Coercion-Resistant Remote Voting using Decryption Mixes

Michael R. Clarkson Andrew C. Myers
Department of Computer Science
Cornell University
{clarkson,andru}@cs.cornell.edu

Abstract

The recently proposed Prêt à Voter election scheme uses decryption mixes to achieve anonymity of votes and verifiability of an election while requiring minimal trust in the component of the election system that performs these mixes. However, this scheme still requires trust in several human and machine components; these requirements make the scheme impractical for use in remote voting. To adapt the scheme for remote voting, new protocols are proposed that preserve anonymity and verifiability and additionally achieve coercion resistance. The scheme is also extended to allow Condorcet voting methods. An implementation is in progress using Jif, a security-typed language.

1 Introduction

Many recent electronic voting protocols have focused on increasing voters' trust in both the integrity of the election results and the confidentiality of the votes that are cast. Protecting these properties is difficult, in part because a voter's vote may be coerced or bought. Voting protocols that protect against coercion are variously called *uncoercible*, *coercion-resistant*, and *receipt-free*. Recent protocols use mix networks and homomorphic encryption to decrypt and anonymize votes. Unfortunately, such schemes are either expensive [5] or require physical assumptions [8].

In contrast, the recent *Prêt à Voter* (PAV) scheme [1] uses a *decryption mix* to decrypt and anonymize votes. The decryption mix is cheap to implement and uses standard public key cryptography rather than more exotic homomorphic schemes. While PAV aims to achieve "minimal dependence. . . [on] the correct behaviour of components of the voting system," it nonetheless assumes that the authentication of voters and the casting of votes takes place in a controlled physical environment where (at least some) trust can be placed in the election officials, the voting booth, and the voting machine. These assumptions make the scheme difficult to use for *remote voting*, in which votes are cast in uncontrolled environments (perhaps over the Internet). For brevity, a summary of PAV is elided here.

This paper presents an extension of PAV intended to make the scheme suitable for remote voting, while preserving verifiability, efficiency, and anonymity (that is, the confidentiality of the map between voters and votes). This extension also attempts to strengthen significantly the coercion resistance of PAV. The new key ideas are the judicious application of homomorphic encryption and the use of decryption mixes for authentication. Our extension requires an anonymous channel and withstands the arbitrary failure of f out of $2f + 1$ tellers in the voting system. Coercion (and violation of anonymity) may become possible when more than f tellers fail, but (as in PAV) tellers have a negligible chance of successfully cheating. We also require a trusted registrar to authorize voters. Proofs that the extension achieves these properties are in progress.

We are also interested in Condorcet voting methods [3], which permit voters to rank the available candidates yet are resistant to strategic voting. This paper demonstrates how to implement such methods without introducing a covert channel that could be used for coercion.

2 Condorcet ballots

In a Condorcet election method, each voter submits an ordering of all the candidates from most preferred to least preferred. Condorcet methods are attractive for two reasons. First, voters can express more information about their preferences, a feature shared with other ranked-ballot voting methods such as the widely used Single Transferable

This work was supported by the Department of the Navy, Office of Naval Research, ONR Grant N00014-01-1-0968; Air Force Office of Scientific Research, Air Force Materiel Command, USAF, grant number F49620-03-1-0156; and National Science Foundation grants 0208642, 0133302, and 0430161. Michael Clarkson is supported by a National Science Foundation Graduate Research Fellowship; Andrew Myers is supported by an Alfred P. Sloan Research Fellowship. Opinions, findings, conclusions, or recommendations contained in this material are those of the authors and do not necessarily reflect the views of these sponsors. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

Vote (STV) method developed by Hare in 1862. Second, Condorcet methods enforce (when possible) the fundamental democratic principle of majority rule: a candidate who would defeat every other candidate in a one-on-one election is declared the *Condorcet winner*. This criterion was defined by Condorcet in 1785.

An election can fail to have a Condorcet winner; this possibility has caused social choice theorists, starting with Condorcet himself, to generate a substantial literature exploring various *completion algorithms*. In practice, the concern about completion seems to be excessive. Users of an online Condorcet voting service that we implemented [7] ran 99 elections with at least 10 voters. Of these, 95 needed no completion algorithm. Other, smaller studies [9] have seen similar results.

No electronic voting schemes of which we are aware have considered coercion-resistant implementation of a Condorcet method. Current schemes support ballot forms with yes/no questions and choices of k out of n. A difficulty with ranked ballots is that they introduce a covert channel: voters can encode information into their vote by changing lower-order preferences. For example, if there are twenty candidates, a voter's lowest ten preferences probably will not influence the outcome of the election, so at least $10!$ distinct values can be covertly encoded into the vote. Since PAV publicly posts votes, voters can be coerced into encoding their identity into their votes.

Rather than representing the candidate ranking with a single ballot, we propose to encode it into C^2 ballots, where C is the number of candidates. Each ballot is a yes/no choice between two candidates i and j; a "yes" choice means the voter prefers candidate i to candidate j. Each ballot has its own onion, denoted $\mathsf{onion}(\langle i,j\rangle)$. To formalize the construction of onions, let each teller i have two asymmetric key pairs A and B, denoted $(k_{i,A},\ K_{i,A})$ and $(k_{i,B},\ K_{i,B})$, and let there be T tellers total. Let g denote generation of a fresh nonce, preferably by a secure random number generator; as in PAV, the size of the nonce is a security parameter. Denote the encryption of value D with key K as $E(D;K)$. An full onion with innermost layer D is denoted by $\mathsf{onion}(D)$; an onion with i layers is denoted by $\mathsf{onion}_i(D)$:

$$\begin{aligned}
\mathsf{onion}(D) &= \mathsf{onion}_T(D)\\
\mathsf{onion}_0(D) &= D\\
\mathsf{onion}_i(D) &= E(E(\mathsf{onion}_{i-1}(D), \mathsf{g}_A; K_{i,A}), \mathsf{g}_B; K_{i,B})
\end{aligned}$$

The voter is given a set of C^2 ballots, one for each possible $\langle i,j\rangle$ pair. As in PAV, it is necessary to audit by using the tellers in an oracle mode to perform random checks to ensure that ballot sets are well-formed (i.e., each onion is correctly constructed and the set contains all possible $\langle i,j\rangle$ pairs); checked ballots must also be discarded. The voter submits the C^2 onions, along with the yes/no choices, in a random permutation. After all submissions from all voters have been received, the decryption mix is then performed as in PAV.[1] Thus, in the first column of the decryption mix, no coercer should be able to determine the voter's preferences. And in the last column of the decryption mix, each of the voter's C^2 yes/no choices has been anonymized, so the coercer should not be able to determine anything about the voter's low-order preferences.

The C^2 submissions from all voters can be totaled to construct a $C \times C$ matrix in which cell (i,j) represents how many voters prefer candidate i to j. The Condorcet completion algorithms in widest use are *summable*: they decide the winner using only this matrix. Because summable Condorcet methods do not need to see all the preferences of a voter at one time, they have a fundamental coercion resistance that other ranked-ballot methods (such as STV) lack.

3 Voter authentication

It is impossible to achieve perfect resistance to coercion in the context of remote voting because a sufficiently powerful coercer can physically restrain a voter from participating in the election. Thus, we settle for *coercion resistance* [5] and assume a threat model in which the coercer does not have (continual) physical control over the voter.[2] The coercer may demand any secrets known to the voter, try to vote on behalf of the voter, demand that the voter abstain, or demand that the voter cast any arbitrary vote. In addition, the voter may actually want to capitulate to the coercer if the voter is trying to sell his vote.

PAV leaves the methods of authorization and authentication unspecified; here, we assume that voters are issued capabilities, which allow them to authenticate and vote, by a registrar. Since the coercer is allowed to demand secrets of the voter, he may demand the voter's capability, which he can then use to vote on behalf of the voter. Thus it is essential that voters be able to lie convincingly about their capabilities. The key idea behind our solution is therefore to allow the voter to choose any random value (of the appropriate length) and present it to the coercer as a capability; this idea was first used in [5]. The coercer will be unable to distinguish it from a valid capability, but ultimately any vote cast with the fake capability will be discarded from the final tally.

Let voter capabilities be constructed like onions. Suppose that the voting system (that is, the collection of tellers) has an asymmetric key pair $(k_{VS},\ K_{VS})$ where private key

[1] We omit PAV's transformation of the choices by a cyclic ordering because it seems unnecessary: the single bit in the choice is insufficient to allow it to be traced through the mix. Thus, the choices are unchanged in each column of the mix.

[2] We are adhering to the spirit of coercion resistance as defined in [5] but not the letter of it. In particular, we do not use the formalization in terms of probability distributions.

k_{VS} has been split in a threshold scheme among all the tellers. Denote the digital signature of D with k as $S(D;k)$. And let a voter capability vc be of the form:

$$vc = \mathsf{onion}(S(\mathsf{valid}, \mathsf{g}; k_{VS}))$$

where valid is a string indicating that the capability is valid. The digital signature functions as a proof of the validity of the capability.

To vote, the voter obtains a set of C^2 ballots via the protocols in Section 4, and a set of C^2 capabilities from the registrar. A *vote* on a ballot with onion $o = \mathsf{onion}(\langle i,j\rangle)$ is a triple $\langle vc, o, c\rangle$, where c is the voter's choice (yes or no) on $\langle i,j\rangle$, and vc is one of the voter's capabilities. To cast a vote, the voter sends it to the voting system:

$$\mathsf{Voter} \rightarrow \mathsf{System}: \quad \langle vc, o, c\rangle$$

The voter casts C^2 such votes. Note that the system makes no attempt to discern whether vc is a valid capability. Instead the tellers, in the tallying phase, send $\langle vc, o, v\rangle$ through the decryption mix (in PAV, they would have sent $\langle o, v\rangle$). At the end of the mix, they are left with:

$$\langle S(\mathsf{valid}, \mathsf{g}; k_{VS}), \langle i,j\rangle, c\rangle$$

The tellers then discard any votes with invalid signatures and proceed with the tallying.

It is now easy for voters to fake capabilities. A voter can pick any random string (of the appropriate length) as fake voter capability fvc. The voter can submit the vote that the coercer wants, under fvc, or present fvc to the coercer and let the coercer submit the vote. Either way, this innermost layer of this capability will, with high probability, be detected as invalid in the final column of the tally—at which point the vote has been anonymized. Thus the coercer is unable to detect whether fvc was fake or valid, beyond what can be inferred from the final tally.

If any votes contain duplicate voter capabilities, it is unsafe to allow them to proceed through the decryption mix. Otherwise, coercers could submit duplicate votes to determine whether a voter capability is valid or coerce voters into submitting all their ballots under the same voter key. Duplicate onions present similar difficulties. Thus, we restrict the first column of the mix to votes whose capabilities and onions are unique. This introduces the possibility that a malicious component of the voting system could attempt to eliminate votes by submitting duplicates. It is possible to defend against this by requiring certain cryptographic commitments from the system, but we omit full details here due to space.

A problem with this scheme is that the voter must trust the registrar who created vc to construct a valid capability and to forget the voter's association with the capability. Eliminating this trust is a topic of current research.

1. Teller $\rightarrow$ Server: $\langle ec * p_T, eb_T, o_T\rangle$ where:
   - $ec = E(i,j;K_{VS})$ is the encrypted candidate
   - p_T is a blinding factor created by Teller
   - $eb_T = E(p_T;K_{VS})$
   - $o_T = \mathsf{onion}_1(\langle i,j\rangle)$ is Teller's part of the onion
2. Server: Ballot is: $\langle ec * p_T * p_S, eb_T * eb_S, o\rangle$ where:
   - p_S is Server's blinding factor
   - $eb_S = E(p_S;K_{VS})$
   - $eb_T * eb_S = E(p_T * p_S;K_{VS})$
   - $o = \mathsf{onion}(\langle i,j\rangle)$ is the full onion

Figure 1. Ballot creation

Finally, note that the voter must submit his votes over an anonymous channel. Otherwise, the coercer could detect the voter submitting votes under multiple capabilities.

4 Ballot handling

Confidentiality of a vote in PAV critically depends on confidentiality of the ballot given to the voter: any agent who observes the voter's ballot, with the plaintext candidate ordering and corresponding onions, can determine the voter's vote using the first column of the decryption mix. To achieve ballot confidentiality, PAV assumes that a trusted component of the system, perhaps election officials at a polling place, gives a random ballot to the voter without observing the ballot in the process. Also, PAV assumes that the voter is unable to show this ballot to any other agent because of isolation in a voting booth, and that the plaintext candidate ordering is destroyed when the voter has cast his vote because of a shredder in the voting machine. In the context of remote voting, these assumptions are unreasonable. We propose the protocols in Figures 1 and 2 to remove the assumptions using cryptography. For simplicity, we assume only two tellers, denoted Teller and Server.

The ballot creation protocol in Figure 1 creates a single ballot for candidate pair $\langle i,j\rangle$. It attempts to ensure that no teller ever learns the mapping from plaintext candidate pair to the onion that encodes the pair, thereby eliminating any necessary trust in the election officials who store the ballots. The protocol uses blinding factors to obscure the plaintext pair. We now assume that (k_{VS}, K_{VS}) is an ElGamal key pair, and $*$ is multiplication in the ElGamal field.

The ballot distribution protocol in Figure 2 is used to give a single ballot for candidate pair $\langle i,j\rangle$ to the voter; at the end of the protocol, the voter has learned that o encodes $\langle i,j\rangle$, but at no point do any of the tellers learn this fact. The pro-

1. Voter $\rightarrow$ Server: K_{sess}, where (k_{sess}, K_{sess}) an ElGamal session key pair created by the voter.

2. Server: Pick a ballot $b = \langle ec * p_T * p_S, eb_T * eb_S, o\rangle$.

3. Server and Teller: $b' = \mathsf{Reencrypt}(b, K_{sess}) = \langle E(i,j;K_{sess}) * p_T * p_S, E(p_T * p_S; K_{sess}), o\rangle$.

4. Server $\rightarrow$ Voter: $b', \mathsf{DS}(b')$

5. Voter:
    (a) Decrypt $E(p_T * p_S; K_{sess})$.
    (b) Unblind $E(i,j;K_{sess}) * p_T * p_S$.
    (c) Decrypt $E(i,j;K_{sess})$.

Figure 2. Ballot distribution

tocol is run C^2 times to obtain all possible candidate pairs; this must be audited as discussed in Section 2. The protocol attempts to eliminate the remaining trust in the election officials, voting booth, and shredder. Reencrypt denotes the distributed reencryption protocol of [11]. As used here, this protocol reencrypts b under the key K_{sess} such that at no time is the plaintext pair $\langle i, j\rangle$ of b available. This should eliminate any necessary trust in the election officials who hand a ballot to the voter. DS denotes the designated signature scheme of [4]. The designation of the signature on b' to Voter ensures that no agent other than Voter can trust that b' truly is Voter's ballot. This should eliminate any need for a voting booth or shredder. Distribution requires an anonymous channel to prevent coercers from learning that a voter has requested more than one set of ballots.

The reencryption protocol used here requires $3f + 1$ servers, where f is the number allowed to fail. We can reduce this to $2f + 1$ by assuming a synchronous system. Regardless, this changes the failure model for anonymity. Whereas PAV requires voters to trust only one out of any number of tellers to preserve anonymity, now voters must trust one out of $2f + 1$ tellers. However, the failure model for verification remains the same as PAV.

5 Conclusion

This paper has described an extension to the *Prêt à Voter* voting scheme. The need for trust in several components of the system has been eliminated through new cryptographic protocols. Coercion resistance has been added to the scheme, and the implementation of Condorcet voting methods has been enabled. Proofs that the extension satisfies these claims are in progress; we are interested in achieving automated verification of the protocols.

We have implemented a prototype voting system called CIVS, with strictly weaker security guarantees than the system described in this paper, and made it available as an online service [7]. We are currently implementing a new voting system based on the protocols described. We are using Jif [6], a security-typed language that provides additional assurance by checking that information flows in the system respect declared confidentiality and integrity policies. How well Jif's policy language can capture the requirements of a voting system is ongoing work that has already resulted in the discovery of *information erasure policies* [2].

Improving the protocols described here is also a current topic of investigation. It would be useful to eliminate the need for $2f + 1$ servers, imposed by the reencryption protocol. Also, eliminating the need for voters to trust a registrar is very important. A possible solution is a distributed construction of capabilities using entropy from all the tellers.

Acknowledgments We thank Stephen Chong, Nathaniel Nystrom, Peter Ryan, Fred B. Schneider, Lantian Zheng, and Lidong Zhou for discussions about this work.

References

[1] D. Chaum, P. Y. A. Ryan, and S. Schneider. A practical voter-verifiable election scheme. In *Proc. ESORICS'05*, Milan, Italy, Sept. 2005. To appear.

[2] S. Chong and A. C. Myers. Language-based information erasure. In *Proc. CSFW'05*, pages 241–254, Aix-en-Provence, France, June 2005.

[3] P. Fishburn. Condorcet social choice functions. *SIAP*, 33(3), 1977.

[4] M. Jakobsson, K. Sako, and R. Impagliazzo. Designated verifier proofs and their applications. In *Proc. EUROCRYPT'96*, volume 1070 of *LNCS*, 1996.

[5] A. Juels, D. Catalano, and M. Jakobsson. Coercion-resistant electronic elections. Cryptology ePrint Archive, Report 2002/165, 2002. `eprint.iacr.org/`.

[6] A. C. Myers. JFlow: Practical mostly-static information flow control. In *Proc. POPL'99*, pages 228–241, San Antonio, TX, Jan. 1999.

[7] A. C. Myers and M. R. Clarkson. Condorcet Internet Voting Service, 2004. `www5.cs.cornell.edu/~andru/civs`.

[8] C. A. Neff. Verifiable mixing (shuffling) of ElGamal pairs. `www.votehere.net/vhti/documentation/egshuf-2.0.3638.pdf`, Apr. 2004.

[9] M. Regenwetter and B. Grofman. Approval voting, Borda winners, and Condorcet winners: Evidence from seven elections. *Manage. Sci.*, 44(4):520–533, 1998.

[10] T. M. Zavist and T. N. Tideman. Complete independence of clones in the ranked pairs rule. *Social Choice and Welfare*, 6(2):167–173, Apr. 1989.

[11] L. Zhou, M. A. Marsh, F. B. Schneider, and A. Redz. Distributed blinding for distributed ElGamal re-encryption. In *Proc. ICDCS*, pages 815–824, Columbus, OH, 2005.