CodeVoting: protecting against malicious vote manipulation at the voter's PC

Rui Joaquim and Carlos Ribeiro

ISEL - INESC-ID
rjoaquim@cc.isel.ipl.pt
carlos.ribeiro@tagus.ist.utl.pt

Abstract. Voting in uncontrolled environments, such as the Internet comes with a price, the price of having to trust in uncontrolled machines the collection of voter's vote. An uncontrolled machine, e.g. the voter's PC, may be infected with a virus or other malicious program that may try to change the voter's vote without her knowledge. Here we present CodeVoting, a technique to create a secure communication channel to a smart card that prevents vote manipulation by the voter's PC, while at the same time allows the use of any cryptographic voting protocol to cast the vote.

Key words: Internet voting, vote manipulation

1 Introduction

Internet voting can be classified as dangerous because the voter votes in an uncontrolled environment. Therefore the voter is vulnerable to coercion/vote selling and the voter's PC is vulnerable to the Internet threats, such as virus, worms, spyware and malware.

There is much work on coercion/vote selling problem [2–9] but, of our knowledge, few on the insecurity of voter's platform. The work on coercion free voting systems follows two main approaches. The first one is the use of a secure and secret communication channel between the voter and a trusted party of the voting system, allowing the voter to cheat on the coercer. Depending on the voting system, such secure and secret communication channel can be required prior to or on the election's day. The second main technique used to prevent coercion is allowing the voter to vote several times. Therefore, since usually the list of voters who voted is public, the only real coercion possible that gives the coercer 100% of guarantees is to coerce the voter to abstain.

We understand that coercion and vote selling is a potential big problem on Internet voting systems. However, we consider that the use of insecure voters' platforms can have a potential higher risk to the elections integrity. We base our opinion on three reasons.

– First, the coercion in large scale, involving possible thousands of voters, is quite unlikely to pass undetected.

Dagstuhl Seminar Proceedings 07311
Frontiers of Electronic Voting
http://drops.dagstuhl.de/opus/volltexte/2008/1299

- Second, with all the security flaws on operating systems and applications, it is easy to write a virus that would foul the voter on election's day and change her vote to a predefined vote.
- Third, we believe that writing a virus and disseminating it would be cheaper and more difficult to traceback to the authors than a coercion/vote selling of a thousand voters, therefore more appealing to an attacker.

CodeVoting addresses some problems of the insecure voter's platform, namely automatic vote manipulation. CodeVoting can be seen as a rearrangement of the ideas presented by Chaum [1] (SureVote). SureVote allows the voter to vote using secret vote and reply codes. These secret codes allow the voter to detect if anyone changed the vote code during the voting process. However, if there is at least one corrupted entity, SureVote does not guarantee that in the counting phase the vote code is translated to the right candidate.

CodeVoting reuses the idea of voting with codes. However, the idea of CodeVoting is to use the codes just as a user interface and not as the entire voting protocol. In CodeVoting we suggest the use of a smart card to translate the vote code to the candidate code. Additionally, we propose the use of the smart card as a trusted party that will participate in a cryptographic voting protocol in behalf of the voter. The use of a cryptographic voting protocol can give the voter guarantees that the voter's vote is counted correctly.

Other proposals to provide a secure voting environments are by Zúquete et al. [11], where the authors propose the use of a special hardware with secure input and output that the voter uses with her PC, and by Volkamer et al. [10], where the authors propose to use trusted computing technology to provide a "controlled" and secure voting environment.

In the next Sect. we will describe in more detail the vote modification attacks. Then, in Sect. 2 we describe how to use CodeVoting to securely cast a vote. Finally, we evaluate CodeVoting in Sect. 4 and conclude in Sect. 5.

2 CodeVoting

The key pieces of CodeVoting are the VoterCard and the CodeCard. The CodeCard is just a paper card with the vote codes printed on it. We will explain later how to get the CodeCard. The VoterCard is a smart card, which will be used to translate the vote codes into a real vote for a specific candidate. The VoterCard is given to the voter at the voters' registration process.

For the voter the voting process is quite simple. The voter just uses a CodeCard to translate the candidate code into a vote code.

For example, a voter with the ballot and CodeCard of Fig. 1 wishes to vote for candidate D she will enter WL764 as the vote code.

Every voter will have a different CodeCard, therefore different vote codes for the same candidate. Each CodeCard is associated to a VoterCard, which is responsible for the translation of the code to the candidate. Only the voter and the VoterCard should know the codes written on the CodeCard. Therefore,

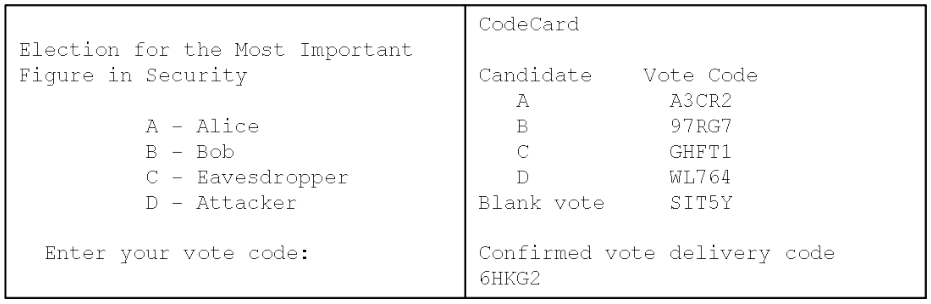

Fig. 1. Example of a ballot (on the left) and a CodeCard (on the right).

CodeVoting protects against a malicious voting application trying to change the voter's vote.

After translating the vote code to the candidate code any voting protocol can be used to cast the vote. One solution is using the VoterCard to process the candidate code accordingly to the cryptographic voting protocol and submit the vote. Another alternative is to create an unchangeable vote and pass it to the voting client application running on the voter's PC so it can proceed with the voting protocol.

When the VoterCard receives a confirmation of a successful vote delivery it releases the confirmed vote delivery code, assuring the voter that her vote was successfully delivered.

Based on this overview of CodeVoting the reader can easily understand that CodeVoting is a kind of a user interface plugin to a voting system that can protect the voter's choice from manipulation.

2.1 CodeCard

The CodeCard is just a paper card with codes printed on it. There should be one CodeCard per VoterCard so that every voter votes with different codes. The voter should be the only one with access to the codes of her CodeCard to prevent manipulation of her vote. Consequently we have a problem to solve: how to create the CodeCard, associate it with the VoterCard and give it to the voter without leaking the codes.

We propose to generate each voter's CodeCard within the VoterCard. This is a good option because the CodeCard becomes automatically associated with the correspondent VoterCard and no other entity besides the VoterCard has access to the codes on the CodeCard. However, we still have the problem of how to secretly print the CodeCard, i.e. how to give it to the voter without leaking the codes. We think the best idea is to have a certified CodeCard printing machine available at the local authorities' offices. Since the codes are generated inside the VoterCard the printing machine would be very simple. It would consist only of a smart card reader, a keypad (for inserting the PIN of the VoterCard and unlock it) and a small printer. We believe that such a simple hardware could be

easily certified and sealed to ensure the secrecy of the codes printed. With the CodeCard printing machines certified and in place a voter could go to any local authority office to generate a CodeCard for her VoterCard. For privacy reasons the printing machine should be inside a private booth, similar to the ones used for traditional paper based voting.

2.2 How to achieve trust on the VoterCard?

Reading the description of CodeVoting, the reader quickly understands that CodeVoting relies on the correct behaviour of the VoterCard. Therefore one can ask: CodeVoting is designed to protect the voter from the insecure voter's PC, but what guarantees are given that the VoterCard does in fact do what it is supposed to do? One way to verify that the VoterCard does in fact what it is supposed to do is to test it. Besides testing the VoterCards in the production phase we believe that would be good to have additional random testing by an independent certification authority.

Additionally we can make voters also part of the certification process. It could be possible for a voter to verify her VoterCard by running a fake election with instant results sometime before the real election.

Nevertheless, one can point out that the application running inside the VoterCard is somehow able to detect that is being subject to a test, and therefore it will act properly in the tests but it will still change the voters vote on the real election day. To prevent this scenario one must be sure of the software running inside the VoterCard. Fortunately, smart cards support signed applications. Therefore, and because it is possible to know which software is inside the VoterCard by verifying its signature, it is possible to use open source certified software. Of course, we can also have certified applications running on a PC. However, since it is possible for an attacker to take control of the voter's PC, a signed application does not guarantee correct behaviour. On the other hand, it is not possible to take control over the smart card. Therefore, an open source signed application can guarantee correct behaviour.

As an outcome of the last two paragraphs, we defend that the use of software running inside a smart card can offer much higher guarantees of correct behaviour than the software running in an insecure/uncontrolled platform such as the voter's PC.

3 Vote modification attacks

A vote modification attack can be performed in two ways: i) changing the vote to a predefined candidate or ii) changing the vote to a random candidate. While the first attack is more powerful the second may be easier to prepare in advance. By other words, to change a vote for a predefined candidate one must have the knowledge of which candidate one wants to change the vote to, while to change the vote to a random candidate there is no need to know the candidates in advance.

If one wants to boost the votes on candidate A it is preferable to perform an attack to directly changing the votes to votes on candidate A. On the other hand, if one wants the decrease the votes for candidate B, in an area known to be much favourable to candidate B, it is enough to perform a random vote modification attack.

Another kind of vote modification attack is to force abstention. In this attack the attacker just has to fake a successful vote delivery. In many voting systems this can be done just by presenting the message "Your vote was successfully delivered. Thank you for voting..

4 Evaluation

We defend that CodeVoting protects against vote manipulations at the voter's PC under the following assumptions: i) the CodeCard is generated in a secure and controlled environment by the VoterCard (the voter is the only person there), ii) the voter keeps her CodeCard secret, and iii) the correspondence between the candidate and its code (letter) cannot be changed. The last assumption can be achieved by publicly exposing the ballot or by any other technique that prevents a ballot change, such as using an image hard to forge/modify as a ballot.

Under these assumptions changing a vote to a predefined candidate is virtually impossible because the corresponding vote code is not known by the attacker and the probability of guessing the correct vote code, with 5 alphanumeric symbols, is 1 in 36^5, i.e. less than 1 in 60 million. A random candidate change attack has almost the same probability of success as the previous attack. If we have n candidates running for the election the probability of guessing a random valid vote code is $n - 1$ in 36^5.

The other possible attack is to fool the voter into believing that she cast a vote, while in reality no vote was cast. To prevent such attack we propose the use of another code to confirm vote delivery. The VoterCard only releases this code after getting a confirmation that the vote was successfully delivered, e.g. could be a message signed by the election server. Therefore, if we use a confirmed vote delivery code with the same length of vote codes, this attack has the same probability of success than the attack of changing the vote to a predefined candidate.

If the voting system does not allow a voter to cast several votes there is also a possible attack to the voters' trust on the voting system. The attack goes as follows: the attacker lets the voter successfully cast a vote and then change the valid confirmed vote delivery code to a fake one. The voter would think that something wrong had happened, however there was nothing wrong. Then the voter would try to vote again and the voting system replies that the voter had already voted and, of course, the voter will protest. If the voting system allows the voter to cast several votes this attack would not be a problem.

5 Conclusions and future work

Internet voting is subject to many dangers, namely vote selling, coercion and vote manipulation by malicious software. While the first two dangers have been subject of much research the last one was not. Here we presented CodeVoting, a solution that with the help of a trusted smart card prevents vote manipulation in an uncontrolled environment. However, there are still some issues to deal with, such as large candidates list and CodeCard reuse.

Large candidate list are a big issue when considering the real application of CodeVoting. As proposed, the CodeCard must have an entry for each possible candidate. If we consider elections with a large number of candidates, lets say above twenty, the size of the CodeCard starts to become to large and usability problems may arise.

Another issue is the CodeCard reuse. If a voter uses the same CodeCard in more than one election it is possible for an attacker to replace the voter's vote by a random one. To be able to perform this attack the attacker must collect the codes used on the first election and be in control of the PCs used by the voter in both elections. If the voter uses different PCs to vote it will be much harder to perform the attack. This attack can be easily prevented if the voter generates a new CodeCard for each election. However, if the elections are running simultaneously this may be harder.

Concluding, we believe that CodeVoting can be successfully used to prevent vote manipulation in uncontrolled environments, however additional work must be done to solve the large candidate list and CodeCard reuse issues. Additionally, it would be interestingly to study alterative solutions to print and secretly delivering CodeCards to voters.

References

1. Chaum, David: SureVote. `http://www.surevote.com/home.html`, September 2007.// International patent WO 01/55940 A1, 02 August 2001.
2. Clarkson, M., Myers, A.: Coercion-Resistant Remote Voting Using Decryption Mixes. In Workshop on Frontiers in Electronic Elections, Milan, Italy, September 2005
3. Estonian Internet Voting System `http://www.vvk.ee`, July 2007
4. Juels, A., Catalano, D., Jakobsson, M.: Coercion-Resistant Electronic Elections. In Workshop on Privacy in the Electronic Society, pages 6170, Alexandria, Virginia, November 2005.
5. Kutylowski, M., Zagórski, F.: Coercion-Free Internet Voting with Receipts. Workshop on e-Voting and e-Govrnment in the UK. Edinburgh, February 2006.
6. Lee, B., Kim, K.: Receipt-Free Electronic Voting Scheme with a Tamper-Resistant Randomizer. In ICISC02: Proceedings of the 5th International Conference on Information Security and Cryptology, LNCS 2587, pages 389–406. Seoul, Korea, November 2002.
7. Hirt, M., Sako, K.: Efficient Receipt-Free Voting Based on Homomorphic Encryption. In International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT), pages 539556, Bruges, Belgium, May 2000.

8. Okamoto, T.: Receipt-Free Electronic Voting Schemes for Large Scale Elections. In Security Protocols Workshop, pages 2535, Paris, France, April 1997.
9. Sako, K., Kilian, J.: Receipt-Free Mix-Type Voting SchemeA Practical Solution to the Implementation of a Voting Booth. In International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT), pages 393 403, Saint Malo, France, May 1995.
10. Volkamer, M., Alkassar, A., Sadeghi, A., Schulz, S.: Enabling the Application of the Open Systems like PCs for Online Voting. In FEE06: Proceedings of the Frontiers in Electronic Elections Workshop. Hamburg, Germany, September 2006.
11. Zúquete, A., Costa, C., Romao, M.: An Intrusion-tolerant e-Voting Client System In WRAITS07: 1st Workshop on Recent Advances on Intrusion-Tolerant Systems Lisbon, Portugal, March 2007.