E-Voting without 'Cryptography'

(Extended Abstract)

Dahlia Malkhi, Ofer Margo and Elan Pavlov

School of Computer Science and Engineering
The Hebrew University of Jerusalem, Israel
{dalia,omargo,elan}@cs.huji.ac.il

Abstract. We present a new, distributed e-voting protocol, which does not require any 'conventional' cryptography, or other means of non-trivial math, and is information-theoretically secure. In our system, the voter receives a receipt confirming her vote, and does not need to trust any pollster to correctly encrypt her vote, or convey it to the talliers. The system can withstand corruption of almost half the number of authorities running the elections. To achieve these goals, we present *enhanced check vectors*, a new weak signature scheme as well as a new protocol for joint creation of weak signatures by multiple servers.

1 Introduction

The need for conducting electronic elections received reiterated emphasis in the recent U.S. presidential elections, and the wide-spread dispute it caused over the Bush-Gore results. There is a clear desire for a technology which makes it easy for every citizen to cast his/her vote and verify the election results, while eliminating the risk of fraud or ambiguous ballots [CM01].

Such electronic voting schemes are required to guarantee the privacy of voters, the verifiability of the results, to prevent ballot stuffing and to ensure that all eligible voters are able to vote. However, all computer-based voting systems rely on some piece of software, called the *pollster*, to interact with the human voter in order to accept his/her vote in the clear. The pollster then transmits the vote (encrypted, broken into pieces, etc.) to the voting system. The easiest attack on any electronic voting system is to modify the pollster software. For example, the pollster may be used for snooping to votes if it is installed on the home PC of voters. Or, it can be used to alter or omit undesired votes (simply by 'pretending' that the voter punched different keys). As pointed by various state and federal committees, [IPI01,**?**] this limitation stands in the way of deployment of electronic voting, particularly over the Internet, as a pollster would need to be installed on the untrusted environment of home PCs.

In this paper, we present an electronic voting scheme that does not require conventional cryptography on the users' side, and hence, does not require a trusted pollster. The solution employs two novel tools: One is an extension of Rabin and Ben-Or's *check vectors* [RB89], which allows one party to hand another party a secret that is guaranteed to be accepted by a group of verifiers. The other is a full fledged electronic voting protocol that makes use of enhanced check vectors and an *anonymous multiparty computation* (AMPC) [MP01] in order to provide a user with a cryptography-free receipt upon voting. The receipt proves to the voter that his/her vote is recorded correctly - no pollster (or any other component) is capable of forging such a receipt if the vote was modified. Our solution decentralizes all phases of the voting process, including registration, such that no collusion of a threshold of authorities can compromise the accuracy of the tally or the privacy of users.

In addition to the usual properties of privacy, democracy, verifiability and accuracy, our approach has the following desirable features:

– It does not use any conventional means of cryptography, polynomial secret sharing, or other non-trivial math. Instead, we rely on AMPC and a new weak signatures primitive, enhanced check vectors, to achieve our goals. The voter is not required to perform complicated computations, and needs only perform $O(b)$ multiplications and additions in order to verify that she holds a valid voting credential, or that her ballot was correctly accepted by the talliers (where b is a security parameter).
– The system can withstand up to $b = \lceil \frac{m}{2} \rceil - 1$ corrupt authorities, where m is the number of talliers or registrars. In particular, we distribute the registration process so that a coalition of b or fewer registrars are not able to compromise voters privacy or to stuff the ballot box.
– The privacy provided by our protocol is information-theoretic, not computational.
– There is no single trusted piece of software that needs to receive a voter's ballot in the clear. Hence, no piece of software can pretend that a different key was "punched" by the voter undetectably.
– A voter can verify that she holds a valid voting credential before the elections and can verify that her vote was properly accepted by the talliers during the election process (that is, a voter gets an election *receipt*). This prevents a faulty pollster from modifying or obstructing the vote.
– A pollster by himself can not tell which candidate the voter voted for (even if he is able to monitor and record all of the voter communication during election day). A pollster requires the cooperation of at least one tallier in order to know which candidate the voter voted for. [1]
– The efficiency of our scheme can be summarized as follows: When there are m tallying and m registration authorities, n voters, and for a security parameter $b \leq \frac{m}{2} - 1$, such that the system can withstand up to b corrupt authorities, the total amount of communication is $O(nb^2)$ during the election phase, and $O(nmb)$ during the registration phase. More specifically, the voter holds one ballot of length $b + 2$, which on election day she needs to send to any b different talliers. Each tallier is required to perform $O(nb)$ multiplications and additions during the election themselves. The voter may perform $O(b^2)$

[1] Our protocol could be enhanced so that a pollster cooperating with up to b talliers would not be able to know which candidate the voter voted for, by splitting each ballot among talliers. For simplicity, this is not pursued further in this paper.

multiplications and additions in order to verify that she holds a valid ballot, or that her ballot was properly received by the talliers. (The computation required is $O(b)$ multiplications and additions per tallier). Other servers may need to perform up to $O(nmb)$ multiplications and additions during the registration stage.

Technical approach: In order to provide cryptography-free ballots and receipts, we make use of a novel information-theoretic checking protocol that extends the *check vectors* of Rabin and Ben-Or [RB89]. In their information checking protocol, there are three parties: A dealer DLR, a receiver RCV and an intermediary INT. INT obtains a secret V from DLR, and would like to give it at a later time to RCV. RCV should be able to confirm that the secret V indeed originated with DLR, in order to accept the secret. Also, INT should know with high probability whether RCV will accept the secret or not.

Our *enhanced check vectors* protocol extends the check vector mechanism in two ways. First, it accommodates a group of receivers such that each receiver can verify independently that the secret indeed originated with DLR, however, no collusion of b or fewer of the receivers can reconstruct the secret V using the information (check vectors) they hold. The intermediary uses the same secret with all receivers, and (unlike in [RB89]) does not need to maintain a separate version of the secret for each potential receiver. Second, we show how the secret can be jointly generated and distributed by multiple dealers, such that intermediaries remain anonymous and such that a collusion of b or fewer of dealers and receivers are not able to recover secrets, forge them, or relate them to intermediaries. We employ an anonymous multiparty computation (AMPC) [MP01] to accomplish the last part.

We proceed to show how the enhanced check vectors protocol can be used in a distributed on-line election protocol whose essence is as follows: Before the elections, each voter receives one vote vector V for each possible ballot value s, e.g., one for Bush and one for Gore. For each vote vector issued to a voter, a corresponding check vector B is issued to each tallier. On elections day, the voter anonymously sends the vote vector V corresponding to her desired ballot s to a minimum of $b+1$ different talliers. Each tallier verifies that $s = BV$ is in S to confirm the eligibility of the voter to vote.

Using the enhanced check vectors protocol, a voter may verify before the elections that any vector it holds is valid and will be accepted by talliers. Moreover, the voter makes use of check vectors to obtain a receipt for her vote. This is achieved as follows. For each possible vote value s, a voter is issued several pairs of vote vectors, $\{\langle V_{1,0}, V_{1,1}\rangle, \langle V_{2,0}, V_{2,1}\rangle, ...\}$. Before the elections, the voter sends one of the vectors, e.g., $V_{1,0}$ to all the talliers, and expects to receive in return the check vector $B_{1,1}$ corresponding to the second vote vector in the pair, $V_{1,1}$. She then verifies that $V_{1,1}B_{1,1} = s$. Receipt delivery during the elections is done in the same manner.

We note that while none of the elements in the protocol requires the use of cryptography, we work in the classic *secure channels* model that requires that the different authorities participating in the protocol should be able to communicate with one another and with the user in a secure manner. In practice, cryptography may be deployed in order to secure the links, or they may be secured by benign means (e.g., regular mail, telephone).

The rest of the paper is organized as follows: In section 1.1 we discuss Related Work. In section 2 we describe the enhanced check vectors protocol and the mechanism used to jointly generate and distribute the secret vote vectors and check vectors. In section 3 we describe the actual voting protocol. We then follow with a security analysis of our work (Section 4) and some concluding remarks (Section 5).

1.1 Related Work

In most general terms, electronic voting protocols can be divided into two categories, centralized and decentralized.

Centralized voting schemes rely on the existence of an anonymous communication channel between voters and either the registration or the tallying authorities in order to maintain voters privacy. The first to propose a method utilizing anonymous communication for electronic voting was Chaum, who has suggested in 1985 [Cha85] the use of *blind signatures* as a tool for privacy in electronic cash and electronic voting systems. The first actual voting protocol employing blind signatures appeared in 1992 [FOO92] and the first published implementation in 1997 [CC97]. A somewhat improved version was offered by He & Su in 98 [HS98]. The basic idea underlying all of these schemes is to employ two logical authorities, a registrar and a tallier. The registrar validates eligible voters and provides them with anonymized certified voting tags, using blind signatures. The voters then cast their (blindly signed) ballots over anonymous communications channels to the tallier. Some of the problems exhibited by these methods that our approach addresses are:

- The voters must fully trust a *pollster* - a piece of software that carries out the (non trivial) encryption functions and anonymous communications for them.
- The registrar, or the registrar in collusion with the tallier, can stuff the ballot box with invalid votes. We note that the idea of using multiple registrars to address this problem has been suggested before by DuRette in [DuR99].
- They require the use of anonymous channels which are costly to maintain, and also exhibit various difficulties in reality (see [WALS02] for a good discussion of the degradation in anonymity protection of various anonymizers).

The second category of electronic voting schemes, decentralized methods, includes various secret-sharing based schemes (e.g., [CFSY96,CGS97,**?**]) (most of them based on the extrapolations of polynomials). In such schemes, a collection of authorities are employed to reduce the trust placed in any single (tallier) authority. The paradigm works as follows: At the voting stage, voters split their ballots into shares using secret sharing techniques, and secretly send each such share to a single authority. At the tallying stage, each authority separately combines the shares of all ballots into a share of the result. Using the homomorphic properties of the secret sharing techniques, the authorities then jointly compute the results of the elections, such that a collusion of fewer than a threshold of the tallying authorities does not reveal information about any single ballot. As these schemes do not rely on anonymous channels, they are quite practical and are in fact deployed in a commercial system, e.g. `VoteHere.Net`. Two of the above problems remain here as well:

- Voters need to trust a pollster that will carry out the necessary encryption and secret sharing for them.
- The issue of registration is generally left outside the scope of such systems, and thus the problem of the registration authority stuffing votes remains.

A different kind of voting method that is not included in either of the categories above is the decentralized protocol in [DmLM82]. This method works without involving any external authority, and moreover, prevents stuffing, but it assumes that voters can know and verify each other and 100% voter participation. Furthermore, it requires a number of rounds of communications and computations among all participants, which is linear in the number of participants. It is therefore not feasible to deploy it in a realistic, large scale elections.

The information checking protocol and check vectors that our approach is based on were first introduced by Rabin and Ben-Or in [RB89] in the context of verifiable secret sharing and secure multiparty computation. Their basic version of check vectors supports only one receiver, who is able to verify the secret as follows. The DLR hands out a pair (s, y) to INT, and a pair (b, c) to RCV (the check vector). RCV verifies when receiving the secret that $s + by = c$. In this (basic) method, multiple receivers using different check vectors are able to recover the original secret by linear extrapolation. Rabin & Ben-Or introduce an extension which supports multiple receivers but requires INT to use a different random value, y_i, for each potential receiver (each receiver verifies $s + b_i y_i = c_i$). Translated to our setting, this would imply that a voter holds a different ballot for each potential tallier. In contrast, in our scheme the voter holds only one ballot, whose length is determined solely by the security parameter.

Franklin and Yung [FY94] treat check vectors as a form of *weak signatures* (signatures that can be verified only by certain checking centers) and show how the protocol can be extended to support blind weak signatures. They proceed to show how these weak signatures can be used for digital cash. They also mention the possibility of using multiple receivers (*checking centers*), using Rabin & Ben-Or's extension to the check vector protocol, and using a majority agreement between the checking centers, in order to decide whether the input (the secret) is valid or not.

AMPC is a new approach for Anonymity without Cryptography, suggested by Malkhi & Pavlov [MP01]. In an AMPC, there are several levels (rows) of players, each row basically serving as the analog of one MixNet server [Cha81]. To use AMPC, users split their input into several shares, and submit each share to one of the players in the the first (top) row. The top row players permute the input shares they receive (using an agreed-upon permutation). Then each player splits his share into several sub-shares, and passes each sub-share to a player in the second level. Each player in the second level combines the sub-shares it received into one share, permutes its shares and again passes split shares to the players in the third level. The process continues until all players in the last (bottom) level send their shares to the receiver, which combines the shares to obtain the original input values. Malkhi and Pavlov briefly discuss in [MP01] an election protocol, which can be used with AMPC. However, their protocol does not enable voters to verify that talliers properly received their ballots during the elections, neither does it enable voters to verify that they hold valid election credentials before the elections. Thus, their scheme does not fully support the requirement of democracy (guaranteeing that each voter is able to vote). Also, their protocol is relatively complicated compared to ours, and requires the talliers and registrars to perform commitments on the voter credentials and ballots before they are computed and published.

2 The enhanced check vectors protocol

In the enhanced check vectors protocol we have a dealer DLR, who would like to give a secret V with a meaning s to intermediary INT, where s is chosen from a small set of possible meanings S (relative to the size of Z_q). At a later time, INT would like to forward the secret to one or more receivers, RCV_1, ..., RCV_m. Each one of the receivers should be able to independently verify the secret meaning and the fact that it indeed originated with DLR. However, a subset of b or fewer receivers (where b is a security parameter of the system) should not be able to reconstruct the secret V. In addition, INT should be able to verify, with high probability, that the receivers would accept her secret.

The basic protocol proceeds as follows (all additions and multiplications are done in Z_q):

- To hand over a secret V with a meaning s, the dealer generates a random vector V of minimal length $b+1$ and gives it to INT. It also generates a group of check vectors $\{B_1..B_m\}$, one vector for each potential receiver. The check vectors are generated so that $VB_i = s$ (scalar product) for all $1 \leq i \leq m$,
- To give V to a receiver, INT sends V to the receiver. Assuming that the receiver holds a check vector B for this secret, it computes $s = BV$ and verifies that s is in S.

In order to enable INT to verify that her secret would be accepted by RCV, we do the following: For each meaning s, the dealer issues several pairs of vectors as described above, say $\{\langle V_{1,0}, V_{1,1}\rangle, \langle V_{2,0}, V_{2,1}\rangle, ...\}$ to INT, and gives each receiver corresponding pairs of check vectors, $\{\langle B_{1,0}, B_{1,1}\rangle, \langle B_{2,0}, B_{2,1}\rangle, ...\}$. All vectors and corresponding check vectors satisfy $V_{k,j}B_{k,j} = s$. To confirm that it holds valid vectors with high probability, INT chooses some $V_{k,j}$ at random, sends it some of the receivers, asks them to reveal the adjoining check vector, $B_{k,((j+1) \mod 2)}$, and verifies $V_{k,((j+1) \mod 2)}B_{k,((j+1) \mod 2)} = s$. It also informs all receivers that the k'th pair is hence invalid for use.

One should note that unlike the group check vectors of [RB89], the amount of information INT is required to hold is proportional to the security parameter b. This is manifested in the length of the vectors V it holds, and reflects the minimum number of colluding receivers which are able to recover the secret using their check vectors. It is not affected by the *potential* number of receivers, which may be higher. Also, in our protocol INT is not required to remember a different value for each receiver, and all of the check vector coordinates are used for verification by all receivers.

2.1 A distributed dealer

We would like to extend our protocol so that the secret V will be jointly assigned by several dealers, and such that any subset smaller than our security threshold is unable to reconstruct it. An additional requisite we have, which is driven by our voting application, is that no collusion of (a threshold of) dealers and receivers will be able to correlate the vectors V or the check vectors to the intermediaries that use them. The intuitive reason for this is that the vectors V are used in our setting as voting credentials. They are initially distributed by the dealers to voters (hence, the dealers must know who they are). However, during elections, the vectors should be used anonymously for voting.

A trivial distribution protocol that one may think of is to let the i'th dealer select the i'th coordinate of the secret V and the i'th coordinate of each of the corresponding check vectors. However, this would maintain a strong correlation between vectors and intermediaries: $log(n)$ collaborating dealers know enough coordinates of each check vector to identify it, with high probability, from a list of vote vectors received by a colluding receiver, and thus they may jeopardize voters anonymity.

In order to anonymize check vectors, we make use of an anonymous multi-party computation (AMPC) [MP01]. The idea is to generate each coordinate in V jointly by dealers and garble all vectors through AMPC so as to prevent correlating the generated value with the intermediaries that received its shares from the dealers. More precisely, recall the definition of an AMPC:

Definition 1. *[MP01]*
Let the input consist of n tuples each consisting of m values from a domain D: $\{\langle X_j^1, X_j^2, ..., X_j^m\rangle\}_{1\le j\le n} \subseteq D^m$. Let a function f be from domain D^m to some domain D'. Let $P^1, \ldots, P^m$ be a set of players. Suppose that each P^i initially knows only $\{X_j^i\}_{1\le j\le n}$. A protocol is an anonymous multi party computation *of f on the inputs $\{X_j^i\}$ with robustness b if it calculates the set of values $\{f(X_j^1, X_j^2, \ldots, X_j^m)\}_{1\le j\le n}$ such that for any set of players G, $|G| \le b$, and any function ϕ, there exists a function ψ such that:*

$$Pr[\phi(\{f(X_j^1, X_j^2, \ldots, X_j^m)\}_{j=1..n}, AMPC^G) = (X_j^1, X_j^2, \ldots, X_j^m)] \le$$
$$Pr[\psi(\{f(X_j^1, X_j^2, \ldots, X_j^m)\}_{j=1..n}) = (X_j^1, X_j^2, \ldots, X_j^m)] \ .$$

The definition above uses $\phi(I, AMPC^G)$ to denote the computation of $\phi(I)$ using any internal values known to G during the AMPC (for any ϕ, $\phi(I)$ alone denotes the computation of ϕ without knowledge of any such internal values). For completeness, a realization of AMPC is depicted in Figure 1 in the appendix.

In our protocol, the function f is simply addition, and the m values in each tuple are the m additive shares $v_{ij}, j = 1..m$ issued by the m dealers, of the v_i coordinate in the secret vector V. For a security parameter b, we require that the minimal length p of V and the corresponding check vectors will be $b + 2$.

The multi-dealer information checking protocol is then as follows:

1. For each intermediary, the j'th dealer DLR$_j$ issues additive shares $v_{1j}, ..., v_{pj}$ of each of the p coordinates of vector V. For purposes of correlating V with its check vector, the j'th dealer also issues an additive share vid_j of a voter-id vid.
2. The intermediary computes the coordinates of her secret vector V, $v_1, ..., v_p$, and her voter-id vid by summing up the shares received from the different dealers: $v_i = \sum_{j=1..m} v_{ij}$, $vid = \sum_{j=1..m} vid_j$.
3. We now employ an AMPC p times as follows. For each coordinate i, $1 \leq i \leq p$, and all intermediaries, the j'th dealer provides v_{ij}, vid_j as input to the j'th AMPC top player (P^j). The AMPC then produces a permuted list of i'th coordinates $v_i = \sum_{j=1..m} v_{ij}$ of all intermediaries along with their corresponding vid. This list is sent securely to an authority that we call A_i (this could be an auxiliary authority, or one of the talliers/registrars). The result of these p AMPCs is that there are p authorities $A_1, ..., A_b$ that each holds $\langle v_1, vid\rangle, ..., \langle v_p, vid\rangle$, respectively, for every intermediary, such that there is no known correlation between the pairs $\langle v_i, vid\rangle$ and the intermediaries known to the dealers. [2]
4. The p authorities now together compute check vectors for V as follows (we present the protocol for computing one check vector for one receiver; the check vectors for the rest of the receivers are computed in a similar manner).
   (a) A_1 selects a random number r_1 and passes it to A_2. In addition, A_1 splits r_1 into p shares, $r_{11}, ..., r_{1p}$, such that $r_{11} + r_{12} + .. + r_{1p} = r_1$, and sends r_{1k} to A_k.
   (b) From here on, the k'th authority A_k, for $2 \leq k \leq p$, selects randomly the k'th component of the check vector, b_k, as well as a random number r_k. It passes b_k, vid to the receiver. It then computes $c_k = c_{k-1} + b_k v_k + r_k$ and sends c_k to A_{k+1} (the last authority A_p passes back to A_1). In addition, A_k splits r_k into p shares, $r_{k1}, ..., r_{kp}$ and sends r_{ki} to A_i.
   (c) Every A_k computes $r'_k = r_{1k} + r_{2k} + .. + r_{pk}$ and sends it to A_1. A_1 computes b_1 as follows: $b_1 = (s - (c_p - \sum r'_k))/v_1$. This computation is feasible if we are working in Z_q, for a prime q, in $O(log(q))$ time (see

[2] To be precise, we should add intermediary-indices to vector components and vids. Thus, the AMPC is used to generate lists of values, e.g., i'th vector coordinates $\{v_i^{INT}\}_{INT}$ for all voters INT, and likewise, voter-id's $\{vid^{INT}\}_{INT}$ for all voters. For simplicity, we omit these indices from the description.

e.g., [CLR89]). This guarantees that $\sum b_i v_i = s$. It then sends b_1, vid to the receiver.

At the end of the enhanced check vectors protocol, each receiver RCV$_j$ holds a list of check vector pairs $\langle vid, B\rangle$ that each corresponds to an anonymous intermediary holding $\langle vid, V\rangle$ such that $VB = s$. This protocol may be repeated multiple times to generate a list of such pairs per vote (as indeed is done to facilitate testing and receipts).

3 Voting Protocol

3.1 Review

The voting protocol is composed of the following phases:

1. Registration - during this stage, the voters identify themselves to $b + 1$ registrars (using IDs established by other means, such as passports or national identity cards), and receive their vote vectors. A voter receives her vote vectors for each possible ballot s. We employ the enhanced check vectors protocol here, so a vote vector corresponds to the secret V that an intermediary INT (the voter) receives from the dealer (registrars), and the check vectors B for a vote vector are computed and distributed to the receivers (talliers).
2. Testing - during this stage the voter verifies (with high probability) that the vote vectors she receives from the registrar(s) would indeed enable her to vote.
3. Voting - The voter sends the vote vector V that corresponds to her desired ballot s to some of the talliers. Each of these talliers holds a corresponding check vector B for that vote vector and verifies that indeed $VB = s$. During this stage the voter is also able to verify that her vote was received properly by the talliers by checking a receipt she receives from each tallier.
4. Counting - each tallier publishes all the ballots it received, and everyone can verify the election results.

3.2 The voting protocol

We now describe the voting protocol in more detail.

Voter Registration The enhanced check vectors protocol of Section 2 is employed by the group of registrars (the dealers) to issue all voters their anonymous secret vectors V, secret meaning s, secret voter-ids vid, and to issue corresponding anonymous check vectors to talliers. For each ballot s, each voter is issued several pairs of vectors $\{\langle V_{1,0}, V_{1,1}\rangle, \langle V_{2,0}, V_{2,1}\rangle, ..., \}$ in order to facilitate testing and receipts.

Testing During this stage the voter verifies that talliers indeed hold a valid check vector for his vote. As mentioned, each voter holds several pairs of vectors for each candidate. She randomly selects one of the vectors, say $V_{1,0}$, and sends it to several talliers. Each tallier sends back the check vector corresponding to the adjoining vector in the pair, in our case $B_{1,1}$. The voter verifies that $V_{1,1}B_{1,1} = s$. She should also publish the vote vector pair she used for verification in order to disqualify it for use in the elections.

Voting The voter anonymously sends one vote vector designating her ballot, along with her *vid*, to at least $b+1$ different talliers. [3] Each tallier verifies that the vote vector received, say $V_{i,1}$, satisfies $V_{i,1}B_{i,1} = s$, and s is in S. The adjoining vote vector in the pair that was used serves as the verification vector. That is, when a tallier receives a vote vector, say $V_{i,1}$, it sends back as a receipt the check vector $B_{i,0}$. The voter then verifies $V_{i,0}B_{i,0} = s$ to verify $B_{i,0}$'s authenticity.

Counting At the end of election day each tallier publishes each vote vector it received along with its *vid*. Other talliers may verify the validity of any published vector (via check vectors) and include it in their tally even if they did not receive it directly from the voter. Each voter may also verify that her vote appears in the list of published votes. The majority's tally then determines the results of the elections.

4 Security Analysis

In this section we analyze the security properties of the electronic voting scheme, roughly according the criteria posed by Cranor and Cytron in [CC97], as detailed below.

For the formal analysis, we will make use of the following notation. We denote by n the number of voters. The number of registrars, auxiliaries and talliers is each m, out of which a total of b corrupt authorities combined of all types is assumed, where $b = \lceil \frac{m}{2} \rceil - 1$. We denote by $p = b + 2$, the typical length of vectors we employ. Finally, let q be the prime size of the finite field Z_q we operate with. Let S be the set of possible valid candidate-ballots. We will use *with high probability* to mean with probability at least $1 - |S|/q$.

4.1 Accuracy

The first requirement of a voting systems is the *accuracy* of the tally it produces. Specifically, it must disallow stuffing of ballots by any participant (including the authorities), and prevent the elimination and the modification of valid ballots.

In order for any vote V to be counted in the final tally in our scheme, there must exist $b+1$ different talliers that accept it. Hence, at least one honest tallier

[3] Anonymized communication can again be achieved via AMPC [MP01], so that the voter does not need to employ cryptographic software for this.

must include the vote in his tally. This means that there is an honest tallier for whom the check vectors B satisfy $VB = s$ for some candidate s. The question is how difficult is it to forge such V. Here, we focus on a particular vote vector V and a corresponding check vector B held by an honest tallier. We show that there is low probability that a collusion of less than $p-1$ authorities (of any combination of registrars, auxiliaries, and talliers) can forge V' for which $V'B = s'$, where s' is an acceptable ballot in S. This proves that forged or modified ballots cannot be stuffed in the ballot box. It remains to argue that valid ballots cannot be eliminated. This is simply a result of the fact that a voter receives a receipt from $b + 1$ talliers.

In order to prove that ballots cannot be forged, we commence by showing that knowledge of at least p check vectors, or t check vectors and u vote vector coordinates where $t + u = p$, is required in order to find a V that will satisfy $VB = s$ for a check vector B unknown to corrupt parties. (Lemma 1). We then show that any set of less than $p-1$ servers does not have sufficient information in order to reveal an additional coordinate of B or V not known to them (Lemma 3 and 4). Together, this implies that no group of $b = p - 2$ colluding authorities can find a vote vector V that satisfies the missing check vector equation.

Lemma 1. *Let B be a check vector held by some honest tallier (hence unknown to anyone but the tallier). Then w.h.p. knowledge of at least p check vectors, or t check vectors and u coordinates of V, where $t + u = p$, is required in order to find V' such that $V'B = s$.*

Proof. Since the check vectors are created independently, the only correlation between the check vectors known to the colluding (corrupt) authorities and the unknown check vector held by the honest authority, is the set of constraints $VB = s$ for all B. Thus in order to find a V' that satisfies $V'B = s'$, the colluding authorities actually have to solve for V. As all vectors are of length p, then in order to solve for V one needs to reduce sufficiently many equations. If the degree of any equation set is p, then clearly knowledge of p items (either p check vectors or t check vectors and u coordinates from the vector V which satisfies all check vector equations) is necessary for solving it.

It is left to argue that degree of the equations set is indeed p. The check vectors are created independently by the auxiliaries for each tallier, and each auxiliary controls the creation of a separate coordinate of the check vectors. Therefore, the creation process is equivalent to the repeated selection of a random vector B that satisfies $BV = s$. W.h.p. the choice of m such vectors ($m >= p$) yields a vector set for which each subset of p vectors are in the general position. Though this is a classic result, we give the intuition here: Every $k < p$ vectors span a sub-space of dimension at most $p-1$. Any k'th vector is chosen randomly from the entire p-dimensional space. Therefore, the probability that the k'th vector belongs to the $(p-1)$-dimensional subspace defined by $p-1$ other vectors is $1/q$. Thus, w.h.p, each subset of size p or less of equations $VB_i = s$ is also independent.

Lemma 2. *Let r_i be a random value used by an honest auxiliary authority in the enhanced check vectors protocol. Then with high probability, a cooperation of at least $p-1$ auxiliary authorities is required in order to reveal r_i.*

Proof. The generation of r_i in the enhanced check vectors protocol induces the following constraints:

1. $r_i = \sum_j r_{ij}$, where r_{ij} is sent to the j'th auxiliary.
2. $r'_k = \sum_k r_{kj}$, where r_{kj} is produced by the k'th authority so that $r_k = \sum_j r_{kj}$, and where all of the r'_k are known to the first auxiliary authority A_1.

A collusion of x auxiliaries, $x < p-1$, that includes A_1 knows all the r'_k's, x of the equations designated in item 2, and x of the equations designated in item 1. Since r_i is a sum of p independent variables chosen uniformly at random, in order to solve for r_i it is necessary to know all of the r_{ij}. However, from the above it is clear that a cooperation of less than $p-1$ auxiliaries has only $p-2$ equations for p variables, and hence leaves every value in Z_q a possible solution for r_i, and guessing has a probability of $|S|/q$ of success.

Lemma 3. *Let b_i be the i'th check vector coordinate chosen by an honest (i'th) auxiliary authority in the enhanced check vectors protocol and held by an honest tallier. Then a cooperation of at least $p-1$ corrupt auxiliaries is required to reveal b_i.*

Proof. The generation of b_i induces the following constraints:

1. $c_i = c_{i-1} + b_i v_i + r_i$, where c_{i-1} is produced by A_{i-1}, c_i is passes on to A_{i+1}, and v_i is known to A_i.
2. $\sum v_i b_i \in S$ (the check vector equation).

By Lemma 2 above, r_i is not known to a collusion of less than $p-1$ auxiliaries. Hence, for a collusion of less than $p-1$ there are 3 unknowns in equation 1 above and two more unknowns in Equation 2 above. Together, they leave every value in Z_q a possible solution for b_i, and guessing has a probability of $|S|/q$ of success.

Lemma 4. *Let v_i be the i'th coordinate of the vote vector V designated in the enhanced check vectors protocol. Suppose that auxiliary authority A_i holding v_i is honest. Then a cooperation of at least $p-1$ corrupt authorities is required to reveal v_i.*

Proof. The generation of v_i induces the following constraints:

1. $v_i = \sum v_{ij}$ where v_{ij} is chosen by the j'th registrar (and passed in the AMPC computation, which is secure by [MP01]).
2. $c_i = c_{i-1} + b_i v_i + r_i$, where c_{i-1} is known by A_{i-1}, c_i is passed to A_{i+1}, and b_i is known to a corresponding tallier. Note that by Lemma 2, r_i is unknown to the collusion.

3. $\sum v_i b_i = s$ (the check vector equations).

A collusion of less than $p-1$ authorities may solve for up to $p-2$ of the vote vector v_j's, as a result of vote vector coordinates known to corrupt auxiliaries and check vectors known to corrupt talliers. This still leaves all possible values for v_i. In addition, they also know at most $p-2$ values in equation 1, which also leaves all possible values for v_i. Finally, they know some equations of the form 2 above for which c_{i-1}, c_i and b_i are known but r_i is not known. Since each such equation adds an unknown r_i, they still are unable to solve for v_i.

4.2 Democracy

The second requirement of a voting system is that it preserves the "one person one vote" rule, allowing every eligible voter and only them one and only one vote. That an eligible voter receives a valid vote vector during registration is guaranteed by our Testing step. Specifically, the probability of detecting corruption at registration (e.g., by providing the voter with invalid vote vectors or some of the talliers with invalid check vector shares) can be made as high as desired by tuning the amount of pre-voting testing. The vote once rule is maintained by having each voter attach her *vid* to the vote and allow each *vid* to appear only once in the final tally. Finally, we already discussed in the accuracy analysis the possibility of producing forged votes, and concluded that the scheme cannot allow ineligible entities to vote.

4.3 Verifiability

The third requirement is that of verifiability. One version of this requirement is individual verifiability, namely that individuals are able to verify that their votes had been counted correctly. A stronger requirement is universal verifiability [BY86], ensuring that any party, including a passive observer, can convince herself that the election is fair, i.e., that the published final tally is computed fairly from the ballots that were correctly cast. Since the individual vote vectors are published by the talliers in our scheme, both individual verifiability and universal verifiability are maintained.

4.4 Privacy and Receipt Freeness

Finally, a fundamental requirement is the need to ensure voters privacy, i.e., that how an individual votes will be kept secret. A stronger requirement of privacy is that the voter is unable to prove how she voted. This requirement is important in order to prevent vote buying and voter coercion. However, this requirement is contradictory to individual verifiability - being able to verify that ones vote is counted correctly usually means also that one can prove how she voted.

In order to link ballots with voters, one actually needs to be able to correlate either vid's or vote vectors with voters. Since ballots are delivered to talliers over an anonymous network, and since we assume the communication channels between the voter and the non-corrupt entry points of the anonymous network are secure, talliers cannot trace ballots back to voters. Since each registrar knows only a share of each vote vector coordinate, any collusion of $p-1$ or fewer registrars knows nothing (in the information theoretic sense) about any vote vector, since they are still missing one share of each coordinate. auxiliary authorities, on the other hand, do know vote vector coordinates. However, they receive these as the results of an AMPC, and hence, by the properties of the AMPC [MP01] they cannot correlate them to any input share which registrars can relate to voters.

The fact that our system provides a receipt for the voter, does not necessarily mean that it is not receipt free: Since the voter (and only her), can produce 'fake' receipts for every possible ballot, a third party, to whom the voter presents her Vote Vectors and receipts, is still unable to know to whom the voter actually voted.

However, a voter can reveal her VID, in order to prove how she voted. Thus our system is currently not receipt free.

5 Discussion and concluding remarks

In this paper, we have presented a new primitive, enhanced check vectors, for the distributed creation and verification of weak signatures, which does not require any cryptography. We showed how this primitive can be used, along with AMPC, to establish a distributed e-voting system which does not require any cryptography. In our suggested voting scheme, the voter also receives a receipt, indicating that her vote was properly received by the talliers. Any software installed on the voter PC (the 'pollster') cannot generate ballots, modify the ballots cast by the voters, or forge receipts. Thus our suggested e-voting system is the first that does not require the use of a trusted piece of software (the pollster) to conduct the encryption and communication on behalf of the voter. The requirement for such a piece of trusted software is one of the major hurdles obstructing the deployment of e-voting systems.

Our system supports safe e-voting from any PC connected to the Internet, and not only from secured, carefully monitored, electronic voting booths. The software required for voting can be distributed by different parties and organizations, and not necessarily by one central authority. Voters do not have to trust the software given to them, and may conduct the series of b multiplications and additions using a simple calculator (or even a scrap of paper) to verify that their ballot was properly accepted by the talliers. The simplicity of the software required makes it feasible to use not only PC's, but also cellular phones or other networked devices, in order to cast one's vote.

We are currently building a prototype of an e-voting system based on our scheme in order to prove its feasibility. We plan to support voting through standard HTML browsers, as well as through any WAP enabled device.

One issue which is not addressed by our scheme (or by any other e-voting scheme) is the need to identify voters during actual voting. If voters are not identified during actual voting, they may sell the voting credentials they received to a third party, who may then vote on their behalf. This probably means that in the foreseeable future, national elections would still be conducted from designated polling stations, in which the voters identity can be verified. Even in this case, in our scheme polling stations can use standard PC's connected by a VPN to the AMPC network and the talliers. Voters do not have to trust the software installed at the polling station, and can use independent means (such as a calculator brought from home) to verify that their ballot was cast correctly. So long as the number of compromised authorities does not exceed the design threshold, the scheme is secure. Note that vote vectors shares can be sent in our scheme to voters by regular mail or e-mail, and only on election day voters identify themselves at polling stations in order to be able to cast their vote.

If deployed in a full Internet voting from home PCs, currently our scheme does not protect voter anonymity against a pollster cooperating with one of the talliers. This vulnerability exists mainly due to our desire to keep the exposition simple. To overcome this, the voter's ballot should be split among different talliers, and never assembled by them separately from other votes. Currently, this would require the voter to contact all participating talliers and, furthermore, require all talliers to function properly during the entire election process. We believe that a quorum system can be naturally employed here in order to relax this requirement. A closely related issue is that of receipt freedom: By splitting her vote, a voter would not be able to prove how she voted, even to a party which controls some of the talliers.

Finally, we believe that enhanced check vectors may have other applications such as anonymous auction systems and the distribution of e-cash.

References

[BY86] J. Benaloh and M. Yung. "Distributing the power of a government to enhance the privacy of voters". In *Proceedings of the 5th ACM Symposium on Principles of Distributed Computing (PODC)*, pp. 52–62, 1986.

[CIV01] California Internet Voting Task Force final report, available at `http://www.ss.ca.gov/executive/ivote/`

[Cha81] D. Chaum. "Untraceable electronic mail, return addresses and digital pseudonyms". *Communications of the ACM* 24(2):84-88, 1981.

[Cha85] D. Chaum. "Security without identification: Transaction systems to make big brother obsolete". *Communication of the ACM* 28(1):1030-1044, 1985.

[CFSY96] R. Cramer, M. Franklin, B. Schoenmakers and M. Yung. "Multi-authority secret-ballot elections with linear work". *LNCS 1070, Advances in Cryptology – EUROCRYPT '96*, pp. 72-83, 1996.

[CGS97] R. Cramer, R. Gennaro and B. Schoenmakers. "A secure and optimally efficient multi-authority election scheme". *LNCS 1233, Advances in Cryptology – EUROCRYPT '97*, pp. 103–118, 1997.

[CC97] L. F. Cranor and R. K. Cytron. "Sensus: A security-conscious electronic polling system for the Internet". *Proceedings of the Hawai'i International Conference on System Sciences*, 1997, Wailea, Hawaii.

[CLR89] T. H. Cormen, C. E. Leiserson and R. L. Rivest. "Introduction to Algorithms". MIT Press, 1989

[CM01] CALTECH-MIT/Voting Technology Project `http://www.vote.caltech.edu/Reports/index.html`

[DmLM82] R. DeMillo, N. Lynch, and M. Merritt. "Cryptographic protocols". *Proceedings of the 14th Annual Symposium on the Theory of Computing*, pp. 383-400, 1982.

[DuR99] B. W. DuRette. "Multiple administrators for electronic voting". B.Sc thesis, MIT, 1999. `http://theory.lcs.mit.edu/~cis/theses/DuRette-bachelors.pdf`

[FY94] M. Franklin and M. Yung. "The blinding of weak signatures (extended abstract)". *LNCS 950, Advances in Cryptology – EUROCRYPT 94*, pp. 67-76, 1995.

[FOO92] B. Fujioka, T. Okamoto and K. Ohta. "A practical secret voting scheme for large scale elections". *LNCS 718, Advances in Cryptology – ASIACRYPT '92*, pp. 244-251, 1992.

[HS98] Q. He and Z. Su. "A new practical secure e-voting scheme". *IFIP/SEC '98 14th International Information Security Conference*, 1998.

[IPI01] Internet Policy Institute, Report of the National Workshop on Internet Voting: Issues and research agenda. Available at: `http://www.netvoting.org/Resources/InternetVotingReport.pdf`

[MP01] D. Malkhi and E. Pavlov. "Anonymity without 'Cryptography' ". *Proceedings of Financial Cryptography '01* (FC '01).

[RB89] T. Rabin and M. Ben-Or. "Verifiable secret sharing and multiparty protocols with honest majority". In *Proceedings of the 21st ACM Symposium on Theory of Computing (STOC)*, pp. 73–85, 1989.

[S91] A. Salomaa. "Verifying and recasting secret ballots in computer networks". *LNCS 555, New Results and New Trends in Computer Science*, pp. 283-289, 1991.

[Sch99] B. Schoenmakers. "A Simple publicly verifiable secret sharing scheme and its application to electronic voting". *LNCS 1666, Advances in Cryptology – CRYPTO '99*, pp. 148-164, 1999.

[WALS02] M. Wright, M. Adler, B. N. Levine, C. Shields. "An Analysis of the Degradation of Anonymous Protocols". In *Proceedings of Network and Distributed System Security Symposium*, 2002.

A An AMPC realization

For completenes, we depict here the construction of an AMPC computation [MP01]. The picture shows a network of m^2 players implementing an AMPC such that any collusion of $m-1$ corrupt players can be tolerated.

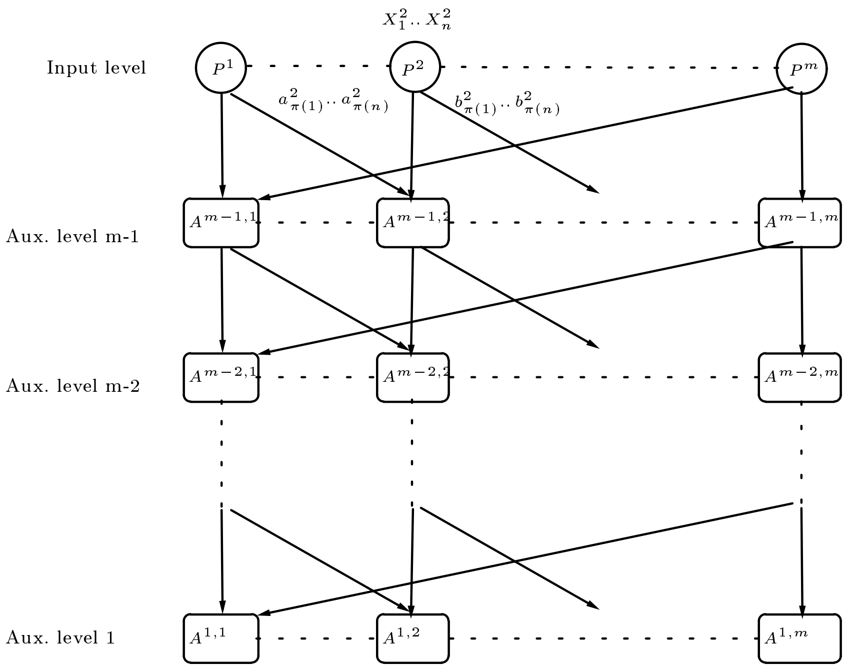

Fig. 1. The communication graph in an AMPC.