Who counts your votes?

Halina Kaminski, Lila Kari, Mark Perry
Computer Science Department
University of Western Ontario, London, Ontario, Canada
{hkaminsk, lila, markp}@csd.uwo.ca

Abstract

Open and fair elections are paramount to modern democracy. Although some people claim that the pencil-and-paper systems used in countries such as Canada and UK are still the best method of avoiding vote rigging, recent election problems have sparked great interest in managing the election process through the use of electronic voting systems. It is a goal of this paper to describe a voting system that is secret and secure as well as verifiable and useable over an existing computer network.

We have designed and implemented an electronic voting system – Verifiable E-Voting (VEV) – with an underlying protocol that secures the election process from malicious practices at the same time as allowing voters and candidates to verify the correctness of their votes.

1. Introduction

Open and fair elections and their appearance as such, are paramount to modern democracy. The citizens of a nation only have an interest in voting because they believe that they truly determine by majority who will be their representatives. Loss of trust in the voting system can cause a whole country to lose faith in their system of government. Although voters and candidates predominantly try to secure the voting process, there is a belief that fraudulent elections have been conducted (e.g. Dominican Republic 1994, Malawi 1999, Sri Lanka 1999, Haiti 2000 [1]). While many computer scientists have been warning of the perils of the use of electronic voting systems, the vendors of such systems have continued to promote their products, claiming increased security and reliability. The quest to find a solution for a secure voting process has encouraged some people to jump on the electronic voting train. Electronic voting systems have much appeal. They can bring ease of use to voters, election candidates, and election officials. Electronic voting machines enhance the election process in the ways: they can handle multiple languages, they can easily adapt to the people with disabilities, and they provide faster results. Although some assume that a computerized voting process is totally reliable and will provide correct results as well as making it impossible to alter the count, this is not the case. As can be seen from the series of research studies as well as recently published discussions in the media, there is opinion that many of the positions put forward by supporters of electronic voting are far from the truth. As a simple example, it is possible that a computer can easily display one set of data to the voter while recording an entirely different vote. This could be caused by a programming error or by a malicious design of the system. In recent months electronic voting machines have been criticized by researchers who say that they can be interfered with to skew results, and it has been shown that there have been inaccuracies in vote counting records. Most of the machines in use leave no verifiable record of the transaction.

2. Overview

While some states trust the pencil and paper voting procedure, others have applied e-voting technology to their democratic elections. As early as 1990, Brazil employed electronic devices to help in the voting process. Ever since then their voting infrastructure has been improved resulting in all-electronic elections held in October 2002. These voting machines used Microsoft Windows NT as an operating system, and touch screen functionality featuring pictures of candidates. [2] Brazil's 2002 e-voting based elections were the world's largest– more than 115 million cast votes – until India's May, 2004 elections when more than half of 670 millions eligible voters cast their vote. India required over 725, 000 [3] voting machines to be delivered to their polling stations, not an easy task under any circumstances, but harder in a developing country.

The United States of America is another country trying to automate the election process. Over the last decade many resources have been dedicated to the provision of secure and reliable voting machines. In the 2004 presidential elections more than two thirds of eligible voters used some kind of electronic machine to cast their votes. Along with their convenience, these voting machines bring a litany of reports about existing problems. There is no major newspaper or magazine that has not published some account about the questionable reliability of direct recording electronics (DREs). For example, the Washington Post [4] reports that in Georgia, some voters found that "when they pressed the screen to vote for one candidate, the machine registered a vote for the opponent". In the state of Alabama, a 7,000-vote error was caused by a computer glitch. The most controversial

dispute occurred over the correctness of the count and inclusion of all voters' ballots in the 2000 U.S. presidential election in Florida. [5] Because the US citizens remember 2000 election debacle, their eyes are turned towards Florida today. But the situation is no different in Utah, Ohio, or any other state for that matter. According to Aviel Rubin, a computer scientist at Johns Hopkins University, "the use of computers puts elections at the mercy of a few companies that make the machines. The threat is that the vendors are in a position to make the election come out any way they want, and it's virtually undetectable". [6] Maybe the electronic processes will record votes correctly, maybe the president in the current round of US elections is the one the electorate wanted, but *maybe* is not good enough for voters to place their trust in a democratic system.

Printing a receipt for each voter might solve the problem. A description of a system that utilizes a verifiable paper trail in the voting process can be also found in [7]. Known as the 'Mercuri method' it outlines the steps that have to be taken in order to include verifiable records in a final tallying. Having printers installed with each voting machine would allow the voter to see that the machine understood his/her intention correctly, but it does not assure that the vote was counted correctly. Moreover, Mercuri's method does not prevent the counting of unauthorized votes nor does it provide a proof that all votes were included in a final tally.

3. VEV Verifiable E – Voting

To include a printer in each voting booth would be difficult to execute: it would require a lot of resources to install and support, and furthermore it would not increase the trust in a system. On the other hand it is clear that voters and candidates should have proof that their votes were counted correctly. We have designed and implemented an electronic voting system – Verifiable E-Voting (VEV) - that allows voters and candidates to verify correctness of their votes, while at the same time the system's underlying protocol secures the election process from malicious practices.

The following voting system requirements were born out of the desire to create a product that would allow modern computer-based technology to truly embody those secure properties that are valued by the public in their voting. The purpose of our work is to make it impossible for voting authorities to engage in a fraudulent behavior, and at the same time the system will provide the secrecy for the voters. It has been an attempt to provide a voting system that would be:

anonymous – only voter knows his/her vote;

correct - it should not be possible for a vote to be altered, or for a valid vote to be eliminated from the final tally, or for an invalid vote to be counted in the final tally;

honest - no one should be able to vote twice or change the vote of another voter;

public - all results should be publicly known, but the connection between votes and the voters should be both **unprovable** and **unknown**. [9]

3.1 Specification of VEV

There are a number of conditions that have to be met in order to provide voters with a secure electronic voting system. Here we include the description of the general steps that need to be taken in the design of the system to provide the user with voting security. The general idea that specifies the underlying protocol for the VEV design has been introduced in [8]. The requirements for the system follow.

Voting takes place over a computer network

VEV is designed to be implemented and used over an existing computer network. The system includes three major parts: the server-side program, client (voter) side application and administrator (administrative user) side software. The server-application should be stored and executed on the main network computer. The client-application could be located either on the main computer or on every network's terminal. It is recommended that, for the security reasons, the administrator's application should be stored on the removable storage device (such as floppy, CD), kept secure, and run only when changes are being made to the voting procedure.

Only authorized voters can vote

Every voter is assigned a username and a unique password. The administrator is responsible for choosing the appropriate values for the name and password, since it depends on the election importance as well as the election settings.

The voter can cast only one vote

It is important for the system to ensure that it allows each voter to cast one and only one vote. VEV also provides the option of a re-vote to each user, and thus it is important that the previous vote cast by a particular voter will be erased when that user votes again.

Only the voter can know his/her vote

In democratic elections only the voter can know his or her voting strategy: This is the secrecy requirement. There cannot be a traceable link left between the voter and the vote, and all processing links should disappear. It should be impossible for anyone to recognize the voter by looking at the ballots cast.

Each voter can check if his/her vote was counted

VEV offers a great enhancement to usual electronic voting processes, in that every user can check if his/her vote is in the ballot (which means it has been counted). The system will provide the option to check the votes (check the ballot), and the voting strategy identification will be displayed. This means that users can count the votes that were cast and can recognize their own vote among the displayed votes.

Voters can change their minds

When the election process progresses, an individual voter can become aware that the desired candidate was not voted for, and thus the system provides the option to re-vote. The system allows users to change their minds multiple times, as VEV supports a multiple re-vote function.

4. The underlying protocol

VEV uses the public-private key paradigm to encrypt information. In this system, the user's identification number (id) and the voting strategy number (v) (which is a numeral representation of the candidate's name) are the two prime numbers that are being used. There are three different algorithms designed to perform calculations with these two prime numbers and returning one large number as a result. It is randomly chosen in the program which one of the three algorithms is used when the voting is performed.

4.1 The algorithms

Function 1

The first function uses the multiplication function as the underlying calculation. As a result, the product of two prime numbers is returned.

Function 2

This function first calculates the product of two prime numbers. Then it swaps the values of the individual bytes within the binary representation of the product (namely copies the value of last byte into the byte before the last, and the value of the second last byte into the last byte). The same swapping operation is done to the third and fourth last byte of the product.

Function 3

This function first calculates the product of two prime numbers. Then it flips (replaces with the complementary value) the values of the individual bits within the binary representation of the product. The algorithm changes the values of bit positions: 3,6,7,12,15.

The fact that both of the prime numbers are randomly generated for each user and for each voting strategy provides great security for the system. The standard RSA cryptosystem uses the same p and q throughout its lifetime where in VEV the probability that the same two numbers will be used twice is very close to zero. The major part of the private key constitutes the fact that there exists a system-defined index that uniquely identifies each candidate. Even if the intruder is able to factor the voting strategy function result, having two prime numbers would not give him any reasonable answer. The secret lies in the knowledge of indexing the candidates and having the function inverses. For this particular reason the usage of 25-bit long prime numbers provides sufficient security to the voting system. The prime numbers are being generated using the constructor for *BigInteger* class from the Java programming language library. The method returns a randomly chosen, 25-bit long positive integer that is a prime number. The probability that the newly generated number represents a prime number will exceed $(1 - 1/2^{100})$. The execution time of this constructor is proportional to the value of the probability parameter (in our system the probability parameter is 100). In addition, each newly created number is checked once again by *isPrime*() function from the Java class library.

5. The Voting Scenario

In the remainder of this paper, 'voter' and 'user' are synonymous, 'server' is used to describe the software implemented and executed on the network's main computer and 'client' represents the computer program that provides the graphical interface to the user, and allows for communication between the server and the voter.

5.1 Phase 1: Preparation

VEV publishes the number of eligible voters and the deadline for the response.

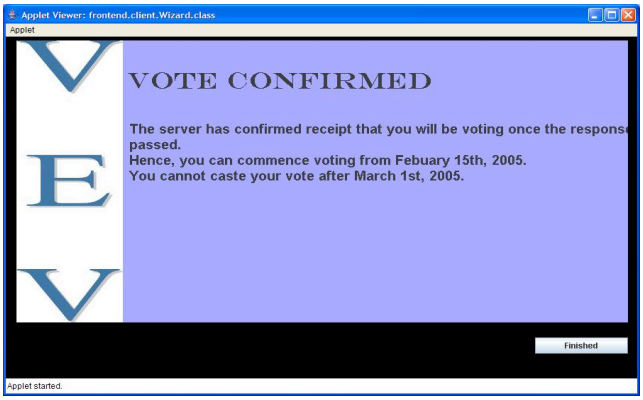

Figure 1 Phase 1 – Confirmation interface

In order to be able to vote, each voter has to confirm (Figure 1) his intention to vote and only those who respond will be allowed to cast the vote later. There will be a specified period of time when the voters can respond.

5.2 Phase 2: Voting Scenario

When the date for the user's response passes, the system enters the phase of the main voting process. The voting system running on the server is constantly waiting for the user to connect. The voter starts using the system by entering the username and the password that was previously obtained from the system's administrator (for example this could be at a pre-voting booth for authentication and password issuing at the voting site). Then the system authenticates the user. If the system recognizes the user it makes all functionality available to this person (such as vote, re-vote or view the existing votes, see: Figure 2). If the voter is not a recognized person (either the username or the password does not match the records) the user is treated as a guest to the system and the only things that are available for viewing are the existing votes.

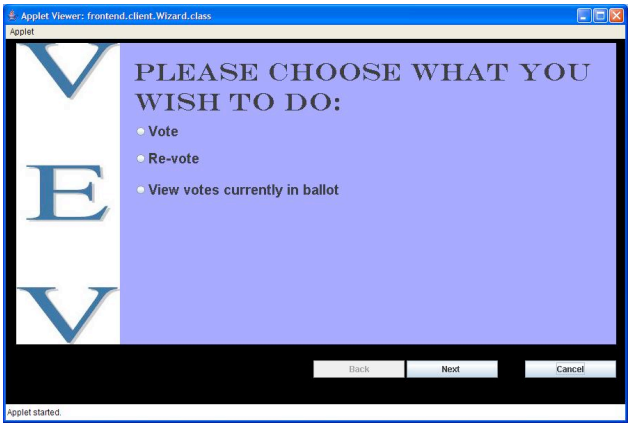

Figure 2 Phase 2 – User can chose the action

If the recognized user chooses to cast the vote for the first time the system creates the identification number for that user.

When the eligible user wants to cast a vote for the first time the client will randomly generate a 25-bit long prime number (*id*) which will be used to uniquely identify that particular user. In the next step of casting a vote the user chooses the candidate that he wants to vote for. The system displays the names of the election candidates and the user chooses one of them.

The numerical encoding for every voting strategy (e.g. name of candidate) should be a large prime number. The voting system is able to handle as many as 24 candidates to be voted for. The number 24 provides the opportunity for the unique encryption of each voting strategy. First, all numbers that end with 1,3,7,9 between 10 and 100 are selected (the underlying reason for that is the fact that the prime numbers end with 1, 3, 7, 9). This way a set of two digit numbers has been created (hereafter called indexes). For every index from the set, an election candidate is assigned. When the user chooses to cast a vote for a particular candidate, a random 25-bit prime number (v) is generated such that the first digit is equal to the first digit of the index and the last digit of v is the same as the last digit of an index. E.g.: Say we have an election candidate Anna S. Initially the system had assigned an index identification number to her that is 51. If the voter decides to cast the vote for Anna S. the client's program will randomly generate the prime number 5…..1 (first and last digit match the index).

Next, the user sends the pair of integers (id, f (id, v)) to the system where f is a randomly chosen encryption function (one of three algorithms that are explained in section 4); id is the identification tag generated for the user, and v is the candidate's name represented in the number; f (id, v) is the result of the encrypting method that takes id and v as its parameters. The system does not know the connection between the username and the id tag (or the voting strategy). The only association that is known to the system is the connection between the id tag and the vote function f (id, v). The user is asked to write down (see: Figure 3) his identification number (id) and the result of the voting strategy function. He is also informed by the system to keep these numbers secret.

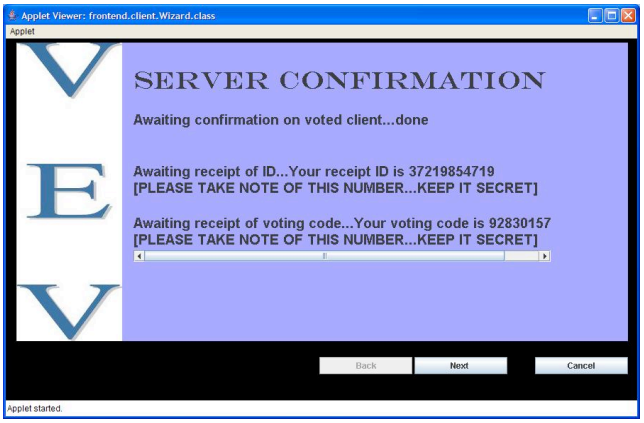

Figure 3 Phase 2 – voting procedure

When the server-side receives the numbers, it publishes the voting function result to the screen.

After each vote is cast, the system publishes the voting results. For each election candidate the system displays f(id,v) to the screen. (see: Figure 4) This way the user can check the correctness of his vote and the distribution of all votes. Publishing the voting strategy will serve an additional function. Every election candidate will be able to check if the votes were counted correctly. This might be of great importance for the candidates, because

elections have been known to be won by a difference of just a few votes.

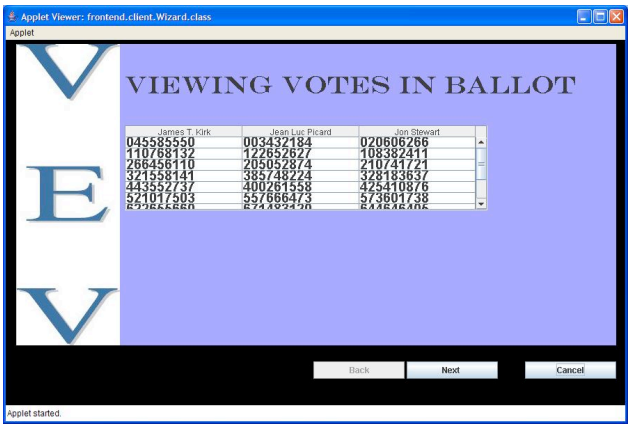

Figure 4 Phase 2,3 – Display of the election results

Furthermore the listing, on a website for example, of the total votes with voting function result for each candidate as they are submitted to the system can prevent vote buying.

6. The underlying algorithm

The primary advantage of public-key cryptography is increased security and convenience. The private key never needs to be transmitted or revealed to anyone. This section explains the major steps that have to be taken in order to implement VEV whose security is based on the usage of the public-private key paradigm.
It has been assumed that the server is running on the main computer and is constantly waiting for a client to connect. It is also anticipated that every user possesses the knowledge of his username and password. The italic type characters will be used to indicate the processes occurring on the server-side of the voting system.

6.1 Step 1: Authentication

1. Voter starts the execution of the client-side program.
2. Client asks the user to enter username. (see: Figure 5)
3. *Server-side application checks if the name exists on the list of users that are eligible to vote.*
4. If the name exists, the user is asked to enter the allocated password; otherwise the user is considered to be system's guest.
5. *In case that the username exists, the server checks if the password matches the username (if the password does not match, the user is considered to be a guest).*

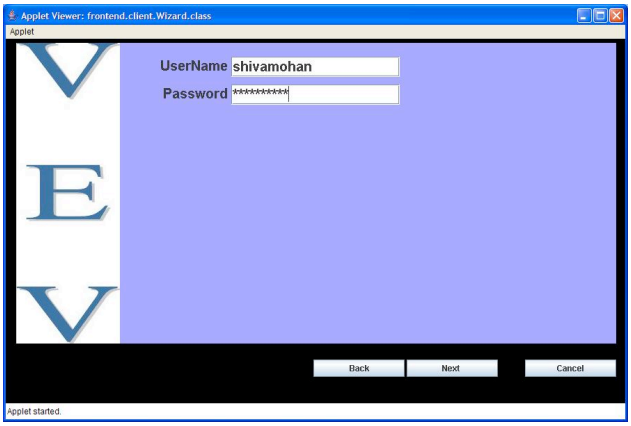

Figure 5 User authentication

6.2 Step 2: User operations

Phase 1 (the time allocated to acknowledge user-responses with the willingness to vote)
1. Client displays the number of users that are eligible to cast a vote.
2. User chooses the option to confirm voting or the option to exit.
3. *If user chooses to confirm voting the server records user's willingness to vote.*
4. Client displays the "Thank you" message and informs the user about voting dates.

Phase 2 (the time allocated for the actual voting)
User chooses to vote
Server checks (using username and password) if the user has voted already.
1. If the user did not cast his vote yet, the client randomly generates a 25-bit long prime number and assigns it as an identification number to that particular voter.
2. The message is displayed on the screen asking the user to take a note of this number and not to reveal it to anyone.
3. Client displays the names of the election candidates, and asks the user to choose one of them.
4. User types in the number of the candidate for whom he wants to cast the vote.
5. Client randomly generates a 25-bit long prime number called voting strategy that meets the index specification.
6. Client performs one of the encrypting functions (called also a voting function; there is a random choice made to use one of the three available encrypting methods) on the user's identification tag and the voting strategy number.
7. Client displays the result of voting function to the user. The user is asked to write the number down and to keep it confidential.

8. Client sends the pair (identification tag, voting function result) to the server.
9. *Server stores the vote information in its database.*
10. Server records that the user voted already. It is done to prevent the user from casting multiple votes.
11. When the user chooses to exit, client disconnects and the link between username and his vote disappears.

If the user previously cast the vote, he is asked to choose the re-vote option.

User chooses to re-vote

1. Client asks the user for his identification tag number.
2. Client asks the user for his voting function.
3. *Server checks if the vote exists.*
4. Client displays the names of the election candidates, and asks the user to choose one of them.
5. User types in the number of the candidate for whom he wants to cast the new vote.
6. Client randomly generates a 25-bit voting strategy that meets the index specification.
7. Client performs one of the encrypting functions on the user's identification tag and the voting strategy number.
8. Client displays the result of voting function to the user. The user is asked to write the number down and to keep it confidential.
9. Client sends the pair (identification tag, voting function result) to the server.
10. *Server stores new vote in its database and erases the old vote.*

User chooses to check the votes

Client displays all the voting functions to the screen. The votes are displayed in such a way, that for every candidate the voting function numbers are displayed in an ascending order. The user can check if his/her vote was counted correctly, and the election candidates can verify the voting results.

User chooses to exit

1. Client displays the "Goodbye" message
2. Client disconnects from the server

7. Conclusion

Voting software cannot be treated in the same way as a word processor or other applications, as we have even less reason to blindly trust the vendor – especially when the whole country's future is at stake. Most of the recent news about harnessing electronics for the election process has been bad. While much work in the USA is aimed at strengthening the ever-tight security around the software source code (it has been suggested that the voting application source code could not be reviewed even if challenged in court), in Australia there is a contrary approach with the voting code being made public. It is often argued [e.g. 9], that the only way to have a trustworthy system is to open the source code of cryptographic functions to the public. The algorithm can really be considered secure when is examined by many experts. Schneier [14] says: "… [t]he only way to have any confidence in an algorithm's security is to have experts examine it."[10] Australian officials believe that elections can benefit from involving the voters in the software development process. The voters can dictate the requirements including security and functionality of the voting system. No matter how many election flaws are found, and despite their severity, electronic voting systems are here to stay and serve us all. The only question remains: "How much, or little, trust can we afford?"

The authors thank NSERC for funding and Shiva Mohan for the screen shots of VEV.

References:

[1] See election publications from the Organization for Security and Cooperation in Europe. http://www.osce.org/odihr/?page=elections&div=reports

[2] Spectrum OnLine 16 August 2004 http://www.spectrum.ieee.org/WEBONLY/resource/nov02/nbraz.html

[3] Baldauf, S., *India's cutting-edge vote: 1st electro-nic election a big deal*. The Seattle Times, April 20, 04

[4] Schulte, B., *Jolted Over Electronic Voting Report's Security Warning Shakes Some States' Trust* Washington Post, August 11, 2003; Page A01

[5] Organization for Security and Cooperation in Europe (OSCE), Office for Democratic Institutions and Human Rights (ODIHR) ELECTION ASSESSMENT MISSION REPORT, 5 November 2002, Retrieved on 25 August, 04 from:,http://www.osce.org/documents/odihr/2003/01/1465_en.pdf

[6] Drinkard, J. *Election officials conflicted on electronic voting machines* USA TODAY Posted 5/5/2004 7:43 AM Updated 5/6/2004

[7] Mercuri, R., A *Better Ballot Box?* , IEEE Spectrum 39, October 2002, Retrieved on: October 9, 2004 from: http://www.spectrum.ieee.org/WEBONLY/publicfeature/oct02/evot.html

[8] Nurmi, H., Salomaa, A., Santean, L.,. *Secret ballot elections in computer networks*. Computers and Security, nr.10, 1991, pp.553-560.

[9] Bruce Schneier is an internationally renowned security technologist and author. Schneier is best known as a security critic and commentator. His books are: "Applied Cryptography", Secrets and Lies", "Beyond Fair".

[10] Schneier, B. Crypto-Gram Newsletter, September 15, 1999, Retrieved on August 26, 2004 from: http://www.schneier.com/crypto-gram-9909.html