Cryptographic election protocols for reweighted range voting & reweighted transferable vote voting

Warren D. Smith
warren.wds@gmail.com

September 27, 2005

Abstract —

We describe a correct, "verifiable," and "coercion resistant" cryptographically secure election scheme which takes $O(NC^k + V)$ (highly parallelizable) steps to process V votes by N voters in a C-candidate election, which for the election systems we shall be considering is best possible if $k = 1$, and we achieve $k = 1$ and/or $k = 2$.

Previous cryptographic election schemes had only been able to handle "additive" election methods such as Plurality, Approval Voting, Condorcet, or Borda count, or methods with "anonymizable" votes such as Instant Runoff Voting with few enough candidates C so that votes were not expected to be uniquifiable (i.e. if $C! \ll V$). The new approach *breaks* those barriers in the three most important special cases. First, we can handle the author's "reweighted range voting" (RRV) at least in the case when the number of possible scores for any given candidate is restricted to a small-enough set (such as the integer interval $[0, 9]$) so that useful votes cannot be uniquely identified by consideration of the sum of its scores on the current winners. Second, we can handle Hare/Droop-reweighted STV (STV=*s*ingle *t*ransferable *v*ote) provided we are willing to use inexact reweighting factors (truncated down to few bits of precision, e.g. to the nearest part in 120) with $k = 2$. (For STV our runtime bound with $k = 2$ probably is still best possible, but we have not proven this.) Third, we can also handle STV without reweighting and "BTR-IRV." However, the new techniques still are not all-powerful because Woodall's DAC (Decreasing Acquiescing Coalitions) voting method still apparently cannot be handled.

There are two previous schemes which I consider practical for performing crypto-secure elections. The first is the "BB-homo" scheme of Cramer, Schoenmakers, et al [4][5] which depends on homomorphic encryptions and bulletin boards. The second is the scheme invented by Juels, Catalano, and Jakobsson [15], as improved to make it practical and linear-time by Smith [23], which depends on "hidden credentials," "plaintext-hashing" for fast all-pairs "plaintext equality testing," and "mixnets." (This scheme is more complicated than the BB-homo scheme, but has the advantage of being applicable to a strictly larger class of election methods and of providing the strongest available security guarantee, namely the "coercion resistance" property introduced by [15].) The scheme we shall now discuss here combines *all* of those old ideas, plus some new ones![1] Although more complicated, its computational resource consumption still is sufficently small that it too seems practically feasible.

1 Cast of characters

Bulletin boards (BBs): Memory which may be read (with random access) by anybody, and which may be written by approved agencies. It is usually convenient to assume that this writing is always of "append" (rather than "random access") style.

Voters: Provide votes (in encrypted form) which are posted on a bulletin board.

Mixers: Accept N encrypted inputs, permute and re-encrypt those inputs, and output the N results along with ZK-proofs[2] that they did so. Several mutually distrustful mixers, one after the other, can thus perform a permutation and simultaneous re-encryption of the data with *nobody* knowing what the product permutation is. (This is called a "mixnet." It is important that the mixers distrust one another because that way at least some will refuse to collude and will not tell each other their secret permutations.)[3]

Talliers: There are several, mutually distrustful, talliers. Each tallier knows some secret information not known to the other talliers. This allows super-threshold subsets of the talliers to perform computations cooperatively that would be infeasible for any subthreshold set. (It is important that they distrust one another[4] because they will refuse to collude and will not tell each other their secrets.)

Verifiers: The voters and talliers and mixers broadcast enough information in the form of "zero knowledge proofs" (preferably "non-interactive" ones[5]) to permit any external verifier, by examining that information, to become confident that the talliers, voters, and mixers are performing the computations they are supposed to

[1]The reader will need to be familiar with [23], otherwise the present paper will be somewhere between "sketchy" and "incomprehensible." Also, familiarity with [4][5] certainly would not hurt. An attempt to survey all of cryptographic voting is [22].

[2]ZK is an abbreviation for "zero knowledge."

[3]It is simplest conceptually to regard the mixers and talliers as disjoint entities. However, they could in fact be the same entities.

[4]Unless the election is a total sham, at least some of the competing *candidates* ought to distrust each other.

[5]To reduce communication needs, and also to allow creation of a permanent record, reviewable at a later time, of the election.

(or confident that some – whose identities will be apparent or irrelevant – are cheating, or that super-threshold sets of cheating talliers and/or mixers are colluding).

For simplified mixer schemes, see [23][22].

2 Informal definitions of Security guarantees

Correctness: ① Each authorized voter's chronologically-last-cast[6] vote is incorporated – correctly – into the computed election result, but ② no unauthorized voter can have his vote counted and ③ no voter can have more than one vote counted.

Verifiable: After the election, the facts that ①-③ were true are proven by a zero-knowledge proof that is available to external verifiers – or (if one or more were false) then the ZK-proof protocol makes it clear which actor first violated the protocol.

Coercion resistant: "We allow the adversary to demand of coerced voters that they vote in a particular manner, abstain from voting, or even disclose their secret keys. We define a scheme to be coercion-resistant if it is infeasible for the adversary to determine whether a coerced voter complies with the demands." [15]

Our schemes shall depend on the assumed difficulty of the discrete logarithm problem in elliptic curve groups of large prime order.

3 Description of Reweighted Range Voting (RRV) [21]

Let there be C candidates from whom V voters are to select W winners, $0 < W < C$, $0 < V$.

```
procedure Reweighted-Range-Vote
 1: Each voter $k$ supplies a $C$-vector $\vec{x}_k$ as his vote, each entry of which is a real number in $[0,1]$. The $c$th entry of this vector expresses that voter's opinion of candidate $c$ (i.e. 1=great, 0.5=middling, 0=terrible);
 2: Each $C$-vector vote has associated with it, a "weight" $w_k \in [0,1]$.
 3: for $r = 1$ to $W$ do
 4:    for $k = 1$ to $V$ do
 5:       Let the weight of vote $k$ be $w_k = 1/(X+1)$, where the *sum* of vote $\vec{x}_k$'s winner-entries is $X$. (Thus, initially, there are no winners and all weights are 1.)
 6:    end for
 7:    Compute the weighted-vote-sum vector $\vec{s} = \sum_{k=1}^{V} w_k \vec{x}_k$ (actually, this step would be best programmed as combined into step 5, but we have written it separately to enhance clarity);
 8:    The candidate $C$ with the largest $\vec{s}$-entry (among candidates who have not yet been declared "winners") is declared to be the $r$th winner.
 9: end for
```

In the 1-winner case, RRV reduces to range voting [24], i.e., it simply adds up all the vote vectors $\vec{s} = \sum_{k=1}^{V} \vec{x}_k$ and then declares the winner to be the index of the largest entry in $\vec{s}$. The first RRV winner in fact is always the same as the range-voting winner, but the second RRV winner is not necessarily the same as the candidate that ordinary range voting would say was in second-place. That is because the reweightings cause the supporters of the first winner to have diminished influence on the choice of the second[7]

4 Description of Hare/Droop-Reweighted (or unweighted) STV

Here is a pseudocode description of the Hare/Droop STV procedure [25][16][13][9]. Let there be C candidates, from whom V voters are to choose W winners ($0 < W < C$, $0 < V$).

```
procedure STV-election
 1: Obtain from each voter a preference ordering (permutation) of the $C$ candidates;
 2: Associate each vote with a real "weight" $w$ with $0 \le w \le 1$, where initially all weights are 1;
 3: Compute the "Droop Quota" $Q = \lfloor V/(W+1) \rfloor + 1$;
 4: loop
 5:    repeat
 6:       for $c = 1$ to $C$ do
 7:          Compute $F_c$, the sum, over all votes ranking candidate $c$ first, of that vote's weight;
 8:       end for
 9:       $g = \text{argmax}\, F_c$;
10:       {$g$ is the "good" canddt with the most 1st-place votes}
11:       if $F_g \ge Q$ then
12:          Multiply the weight of each vote which ranks $g$ first, by $(F_g - Q)/F_g$;
13:          Declare $g$ to be a "winner" and eliminate $g$ from all preference orderings;
14:       end if
15:       exitwhen $W$ canddts have been declared winners;
16:    until $F_g < Q$
17:    $b = \text{argmin}\, F_c$;
18:    {$b$ is the "bad" canddt with fewest 1st-place votes}
19:    Declare $b$ to be a "loser" and eliminate $b$ from all preference orderings;
20: end loop
```

This is a fairly complicated procedure. Many variants of it, both less and more complicated, also exist.

To explain the concepts inside Hare/Droop STV in English: there are two things going on: *elimination* of loser-candidates top-ranked by the fewest voters (in step 19), and *winner declarations* for candidates top-ranked by enough voters (exceeding the "Droop quota"). After either move, the winner or loser is eliminated from all votes. Votes then are *reweighted* (step 12)

[6]Other vote-selection conventions could also be considered.

[7]The reweighting formula again was carefully designed [21] to force "proportionality." It can also confer additional benefits of "encouraging voter honesty." For example, a voter has incentive not to exaggerate his high opinion of some candidate Y too greatly, because then (if Y wins) that exaggeration will decrease the weight of the voter's vote in later RRV rounds.

so that anybody who just voted for a winner will have smaller vote-weight in the next round.

In *unweighted* STV, there are only eliminations; there are no winner-declarations and all weights always are 1. Unweighted STV is simpler and in the single-winner case is completely equivalent. The reweighting assures "proportionality" theorems hold in the multiwinner case; unweighted STV used as a multiwinner election system can suffer from severe *proportionality failures*. For example,[8] let $C = 2W$, let there be two kinds of people ("Democrats" and "Republicans"), and let 49.99% of the voters be Democrat and 50.01% Republican. Let there be W Republican and W Democratic candidates, with all the Republicans being indistinguishable "clones" (whom the voters order randomly) but with one special Democrat being more attractive to each of the Democratic voters, than any of the other $W - 1$ Democratic candidates. Then, no matter how large W is, elimination-only STV (i.e. unweighted STV, also called Instant Runoff Voting) will elect a committee consisting of the one special Democrat and $W-1$ Republicans (very unrepresentative). That is because the unspecial $W - 1$ Democrats will be eliminated since each has zero first rank votes.

In contrast, the full `STV-election` procedure would have elected the special democrat immediately (exceeds quota), and then eliminations would have roughly alternated between declaring Republican and Democrat losers, until at last a committee with 50-50 composition was attained.

5 BTR-IRV and IRV

IRV (Instant Runoff Voting) is the single-winner version of STV. (The reweighting rules are irrelevant since no reweighting ever occurs before the winner has been identified.)

"BTR-IRV" is a voting method[9] intended to try to combine the virtues of IRV with Condorcet: BTR-IRV will always elect a Condorcet-winner if one exists. The BTR-IRV method is this. Each "vote" is a rank-ordering of the C candidates. The two candidates with the fewest top-rank votes ("the two worst") are compared head-to-head based on all the votes with the remaining $C-2$ candidates erased from the picture. Whoever loses this comparison ("the worst") is eliminated from the election and from all votes. We then continue doing such eliminations until only one candidate remains, and declare him winner.

6 How to do (reweighted or unweighted) STV elections crypto-securely

1. Each *vote* in a C-candidate election consists of C different C-vectors. The kth C-vector is a vector containing one 1-entry and $C - 1$ zero-entries, where the position of the "1" indicates the kth preference vote. (Thus there are C^2 numbers in the vote in all. All are encrypted using homomorphic ElGamal randomized encryption.)

2. To assure that this collection of C-vectors genuinely represents a permutation (rank ordering) of the C candidates, the voter must also provide a zero-knowledge (ZK) proof that the $C \times C$ matrix formed by these vectors is a permutation matrix, i.e. has exactly one 1-entry in each row and in each column.[10]

3. For simplicity of exposition, we shall suppose we are going to use reweighting factors rounded off to the nearest part in 120. In that case we need to pre-multiply all votes by (which in the homo-encrypted world means pre-exponentiate the votes to the power) 120^C.

4. These votes are initially handled just like in the JCJ-Smith scheme [23] using mixnets, ZK-proof checking, and "plaintext-hashing" of "credentials" to get rid of unauthorized, fake, invalid, and double votes while preserving vote-anonymity and secrecy.

5. However, unlike in [23], the votes now are *not* decrypted at the end with the aim of allowing a trivial plaintext vote-processing step as a finale. Instead we now proceed to a different kind of vote processing.

6. To add up all the first-preference votes, we *multiply* all the first-preference C-vectors elementwise to get (because of the homomorphic nature of the encryption) the *encrypted* C-vector of vote *totals*. The elements of this vector may then be cooperatively-decrypted by the mutually distrustful election authorities (none of whom know the decryption key individually).

7. With the first-round totals now broadcast in plaintext form, we now publicly eliminate a candidate or declare one a winner (whichever follows from that set of totals). If candidate j is eliminated, we erase all the jth entries of all C-vectors in all votes, thus decreasing C to $C - 1$, and skip to step 11.

8. If candidate w is declared a winner, then we compute and broadcast the reweighting factor (which depends on how many votes w got above and beyond the "Droop quota") and round it off to the nearest part in 120, with the result being, say, $F/120$ for some integer $F \geq 0$.

9. We then go through all votes: for each, if it voted for w, we reweight it by multiplying all numbers by $F/120$, i.e. in the homo-encrypted world by exponentiating everything to the power $F/120$. Note the "division" by 120 here is performed mod P. Throughout, we assume we are doing everything in a publicly known elliptic curve group of large publically known prime order P where $P \gg 120^C$.

10. Finally, we "rotate" the votes for w by making their first C-vector (which was their first preference vote) now be their last by moving the vector data around. (This can be done obliviously and regardless of the encryptions.)

[8] Brian A. Wichmann showed me this example.

[9] "BTR" stands for "*b*ottom *t*wo *r*anks" and it also has been suggested that it be pronounced "better." It was invented by Rob LeGrand, a graduate student at Washington University in St. Louis. However, there are some reasons to claim that BTR-IRV is in fact a *less* desirable voting system than ordinary IRV, e.g. it seems more vulnerable to "strategic voting."

[10] Such a ZK-proof could consist of single-bit ZK-proofs (0 or 1) for each entry, and then multiply all the homomorphic encryptions of the entries in each row (or column) together to get the homomorphic encryptions of the row (column) totals, and finally ZK-prove each of those totals are encryptions of 1. Both of these ZK-proofs can be done via a "ZK-proof of discrete log equality" [23][22] if we are using ElGamal encryptions.

11. We then *mix* and *re-encrypt* all votes using a mixnet.

12. We are now ready to perform the second STV round (by looping back to step 6). We continue on for $C-1$ rounds (or if it is a multiwinner election with M winners, for $C-M$ rounds) in all, at which point we are done. Each round, the weighted-total first-rank votes for all C candidates are announced, and the eliminated or declared-winner candidate for that round is announced.

Discussion. The votes themselves are never decrypted – only the vote-total vectors are decrypted – *except* that each round, each top-rank-vote (or nonvote) for that round's winner (if there *is* a Droop-quota-exceeding "winner" that round) is decrypted thus revealing a single plaintext bit per vote.

The author of that vote is not known at the time the bit is revealed (due to previous mixnet steps) and then the revealed bits are erased and forgotten and the votes are modified, remixed, and re-encrypted (in step 11) before the next STV round. If the bits were *not* "erased and forgotten" that is OK because the votes have now been mixed so that the association of the bits with the (new mixed) votes is *effectively* erased and forgotten.

Finally: the scheme we have just outlined also can handle "IRV" (instant runoff voting) [25], i.e the unweighted variant of STV, and "BTR-IRV" elections with appropriate slight modifications in the latter case.

7 How to do RRV elections crypto-securely

We shall initially assume that the "real number" scores that each voter awards to each candidate in RRV voting, in fact are, *integers* in the interval $[0, 9]$ (or $[0, 99]$), i.e. we are using "low precision *fix point*" reals. That leads to the simplest protocols. However, it is also possible (with more effort and complexity) to handle reals with *arbitrary* (but fixed) precision by use of "bit splitting" or "radix splitting" techniques (provided we then agree to round all reweighting factors to, e.g. the nearest part in 120 instead of using the exact RRV reweighting formula).

1. In a C-candidate election, each vote is a C-vector of scores, where the scores are normally considered to be reals in $[0, 1]$, but we shall, for the purpose of considering this algorithmic implementation, assume they are integers in $[0, 9]$. We assume the voters provide all these scores in homomorphic-ElGamal-encrypted form with ZK-validity proofs (integer-interval-membership ZK-proofs).

2. These votes are initially handled just like in the JCJ-Smith scheme [23] using mixnets, ZK-proof checking, and "plaintext-hashing" of "credentials" to get rid of unauthorized, fake, invalid, and double votes while preserving vote-anonymity and secrecy. However, unlike [23], the votes now are *not* decrypted at the end with the aim of allowing a trivial plaintext vote-processing step as a finale. Instead we now proceed to a different kind of vote processing.

3. We begin by *multiplying* all scores by $(1+9C)!$ (i.e. in the homo-encrypted world, *exponentiating* them all to the power $(1+9C)!$) to allow easy reweightings by reciprocals of integers in $[1, 9C+1]$ later. If the weights are to be *approximate* and rounded to the nearest part in 120 (say) then we could use the factor 120 instead of the factor $(1+9C)!$ here.

4. At all times each vote will be publicly associated with a publicly known reciprocated-weight value ("$X+1$" in step 5 of the RRV algorithm description in §3). We begin by setting all vote-weights to 1. Each weight will monotonically decrease (and hence its reciprocal will increase) as the election proceeds.

5. We add up all the weighted votes by *multiplying* their homomorphic encryptions (each exponentiated to the power $1/F_j$ where the weight $1/F_j$ is associated with vote j). Due to the homomorphic nature of the encryption scheme, we now have the C-vector of *totals* – in *encrypted* form.

6. We now apply cooperative decryption to decrypt the totals.[11] These can now be broadcast publicly. That tells us both the RRV first round results, and in particular it tells everyone the first-round's winner w.

7. We now find the new public weight (see step 5 in the RRV algorithm in §3) to give to each vote by going through all the votes, for each vote cooperatively-decrypting that vote's score for w and using it to additively-adjust that vote's public reciprocated weight, i.e. to adjust the X-value in step 5 of the RRV algorithm description in §3.

8. We then *mix* and *re-encrypt* all votes using a mixnet[12] – and eliminate all the scores for w within each vote, i.e. turning all votes into $(C-1)$-vectors instead of C-vectors.

9. We are now ready to perform the second round by looping back to step 5. We continue on for M rounds in a multiwinner election with M winners, $1 \le M < C$, at which point we are done.

Discussion. The votes themselves are never decrypted – only the vote-total vectors are decrypted – *except* that each round, each vote's score for that round's winner is decrypted revealing a single integer in $[0, 9]$ per vote, and used to update the public weight of that vote. The author of that vote is not known at the time this integer is revealed (due to previous mixnet steps) and then the scores are erased and forgotten and the votes are modified, re-mixed, and re-encrypted before the next round. If the scores were *not* "erased and forgotten" that is OK because the votes have now been mixed so that the association of the scores with the (new mixed) votes is *effectively* erased and forgotten. If the scores and score-sums on the winners-so-far for any given vote lie in a small-enough-cardinality set so that *many* voters are expected to have votes with those same scores and score-sums-so-far (i.e. if $V \gg 9C$ in our case with $[0, 9]$ score range...), then this revelation is not enough to reveal the identity of any voter, so that anonymity is preserved.

That ends the description of the simplest version of the crypto-secure RRV election scheme. But we can do better.

[11] See [22] for discussion of how to use Pollard-λ or the Shanks "baby-step giant-step" method to performthe final step of this decryption in public in time sublinear in the number V of votes, if the allowed score range, such as $[0, 99]$, is sublinear in V.

[12] Note that the "mixing" permutation will no longer be perfectly zero-knowledge hidden, in the sense that everybody will know that a vote with weight k could not have been permuted into a vote with weight $\neq k$.

By "bit-splitting" the reals into their binary bits, and updating weights one bit at a time, we can reduce the amount of publicly revealed information about each vote, to a single bit (in addition to its current public weight), and we also can handle arbitrary-precision real numbers.

References

[1] Masayuki Abe: Mix-networks on permutation networks, ASIACRYPT (1999) 258-273, Springer LNCS #1716. Corrections and extensions: Masayuki Abe & Fumitaka Hoshino: Remarks on Mix-Network Based on Permutation Networks, Public Key Cryptography (2001) 317-324, Springer (LNCS #1992).

[2] I.F.Blake, G.Seroussi, N.P.Smart: Elliptic Curves in Cryptogaphy, London Math'l Society Lecture Notes #265, Cambridge University Press 1999. Errata `www.hpl.hp.com/infotheory/errata082900.pdf`.

[3] Fabrice Boudot: Efficient Proof that a Committed Number Lies in an Interval, pp.431-444 in Proc. of EuroCrypt 2000, Springer Verlag LNCS #1807.

[4] Ronald Cramer, Matthew Franklin, Berry Schoenmakers, Moti Yung: Multi-Authority Secret-Ballot Elections with Linear Work, pp.72-83 in EuroCrypt 1996, Springer LNCS #1070.

[5] R. Cramer, R. Gennaro, B. Schoenmakers: A secure and optimally efficient multi-authority election scheme, pp.103-118 in Advances in Cryptology EUROCRYPT 1997, Springer LNCS#1233. Journal version appeared in: European Transactions on Telecommunications 8 (Sept.-Oct. 1997) 481-490.

[6] Yvo G. Desmedt & Yair Frankel: Threshold cryptosystems, Crypto 89 (1990) 307-315 (Springer LNCS #435).

[7] Yvo G. Desmedt: Threshold cryptography, European Trans. Telecommunications 5,4 (1994) 449-457.

[8] Yvo G. Desmedt: Some recent research aspects of threshold cryptography, Information Security Proceedings (1997; Springer LNCS #1396) 158-173.

group, 457-469.???

[9] Henry R. Droop: On methods of electing representatives, J.Statistical Society London 44,2 (1881) 141-196, comments 197-202.

[10] R.Gennaro, M.O.Rabin, T.Rabin: Simplified VSS and fast-track multiparty computations with applications to threshold cryptography, Proc. ACM Symposium on Principles of Distributed Computing PODC 7 (1998) 101-111.

[11] Jens Groth: A Verifiable Secret Shuffle of Homomorphic Encryptions, pp. 145-160 in *Practice and Theory in Public Key Cryptography - PKC 2003* (Springer LNCS #2567). LATER UPDATE???

[12] Jens Groth: Non-interactive Zero-Knowledge Arguments for Voting, pp. 467-482 in *Applied Cryptography and Network Security - ACNS 2005* (Springer LNCS #3531).

[13] Clarence G. Hoag & George H. Hallett: Proportional Representation, Macmillan, New York 1926; Johnson reprint corp. 1965.

[14] J. Economic Perspectives 9,1 (1995) is a special issue on voting methods and is a recommended way to learn about them.

[15] Ari Juels, Dario Catalano, M.Jakobsson: Coercion-resistant electronic elections. One older version was at Cryptology ePrint Archive: Report 2002/165 `http://eprint.iacr.org/`; latest version on Juels web page: `http://www.rsasecurity.com/rsalabs/node.asp?id=2030` as of June 2005. Now accepted by Workshop on Privacy in the Electronic Society (WPES) Alexandria, VA, USA. Nov. 2005.

[16] W.J.M. Mackenzie: Free elections, George Allen & Unwin Ltd. London 1958.

[17] Hannu J. Nurmi: Voting procedures: A summary analysis. British Journal of Political Science 13,2 (1983) 181-208.

[18] T.P.Pedersen: A threshold cryptosystem without a trusted party, Eurocrypt 91 (Springer LNCS #547) 522-526.

[19] Claus-Peter Schnorr: Efficient Signature Generation by Smart Cards, J. Cryptology 4,3 (1991) 161-174.

[20] Adi Shamir: How to share a secret, Commun.ACM 22,11 (1979) 612-613.

[21] Warren D. Smith: Reweighted Range Voting – new multiwinner election scheme, #78 at `http://math.temple.edu/~wds/homepage/works.html`.

[22] Warren D. Smith: Cryptography meets voting, #80 at `http://math.temple.edu/~wds/homepage/works.html`.

[23] Warren D. Smith: New cryptographic election protocol with best-known theoretical properties, #89 at `http://math.temple.edu/~wds/homepage/works.html`. Frontiers in Electronic Elections (FEE) 2005 Milan.

[24] Warren D. Smith: Range voting, #56 at `http://math.temple.edu/~wds/homepage/works.html`.

[25] Nicolaus Tideman: The Single transferable Vote, J. Economic Perspectives 9,1 (1995) 27-38. (Special issue on voting methods.)

[26] Douglas R. Woodall: Monotonicity of single seat preferential election rules, Discrete Applied Maths. 77,1 (1997) 81-98.