Ann. Telecommun. (2016) 71:299–307
DOI 10.1007/s12243-016-0509-8

An experiment on the security of the Norwegian electronic voting protocol

Kristian Gjøsteen[1] · **Anders Smedstuen Lund**[1]

Received: 30 June 2015 / Accepted: 17 March 2016 / Published online: 31 March 2016

Abstract Even when using a provably secure voting protocol, an election authority cannot argue convincingly that no attack that changed the election outcome has occurred, unless the voters are able to use the voting protocol correctly. We describe one statistical method that, if the assumptions underlying the protocol's security proof hold, could provide convincing evidence that no attack occurred for the Norwegian Internet voting protocol (or other similar voting protocols). To determine the statistical power of this method, we need to estimate the rate at which voters detect possible attacks against the voting protocol. We designed and carried out an experiment to estimate this rate. We describe the experiment and results in full. Based on the results, we estimate upper and lower bounds for the detection rate. We also discuss some limitations of the practical experiment.

Keywords Usability experiment · Attack detection · Electronic voting

1 Introduction

Remote Internet voting can make voting significantly more accessible for many voter groups, especially in sparsely populated countries such as Norway. Younger generations increasingly expect government services to be available on the Internet, and this seems to include voting. Several countries have either conducted trials or are using remote Internet voting as an option in government elections. Estonia is cited as the primary example, but countries such as Switzerland also use internet voting.

Norway conducted two trials of remote internet voting, the first during the 2011 municipal elections and the second during the 2013 parliamentary elections. Both trials covered roughly 160 000 voters. The Norwegian central government has since decided not to continue with trials of electronic voting of any kind.

It is important when using any kind of voting system to be able to argue convincingly that no attack or other irregularity that changed the outcome of the election has occurred. A crucial part of that argument will be an understanding of how well voters are able to use the voting system to detect attacks, in addition to arguments for why voters should be able to detect attacks.

The system used in the Norwegian trials had a limited form of verifiability, where voters were sent a return code (via SMS message) that could be used to verify that the ballot had been received correctly.

Even though the system is fairly simple to use, one can not expect that every voter will be able or even want to verify that the ballot was correctly received by the system.

What we instead might hope to achieve is to use the limited verifiability to be able to argue convincingly that all but a negligible fraction of the votes are counted as cast, when this is in fact the case. Also, in the event of a significant attack, we want to be able to estimate the size of the attack with some certainty.

The cryptographic analysis [3] of the Norwegian voting protocol makes certain assumptions. It has always been well-known that if these assumptions are violated, many

✉ Kristian Gjøsteen
kristian.gjosteen@math.ntnu.no

Anders Smedstuen Lund
anders.lund@math.ntnu.no

[1] Department of Mathematical Sciences, NTNU, 7491 Trondheim, Norway

attacks are possible, and these attacks cannot in general be detected by voters. The Norwegian government was also well aware from before the system was designed that these assumptions would not remain valid indefinitely.

Some of the (well-known) ways these assumptions fail have subsequently been remarked upon by Koenig et al. [7], but their proposed countermeasures are either ineffective or inapplicable for Norway. Note that in this work, we are only interested in attacks exploiting (or causing) mistakes made by voters. Attacks violating the underlying assumptions are therefore not relevant for this paper.

Our contribution We have designed a simple statistical model for answering this question based on the number of reported cases of fraud. We briefly discuss the effectiveness of this model. The central parameters used in the model is the detection rate for fraud, and the false alarm rate. We should also stress that these parameters are not artefacts of the statistical model, but are of independent interest for anyone trying to understand the security of electronic voting systems.

We have designed a number of plausible attacks against the system used in the Norwegian elections, all predicated on the voter making some kind of mistake, either unprovoked or provoked. Many of these attacks are easily adapted to other voting schemes, such as those used in Estonia and Switzerland.

Studying security mechanisms in the laboratory is difficult, because a laboratory setting tends to be very different from the real world in which security mechanisms are used. With this in mind, we have designed a fairly simple laboratory experiment for measuring the detection rate and the false alarm rate for attacks. The software we wrote for the experiment are specific to the Norwegian election and Norway. But the general design ideas can be used to design similar experiments for other voting systems, such as those in Estonia and Switzerland. In fact, the techniques are not even restricted to voting systems.

At the time of the experiment, no further trials of internet voting were planned in Norway and the exact detection rate in Norway was of limited interest. It was therefore clear from the outset that our experiment would be fairly small, which meant that its power would be limited. With this limitation in mind, we chose a single attack to test.

While the exact outcome of the experiment is of limited interest, the most interesting outcome is the fact that the experiment worked well. We have shown that this approach to measuring detection rates works. During the experiment, we observed exactly the features we expect to encounter in the real world. This means that countries using remote internet voting can run laboratory experiments based on our techniques to gain a better understanding of the performance of their verification mechanisms.

We stress that a real experiment to properly understand an attacker's capabilities will require much larger experiments, and will also have to explore many more of the options available to an attacker. Our catalogue of attacks illustrates the magnitude of the effort, and may also be of some assistance when assessing other systems.

Related work The protocol used in the trials in Norway is described in detail in [2, 3]. The protocol has been improved in [4], but these improvements has not changed the way the voter experiences the voting process.

One of the security measures in the protocol is the ability to verify that your vote was cast as you intended it through the use of return codes, sent in the form of text messages to the your cell phone. Karayumak et al. [5] and Sherman et al. [11] has conducted mock elections using, respectively, Helios and Scantegrity, and found that voters are confused about the motivation of having verifiability.

Weber and Hengartner [13] found that many voters are not able to verify that their ballot was cast as intended. Karayumak et al. [6] has analyzed Helios using the cognitive walkthrough method [12], and found that it was likely that voters using Helios would not be able to verify their ballot. Experiments have been designed in order to increase the understanding of the use of verification among voters through mental models [1, 8, 10, 14].

Olsen and Nordhaug [9] has conducted an experiment testing the detection rate of certain attacks on the Norwegian voting protocol, but very few details about the experiment is described.

Overview of the paper We discuss how we can use attack reports to argue whether or not an attack that changed the election outcome has occurred in Section 2, while also using previously obtained data to estimate the false alarm rate.

In Section 3, we describe briefly the Norwegian internet voting system and a number of possible attacks against the system relying solely on the voter making a mistake.

In Section 4, we describe the experiment, and discuss some possible sources of errors. Finally, Section 5 presents the results, and Section 6 discusses the results and what conclusions we draw from these results.

2 Attack detection

The election authority's goal after any election is *to argue convincingly that no attack took place that changed the outcome of the election*, in the event that no such attack took place. We will investigate one possible approach of arguing convincingly, based on attack detection rates.

Our main observable statistic will be the number of reported attacks. Note that this number will almost certainly be different from the number of attacks. First of all, many attacks will go unreported. And since the exact nature of the attack probably will be unknown, we must also expect unfounded attack reports.

Let N be the number of voters submitting ballots electronically. Define the events $E_1, \ldots, E_N$ to be that the ith voter was attacked, and let $F_1, \ldots, F_N$ be the events that the ith voter reported an attack. The events $E_i \wedge F_i$ and $\neg E_i \wedge \neg F_i$ indicate correct observations, the event $E_i \wedge \neg F_i$ is an unreported attack (it may or may not be detected), while the event $\neg E_i \wedge F_i$ is a false alarm.

We shall assume that the reporting events only depend on the corresponding attack events, and that they otherwise are independent and identically distributed. Under this assumption, we can define:

$$\delta_0 = \Pr[F_i \mid E_i] \text{ and } \delta_1 = \Pr[F_i \mid \neg E_i].$$

That is, δ_0 is the probability of correctly reporting an attack, the attack *detection rate*, and δ_1 is the probability of reporting an attack when there is no attack, the *false alarm rate*.

Remark 1 It is quite clear that reporting events are not independent. However, we believe independence is a fairly good approximation to reality.

Remark 2 It is probably incorrect to assume that δ_0 and δ_1 do not change during the voting period. For instance, if ongoing attacks get a lot of media attention, we expect both δ_0 and δ_1 to increase. It is therefore assumed that the data used for this analysis is gathered under fairly uniform conditions.

Let X be the number of attacks, and let Y be the number of attack reports. From our point of view, each ballot submission is essentially a Bernoulli trial. The number of correct attack reports will therefore follow a binomial distribution with parameters δ_0 and X, while the number of incorrect attack reports will follow a binomial distribution with parameters δ_1 and $N - X$.

If we let Y_0 be the number of correct attack reports and Y_1 be the number of incorrect attack reports, we get that $Y = Y_0 + Y_1$ and

$$\Pr[Y = y \mid X = x] = \sum_{i=0}^{y} \Pr[Y_0 = i \mid X = x] \Pr[Y_1 = y - i \mid X = x].$$

Since we want to argue convincingly that no attack that changed the election outcome occurred, our null hypothesis H_0 should be that there was an attack that changed the election outcome. The alternative hypothesis H_1 is that there was no attack that changed the election outcome. We want to know if our evidence strongly rejects the null hypothesis.

Given the election result, we can deduce the minimum number of attacks x required to change the election result. Our null hypothesis H_0 should be $X \geq x$. Since we have no hypothesis concerning the distribution of X, we instead use the null hypothesis $X = x$ and construct a one-tailed test. The alternative hypothesis H_1 is then that $X < x$.

To compute the size of the critical region for our one-tailed test, we choose a significance level α and find the largest y_α such that

$$\sum_{y=0}^{y_\alpha} \Pr[Y = y \mid X = x] \leq \alpha.$$

To summarize, y_α is the largest number of observed attack reports that allow us to conclude that there was no attack.

Remark 3 Our test will tend to have a large false acceptance rate, especially if the false alarm rate is large. It seems difficult to avoid this while still achieving the goal of arguing convincingly that there was no attack that changed the election outcome. Other tests with other goals can be designed, of course.

2.1 The false alarm rate

During the two trials, with tens of thousands of electronic votes, there were no false errors reported as far as we have determined. A number of real errors were reported, however, caused by misprinted return codes. We may assume that there was no attack during the two trials.

Based on this observation, we place an upper bound on the false alarm rate. Using a standard test (based on zero false alarms and 70,000 internet voters), we can reject the hypothesis that the false alarm rate is 0.0001 or greater (Table 1). While we cannot reject a non-zero false alarm rate, our best estimate for the false alarm rate must be $\delta_1 = 0$.

Remark 4 Note that δ_1 strongly influences the size of the critical region for the null hypothesis in Section 2, and also the probability of false acceptance. What happens is that as the false alarm rate increases, the size of the critical region increases, but the probability of false acceptance increases.

Table 1 The probability p of zero false alarms over N = 70,000 votes for various false alarm rates δ_1

δ_1	0.01	0.005	0.001	0.000 1	0.000 05
p	0.0	0.0	0.0	0.0	0.030 19

Remark 5 We can also consider the detection rate and the alarm rate *in hindsight*, where the voter some time after voting is informed of the fact that some voters were attacked and asked if he believes he was attacked.

In this situation, it is reasonable that the false alarm rate *in hindsight* would be much larger than the false alarm rate. If possible, we want to confirm or reject this hypothesis.

2.2 Discussion

We have computed the size of the critical region for various attack sizes and detection rates. We have used a false alarm rate of 0.

As we can see from Table 2, the critical region will be small or *non-existent* for low detection rates and small attacks. This means that our false acceptance rate will be very large.

However, in the 2013 parliamentary elections, the lowest margin was several hundred votes. So even with very small detection rates it seems safe to conclude that there was no attack.

For municipal elections in 2011, there are a number of municipalities where our test has a non-existent critical region, which means that the test cannot always be applied. We must expect such results for parliamentary elections too, every now and then, even if none were observed in 2013.

This does not imply that our test is useless, of course, but the underlying statistic is insufficient. For very close elections, it is likely that one could never argue convincingly, purely based on the attack report statistic, that there was no fraud. This means that other approaches involving more data must also be studied, in preparation for close elections.

In summary, based on data from the 2011 and 2013 elections, we believe that the test could be quite effective tool for most elections, even when the detection rate is very small. But it cannot be the only tool. We also expect that similar tests could be useful tools also outside of Norway.

Table 2 The critical region for $\delta_1 = 0$ and various attack sizes X and detection rates δ_0. A $-$ means that the critical region is empty

X	30	100	300	1000
$\delta_0 = 0.025$	–	–	2	16
$\delta_0 = 0.050$	–	1	8	38
$\delta_0 = 0.10$	0	4	21	84
$\delta_0 = 0.25$	3	17	62	227

3 Attacks

In order to estimate the attack detection rate, we need to consider possible attacks.

In the Norwegian voting system, immediately after successful ballot submission, the voter is sent so-called return codes by SMS, so that he can verify, using a table printed on his voting card, that the correct ballot was received. (In the event that the voter notices an error, he may vote again electronically or vote on paper.)

The procedure the voter must follow when voting can be summarized as follows:

1. Log in to the voting application.
2. Enter the correct ballot into the voting application.
3. Wait for the voting application's confirmation screen.
4. Wait for the SMS message with the return code.
5. Verify that the return code is correct.

Only if all of the steps are performed correctly should the voter accept the ballot as cast.

What follows is a (non-exhaustive) list of a number of possible attacks against the voting system. We have assumed that the attacker controls the voter's terminal, but all the attacks are possible even if the attacker merely is able to fool the voter into visiting a fake voting web site.

While this catalogue is not needed for our experiment, we believe that it usefully illustrates the number of options available to an attacker, and the magnitude of the task facing a defender that wants to understand the attacker's options.

3.1 Ballot deletion attacks

These attacks try to delete undesired ballots. Note that this is stronger than random deletion attacks, because the attacker gets to see the ballot before deciding to delete it.

A1. Straight-forward vote deletion For ballots the attacker does not like, the attacker does not submit the ballot.

The voter needs to notice that he did not receive something, which is much more difficult than noticing that he received something incorrect.

A2. Ask If the User Wants Return Codes For ballots the attacker does not like, the attacker does as follows: He does not submit the ballot. On the fourth screen, a button is included labeled with ≪Press here to get return codes per SMS≫. If the voter pushes the button, the vote is submitted and the voter gets his return codes, otherwise nothing is done.

Compared with the previous attack, the attack yield is reduced, but the detection rate should be reduced.

3.2 Ballot modification attacks

These attacks try to modify undesired ballots.

A3. Incorrect Return Codes The attacker changes the ballot. The voter gets an incorrect return code.

The voter may not verify the return code, or may accept the vote as cast even if the return code does not verify it.

A4. Incorrect Return Codes II The attacker changes the vote. On the fourth screen, the attacker displays a message saying that (a) ≪the return code system is experiencing instability≫, or (b) ≪due to telecom errors, some users are receiving return codes intended for others≫. Does this tend to alert users, or does it stop the user from complaining?

This may depress detection rates.

A5. Incorrect Return Codes on First Try The attacker changes the vote. On the "thank you for voting" screen, the attacker discretely notifies the user that he should vote again if he gets the wrong return codes. If the voter tries to vote again, the attacker does not change the vote.

If the user detects the attack, but does not report the incident, the election supervisors may not correctly estimate the size of the attack.

This also affects incident response. For instance, if the first step of incident response is to ask the user to vote again, it is vital that the incident response team records this event as a security event, even though the problem may seem to be solved.

A6. Ask the User to Enter Return Codes For ballots the attacker does not like, the attacker does as follows: He does not submit the ballot. After ballot submission, a new confirmation screen asks the voter to enter his return codes to verify his vote.

The idea is that the user knows he is supposed to do something with the return codes, but he does not know what. Entering them to verify the choices seems to be sensible.

A7. Ask the User to Enter Return Codes II Send the entered return codes to the voter via SMS.

This is technically feasible, but somewhat challenging.

The advantage is that the voter now gets the correct return code via SMS, as he is supposed to.

4 The Experiment

Based on the attacks described in Section 3, we designed an experiment to estimate the detection rate. Obviously, it would be unethical to run this experiment during a real election, so we need to run a laboratory experiment. We could not easily use the real voting system, so we had to build a mock-up of the voting system.

Our idea was to expose the participants to some of the attacks described in Section 3, and then record the number of attack reports for each attack. This would give us an estimate for the detection rate δ_0.

Experiment size Given the resources available to conduct the experiment, we chose to recruit students on our university campus. Our strategy was to recruit participants so that the average level of computer skills among the participants was at least as high as the population in general.

With this in mind, we decided to recruit only from engineering or science students. The assumption is that given the technical aspect of their studies, the average student attending these lectures should have at least as high a level of computer skills as the population in general.

As a way to motivate the students to participate, we announced that two of the participants who completed the experiment would be randomly chosen to receive a gift card with the value of 1000 NOK. We recruited 46 participants in total.

Choices We felt that it was entirely plausible that we would measure a zero detection rate. With this in mind, we selected our subjects and the attacks we studied such that if we measured a zero detection rate, our experiment would constitute strong evidence that the verification mechanism in the Norwegian system would not work.

Given the number of participants we were able to recruit, we chose to only test one attack. The attack chosen was Attack A1 described in Section 3. It was modeled by not sending a return code to the voter.

Experiment setting It is more or less axiomatic in security usability that users are not trying to use a security mechanism, they are trying to use the system protected by the security mechanism. In general, we must therefore expect voters to focus on the job of submitting a vote, not on the security mechanisms that protect the voting process.

Participants in a laboratory experiment will probably be more alert than real voters in a real voting situation. For instance, if the participants are told that they will be exposed to social engineering attacks, we should expect a very high false detection rate. If the participants are told that they are merely testing usability, not security usability, we should expect a negligible false alarm rate, and probably also a very low detection rate.

We chose to expose the participants to three separate tasks, and emphasized to the participants that all tasks where equally important. The idea was that this would make the subject focus on carrying out the tasks at hand, and at the same time worry less about the security mechanisms involved in each task.

Since we could not draw attention to the electronic voting task, we could not ask if any of the participants actually participated in the trials in 2011 or 2013. However, statistically speaking, it is unlikely that any of them did, and none of them mentioned this to us.

Practical issues Any communication between the participants in the experiment will compromise subject independence, which is essentially the same as reducing the size of the experiment. It is therefore important to prevent the participants from communicating, both before and after measurements. This requires careful management of the experiment.

Also, the act of measuring should preferably not influence the measurement. It is therefore important that any questions asked to the participant are as unspecific as possible, probably along the lines of: *How confident are you that the correct vote was recorded?*

Ethical considerations Our experiment involved human participation, and therefore we had to take care when designing the experiment. We decided to design the experiment so that no information about the participants was stored electronically. The information recorded during the experiment was completely anonymous, which was also stated clearly to every participant.

Experiment description Our experiment was divided into two phases. The first phase was the actual experiment. In the second phase, conducted some time after the first phase, we simply asked each participant a few questions about the experiment.

In the first phase, we made appointments with each participant at different times, so that they would avoid running into each other, thereby minimizing communication between participants. The experiment was conducted in a room with only the participant and the experiment supervisor (one of the paper authors) present. The author supervising the experiment was seated behind the participant. While the participants carried out their tasks, the supervisor only answered questions the participants had about how to complete the tasks.

At the start of the experiment, the participants received written instructions describing the tasks, how to perform them, and the mock information to be used for each task. They were given the following tasks:

1. Log on to a mock e-voting webpage and submit a ballot for a specified party with specified changes to the party ballot.
2. Log on to a mock bank webpage, pay one invoice and transfer money from one account to another.
3. Log on to a mock government webpage and submit an application for changing your official postal address.

To log on to the different sites, we made mock versions of three different federated identity systems. The systems being used was respectively *MinID*, *BankID på mobil*, and a smartcard solution. Both *MinID* and *BankID på mobil* involves the use of a mobile phone; this mobile was modeled on the top right part of the screen. The action of inserting the smartcard was modeled at the same place on the screen. It turned out that the only questions not concerning attacks against the e-voting systems was where to find either the mobile or how to insert the smartcard.

After the participants had completed the three tasks, they continued to a web page asking the five following questions:

1. On a scale from 1 to 5, where 5 is best, how good are your computer skills?
2. On a scale from 1 to 5, where 5 is best, how good are your practical computer security skills?
3. For each of the three tasks: On a scale from 1 to 5, where 5 is best, how confident are you that nothing went wrong during this task?

Answering these questions completed the first phase of the experiment. After all participants had finished phase one, they where asked to come back for the second phase.

In the second phase of the experiment, the participants were informed that some of the participants where subject to an attack on the e-voting system and some where not. Then they were asked the following two questions:

1. Do you think you were subject to an attack when using the e-voting system?
2. If yes, why do you think you were subject to an attack?

On the first question the participants where given three alternatives, ≪Yes≫, ≪No≫, and ≪I don't know≫. We made sure that participants kept their answer hidden from each other and did not in any way communicate with other participants.

5 Results

As mentioned in Section 4, we were able to recruit 46 participants. We chose to divide them randomly into one group of 30 participants, which would be subject to Attack A1 as described in Section 3, and one group of 16 participants, which would serve as a control group.

Detection rate During the first phase of the experiment, six participants from the group exposed to an attack told us they

Table 3 Number of participants in the attack group answering the different options for each of the five questions asked

Answer	1	2	3	4	5
Computer skills	0	1	11	13	5
Computer security skills	0	5	17	7	1
Bank webpage	2	1	4	9	14
E-voting webpage	5	5	5	5	10
Government webpage	1	1	10	10	8

did not get a return code. Of these six participants, only three thought it was a problem and said they would have informed someone about it if this was a real election. The other three did not consider it a problem. Two of them stated that they would assume everything was in order if this was a real election. The last one of the three stated that he/she would log out, and then assume everything was in order. We also observed that several other participants seemingly noticed the missing return code (some even voted once more), but none of these mentioned anything and continued to the next task.

Our main goal is to estimate the fraction of the population that would alert the election authority. Therefore, we only count the participants that both notified us of the missing return codes and told us that they would contact someone about this in a real life situation.

Comparing groups Table 3 summarizes the answers of the participants in the group exposed to an attack. The participants were fairly confident in their computer skills. Given these results, we keep our assumption that the average level of computer skills of this group is at least as high as that of the general population.

Table 4 summarizes the answers of the control group participants. The control group also seems to be fairly confident in their computer skills, and they also seem to have less confidence in their practical security skills. As with the group subject to an attack, we keep our assumption that the average level of computer skills of the control group is at least as high as that of the general voter population.

Table 4 Number of participants in the control group answering the different options for each of the five questions asked

Answer	1	2	3	4	5
Computer skills	0	1	3	10	2
Computer security skills	0	2	7	7	0
Bank webpage	0	5	1	5	5
E-voting webpage	0	5	2	5	4
Government webpage	0	3	2	7	4

Table 5 Number of participants in the control group answering the different options for each of the five questions asked

	≪Yes≫	≪No≫	≪I don’t know≫
Group subject to attack	3	17	3
Control group	1	12	1

More importantly, we believe both groups are reasonably equal, both in terms of general computer skills and practical computer security skills.

Discussion of phase 1 results It seems like the participants exposed to an attack were less confident that nothing had gone wrong when using the e-voting webpage, than they were when using the other two systems. This could be because they are more accustomed to using the two other systems, but we suspect it was because more participants than those who told us, noticed the lack of return codes.

It is also important to notice that the answers to these questions might be influenced by how secure they belive each government federated log in system is. One participant said after finishing phase two, when asked how secure each of the three systems felt, that he/she felt a smartcard solution was less safe than the other two solutions.

The participants in the control group seem to be as confident that everything was correct when they voted, as they were when they performed the tasks on the bank webpage. This seems to differ from the attack group, which supports our suspicion that not everyone who noticed the lack of return codes told us.

Phase 2 Of the 46 participants in phase one, 37 returned to complete phase two. The answers on question one of phase two is listed in Table 5.

None of the participants from the group subject to an attack answering ≪I don’t know≫ made any comments suggesting they had noticed the lack of return codes, while everyone answering ≪Yes≫ commented that they had noticed a lack of return codes. One of the participants answering ≪No≫ chose to answer the second question. He/she commented that he/she seemed to recall not receiving return codes, but did not think this was a problem. This implies that even when being told an attack has occurred and noticing a deviation from how the system is supposed to work, we cannot expect all voters to report the deviation to the election authority.

From the control group, one participant believed he/she had been subject to an attack. He/she commented that he/she thought it might be the case that some of the messages or numbers he/she had seen on the screen had been tampered with. He/she could not point to which message or number this might be.

Table 6 Lower bound of δ_0 for different values of X at significance level 2.5 %

X	2	3	4	5	6	7
δ_0	0.009	0.022	0.038	0.057	0.078	0.1

6 Discussion

We want to find upper and lower bounds on the detection rate δ_0 based on the results from our experiment. Using hypothesis testing, we can achieve this. To find the bounds, we will use the random variable X (the number of attack reports from the attacked group) as test statistic. For the lower bound, we will use a one-tailed test with the following hypotheses:

$H_0 : \delta_0 = \mu$ and

$H_1 : \delta_0 > \mu$.

To compute the critical value x for a given α, we find the smallest x such that

$$\Pr[X \leq x \mid \delta_0 = \mu] \geq 1 - \alpha,$$

and reject the null hypothesis H_0 if $X > x$. We want to find a lower bound on δ_0, so for different values of X, we find the smallest value of δ_0 that allows us to keep the null hypothesis H_0. Similarly to find the upper bound, we use the hypotheses:

$H_0 : \delta_0 = \mu$ and

$H_1 : \delta_0 < \mu$,

compute the critical value x, given α, by finding the smallest value of x such that

$$\Pr[X \leq x \mid \delta_0 = \mu] > \alpha,$$

and reject the null hypothesis if $X < x$. Then we find the largest value of δ_0 that allows us to keep the null hypothesis for various sizes of X. Estimates on the lower bound and the upper bound of δ_0 given $\alpha = 0.025$ for different values of X are listed in, respectively, Tables 6 and 7.

In our experiment, we registered three attack reports, hence it seems reasonable to adopt a lower bound on δ_0 of 0.022 and an upper bound of 0.265. If we had chosen to let all 46 participants be subject to an attack, we might have achieved tighter bounds, but only slightly. To get the bounds significantly improved, a larger experiment must be carried out.

Table 7 Upper bound of δ_0 for different values of X at significance level 2.5 %

X	2	3	4	5	6	7
δ_0	0.219	0.265	0.306	0.346	0.384	0.421

We also wanted to test the hypothesis that when voters are told an attack has occurred, the false alarm rate will increase. For the purposes of this discussion, we have adopted the hypothesis $\delta_1 = 0.001$. Our null hypothesis should be that the false alarm rate did not change, and the alternative hypothesis should then be that the false alarm rate increased.

Our test statistic Y is the number of false attack reports registered during the experiment. The critical value is the smallest value of Y we can observe while still having

$$\Pr[Y > y \mid \delta_1 = 0.001] = 1 - \sum_{i=0}^{y} b(i, 16, 0.001) \leq \alpha$$

Using $\alpha = 0.05$, we obtain a critical value of 0, and since we observed one false attack report, we are forced to reject our null hypothesis and accept the alternative hypothesis of $\delta_1 > 0.001$. If we decrease α to 0.025, we can still reject the null hypothesis. For $\alpha = 0.05$, we are forced to keep the null hypothesis if we alter the false alarm rate of the null hypothesis to $\delta_1 = 0.005$.

It is hard to say, based on the participant's comments, if he/she would have reported an attack in a real situation, therefore it would be preferable to test the hypothesis in a larger control group. Clearly testing more attacks, and in larger groups, would have been preferable, but this was not possible given the resources and time available.

7 Conclusion

We have designed and described an experiment with the goal of estimating the detection rate when deleting ballots (not delivering return codes). Based on this, we have obtained an upper bound on δ_0 of 0.265 and a lower bound on δ_0 of 0.022. Furthermore, we have tested whether the false alarm rate increases when voters are told that an attack has occurred. Based on our results, it seems the false alarm rate does increase, but we would like to test this in a larger group to confirm our results.

We note that the students we recruited for our experiment will be much more familiar with computers in general than the general Norwegian population. Our detection rate estimate is probably an upper bound on he detection rate for the Norwegian population as a whole.

References

1. Campbell BA, Byrne MD (2009) Straight-party voting: what do voters think? IEEE Trans Inf Forensics Secur 4(4):718–728. doi:10.1109/TIFS.2009.2031947
2. Gjøsteen K Analysis of an internet voting protocol. Cryptology ePrint Archive, Report 2010/380 (2010). http://eprint.iacr.org/

3. Gjøsteen K The Norwegian internet voting protocol. Cryptology ePrint Archive, Report 2013/473 (2013). http://eprint.iacr.org/
4. Gjøsteen K, Lund AS The Norwegian internet voting protocol: a new instantiation. Cryptology ePrint Archive, Report 2015/503 (2015). http://eprint.iacr.org/
5. Karayumak F, Kauer M, Olembo MM, Volk T, Volkamer M (2011) User study of the improved Helios voting system interfaces. In: 1st workshop on socio-technical aspects in security and trust, STAST 2011, Milan, pp 37–44. doi:10.1109/STAST.2011.6059254
6. Karayumak F, Olembo MM, Kauer M, Volkamer M (2011) Usability analysis of Helios - an open source verifiable remote electronic voting system. In: 2011 electronic voting technology workshop / workshop on trustworthy elections, EVT/WOTE '11, San Francisco. https://www.usenix.org/conference/evtwote-11/usability-analysis-helios-%E2%80%94-open-source-verifiable-remote-electronic-voting
7. Koenig RE, Locher P, Haenni R (2013) Attacking the verification code mechanism in the Norwegian internet voting system. In: Heather J, Schneider SA, Teague V (eds) Proceedings of 4th international conference of e-voting and identify. Vote-ID 2013. Lecture Notes in Computer Science, vol 7985. Springer, Guildford, pp 76–92. doi:10.1007/978-3-642-39185-9_5
8. Olembo MM, Bartsch S, Volkamer M (2013) Mental models of verifiability in voting. In: Proceedings of 4th international conference of e-voting and identify, Vote-ID 2013, Guildford, pp 142–155. doi:10.1007/978-3-642-39185-9_9
9. Olsen KA, Nordhaug HF (2012) Internet elections: unsafe in any home? Commun ACM 55(8):36–38. doi:10.1145/2240236.2240251
10. Schneider S, Llewellyn M, Culnane C, Heather J, Srinivasan S, Xia Z (2011) Focus group views on Prêt à Voter 1.0. In: 2011 international workshop on requirements engineering for electronic voting systems, REVOTE 2011, Trento, pp 56–65. doi:10.1109/REVOTE.2011.6045916
11. Sherman AT, Carback R, Chaum D, Clark J, Essex A, Herrnson PS, Mayberry T, Popoveniuc S, Rivest RL, Shen E, Sinha B, Vora PL (2010) Scantegrity mock election at Takoma park. In: 4th international conference of electronic voting 2010, EVOTE 2010. Co-organized by Council of Europe, Gesellschaft für Informatik and E-Voting.CC. Castle Hofen, Bregenz, pp 45–61. http://subs.emis.de/LNI/Proceedings/Proceedings167/article5683.html
12. Stone D, Jarrett C, Woodroffe M, Minocha S (2005) User interface design and evaluation. Morgan Kaufmann
13. Weber JL, Hengartner U (2009) Usability study of the open audit voting system Helios. http://www.jannaweber.com/wp-content/uploads/2009/09/858Helios.pdf
14. Yao Y, Murphy LD (2007) Remote electronic voting systems: an exploration of voters' perceptions and intention to use. EJIS 16(2):106–120. doi:10.1057/palgrave.ejis.3000672