SpeakUp: remote unsupervised voting

Stefan Popoveniuc

KT Consulting
stefan@popoveniuc.com

Abstract. We present SpeakUp, a novel way to cast a ballot remotely, using a personal computer connected to the Internet. SpeakUp is resistant to potential malware installed on the voter's machine, addresses voter authentication, and offers some degree of protection against massive coercion.
The main idea is that voters are required to read out loud a short text that is associated with the candidate for whom they wish to vote. The voting server identifies the voter by the characteristics of their voice and identifies which text was read. The text, along with the voter's voice, is publicly posted on a bulletin board, and may serve as a receipt for a universally verifiable method of tallying. SpeakUp is based on the biometric characteristics of the voter's voice. Speaker verification is used to authenticate the voters. Similar to voice biometrics, it is assumed that it is difficult for computers to spoof the voter's voice.

Keywords: Internet voting, CAPTCHA voting, speaker verification, voice biometrics, end-to-end voting.

1 Introduction

Remote voting is often cited as a potential solution to make elections more convenient, increase voter participation, and reduce administrative costs. In recent years, jurisdictions around the world, such as Switzerland, Estonia and Australia, have expressed interest in moving to Internet voting as an alternative to absentee or poll site voting. In the United States, one type of remote voting, vote-by-mail, has allowed the states of Washington and Oregon to nearly eliminate poll sites for elections. While Internet voting will probably not replace polling place voting in the near future, there may be instances in which Internet voting has certain advantages, e.g. for deployed military personal or overseas citizens (UOCAVA). SpeakUp was mainly designed for military personal and overseas citizens which do not have access to a regular polling site.

Internet voting may introduce vulnerabilities that are different from the potential problems in a polling place setting, or in mail voting. Security analyses of Internet voting systems have identified a number of potential security issues with using personal computers to cast votes online [5, 23, 13]. The computer used by the voter is not under the control of the election authority. It could be infected with malicious software (malware) that can spy on how the vote is cast,

or change the vote during the casting process. Since a private voting booth does not exist, voters could be subject to improper influence from human coercers.

In addition, voter authentication may be problematic, since voters may not have credentials that can safely be used for authentication in a remote setting. In particular, voters may give their voting credentials to someone else, effectively selling their votes.

SpeakUp provides a novel technique that addresses the malware problem, the voter authentication problem, as well as the possibility of massive coercion via the selling of credentials. Small scale coercion, such as coercion by a family member or someone being physically present next to the voter, is still possible.

Ensuring that a computer is virus-free is virtually impossible. The constant battle between the anti-virus industry and programmers that write malware is well-known. There are a number of techniques that malware developers use to avoid detection, including self-modification and disabling antivirus software. Computer viruses can go undetected, because they can minimally impact the computer's operation and can delete themselves immediately after they ran the first time. Operating systems, bootstrapping software, and other essential software can come with such a virus already installed, and would avoid detection, since such a program has complete control over the computer, preventing any anti-virus software from inspecting it.

SpeakUp bypasses the virus problem. It allows the voter to confidently cast a ballot from any computer, without being afraid that a virus or any software can adversely influence her choices. SpeakUp is radically different from traditional methods to secure client computers, such as installing and constantly updating antivirus software, or booting from a secure read-only media. Even if such methods would be effective, the election officials do not have any means of verifying that the voter complied and used the recommended secure platform. As the election officials are responsible for ensuring that only legitimate voters cast ballots, and the true intentions of the voters are collected, their security assumptions cannot be based on the average voter being an expert at running a secure computational platform.

1.1 Previous work

In the recent years, a series of proposals have emerged, which allow vote casting using the Internet. Most of them use the Internet in some steps of the voting process, but other Internet-independent channels must be used to either initiate or complete the voting process.

Paul et al. [18] suggest an approach that uses visual cryptography, similar to a voting system proposed by Chaum [9]. The voter receives by regular mail a piece of paper which is physically placed on top of the computer screen in order to decipher an image sent by the server. The image contains the "encrypted" list of candidate. To cast a ballot, the voter clicks on the name of the candidate. Because the candidates are listed in a random order and the piece of paper is needed to decipher the order, no malware can modify the voter's vote or find out how she voted.

Kutylowski and Zagórski [14] also suggest sending the voter a code sheet by mail. The main idea is based on code voting: the voter gets a code book via an Internet independent channel, and she enters the codes associated to the candidates they want to vote for. This way the voter's computer does not get to see the candidates that the voter selects, and cannot generate valid codes if it wants to switch the voter's vote to another candidate.

Adida proposes Helios [1], a Internet voting system that does not use a secondary channel. He acknowledges that Helios does not protect against potential malware on the voter's computers. Thus the virus can both find out how the voter wants to vote, and switch her vote to another candidate, as demonstrated by Desment and Estehghari [12].

Remote supervised voting over the Internet using dedicated voting kiosks has been another recent proposal [11]. In this setting, the kiosk is similar to a regular voting machine, the only difference being a live connection to the voting server via the Internet. The kiosk is under the control of voting officials and kiosk workers ensure that the voter is alone when casting a vote.

Systems proposed by Oppliger [17], and Popoveniuc and Vora [21], use CAPTCHAs (Completely Automated Public Turing tests to tell Computers and Humans Apart [26]) to create a secure channel from the voting server to the human voter. The Internet and the voter's computer do not have access to the data from this channel because it is assumed that the CAPTCHA is secure. These systems are the closest to SpeakUp, which also uses this channel to communicate information from the server to the human voter.

For the scope of this paper, we refer to Internet voting as being the process by which the voters communicate their intentions to some election server using their own equipment and the Internet only. No other communication channel should be used during ballot distribution or vote casting (i.e., no phones or postal mail). Internet voting is also viewed as an unsupervised activity. Figure 1 shows the general architecture of an Internet voting systems, including the legitimate users of the systems and the attackers.

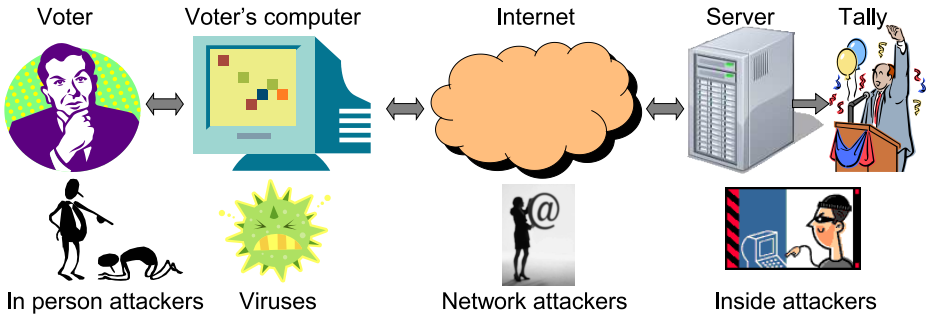

Fig. 1. General architecture for Internet voting, including the legitimate players and the potential attackers

1.2 Assumptions and properties

While do not perform a complete, formal analysis of the security offered by SpeakUp, we do try to present an intuitive explanation of why SpeakUp has some of the properties which are desired for an Internet voting system. These properties are based, in part, on the following assumptions:

1. There exist a class of problems which can be generated by a computer, such that, given a problem, it is easy for a human to find its solution, but it is essentially impossible for a computer program to find its solution. We refer to this general class as CAPTCHAs.
2. Given a set of CAPCHAs and a solution which is known to be for one of the CAPTCHAs, a computer program cannot associate the given solution to any of the CAPTCHAs, with a probability better than random guessing.
3. Given a sample human voice, a fixed text and a human voice reading the given text, there exists a computer program which can say if the two voice samples come from the same human or not (speaker verification).
4. It is difficult for a computer program to synthesize the voice of a given human reading a given text.
5. There exists a computer program that takes as input a set of texts and a voice recording of a human reading one of the given texts, and outputs the text which is read in the recording (voice recognition).
6. The list of registered voters is accurate and public.
7. The voters' computers have a working microphone.

SpeakUp offers the following properties:

1. Voters can cast their ballots using their personal computers connected to the Internet.
2. No computer program can modify the vote cast by the voter, regardless if it is running on the voter's computer, on the server of the election authorities or on any other computer.
3. No human can modify the vote cast by the voter. In particular, election officials cannot modify or inject votes.
4. Only authorized voters can cast ballots. In particular, an authorized voter cannot pass her voting credentials to another entity. Also, the election officials cannot stuff the ballot box.
5. No computer program can figure out how the voter voted without the help of a human, with one exception: the server of the election authorities can figure out how a voter voted.
6. The voter cannot loose or forget her voting credentials.
7. SpeakUp can be combined with a universally verifiably tallying method (e.g. decryption mixnet [10], homomorphic tallying [2], punchscanian mixnet[19]) which allow anyone to check that all the votes have been tallied as recorded.

2 Protocol Description

The basic idea behind SpeakUp is the construction of two channels, one from the server of the election authority (a machine) to the voter (a human), and one from the human voter back to the server. These two channels can be divided up into six separate sub-channels, as shows in Figure 2:

1. from the voting server to the voting application on the voter's computer
2. from the voting application on the voter's computer to the voter's screen
3. from the voter's screen to the human voter
4. from the human voter to the input device of the voter's computer
5. from the input device to the voting application on the voter's computer
6. from the voting application on the voter's computer to the voting server

A seventh channel is from the voting server to the announced final tally. Figure 2 shows how these channels can be protected and what channels are secured by SpeakUp.

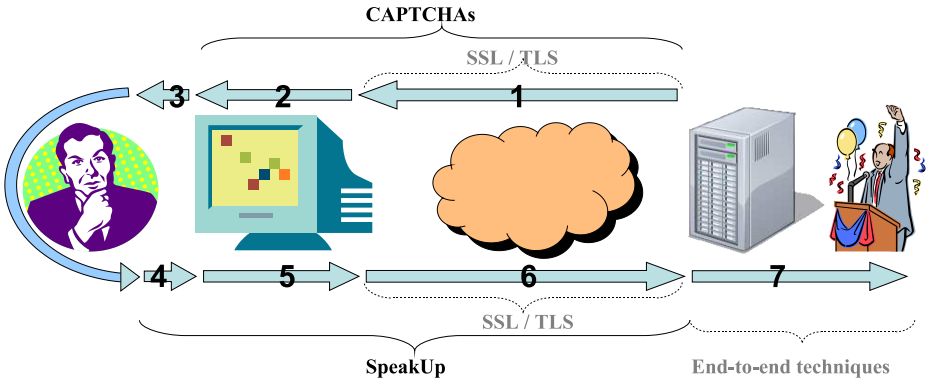

Fig. 2. Channels in an Internet voting system.

2.1 Voter Registration

To enroll in SpeakUp, the voter goes in person to a registration authority and presents the required identification documents. She selects the language or languages that she believes she will use when casting a ballot. She is then asked to read out loud into a microphone a fixed text in the language she selected (if more than one language, she is asked to read one text per language). This reading takes place in a quiet room, with ideal acoustic conditions. This constitutes the fingerprint of the voter's voice. All voters may be required to read the exact same text.

The registration authority stores the voice template and allows the voter to verify it from time to time (e.g., every couple of years) to see that the voice characteristics did not change. The voter may be required to re-register in person when the voice template that the election authority has becomes obsolete. However, the voter does not need to re-register in person if she moves to a different electoral jurisdiction, as the new jurisdiction may be able to obtain the existing voice template from the previous jurisdiction. The voice templates may be made public by the registration authority, such that anyone can access them.

2.2 Ballot Preparation

Each election authority prepares a sufficient number of ballots. On each ballot, a random set of words are associated with each possible selection (e.g., each candidate in each race). This general technique is often called *code voting*.

Each ballot may be identified by a unique serial number. The election authority encrypts each ballot with a different symmetric key and publishes all the encryptions (or, more generally, a commitment for each ballot is published).

2.3 Ballot Casting

Using her own computer (or some other personal computing device, like a smartphone, PDA, etc.) the voter goes to a web page that is publicly known to belong to the election authority responsible for her electoral jurisdiction (e.g., the local county webpage). The voter types in her full name and address, and selects the language she wants to vote in. The server checks if the selected language is valid for the given name and address. If not, the voter is informed and the process is aborted. Otherwise, the server locates the ballot style specific for the voter.

The server sends the voter a ballot constructed in the following way: next to each candidate, there is a moderately short text consisting of valid words from the dictionary corresponding to the language the voter selected. The association between these random looking texts and candidates has been previously committed to, by publishing the encryptions of the ballots. The random words should be unpredictable to anyone else but the server. The text next to each candidate is written into a CAPTCHA. Each ballot may have a unique serial number. A sample ballot is presented in Figure 3.

To cast a vote for a candidate, the voter is asked to read out loud into the microphone attached to her computer the text next to the candidate for whom she wishes to vote, along with the serial number of the ballot and the race identifier. For example the voter reads out loud: "Serial number 4711, presidential race: *exactly telephone valley group*". Note that she does not read the name of the candidate she wants to vote for.

The audio recording that is captures is sent to the election server via the voter's computer and the Internet. Three things happen:

1. speaker verification: the server retrieves the template provided by the voter when she registered and compares it with the received recording. Using a

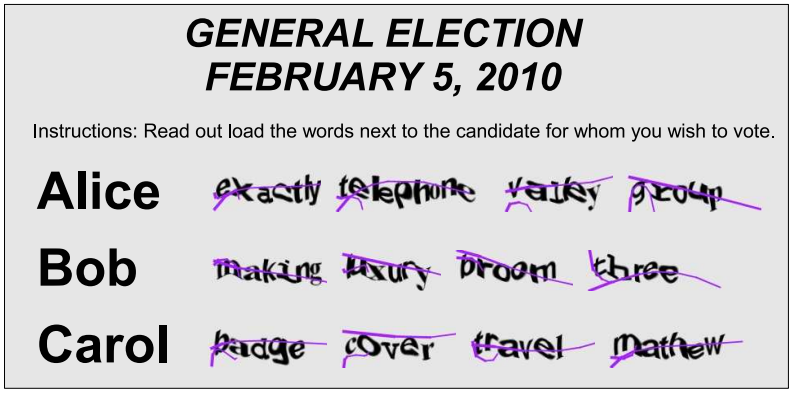

Fig. 3. Sample SpeakUp Ballot Screen

speaker verification technique the voter is authenticated [15, 25, 16]. If the speaker verification fails, the server informs the voter and aborts the process.
2. text identification: the server identifies the race, and matches the text read by the voter to one of the texts that were send to the voter. If analyzing the voice of the voter results in no match, it informs the voter (and maybe allows her to read the text again, up to a maximum number of times).
3. receipt publishing: the server publishes the audio recording it got from the voter on a public bulletin board, next to the identity of the voter and the ASCII text of how the voice was interpreted (but not what candidate was associated with that text).

The voter is not given any explicit receipt by the server, but her published voice is an unforgeable receipt of her cast ballot.

2.4 Checking the Bulletin Board

At any time after ballot casting, the voter can check the public bulletin board. There are three scenarios:

1. her voice is correctly posted on the bulletin board. In this case, the voter just checked that her vote is recorded correctly and does not have to do anything else.
2. her voice is not posted on the bulletin board. In this case, the voter tries to vote again. The voting system should allow her to cast a new ballot. A denial motivated by double voting is not possible, since the first vote is not posted on the bulletin board.
3. her voice appears modified on the bulletin board, i.e., other words are pronounced. Based on the security of voice biometrics, this should be very difficult to achieve. This case is discussed in more detail in section 4.

Anybody can check that all the voices posted on the public bulletin board sound human and do correspond to the the clear text that is posted next to them. If a

voice does not sound human, a complaint can be filed and further investigations can be done. An automated way of distinguishing human voice from computer generated noise is an interesting future research topic.

As discussed below, the security of SpeakUp is derived in part from the security of biometric authentication based on the voter's voice. As long as voice biometrics are considered secure, SpeakUp is resistent to ballot injection or modification attacks.

3 Potential Attacks and Informal Security Analysis

We differentiate between three types of attacks:

1. automated attacks
2. human attacks
3. insider attacks

Automated attacks have a very large potential impact, since a single attacker or a small coalition can mount automated attacks that have very large effects and can change the outcome of the election. SpeakUp is primarily designed to deter automated attacks.

Human attacks are usually on a much smaller scale. SpeakUp addresses some of the human attacks, such as the voter selling her voting credentials, but does not address the basic problem of in person coercion. All remote voting systems currently in use (e.g. Internet, postal mail, telephone, fax etc.) are vulnerable to in person coercion.

Insider attacks are attacks which come from within the voting that is under the control of the election officials. They have been addressed lately by end-to-end verifiable techniques such as Prêt à Voter[8], Scratch&Vote[2] or Punch-Scan[19]. SpeakUp is an alternative front-end [20] (a way of capturing a coded vote) for such end-to-end verification techniques, and is compatible to any back-end that allows for the detection of insider attacks.

We only look at attacks that target the integrity of the reported tally and the confidentiality of the cast ballots. Attacks against availability (e.g. denial of service attacks) or attacks aimed at biasing voters towards a particular candidate via methods that are independent of the voting technology (e.g. electoral pop-ups), are outside the scope of this paper.

3.1 Automated Attacks

One unique threat to any form of electronic voting is that of automated, large-scale attacks. The scale of such attacks for Internet voting systems can be much greater than for poll site voting, as a single attacker can impact a very large number of voters or jurisdictions.

Suppose there is some malware installed on the voter's computer. The voter either does not know or does not care. While operating antivirus software is useful, we cannot assume that it is up-to-date, or even that it has the ability to

identify any piece of malware on the system that may attempt to disrupt the election. We will assume the malware has unrestricted access to all the data the computer has access to.

There are two tricks to make the vote casting technique resistant to malware-based attacks:

1. make the data processed by the computer incomprehensible to any malware, while at the same time the data makes perfect sense to the human voter.
2. design the cast ballot in a way that only a legitimate voter could have produced it.

In effect, a channel is constructed between the voting server and the voter itself, instead of being between the voting server and the voter's computer (classically done by SSL/TLS or some other form of encryption). This is done via CAPTCHAs and voice biometrics.

CAPTCHAs are essentially puzzles that humans can (hopefully easily) provide the solution to, while the same time being very difficult for a computer to solve. While we use CAPTCHAs in which voters are required to "decipher" a wavy text, this should only be viewed as an example of a way to "tell computers and humans apart".

The malware may have two targets:

1. Determine which candidate the voter cast a vote for. This is a privacy violation.
2. Using voter's credentials, it wants to cast a vote different from the one the voter intends to cast, and not get detected. This is an integrity violation.

A third possible target may be not to allow the voter to cast a ballot at all, regardless of the vote the voter wants to cast. This is a non-adaptive denial of service attack. This can be done by blocking some network messages, bombarding the voter with a large number of pop-ups, resetting the machine, etc. SpeakUp does not protect against denial of service attacks.

Ballot Secrecy A piece of malware on the voter's machine may attempt to determine for whom the voter is voting. The goal of the attacker may be to conduct a selective denial of service attack if that selection is not in favor of a particular candidate, or it may be as part of an attempt to coerce the voter into voting a particular way.

Ballot secrecy in SpeakUp depends on the security of the CAPTCHA. Cast votes in SpeakUp are audio recordings encoded using code words hidden in a series of CAPTCHAs. To violate ballot secrecy, a piece of malware must match the words spoken by the voter to the code words of a particular candidate. This requires the malware to process both the spoken words of the voter using voice recognition, as well as the code words hidden in the CAPTCHA.

Advances in reliable voice recognition technology have made it technically feasible to recover the spoken code words, leaving the CAPTCHA as the only defense. The CAPTCHAs are constructed to make it difficult for computers to

find their solutions. Given a set of problems and a solution known to belong to one of the problems, associating the solution with the problem is a somewhat different property. This is an interesting research questions in itself and a good area for future work.

Note that the voting server only needs to match the voter's voice to a small set of known texts, whereas a virus must contain a full blown voice recognition engine, since it needs to recognize random words the voter is pronouncing.

Vote Stealing We note that the words next to the candidates are embedded in CAPTCHAs, which by definition, are used to tell computers and humans apart. Humans can (hopefully easily) provide the solution to the capcha, while the same operation should be difficult for any computer program, and thus for the malware too. Therefore the malware should not have access to the text contained in the CAPTCHAs. To try to cast a ballot for a candidate other than the voter's chosen candidate, the malware has to break the CAPTCHA.

Even if we assume that the malware breaks the CAPTCHA or otherwise gains access to the text encoded in the CAPTCHA, the malware would have to synthesize the voter's voice for that particular text. SpeakUp assumes that voice is used as a biometric identification technique, i.e. given a random person (a voter able to speak) it is difficult for a computer program to generate an audio recording that represents the voice of that person while reading some given random text. As long as biometrics based on voice verification are considered secure, SpeakUp inherits its security properties.

Voter Impersonation The speaker verification algorithm used to match the audio recording from the voter to the voice sample captured during registration should be public. A computer program, knowing the algorithm that is used, and having access to the text that should be read (by breaking the CAPTCHA), can try to produce an audio file that would fool the matching algorithm into thinking that the voter is the one reading it. The computer generated audio file will most likely sound very different from a human voice. While this may seem like a reasonable attack, SpeakUp publishes all the audio files on a public bulletin board. Since anyone can listen to the files from the bulletin board, such strange sounding files would be detectable.

An option that would automatically filter these strange sounding files is to have a computer algorithm on the server that is able to differentiate between a computer generated voice and a voice produced by a human. This is left as an open research question.

Vote Flipping The malware on the voter's computer may try to trick the voter into casting a vote for a candidate chosen by the malware instead of the voter. The assumption is that the malware knows how the voter is going to vote. It may have this information from tracking the web pages that the voter visits, or from seeing the donations that the voter made for a particular candidate (yes, this

would be a very knowledgeable virus). The malware can take the CAPTCHA next to the malware's candidate and put it next to the candidate that the voter is likely to vote for.

Fig. 4. Sample Meta CAPTCHA Ballot Screen

To avoid such an attack from the malware, a meta CAPTCHA can be constructed: a CAPTCHA that contains CAPTCHAs, such that no software can rearrange the individual CAPTCHAs. For example, there may be some structure in the first letters of all CAPTCHAs (alphabetically ordered), or all CAPTCHAs in a single race may be placed on common meaningful background (e.g a picture of a landscape, as shown in Figure 4). The malware cannot clip only the text from the CAPTCHA and paste it somewhere else in the background picture, because it will not fit in the new place. Trying to clip a regular area (such as a rectangle) is going to perturb the background and the voter may notice it.

Like in any other voting system where the order of the candidates is fixed, we assume that the order of the candidates is publicly known (perhaps alphabetical) and the voters would notice if the order of the candidates is different from the official posted one. This would prevent the malware from rearranging the candidates.

This meta CAPTCHA technique allows for deterring attacks against the integrity of the election (the malware tricking the voter into voting for some candidate), but does not address the privacy concerns: the malware can still recover the text from analyzing the voter's voice and, it may be able to associate this text with the a candidate (by breaking the assumption that it is difficult for a computer program to associate a given solution to a problem from a given set of problems). To address this somewhat complicated attack on privacy, we describe a technique based on indirection. For example, next to each candidate there can be a picture containing some objects. Bellow the list of candidates, there can be a set of CAPTCHAs, each containing the words that identify the objects in the picture. The order of the CAPTCHAs is different from the order

of the pictures (and different for each ballot). Figure 5 shows an example. This is very much like PunchScan [19] ballots looks like. To vote, the voter would first look at the picture next to her favorite candidate, and read out loud the CAPTCHA that contains the list of objects in the picture. The malware has access to the list of objects, but is going to have a hard time associating this text to one of the pictures and therefore to one of the candidates. This protects the privacy of the vote in the case in which the virus is able to recognize the text from the human voice and match it with one of the CAPTCHAs (a fairly complicated attack).

The above suggestion is only a particular example of a technique that can be very general: have a concept next to each candidate and then a list of related concepts in some random order from which the voter can choose. It should be difficult for any computer program to associate the voter's choices with the concepts next to the candidates, partially because of the difficulty to associate the concepts in the first list with the concepts in the second list.

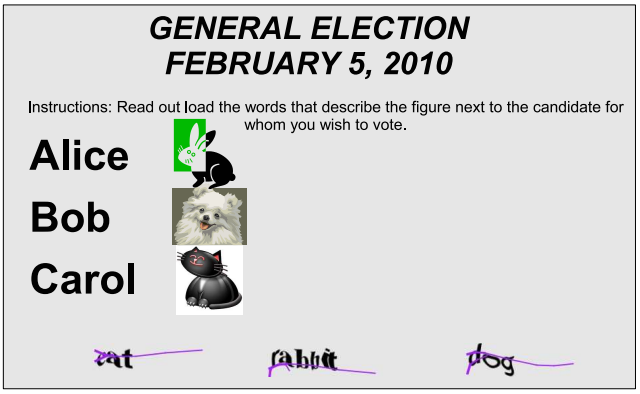

Fig. 5. A SpeakUp ballot with indirection

Another simple technique which would allow the voter to detect if the virus re-arranged the CAPCTHAs is to have a meaningful text in the presented CAPTCHAs. Re-arranging the CAPTCHAs would make the text meaningless and the voter would detect it. For example, a meaningful paragraph (e.g. taken from a literature book or provided by reCAPCTHA) could be broken up into four word segments and each segment is placed next to a candidate in order. Poems may also be useful. Note that, given a set of words, it should be difficult to figure out if the words come from the beginning, the middle or the end or a meaningful text.

3.2 Human-Aided Attacks

The malware may try to communicate the data it sees to some remote location where a number of humans are actively trying to attack the system (a

CAPTCHA farm or Mechanical Turk). The humans from the farm can solve the CAPTCHAs and find out how the voter voted. However, the voter's vote cannot be modified, since the voice of the legitimate voter is used as a biometric way of authentication.

If the virus does not show the ballot to the voter immediately, but instead sends the ballot to the farm, the farm worker can decode the CAPTCHA and generate a new, farm-brewed ballot. Assuming the virus knows for which candidate the voter wants to vote for, the new ballot can re-assign the text that was initially associated to the attacker's favorite candidate, such that now it appears next to the voters favorite candidate. The voter would be tricked into reading the text, thinking it casts a vote for her favorite candidate.

3.3 Insider Attacks

Election officials and other users that have special privileges in administering and running the election may themselves be the source of attacks which can target ballot confidentiality or election integrity.

Ballot Confidentiality To be able to deliver the correct ballot format to the voter, the voting server (viewed as a general entity) must have access to the voter's identity, as well as the voter's voice template used for biometric identification. The server also has access to the clear text ballot (the solutions to CAPTCHAs) to be able to check the solutions that the voter provided. Therefore the server may have access to the association between the voter's identify and her choices.

The voting server has to be trusted for ballot secrecy. Solving this problem remains an open research question.

End-to-End Cryptographic Voting Protocols We now focus on the integrity aspect. Insiders may modify the cast ballots and produce a tally that does not reflect the sum of the validly cast votes. Such attacks are addressed in currently deployed voting systems by extensive checking and certification programs. Such programs can be extended to the voting servers, as they are under the control of the election authority.

In this paper we go one step further and show that some existing techniques such as end-to-end verifiable voting systems, can be easily applied to SpeakUp to protect the integrity of the votes and to address the integrity threats that come from insiders. It is common to decompose the end-to-end verifiability in three parts: cast as intended, recorded as cast and counted as recorded. We address each part, in the order of difficulty.

Recorded as Cast Verification To check that her vote is recorded as cast, SpeakUp allows the voter to check that her voice is correctly posted on a public bulletin board. She can complain if it isn't. The voter cannot issue false complaints, since her voice is a biometric way of authenticating her and it is difficult

for someone else to have generated it. SpeakUp is different in this aspect from all the other end-to-end systems: instead of putting the burden of proof on the voter to produce the valid receipt (e.g. signed by the election authority), the burden of proof is now on the election authority to prove that the voter's voice corresponds to the sample voice obtained at registration. This comparison can be done by any member of the public if the voice templates are public.

Counted as Recorded Verification To check that all the posted votes are counted as recorded, a publicly verifiable tallying method is used. Most example of such techniques from the end-to-end voting literature are compatible with SpeakUp (homomorphic schemes [4], generic decryption mixents [10], re-encryption mixnets [24], punchscanian mixnets [19], etc.). It is out of the scope of this paper to describe such techniques.

Cast as Intended Verification One possible scenario is that the server of the election authority may send the voter a ballot that has the texts associated differently than it was committed to ahead of time. This is similar to misprinting a ballot in some end-to-end voting systems [2],[6]. The typical solution is to allow the voter to get two ballots, one to cast, and one to spoil. The spoiled one is going to be checked that it is well formed (i.e. according to the commitments previously made). We present a similar technique adapted for a remote setting. The techniques is partially inspired by eScantegrity [7].

During the ballot preparation, the election authority published the encryption of each ballot on the bulletin board. The voter is allowed to choose any two unused ballots from the bulletin board and the server sends the voter both ballots (in clear text). This means that the voter gets the ordered list of CAPTCHAs for both ballots. The ballots are not signed, so the voter cannot prove to anyone it got them from the server.

Using an 1-out-of-2 oblivious transfer protocol [22], the voter gets from the server one of the two keys used to encrypt the two ballots (the encryptions were posted on the bulletin board). The voter uses the key it got during the oblivious transfer to encrypt one of the two clear text ballots and checks that the encryption corresponds to the one posted on the bulletin board. If this is the case, the voter reveals to the server the key it got from the oblivious transfer protocol and uses the other ballot (the one for which the key was not obtained by the voter) to cast a vote. Note that the server may still have cheated on the unchecked ballot, with a probability of 50%. Cheating on many ballots (i.e. on many voters) becomes impractical because of the probability of not being detected drops exponentially with the number of ballots cheated on.

4 Additional Properties of SpeakUp

We briefly mention some desirable properties for any Internet voting system, such as voter authentication, resistance to vote selling and protection against ballot box stuffing.

4.1 Voter Authentication

SpeaksUp addresses the problem of voter authentication using speaker verification. This is a particular form of biometric authentication, which has some advantages. First, the voter cannot give her voting credentials to someone else. Unlike shared secrets or security tokens, it is difficult for a voter to lend her voice to someone else. Since the authentication mechanism uses a challenge-response protocol (the server gives the voter randomly-generated prompts), recording some authentication phrase and trying to replay it will not be an effective attack. Therefore an attacker cannot collect credentials of voters and use them to cast votes at will.

Providing a simple technique for the authentication of remote voters is of value in itself. Speaker verification seems to be the easiest out of the possible biometric authentication choices. Fingerprints, retina scan or DNA matching are not challenge response protocols in themselves, thus replay attacks are possible. In a challenge-response protocol, like SpeakUp, the virus cannot capture the authentication credentials and reuse them in a latter session. Moreover, capturing such biometrics from the voter implies a specialized reader attached to a voter's computer. It is unlikely that the voters have such readers.

Speaker verification biometrics are simple to capture, since many of the personal computing devices are equipped with microphones (phones, PDA's, many laptops) and microphones are usually easily added if they are missing.

Speaker verification is an area of ongoing research. Current implementations of speaker verification products may be acceptable for immediate use [15]. In speaker recognition, the voice sample is compared against a large number of samples and the closest one is identified. In speaker verification, a voice sample is compared with another voice sample and the output is a binary value: the two samples have been produced by the same person or not.

While the election authority does not have a pre-recording of the text the voter is reading when casting a ballot (which is the case in text dependent speaker recognition), the election authority expects to hear a text from a very small set of possible texts which are chosen by it. Thus the election authority knows what text the voter is reading.

One example where speaker verification is currently used in practice is authentication of welfare recipients undertaking telephone transactions at Centrelink in Australia [25].

This relatively simple authentication technique solves an important problem in remote voting. While some voters may have government-issued smart-cards (e.g. Estonian citizens), others may not have any sort of government ID (e.g. some U.S. citizens). Voice verification seems to be something that is already "deployed" (something that voters already have), handy, cheap and user friendly.

Moreover, managing voter credentials such as voice templates should be easier that managing usernames and passwords, since voter may loose their username or forget their password. Doing password recovery may be either unsecure, expensive, or unfriendly. It is less likely that the voter looses her voice.

One way to lower the error rate for speaker verification is to capture a video of the voter while providing the solution to the CPATCHA, and to use face recognition techniques to match the picture of the voter (e.g. from her driver licence) to the face in the video. We suggest that this should be done in addition of speaker verification, and not as a replacement. Alternatively, instead of video, a set of still photos can be taken at random moments in time while the voter is working on trying to cast her ballot. Most laptops and smart-phones now come with incorporated cameras. More advanced techniques may detect if the way the voter's mouth and lips are moving are consistent with the words captured via the microphone (a liveness test).

4.2 Resistance to Vote-Selling

In a remote setting, the voter may be tempted to give her voting credentials to someone else. She may not care to vote at all, she may gain something, or she may avoid punishment. Using SpeakUp, the voter cannot give her voting credentials to someone else, since she authenticates herself by using her voice in an challenge-response protocol. The voter cannot give a recording of her to someone else since she is is asked to read a random text that changes every time.

To coerce the voter, the attacker has to either be physically next to the voter. This small scale coercion is still possible with SpeakUp.

4.3 Layered security

To protect the integrity of the vote, SpeakUp offers a layered security approach by combining CAPTCHAs with biometrics. A virus on the voter's computer would have to break both techniques to be able to cast a vote different than the one the voter intended. If only the CAPTCHA is broken and the virus can recognize the text the voter is pronouncing, privacy is compromised, but integrity isn't because the malware is not able to synthesize the voter's voice to be able to impersonate the voter. If the virus is able to synthesize the voter's voice but cannot break the CAPTCHA, it is not able to find what text is next to its favorite candidate and thus what text to synthesize.

4.4 Accessibility

Voters who are visually impaired will not be able to read the text next to the candidates. Voters that cannot speak (or have speech impediments) will not be able to read the text out loud . For the first category, audio CAPTCHA can be used.

Voters with speech impediments may use a touch-screen to handwrite the text corresponding to her favorite candidate. Instead of doing speaker verification, the server will do handwriting verification (this is arguably more difficult to do from a technical point of view). In the registration phase, voters are asked to write by hand a given text (instead of reading it out loud). Their handwriting specimen is compared with the handwriting they provide when the vote is cast.

Another option for voters that cannot speak is to use sign language. Their personal computers must be equipped with a web camera and the sign language is interpreted by a human at the receiving side. The movie captured is published on the bulletin board and can be checked by anyone. The movie must also capture the face of the voter, such that the voter is identified. Interpreting the sign language and doing voter authentication can be done by human election officials and need not be automated.

5 Future Work

We have identified several general research areas that are not necessary particular to SpeakUp, but would contribute to a better security of the system:

1. an automated way of distinguishing if a given audio recording was generated by a computer or by a human.
2. better ways to distinguish humans from computers, i.e. better CAPTCHAs.
3. preventing denial of service attacks and/or quickly recover from such attacks

We note that recently, there appeared some attacks [29, 28] that solve with some probability some forms of visual CAPTCHAs based on the user recognizing the distorted text on an image. Other attacks have been aimed at audio CAPTCHAs [27]. At the same time, new forms of tests that are claimed to be solvable only by humans appeared [3]. We urge the reader to treat the specific CAPTCHAs used by us only as examples of problems which can be easily solved by humans, but are difficult for the computers to solve. Such techniques may be radically different from reading some distorted text. We expect that any implementation will use the most secure CAPTCHAs available at the time an election is run.

There is also SpeakUp specific research that needs to be addressed by future work. First, an implementation of the protocol along with a performance measure that would include the rate of false positives and false negatives for speaker verification with a known text. In general, the evaluation process [15] has been focused on text independent speaker verification rather than speaker verification with a known text.

Second, an usability study could focus on how difficult is for voters to complete a voting task using SpeakUp.

Third, a technique that would prevent the voting server from being able to link voters to votes is needed to protect the ballot secrecy from being breached by the voting server.

6 Conclusions

We presented SpeakUp, a voting system that goes back to voting via voce. Speaker verification along with voice recognition are used to authenticate a voter and identify the vote she wants to cast. Even if the voter's computer is infected

with viruses, it cannot synthesize the voice of the voter for the random text that is associated with a candidate, and thus cannot modify the voter's vote.

From the available biometric authentication techniques, speaker verification was used because the challenge response nature of it ensures that replay attacks are not possible.

SpeakUp can be coupled with any end-to-end publicly verifiable scheme that allows voters to check that their vote is recorded correctly and allows everyone to check that all votes were correctly tallied.

References

1. B. Adida. Helios: Web-based open audit voting. In *Proceedings of the Fourteenth USENIX Security Symposium (USENIX Security 2008)*. Usenix, July 2008.
2. B. Adida and R. L. Rivest. Scratch & Vote: self-contained paper-based cryptographic voting. In *WPES '06: Proceedings of the 5th ACM workshop on Privacy in electronic society*, pages 29–40, New York, NY, USA, 2006. ACM Press.
3. alipr.com. Next-generation captcha exploits the semantic gap. `http://tech.slashdot.org/article.pl?sid=08/04/23/0044223` [Online; accessed 5-February-2010].
4. J. C. Benaloh. *Verifiable Secret Ballot Elections*. PhD thesis, Yale University, 1987.
5. California Secretary of State. California Internet Voting Task Force Report. `http://www.sos.ca.gov/elections/ivote/`, 2000.
6. D. Chaum, R. Carback, J. Clark, A. Essex, S. Popoveniuc, R. L. Rivest, P. Y. A. Ryan, E. Shen, and A. T. Sherman. Scantegrity ii: End-to-end verifiability for optical scan election systems using invisible ink confirmation codes. In *EVT'07: Proceedings of the USENIX/Accurate Electronic Voting Technology on USENIX/Accurate Electronic Voting Technology Workshop*. USENIX Association, 2008.
7. D. Chaum, S. Popoveniuc, and P. Vora. eTegrity and ePunchScan. *End-to-End Voting Systems Workshop*, Oct. 2009.
8. D. Chaum, P. Y. A. Ryan, and S. Schneider. A practical voter-verifiable election scheme. In *In Sabrina De Capitani di Vimercati, Paul F. Syverson, and Dieter Gollmann, editors, ESORICS, volume 3679 of Lecture Notes in Computer Science*, pages 118–139. Springer, 2005.
9. D. Chaum, J. van de Graaf, P. Y. A. Ryan, and P. L. Vora. Secret ballot elections with unconditional integrity. Technical report, IACR Eprint, 2007. `http://eprint.iacr.org/` or `http://www.seas.gwu.edu/~poorvi/cgrv2007.pdf`.
10. D. L. Chaum. Untraceable electronic mail, return address, and digital pseudonym. *Communication of ACM*, February 1981.
11. A. Contorer. Secure remote voting for overseas and disabled votersw. `http://www.everyonecounts.com/uploads/SecureRemoteVoting-NASS.pdf`, 2009.
12. Y. Desmedt and S. Estehghari. Hacking helios and its impact. *Crypto Rump Session*, August 2009.
13. D. Jefferson, A. D. Rubin, B. Simons, and D. Wagner. Analyzing internet voting security. *Communications of the ACM*, 47(10):59–64, 2004.
14. M. Kutylowski and F. Zagrski. Scratch, click & vote: E2e voting over the internet. *End-to-End Voting Systems Workshop*, Oct. 2009.

15. NIST. The 2008 NIST speaker recognition evaluation results. http://www.itl.nist.gov/iad/mig/tests/sre/2008/official_results/index.html, August 2008.
16. NSTC Subcommittee on Biometrics. Speaker recognition. http://www.biometrics.gov/Documents/SpeakerRec.pdf, August 2006.
17. R. Opplinger, J. Schwenk, and C. Löhr. Captcha-based code voting. In *3rd International Conference on Electronic Voting*, EVOTE08, 2008.
18. N. Paul, D. Evans, A. Rubin, and D. Wallach. Authentication for remote voting. *Workshop on Human-Computer Interaction and Security Systems*, April. 2003.
19. S. Popoveniuc and B. Hosp. An introduction to PunchScan. In *IAVoSS Workshop On Trustworthy Elections (WOTE 2006)*, Robinson College, Cambridge UK, June 2006.
20. S. Popoveniuc and P. Vora. A framework for secure electronic voting. In *IAVoSS Workshop On Trustworthy Elections (WOTE 2008)*, Katholieke Universiteit, Leuven, Belgium, July 2008.
21. S. Popoveniuc and P. Vora. Remote ballot casting with captchas. In *3rd Benelux Workshop on Information and System Security*, Eindhoven, The Netherlands, November 2009.
22. M. O. Rabin. How to exchange secrets by oblivious transfer. http://eprint.iacr.org/2005/187.pdf, 1981.
23. A. Regenscheid and N. Hastings. NISTIR 7551: A threat analysis on uocava voting systems. http://www.vote.nist.gov/uocava-threatanalysis-final.pdf, 2008.
24. P. Y. A. Ryan. Pret A Voter with Paillier encryption. Technical Report CS-TR-1014, University of Newcastle upon Tyne, School of Computing Science, April 2007.
25. R. Summerfield, T. Dunstone, and C. Summerfield. Speaker verification in a multi-vendor environment. http://www.w3.org/2008/08/siv/Papers/Centrelink/w3c-sv_multi-vendor.pdf.
26. L. von Ahn, M. Blum, N. Hopper, and J. Langford. Captcha: Using hard AI problems for security. In *Proceedings of Eurocrypt*, pages 294–311, 2003.
27. WintercoreLabs. Breaking gmails audio captcha. http://blog.wintercore.com/?p=11 [Online; accessed 5-February-2010].
28. J. Yan and A. S. E. Ahmad. Is cheap labour behind the scene? - low-cost automated attacks on yahoo captchas. http://homepages.cs.ncl.ac.uk/jeff.yan/yahoo.htm [Online; accessed 5-February-2010].
29. J. Yan and A. S. E. Ahmad. A low-cost attack on a microsoft captcha. http://homepages.cs.ncl.ac.uk/jeff.yan/msn_draft.pdf [Online; accessed 5-February-2010].