

To open a commitment, we reveal $m \in \mathcal{M}_K, r \in \mathcal{B}_K$ so that $ver_K(m,r,c) = 1$. Note that we do allow for openings not corresponding to correctly formed commitments since the opening space and the randomizer space do not need to be identical. However, we still require that the binding property be satisfied, i.e., that nobody can find a commitment in $\mathcal{C}_K$ and two correct openings of it with different messages m_1 and m_2.

In order for the commitment schemes to be useful in our voting protocol we have some additional requirements. One important thing is that the spaces associated with the commitment scheme shall be abelian groups[1], and furthermore that the message space is the entire set of integers. That means we have groups $\mathcal{M}_K = \mathbf{Z}, (\mathcal{R}_K, +) \leq (\mathcal{B}_K, +)$ and $(\mathcal{C}_K, \cdot)$.

Homomorphic property: The commitment schemes we look at must be

[1] We assume that both the group and the elements in the groups we work with can be represented in a suitable manner, the binary operations and inversions can be computed efficiently, and that we can readily recognize whether an element belongs to a particular group.

homomorphic, meaning that for all $m_1, m_2 \in \mathbf{Z}$ and all $r_1, r_2 \in \mathcal{B}_K$:

$$com_K(m_1; r_1) com_K(m_2; r_2) = com_K(m_1 + m_2; r_1 + r_2).$$

Root opening: We demand that for any $c \in \mathcal{C}_K$, if we can find $e \in \mathbf{Z} \setminus \{0\}$ and $m \in \mathbf{Z}, z \in \mathcal{B}_K$ so that $com_K(m; z) = c^e$ then we can compute an opening of c.

An example of such a commitment scheme is the following variant of the Damgård-Fujisaki commitment scheme from [10]. Here, the key consists of n chosen as a product of two large safe primes, and two squares g, h so that $\log_g h$ and $\log_h g$ is not known to the sender who is making the commitment.

A commitment to an integer m is formed by choosing r at random from a sufficiently large interval of integers and letting the commitment be $com_{(n,g,h)}(m; r) = g^m h^r \bmod n$.

To open a commitment c we produce b, m, r such that $1 = b^2 \bmod n$ and $c = bg^m h^r$.

The ElGamal style encryption scheme we presented before satisfies these requirements too, except for the root opening property. It satisfies a weaker root opening property. Given a valid ciphertext (u, v) we may extract the plaintext of (u, v) from an opening of (u^e, v^e), where $0 < e < p, q$. In addition, we can simply check whether a ciphertext is valid by computing the Jacobi symbols of u and v. In the following, any homomorphic public-key cryptosystem with the above properties will work, even if the root opening property is only satisfied for $e \in \{0, \ldots, 2^t - 1\}$, where t is some security parameter. Therefore, we describe the protocols in general terms in what follows. We shall always write pk for the public key of the cryptosystem, and let $\mathcal{C}_{pk}$ be the corresponding ciphertextspace, consisting of only *valid* ciphertexts.

Given a homomorphic integer commitment scheme, we can now use the following Σ-protocol for proving knowledge that a commitment and a ciphertext contain the same element modulo n where the message space for the cryptosystem is $\mathbf{Z}_n$.

Proof of commitment and encryption holding same element modulo n

Common input: A commitment $c \in \mathcal{C}_K$ and an encryption $E \in \mathcal{C}_{pk}$.
Private input for the prover: $m \in \mathbf{Z}_n$, $r_c \in \mathcal{R}_K$ and $r_E \in \mathcal{R}_{pk}$ so that $c = com_K(m; r_c)$ and $E = E_{pk}(m; r_E)$.

Initial message: Pick $d \in \mathbf{Z}$ as a shadow[2] of em, $r'_c \in \mathcal{R}_K$ as a random shadow of er_c and $r'_E \in \mathcal{R}_{pk}$ as a random shadow of er_E. Let $a_c = com_K(d; r'_c)$ and $a_E = E_{pk}(d \bmod n; r'_E)$. The initial message is (a_c, a_E).

Challenge: The challenge consists of e chosen at random from $\{0, \ldots, 2^t - 1\}$.

Answer: Set $D = em + d, z_c = r'_c + er_c, z_E = r'_E + er_E$. The answer to the challenge is (D, z_c, z_E).

Verification: The verifier checks that $(D, z_c, z_E) \in \mathbf{Z} \times \mathcal{R}_K \times \mathcal{R}_{pk}$, $com_K(D; z_c) = a_c c^e$ and $E_{pk}(D \bmod n; z_E) = a_E E^e$.

Having an integer commitment to the vote, the next question is how to prove that it has the correct form. Lipmaa [18] suggests selecting M as a prime and using a zero-knowledge proof of knowledge to demonstrate that the following three commitments $c_v = com_K(v; r_v), c_b = com_K(M^L/v; r_b)$, $c_c = com_K(M^L; 0)$ satisfy a multiplicative relationship. This implies that the absolute value of the content in c_v, $|v|$, is a divisor in M^L. Subsequently using a range proof, see [18] or Boudot's article [2], we can then prove that $v \geq 0$. Combining these two pieces of information we see that v is of the desired form.

This idea can be improved upon. Proving that a committed integer is positive is not that simple. In [18], the fact that all positive integers can be written as a sum of four squares, and, of course, no negative number can be written as such a sum, is used. In other words four commitments are provided and it is proven that all of them contain squares. The commitment to the vote v is the product of these four commitments, by the homomorphic property giving us that the commitment contains a non-negative integer.

[2] Let us informally explain the concept of shadowing and random shadowing. In this proof we will at some point reveal $D = d + em$ where $e \in \{0, \ldots, 2^t - 1\}$. To preserve the zero-knowledge property we must therefore choose d such that revealing D does not give away any knowledge about m. In the particular case here we know that $m \in \mathbf{Z}_n$ and $0 \leq e < 2^t$. Thus by selecting d at random from $\{-2^{k+2t}, \ldots, 2^{k+2t}\}$, where $k = |n|$, we ensure that the secrecy of m is preserved. Similarly we will at some point reveal an element $z_c = r'_c + er_c \in \mathcal{R}_K$. This should not give away knowledge about r_c. In addition r'_c should be chosen such that we cannot distinguish it from a properly chosen random element from $\mathcal{R}_K$. We call r'_c chosen in this way a random shadow for er_c. We can speak of computational, statistical and perfect shadowing depending on how the shadow hides the underlying element. In the protocols we know of the most common case is statistically hiding shadows and random shadows.

According to [18], the range proofs in [2] are 20% more efficient but still in the same ball park.

As an alternative, we propose letting M be the square of a prime. Any legal vote is now a square, and we simply have to prove it a square in order to show that it is non-negative. So let $M = p^2$ with p prime and provide a commitment c_v to M^j. We show that c_v contains the square of the contents of a commitment c_a to p^j. Furthermore, we prove that the content of c_a multiplied by the content of another commitment c_b equals p^{L-1}. All in all this proves that c_v contains a vote of the correct form, and it replaces the somewhat complex range proof with a single squaring proof.

Further improvements can be achieved but they require that we dig into the proof system we use for proving multiplicative relationships between commitments. Let us therefore first present a general Σ-protocol for making proofs of the contents of some commitments having a multiplicative relationship with each other.

Proof of multiplicative relationship
Common input: $c_a, c_b, c_c \in \mathcal{C}_K$.
Private input for the prover: $a, b \in \mathbf{Z}, r_a, r_b, r_c \in \mathcal{R}_K$ such that $c_a = com_K(a; r_a), c_b = com_K(b; r_b), c_c = com_K(ab; r_c)$.

Parallel proof: Make in parallel with the rest of the protocol a proof of knowledge of commitment opening of c_b or c_c using a Σ-protocol.

Initial message: Select d such that it shadows ea. Choose $r_d, r_{db} \in \mathcal{R}_K$ as random shadows of er_a and $-(ea + d)r_b + er_c$, and send $c_d = com_K(d; r_d)$ and $c_{db} = com_K(db; r_{db})$ to the verifier.

Challenge: Select at random $e \in \{0, \dots, 2^t - 1\}$.

Answer: Respond with $f = ea + d, z_1 = er_a + r_d, z_2 = fr_b - er_c - r_{db}$.

Verification: Accept if and only if $f \in \mathbf{Z}, z_1, z_2 \in \mathcal{R}_K, com_K(f, z_1) = c_d c_a^e$ and $c_{db} c_c^e com_K(0; z_2) = c_b^f$ and the parallel proof of knowledge is acceptable.

The proof of this being a Σ-protocol is standard and we do not go through it here. We would, however, like to point out the little detail that we allow the parallel proof to be a proof of knowledge of an opening of c_c. The reason for this is that we make a multiplication proof where we already know the opening of $c_c = com_K(p^{L-1}; 0)$ and thus we can save ourselves from having to do the parallel proof. The price for this change is that the root opening assumption on the commitment scheme needs to be slightly stronger than usual. Usually, one only requires that knowing an opening of c^e for a commitment c with $e \in \{1, \ldots, 2^t - 1\}$ makes it possible to open c itself. We require that knowing an opening of c^f with $f \in \mathbf{Z} \setminus \{0\}$ makes it is possible to find an opening of c.

Another thing worth noting is that in the proof of the commitment c_v containing M^j the commitment c_a to p^j is involved in two multiplication proofs. It is used both in the multiplication proof that shows it is a factor in p^{L-1} and in the multiplication proof where it is shown that the square of its content is contained in c_v. Selecting the same challenge e in both the multiplication proofs, something that we can do and still preserve zero-knowledge because the proof system is special honest verifier zero-knowledge, allows us to recycle d, c_d, f and z_1 in the two proofs.

As a final improvement, we shall see that we do not at all need c_v in the proof of the correctness of the vote. We may entirely skip this commitment and jump directly to proving that the encryption of the vote contains the square of the content in c_a. This is due to the fact that on the commitment side, we use c_v to hold M^j as the result of squaring the content of c_1. However, we may as well use $c_1^f = c_1^{ep^j+d}$ directly since this by the homomorphic property of commitments is a commitment to $ep^{2j} + dp^j$ and thus contains the interesting $p^{2j} = M^j$ itself.

It is time to combine all our ideas into an actual protocol.

Proof of knowledge for a ciphertext containing a valid vote
Common input for prover and verifier: Prime p such that $M = p^2$ and an encryption $E \in \mathcal{C}_{pk}$.
Private input for the prover: $0 \leq j < L$ and $r_E \in \mathcal{R}_{pk}$ such that $E = E_{pk}(M^j; r_E)$.

Initial message: Choose first r_a, r_b at random from $\mathcal{R}_K$ and form commitments $c_a = com_K(p^j; r_a)$ and $c_b = com_K(p^{L-j-1}; r_b)$.

Choose d such that it shadows p^j. Choose γ such that it shadows $eM^j + dp^j$. Choose $r_d, r_{db}, r_\gamma \in \mathcal{R}_K$ and $r'_\gamma \in \mathcal{R}_{pk}$ as random shadows of $er_a, (ep^j + d)r_b, (ep^j + d)r_a, er_E$ respectively.

Send $c_d = com_K(d; r_d), c_{db} = com_K(dp^{L-j-1}; r_{db}), c_\gamma = com_K(dp^j + \gamma; r_\gamma)$ and $E_\gamma = E_{pk}(dp^j + \gamma \bmod n; r'_\gamma)$ to the verifier.

Challenge: Select e at random from $\{0, \ldots, 2^t - 1\}$.

Answer: Send $f = ep^j + d, z_1 = er_a + r_d, z_2 = fr_b - r_{db}, z_3 = fr_a + r_\gamma, z_4 = er_E + r'_\gamma$ and $D = eM^j + dp^j + \gamma$ to the verifier.

Verification: Check that $c_d, c_{db}, c_\gamma \in \mathcal{C}_K, E_\gamma \in \mathcal{C}_{pk}, f, D \in \mathbf{Z}, z_1, z_2, z_3 \in \mathcal{R}_K$ and $z_4 \in \mathcal{R}_{pk}$.

Verify that $com_K(f; z_1) = c_d c_a^e, c_{db} com_K(p^{L-1}; 0)^e com_K(0, z_2) = c_b^f$, $com_K(D; z_3) = c_a^f c_\gamma$ and $E_{pk}(D; z_4) = E^e E_\gamma$.

Theorem 1 *The proof system above is a Σ-protocol for proving that E is a ciphertext holding a vote on the correct form. It is statistical special honest verifier zero-knowledge if the commitment scheme is statistically hiding and the shadows are statistically hiding.*

Proof. Theorem 1 follows as a corollary to Theorem 2 proven later.

□

Compared to the scheme from [11], which until now is the most efficient voting scheme based on homomorphic encryption, we asymptotically get an improvement in the order $\log L$ both in terms of communication complexity and computational complexity on the voter's side. Furthermore we note that the constants in this scheme are smaller than the constants in the schemes of both [11] and [18].

An additional advantage of the approach is that it can be extended to cover the situation where each voter is allowed to cast several votes in the same session. We define a new election parameter N to be the number of candidates a voter may vote for. Moreover, we demand that the votes must be cast on different candidates. A simple approach would be to cast N votes and proving them all to be different, but we can do much better than this. The first thing we notice is that it is sufficient for the voter to provide an encryption of the sum of his votes and proving this sum correct. We write the candidates in increasing order $0 \leq j_1 < \ldots < j_N < L$. We encrypt $M^{j_1} + \ldots + M^{j_N}$ and wish to have a Σ-protocol for proving that a ciphertext E contains a vote of this form.

To do so, we may form commitments $c_1, \ldots, c_N$ to $p^{j_1}, \ldots, p^{j_N}$, and furthermore make commitments $c'_1, \ldots, c'_N$ to $p^{j_2 - j_1 - 1}, \ldots, p^{L-1-j_N-1}$. Using multiplication proofs we can demonstrate knowledge that for $i = 1, \ldots, N$, the contents of c_i and $(c'_i)^p$ multiplied with each other equals the content of c_{i+1}, where we let $c_{N+1} = E_{pk}(p^L; 0)$. This shows that all the commitments $c_1, \ldots, c_N$, except for a sign difference, contain powers of p, that all the exponents are different, and that the exponents lie in the interval $\{0, \ldots, L-1\}$.

We can proceed by forming commitments $c''_1, \ldots, c''_N$ to $M^{j_1}, \ldots, M^{j_N}$. We prove for $i = 1, \ldots, N$ knowledge that the contents of $c''_1, \ldots, c''_N$ contain the square of the content of $c_1, \ldots, c_N$. Finally, we form the commitment

$c''_1 \cdots c''_N$. This is a commitment to the intended vote, which proofs show contains an element on the form $M^{j_1}+\ldots+M^{j_N}$, where $0 \leq j_1 < \ldots j_N < L$. What is left is to encrypt this vote to a ciphertext E and prove knowledge of the equality with the content of $c''_1 \cdots c''_N$.

We can make similar improvements as we did in the voting scheme for the single candidate scenario. We note that the commitments $c_1, \ldots, c_N$ are all involved in two multiplication proofs and obtain a more efficient proof system by using the same challenge e in all the proofs allowing us to recycle the d, c_d, f, z_1 parts in the multiplication proofs.

Furthermore, we do not need to start each multiplication proof with a parallel proof of knowledge for some opening. Throughout the proofs, we do have knowledge of an opening to the commitment to the product of the contents, since we know how to open $c_{N+1} = com_K(p^L; 0)$ of course. The multiplication proof involving c_N and c'_N proves knowledge of how to open c_N. This in turn means that the multiplication proof involving c_{N-1} and c'_{N-1} proves knowledge of how to open c_{N-1}, etc.

Finally, since we use the same challenge in all the proofs, we may avoid supplying the commitments $c''_1, \ldots, c''_N$ in a manner similar to the single candidate scheme .

Let us write the entire scheme down

Proof of knowledge for a ciphertext containing a valid vote on multiple candidates

Common input: Prime p such that $M = p^2$ and $E \in \mathcal{C}_{pk}$.

Private input for the prover: $0 \leq j_1 < \ldots < j_N < L$ and $r_E \in \mathcal{R}_{pk}$ such that $E = E_{pk}(M^{j_1} + \ldots + M^{j_N}; r_E)$.

Initial message: Choose at random $r_1, \ldots, r_N, r'_1, \ldots, r'_N$ from $\mathcal{R}_K$, and form commitments $c_1 = com_K(p^{j_1}; r_1), \ldots, c_N = com_K(p^{j_N}; r_N), c'_1 = com_K(p^{j_2-j_1-1}; r'_1), \ldots, c'_N = com_K(p^{L-j_N-1}; r'_N)$.

Choose $d_1, \ldots, d_N$ such that they shadow $ep^{j_1}, \ldots, ep^{j_N}$, and γ such that it shadows $ep^{2j_1} + d_1 p^{j_1} + \ldots + ep^{2j_N} + d_N p^{j_N}$.

Choose $r_{d_1}, \ldots, r_{d_N}$ as random shadows of $er_1, \ldots, er_N$. Choose $r_{d_1 b}, \ldots, r_{d_N b}$ as random shadows of $-p(ep^{j_1}+d_1)r'_1+er_2, \ldots, -p(ep^{j_N}+d_N)r'_N + er_{N+1}$, where $r_{N+1} = 0$. Choose r_γ as a random shadow of $(ep^{j_1}+d_1)r_1 + \ldots + (ep^{j_N}+d_N)r_N$,and $r'_\gamma \in \mathcal{R}_{pk}$ as a random shadow of er_E.

Send $c_{d_1} = com_K(d_1; r_{d_1}), \ldots, c_{d_N} = com_K(d_N; r_{d_N})$, $c_{d_1 b} = com_K(d_1 p^{j_2-j_1}; r_{d_1 b}), \ldots, c_{d_N b} = com_K(d_N p^{L-j_N}; r_{d_N b})$,

$c_\gamma = com_K(\gamma; r_\gamma)$ and $E_\gamma = E_{pk}(d_1 p^{j_1} + \ldots + d_N p^{j_N} + \gamma \bmod n; r'_\gamma)$ to the verifier.

Challenge: Select e at random from $\{0, \ldots, 2^t - 1\}$.

Answer: Send $f_1 = ep^{j_1} + d_1, \ldots, f_N = ep^{j_N} + d_N, z_{1,1} = er_1 + r_{d_1}, \ldots, z_{1,N} = er_N + r_{d_N}, z_{2,1} = pf_1 r'_1 - er_2 - r_{d_1 b}, \ldots, z_{2,N} = pf_N r'_N - er_{N+1} - r_{d_N b}, z_3 = f_1 r_1 + \ldots + f_N r_N + r_\gamma, z_4 = er_E + r'_\gamma, D = e(M^{j_1} + \ldots + M^{j_N} + d_1 p^{j_1} + \ldots + d_N p^{j_N} + \gamma$ to the verifier.

Verification: Check that $c_{d_1}, \ldots, c_{d_n}, c_{d_1 b}, \ldots, c_{d_N b}, c_\gamma \in \mathcal{C}_K, E_\gamma \in \mathcal{C}_{pk}$, $f_1, \ldots, f_N, D \in \mathbf{Z}, z_{1,1}, \ldots, z_{1,N}, z_{2,1}, \ldots, z_{2,N}, z_3 \in \mathcal{R}_K$ and $z_4 \in \mathcal{R}_{pk}$. Verify that $com_K(f_1; z_{1,1}) = c_{d_1} c_1^e, \ldots, com_K(f_N; z_{1,N}) = c_{d_N} c_N^e$, $c_2^e c_{d_1 b} com_K(0; z_{2,1}) = (c'_1)^{pf_1}, \ldots, c_{N+1}^e c_{d_N b} com_K(0; z_{2,N}) = (c'_N)^{pf_N}$, $com(D; z_3) = c_1^{f_1} \cdots c_N^{f_N} c_\gamma$, where $c_{N+1} = com_K(p^L; 0)$. Finally check that $E_{pk}(D \bmod n; z_4) = E^e E_\gamma$.

Theorem 2 *The proof system above is a Σ-protocol proving that E encrypts a correct vote on multiple candidates. If the commitments are statistically hiding and the shadows and random shadows are statistically hiding, then the proof system is statistical special honest verifier zero-knowledge.*

Proof.
Completeness: Easy to see.
Special Soundness: Assume that we have two acceptable proofs to two different challenges e and e' to the same initial messages. This means we have answers $f_1, \ldots, f_N, z_{1,1}, \ldots, z_{1,N}, z_{2,1}, \ldots, z_{2,N}, z_3, D, z_4$ and $f'_1, \ldots, f'_N, z'_{1,1}, \ldots, z'_{1,N}, z'_{2,1}, \ldots, z'_{2,N}, z'_3, D', z'_4$ to the respective challenges satisfying the criteria specified in the verification step.

Starting with the encryption side of the proof we have

$$E_{pk}(D; z_4) = E^e E_\gamma \wedge E_{pk}(D'; z'_4) = E^{e'} E_\gamma.$$

This gives us

$$E_{pk}(D - D'; z_4 - z'_4) = E^{e-e'}.$$

Using the root opening assumption of the homomorphic cryptosystem we may now extract the plaintext of E. We call this plaintext v.

Going to the commitments we see that

$$com_K(f_1; z_{1,1}) = c_{d_1} c_1^e \wedge com_K(f'_1; z'_{1,1}) = c_{d_1} c_1^{e'}.$$

This gives us

$$com_K(f_1 - f_1'; z_{1,1} - z_{1,1}') = c_1^{e-e'}.$$

Using the root opening assumption on the commitment scheme we may from this extract an opening of c_1. In a similar manner we can extract openings of $c_2, \dots, c_N$. We call the contents of the commitments for $a_1, \dots, a_N$.

From the other part of the multiplication proofs we see that

$$c_{N+1}^{e} c_{d_N b} com_K(0, z_{2,N}) = (c_N')^{pf_N} \wedge c_{N+1}^{e'} c_{d_N b} com_K(0, z_{2,N}') = (c_N')^{pf_N'}$$

giving us

$$com_K(0; z_{2,N} - z_{2,N}') com_K(p^L; 0)^{e-e'} = (c_N')^{p(f_N - f_N')}.$$

We now know an opening of the commitment on the left hand side. We have $f_N' \neq f_N$ since $1 = com_K(0; 0)$, and the left hand side cannot be opened as zero by the binding property of the commitments. Accordingly we argue by the root opening assumption on the commitment scheme that we can extract an opening of c_N'. The opening must furthermore be non-zero since the left hand side opens to something non-zero. We can now in a quite similar manner go backwards finding non-zero openings of $c_{N-1}', \dots, c_1'$. We call the contents of the commitments for $b_N, \dots, b_1$.

We now have openings of the commitments $c_1, \dots, c_N, c_1', \dots, c_N'$ and E. Furthermore, by the binding property of the commitment scheme, these openings must be the only ones that the prover can produce. Therefore, we can now speak of *the* content of $c_1, \dots, c_N, c_1', \dots, c_N'$ and E in the rest of the proof.

What is left to argue is that the opening of the encryption satisfies the requirements of the proof. In that case, we have extracted a witness for the vote being on the correct form.

We get from

$$com_K(f_N - f_N'; z_{1,N} - z_{1,N}') = c_N^{e-e'}.$$

that

$$f_N - f_N' = a_N(e - e') \Rightarrow a_N = \frac{f_N - f_N'}{e - e'} \in \mathbf{Z}.$$

From

$$(c_N')^{p(f_N - f_N')} = com_K(0; z_{2,N} - z_{2,N}') com_K(p^L; 0)^{e-e'}.$$

we see that

$$p(f_N - f_N') b_N = (e - e') p^L.$$

This implies that

$$a_N b_N = p^{L-1}.$$

This means that $|a_N| = p^{j_N}$ where $0 \leq j_N < L$. In a similar fashion, we deduce $|a_1| = p^{j_1}, \ldots, |a_{N-1}| = p^{j_{N-1}}$ with $0 \leq j_1 < \ldots j_{N-1} < j_N$.

We proceed to the link between the commitments and the encryption. We have

$$com(D; z_3) = c_1^{f_1} \cdots c_N^{f_N} c_\gamma \wedge com(D'; z_3') = c_1^{f_1'} \cdots c_N^{f_N'} c_\gamma$$

implying that

$$com(D - D'; z_3 - z_3') = c_1^{f_1 - f_1'} \cdots c_N^{f_N - f_N'}.$$

Recall that for all i we have $a_i = \frac{f_i - f_i'}{e - e'}$. This means that the equation above gives us

$$D - D' = (e - e')(p^{2j_1} + \ldots + p^{2j_N}).$$

On the encryption side the equation

$$E_{pk}(D - D'; z_4 - z_4') = E^{e-e'}$$

shows that the content v satisfies

$$D - D' \equiv (e - e')v \bmod n.$$

Since $(e - e')$ is invertible modulo n we deduce that

$$p^{2j_1} + \ldots p^{2j_N} = v \bmod n.$$

In other words, the witness (v, r_E) consists of a correctly formed vote on the form $M^{j_1} + \ldots + M^{j_N}$, where $0 \leq j_1 < \ldots < j_N < L$, and the randomness involved in the encryption. This concludes the demonstration of the special soundness.

Special honest verifier zero-knowledge: Given the common input and a challenge $e \in \{0, \ldots, 2^t - 1\}$ we wish to simulate a proof of the encryption containing a vote on the right form.

We start by picking at random $r_1, \ldots, r_N, r_1', \ldots, r_N'$ from $\mathcal{R}_K$. We form the commitments $c_1 = com_K(p^0; r_1), \ldots, c_N = com_K(p^{N-1}; r_N), c_1' = com_K(1; r_1'), \ldots, c_N' = com_K(1; r_N')$. Due to the hiding property of the commitment scheme these commitments are indistinguishable from properly formed initial message commitments to $p^{j_1}, \ldots, p^{j_N}$ and $p^{j_2 - j_1 - 1}, \ldots, p^{L - j_N - 1}$.

We now pick $f_1, \ldots, f_N$ as shadows for $ep^{j_1}, \ldots, ep^{j_N}$ and D as a shadow for $f_1 p^{j_1} + \ldots + f_N p^{j_N}$. With this choice of $f_1, \ldots, f_N, D$ they are indistinguishable from the $f_1, \ldots, f_N$ and D of a real proof by the definition of shadows.

We may also pick $z_{1,1}, \ldots, z_{1,N}, z_{2,1}, \ldots, z_{2,N}, z_3 \in \mathcal{R}_K$ and $z_4 \in R_{pk}$ as random shadows so that they are indistinguishable from those in a real proof.

We compute $E_\gamma = E_{pk}(D; z_4)E^{-e}$ and $c_\gamma = c_1^{-f_1} \ldots c_N^{-f_N} com_K(D; z_3)$.

We set $c_{d_1} = com_K(f_1; z_{1,1})c_1^{-e}, \ldots, c_{d_N} = com_K(f_N; z_{1,N})c_N^{-e}$ and $c_{d_1 b} = com_K(0; z_{2,1})^{-1} c_1^{pf_1} c_2^{-e}, \ldots, c_{d_N b} = com_K(0; z_{2,N})^{-1} c_N^{pf_N} c_{N+1}^{-e}$.

With these choices, we have a simulated proof that due to the hiding property of the commitment scheme and the semantic security of the cryptosystem looks entirely like a normal proof with challenge e. This means that we have demonstrated the special honest verifier zero-knowledge property of the proof system.

Finally, we see from the proof of special honest verifier zero-knowledge that if the commitments $c_1, \ldots, c_N, c'_1, \ldots, c'_N$ are statistically hiding and that all the shadows and random shadows are statistically hiding, then the entire proof system is statistical special honest verifier zero-knowledge.

□

The possibility of a voter voting on less than N candidates is obtained by including N dummy candidates. If we remove the exponentiation to the power p and let $c'_1, \ldots, c'_N$ be commitments to $p^{j_2 - j_1}, \ldots, p^{L - j_N}$ instead, we get a proof system for the correctness and knowledge of the vote where the voter does not need to vote on different candidates. Here, p must, of course, be chosen large enough to accommodate for the larger number of votes a candidate can obtain.

So far, we have presented methods to make the zero-knowledge proofs that accompany an encrypted vote easy to form for the voter. On the server side, things are also much easier since the verification of these proofs is much easier than the more involved Σ-protocols used in [11]. We present a further speedup by presenting a randomized verification algorithm where we only need to compute one commitment instead of several of them.

One thing that is common in the verification procedure of the proofs we have presented above is that we compute two elements in $\mathcal{C}_K$ in two different ways, for instance as $com_K(f, z_1)$ and ac^e, and then, after this computation, we check whether they are identical. Since the computations involved in the computation of the two elements may be complicated, for instance requiring large exponentiations, we wish to reduce the time used in this process. When having many such pairs of elements, we may reduce the computational time involved in the verification of the proofs, taking advantage of the fact that we are working in a group.

Let us say we are given multiple pairs $(c_1, d_1, \ldots, c_n, d_n)$ in $\mathcal{C}_K$. We wish to check that the elements are pairwise identical. Choose $s_1, \ldots, s_n$ at random from $\{0, \ldots, 2^t - 1\}$. Here t may be a smaller security parameter than in the Σ-protocols since the computation happens only on the verifier's side and thus the prover is incapable of trying actively to cheat. Provided $\mathcal{C}_K$ is a group with no non-trivial elements of order less than 2^t we have with probability at least $1 - 2^{1-t}$ that $c_1^{s_1} \ldots c_n^{s_n} \neq d_1^{s_1} \ldots d_n^{s_n}$ if $\exists i : c_i \neq d_i$.

The reason why this is interesting is the homomorphic group structure of the commitments we are investigating. Note that the proofs we presented are in a form where one side has the form of a commitment $c_i = com_K(m_i; r_i)$, with m_i and r_i known to the verifier. Let us say that $c_1, \ldots, c_n$ are commitments. We can compute $c_1^{s_1} \ldots c_n^{s_n}$ as $com_K(s_1 m_1 + \ldots + s_n m_n; s_1 r_1 + \ldots + s_n r_n)$. If the binary operations in the groups $\mathcal{M}_K$ and $\mathcal{R}_K$ are faster to compute than the binary operations in $\mathcal{C}_K$ this makes verification more efficient.

Furthermore, depending on the groups in use we may take advantage of exponentiation techniques allowing us to compute $d_1^{s_1} \ldots d_n^{s_n}$ roughly at the price of one exponentiation. This was the emphasis in [5] where a somewhat similar technique for fast batch verification of signatures was investigated.

Since the probability of catching any cheating grows exponentially with t, we can typically choose t reasonably small. Accordingly, the extra computational effort required to compute the additional exponentiations to $s_1, \ldots, s_n$ is dwarfed by the savings we get by not having to verify each commitment opening by itself.

The technique is presented in quite general terms above since indeed it can be used in many contexts. Furthermore note that it works in all contexts where the message space is some group without small annihilators, not just where the message space is the integers.

In the voting scheme for multiple candidates, the verification procedure after some calculating becomes the following:

Verification: Check that $c_{d_1}, \ldots, c_{d_n}, c_{d_1 b}, \ldots, c_{d_N b}, c_\gamma \in \mathcal{C}_K, E_\gamma \in \mathcal{C}_{pk}$, $f_1, \ldots, f_N, D \in \mathbf{Z}, z_{1,1}, \ldots, z_{1,N}, z_{2,1}, \ldots, z_{2,N}, z_3 \in \mathcal{R}_K$ and $z_4 \in \mathcal{R}_{pk}$.

Select at random $s_1, \ldots, s_N, s'_1, \ldots, s'_N, s \in \{0, \ldots, 2^t - 1\}$. Verify that

$$com_K(s_1 f_1 + \ldots + s_N f_N + sD + es'_N p^L; s_1 z_{1,1} + \ldots + s_N z_{1,N}$$

$$
\begin{aligned}
&+s'_1 z_{2,1} + \ldots + s'_N z_{2,N} + z_3) \\
&= c_\gamma^s c_{d_1}^{s_1} \cdots c_{d_N}^{s_N} (c_{d_1 b}^{-1})^{s'_1} \cdots (c_{d_N b}^{-1})^{s'_N} c_1^{es_1+sf_1} c_2^{e(s_2-s'_1)+sf_2} \cdots \\
&\quad c_N^{e(s_N-s'_{N-1})+sf_N} (c'_1)^{pf_1 s'_1} \cdots (c'_N)^{pf_N s'_N}
\end{aligned}
$$

Finally, check whether $E_{pk}(D \bmod n; z_4) = E^e E_\gamma$.

4 Securing an Implementation in Practice

We have had the opportunity to work with practical aspects of implementing an e-voting system in connection with the EU project, e-Vote. Several challenges beyond the scope of the cryptographic protocols have been identified and solutions have been found. These challenges are partly due to security aspects special to voting solutions, which cannot be solved by technical means alone, and partly due to standard problems with providing a satisfactory combination of security and usability of the authentication mechanisms used. We dedicate some subsections to the individual problems and solutions.

We use the bulletin board model. All entities, persons as well as servers, will have at least one public/private key pair to enforce the model. However, we will only include the aspects of PKI having to do with authentication of voters here.

We do not propose a total solution, but give solutions to sub-problems, some of which can be adopted for any particular voting system according to relevant tradeoffs for each individual system.

4.1 Taking Requirements Seriously.

In the newspapers, a considerable amount of the debate on electronic voting is dedicated to suggestions for voting systems tailored at selling large amounts of expensive equipment of one kind or the other. Examples are chip-cards and biometric devices.

It is, however, our belief that the competition will rapidly make such approaches obsolete. In order to have success with voting technology, it has to be tailored to meet the requirements of election organizers and voters rather than those of the vendors.

Shortly expressed, voting systems are like all other systems. In order to implement a successful system it is not only necessary to understand the requirements correctly; it is also necessary to respect them.

4.2 Deployment of a PKI.

After having stated some concerns in Section 4.1, we must, however say that we see no alternative to using PKI for authentication of voters. The main reason is that unless a public/private key pair is used, anybody who can verify authentication information can also fake it. In particular, universal verifiability of an election with a decent level of security is very significantly simplified by using PKI.

If a public PKI is in place and most voters have signature keys, it will be most natural to use that PKI. In practice, this is usually not the case today though.

We are working with two approaches to overcome the limitation of a potentially lacking PKI:

- Having the voters generate one-time key pairs on their web browsers and having certificates on those keys issued on-line. In practice we work with the model that each voter receives a cryptic user identity and a one-time password, based on which the certificate is issued on-line. The user identity and the password must be received through two different channels in order to provide a decent level of security. We consider two physical letters with some days in between as the most realistic option.

- Using a virtual chip-card. This means that the keys of the voters are stored in secure hardware by a pair of trusted organizations. Usage of the keys can be requested by providing two means of authentication to two servers located in different organizations. Again, both means of authentication can be very cheap and simple-minded.

As for the approaches mentioned above, the first one is appropriate if there is a long period of time between elections, and the PKI is not used for other purposes. The last one is appropriate if regular elections tke place or the PKI is to be used for other purposes as well. The last approach has the advantage that voters already in the system can be notified about new elections by means of an insecure email only. Thus the cost of arranging additional elections is very low.

4.3 Protection against Hackers.

In [11] Damgård and Jurik proposed a scheme for protecting internet voters against hackers. We will understand the word hacker in a broader sense so that it includes system administrators, who can completely legitimately observe and control computers of voters remotely, as well as hackers breaking in without permission.

The proposed solution is to provide each voter with a paper ballot with a list of candidates listed in some natural ordering, and, in addition, numbered corresponding to a permutation π of the candidates. The voter then enters the number $\pi(c)$, where c is the number of the candidate selected according to the natural ordering.

If a hacker observes the voting process, he will not gain any information about the candidate chosen, even if he has full control of the computer of the voter. Furthermore, if he tampers with the vote, the outcome will be uniformly distributed on all candidates.

Combining this protection with homomorphic encryption in an efficient way is quite difficult. The scheme suggested in [11] for the generalized Paillier system is too slow to be feasible with the current band-with on the Internet and performance of computers.

In [16] we will propose another scheme, where we trade security and performance. In short, by restricting the possibilities of the hacker slightly less than for the original scheme, performance properties of the integration with the homomorphic crypto system improves sufficiently to make this sort of protection feasible.

The paper ballot with permutations can also contain one piece of authentication information and possibly more.

4.4 Server Authentication.

Server authentication is normally obtained by a SSL connection between a web server and a web browser.

Technically, this works well, but in practice most web browsers are wrongly configured and most voters will be unable to tell, whether a server has been correctly authenticated or not.

As a solution to this problem we propose that the paper ballot with permutations and one piece of authentication information shall also contain a piece of graphics, different for each voter. Furthermore, it will include instructions for the voter about how to verify that the same graphics appears on the web page from where he votes.

When the voter enters the first piece of authentication means, he will be confronted with some graphics on the screen. If it is not identical to the graphics on his paper ballot, he will have instructions to exit the faked web server.

4.5 Voters Being Looked over the Shoulder.

A concern that for example journalists have expressed to us, and that we will have to take very seriously, is that of a voter being looked over the shoulder while he/she votes. The person looking over the shoulder can, for example, be a husband or an employer. This problem is not solved by the solution that protects against hackers because also physical items can be seen by the person looking over the shoulder.

The best solution to this problem that we have encountered was suggested by the local community of Høje Tåstrup, a suburb of Copenhagen, which had worked with the problem in connection with an early voting pilot. The solution is to provide a facility where the voter can go discretely to have his/her electronic vote replaced by a manual vote. In a transition period, where manual voting (voting at election sites) exist side by side with Internet voting, this can be by providing the opportunity for voters to vote at election sites before and after they have cast their Internet vote.

In order to integrate this with a voting scheme based on homomorphic encryption and protection against hackers, cancellation/replacement of votes must be implemented in such a way that it cannot be detected, which votes have been replaced. When this is combined with universal verifiability, the need for new cryptographic primitives arises. We will treat this subject in a separate paper.

4.6 Long Term Privacy.

The universal verifiability means that anybody can connect each voter to the ciphertext. Security is based on the assumption that it is infeasible to decrypt the ciphertext and see what the voter has voted. In order to protect the privacy of the voter, not just at the time of the election but also several years into the future, the keys used for the cryptosystem must be large.

For the same reason, we suggest proving the correctness of the vote using a zero-knowledge proof that is statistical zero-knowledge. If this suggestion is followed, the zero-knowledge proof will not reveal which vote has been cast even if the commitment scheme is broken.

We summarize this: The key used for the homomorphic crypto system must be sufficiently strong to be supposed to remain unbroken for an extended period, whereas the strength of the key for the commitment scheme will only have to be strong enough to remain unbroken for a shorter period, provided that the zero-knowledge proofs are statistical zero-knowledge.

4.7 Legal Considerations.

Most countries have rather precise regulations, specifying how public elections of various types must be performed. Thus laws, but usually not constitutions, may have to be changed before an electronic voting system can be used in elections covered by these laws. For elections performed internally in an organization other than a state, similar challenges may be encountered - parts of the internal rules of the organization may have to be changed.

Today, most advanced countries have a signature law. It seems to be a wise decision to study, which messages in a voting system must be secured in particular ways in order to make the decisions imposed by election organizers legally binding. For example, in order to provide non-repudiation, it may be necessary to have some messages independently time stamped.

This can be reformulated in the way that the system must be designed so that predictable conflicts can be resolved successfully in court using the local signature law.

We refer to [19] and the national signature laws for more details.

References

[1] Abe: *Universally verifiable MIX net with verification work independent of the number of MIX centers*; proceedings of EuroCrypt 98, Springer Verlag LNCS.

[2] Boudot: *Efficient Proof that a Committed Number Lies in an Interval*, Proc. of EuroCrypt 2000, Springer Verlag LNCS series 1807.

[3] J. Bar-Ilan, and D. Beaver: *Non-Cryptographic Fault-Tolerant Computing in a Constant Number of Rounds*, Proceedings of the ACM Symposium on Principles of Distributed Computation, 1989, pp. 201-209.

[4] Baudron, Fouque, Pointcheval, Poupard and Stern: *Practical Multi-Candidate Election Scheme*, manuscript, May 2000.

[5] Bellare, Garay, Rabin: *Fast Batch Verification for Modular Exponentiation and Digital Signatures*; proceedings of EuroCrypt 98.

[6] B. Schoenmakers: *A simple publicly verifiable secret sharing scheme and its application to electronic voting*, Advances in Cryptology - Crypto '99, vol. 1666 of LNCS, pp. 148-164.

[7] R. Cramer, I. Damgård and J. Nielsen:*Multiparty Computation from Threshold Homomorphic Encryption*, Proceedings of EuroCrypt 2001, Springer Verlag LNCS series 2045, pp.280-300.

[8] R. Cramer, M. Franklin, B. Schoenmakers & M. Yung: *Multi-authority secret ballot elections with linear work*, Advances in Cryptology - EuroCrypt '96, vol. 1070 of LNCS, pp. 72-83.

[9] R.Cramer, R.Gennaro, B.Schoenmakers: *A Secure and Optimally Efficient Multi-Authority Election Scheme*, Proceedings of EuroCrypt 97, Springer Verlag LNCS series, pp. 103-118.

[10] Damgård and Fujisaki: *An Integer Commitment Scheme based on Groups with Hidden Order*, Manuscript, 2001, available from the ePrint archive.

[11] Damgård and Jurik: *A Generalisation, a Simplification and some Applications of Paillier's Probabilistic Public-Key System*, Proc. of Public Key Cryptography 2001, Springer Verlag LNCS series.

[12] Damgård and Jurik: *Client/server tradeoffs for online elections*; proceedings of PKC'02.

[13] A. Fujioka, T. Okamoto & K. Otha: *A practical secret voting scheme for large scale elections.*, Advances in Cryptology - AusCrypt '92, pp. 244-251.

[14] Fujisaki and Okamoto: *Statistical Zero-Knowledge Protocols to prove Modular Polynomial Relations*, proc. of Crypto 97, Springer Verlag LNCS series 1294.

[15] Oded Goldreich, Silvio Micali, and Avi Wigderson: *How to play any mental game or a completeness theorem for protocols with honest majority*, in *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, pages 218–229, New York City, 25–27 May 1987.

[16] J. Groth, G. Salomonsen: *A practical Protocol for protecting Internet Voters against Hackers*, Work in progress.

[17] M.Hirt and K.Sako: *Efficient Receipt-Free Voting based on Homomorphic Encryption*, Proceedings of EuroCrypt 2000, Springer Verlag LNCS series, pp. 539-556.

[18] Lipmaa: *Statistical Zero-Knowledge Proofs from Diophantine Equations*; Cryptology ePrint Archive, Report 2001/086.

[19] Mitrou, Gritzalis, Katsikas, S. *Revisiting legal and regulatory requirements for secure e-voting.* Proc. of the 16'th International Information Society Conference (IFIP/SEC-2002) M. el Hadidi, et al. (Eds.), Egypt, 6-8 May 2002. Kluwer Academics Publishers.

[20] Ohkubo and Abe: *A Length-Invariant Hybrid Mix* Proceedings of AsiaCrypt 00, Springer Verlag LNCS.

[21] P.Pallier: *Public-Key Cryptosystems based on Composite Degree Residue Classes*, Proceedings of EuroCrypt 99, Springer Verlag LNCS series, pp. 223-238.