Attacking the Verification Code Mechanism in the Norwegian Internet Voting System

Reto E. Koenig, Philipp Locher, and Rolf Haenni

Bern University of Applied Sciences, CH-2501 Biel, Switzerland
{reto.koenig,philipp.locher,rolf.haenni}@bfh.ch

Abstract. The security of the Norwegian Internet voting system depends strongly on the implemented verification code mechanism, which allows voters to verify if their vote has been cast and recorded as intended. For this to work properly, a secure and independent auxiliary channel for transmitting the verification codes to the voters is required. The Norwegian system assumes that SMS satisfies the necessary requirements for such a channel. This paper demonstrates that this is no longer the case today. If voters use smartphones or tablet computers for receiving SMS messages, a number of new attack scenarios appear. We show how an adversary may exploit these scenarios in systems providing vote updating and point out the consequences for the vote integrity in the Norwegian system. We also give a list of possible counter-measures and system enhancements to prevent and detect such attacks.

1 Introduction

In the design and implementation of secure Internet voting systems, the *secure platform problem* is one of the most challenging obstacles to overcome [23]. Given the manifold vulnerabilities of today's computers, particularly those caused by malicious software, it is inappropriate to assume that voters will have access to a reliable machine that works correctly under all possible circumstances. Voting protocols must therefore be designed to deal with the possibility that some voters will use machines that are infected by various types of possibly very sophisticated malware. In a worst-case scenario, the malware is designated to attack particular voting events, while remaining completely silent and therefore hard to detect at other times. Attacks of such a type can be launched with a few mouse clicks. Since the correct outcome of an election is of great significance for the whole electorate, infected computers become immediately a problem for everybody.

Recent malware attacks in other application areas have demonstrated that they represent a real and serious threat today. In 2012, for example, estimated 36 million Euros were stolen from several ten thousand bank customers all across Europe by a smart Trojan called *Eurograbber* [14]. In a recent report, the number of new Windows-based malware in 2012 is estimated as almost 1.4 million [2]. Increasing numbers of new malware are reported for other platforms, in particular in the emerging area of mobile devices (smartphones and tablet computers).

1.1 Existing Approaches

A malware attack against an Internet voting system may aim at violating either the secrecy or the integrity of the vote (or both). Full protection against both types of attacks is very hard to achieve. Possible full protection approaches are based on distributing trustworthy hardware devices to the voters, but this is very expensive [8]. Some other approaches suggest using trusted out-of-band channels such as regular postal mail. The idea is to securely exchange additional information between the voting system and the voters, which allows them to protect the privacy or to verify the integrity of the vote. There are two related ways of using such an auxiliary channel, each of which with its own pros and cons.

Code Voting [10,13,19,22]. The idea of code voting is to enter candidate codes instead of candidate names when casting a vote. These codes are distributed over a separate secure channel in form of personalized code sheets and differ from voter to voter. This prevents the voter's insecure platform from learning the actual candidate choice and from guessing the codes of other candidates. This simple mechanism thus provides privacy and integrity in the presence of designated malware, but it does not prevent the malware from not casting the vote at all or from casting votes with invalid codes. Another major problem of code voting is the restricted usability.

Verification Codes [3,9,11,16,21]. The setting here is similar to code voting, but instead of entering the codes of the selected candidates as printed on the personalized code sheet, voters only need to check if the codes match with what is displayed by the voting system after casting the vote. If the codes match, the voter has a strong indication that the vote has been cast and recorded as intended.[1] Since verifying the codes is an optional step, usability is not so much of an issue as in code voting. On the other hand, verification codes alone cannot prevent malware from breaking the secrecy of the vote.

Code voting and verification codes can be applied separately or in combination. If applied in combination, the vote secrecy is protected and a strong indication is given that the vote has been cast and recorded as intended. A general problem that affects all possible scenarios is the secure printing of the code sheets. It is usually solved by organizational and non-cryptographic technical measures. Another general problem is the possibility that malware can learn the codes in a system that supports vote updating. As soon as several codes are known to the malware, it can start fooling the voter and possibly cast a final vote that is different from the voter's intention.

[1] Some existing systems, for example the system used in the canton of Geneva in Switzerland [4, 27], use a simplified type of verification code in form of a picture, which differs from voter to voter, but not from candidate to candidate. In such a case, a correct verification code only implies that some vote has reached the voting server, but it does not guarantee its integrity.

The Norwegian Internet voting system is based on verification codes and supports vote updating [7, 18, 25].[2] To avoid that malware learns the verification codes, the existence of two separate out-of-band channels is assumed: Initially, the code sheets are sent by postal mail to the voter's home address (pre-channel), and after vote casting, an SMS message with the verification codes of the selected candidates is sent to the voter's mobile device (post-channel). This introduces additional trust assumptions, for example that SMS provides a sufficiently secure and truly out-of-band channel, which is strictly detached and completely independent from the Internet (voting channel). Since SMS-based one-time passwords are used for voter authentication, these assumptions are even more critical and may have much farther-reaching consequences when violated.

1.2 Contribution and Overview

In this paper, we provide evidence that SMS is no longer a sufficiently secure candidate for the out-of-band post-channel in the Norwegian Internet voting system. The main problem is the widespread use of smart mobile devices today, which provide all sorts of new attack scenarios. Eurograbber is a prominent example that illustrates the practicability and efficiency of such attacks. In a recent report [15], two students of ours demonstrated that such attacks can be executed using even less infrastructure than was required by Eurograbber. Not surprisingly, the attacks presented in the report are directly transferable to the Norwegian system, as they rely on equivalent trust assumptions.

Based on these findings, we provide a systematic overview and detailed description of the attacks that result from using the SMS channel—as proposed in the Norwegian system—in combination with vote updating. Our analysis gives enough technical detail to understand the attacks not only from a conceptual point of view, but also from the perspective of implementing corresponding malware. For each attack scenario discussed in our overview, we point out the consequences with respect to the secrecy and integrity of the vote. We also discuss the effectiveness of each attack in terms of practicability, scalability, and detectability. Finally, we propose some counter-measures for preventing or detecting such attacks, and therefore contribute to the improvement of the Norwegian system.

2 The Norwegian E-Voting System

In this section, we provide a high-level description of the Norwegian voting system as described in the available literature [5–7, 9, 25]. The system overview is given from the voter's perspective, as the attacks presented in Section 3 are not targeted towards the server infrastructure. Then we explain precisely the role of the verification codes and the underlying trust assumptions.

[2] In the available documents about the Norwegian system, verification codes are called *return codes* [6] or *receipt codes* [7]. To avoid any ambiguity with the concept of a receipt in the sense of receipt-free voting systems, we prefer to call them verification codes. Furthermore, code sheets are called *poll cards* [6] or *voting cards* [7]. We prefer to call them code sheets, as it is common in the code voting literature.

2.1 Overview of the Voting Process

According to the analysis of the Norwegian system in [7], the main actors involved in the vote casting process are the voter V, the voter's computer P, the ballot box B, the receipt generator R, the decryption service D, and the auditor A. It is assumed that they communicate over secure, authenticated channels, as shown in Figure 1 (or in [7]). Unidirectional channels are represented by single-ended arrows. All other channels are bidirectional. The offline pre-channel for sending the personalized code sheets to the voters by postal mail is not shown. In this paper, we adopt the assumptions that both the printing service and postal mail are sufficiently reliable and secure, i.e., we presume that voters receive their code sheet before the voting period begins, and that the candidate codes remain secret during the printing and transmission process.

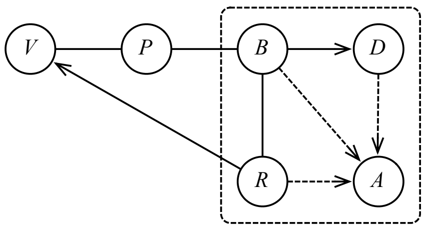

Fig. 1. The simplistic view of the communication channels of the Norwegian system as presented in [7]. This picture ignores the fact that most voters today receive SMS messages on smartphones or other mobile devices, which may be infected by malware. The direct arrow from the receipt generator to the voter is therefore an oversimplification of today's reality.

During the voting period, the voter first initiates the voter authentication mechanism provided by MinID, a two-level authentication service that was created to offer standardized authentication for various governmental services in Norway.[3] For this, the voter enters the personal credentials consisting of an identification number and a secret password. The MinID server then generates a one-time password (a five-digits code called *SMS code*) and sends it to the voter by SMS.[4] Finally, if the voter enters the correct SMS code, the authentication process succeeds.

After authentication, the voter selects the voting options using the computer and submits the vote. The submitted vote is encrypted and signed by the computer and sent to the ballot box. The ballot box blinds the vote and passes the blinded vote together with voter's personal identification number to the receipt generator. Based on the identification number the receipt generator computes the personalized verification codes of the blinded vote and sends them back to

[3] In upcoming Internet voting pilots, additional external authentication mechanisms will be supported (BankID, Buypass, Commfides).

[4] MinID users can order a sheet with multiple one-time passwords by postal mail.

the voter by SMS. Finally, the voter checks the received verification codes with those on the code sheet. At the end of the voting period, all encrypted votes are sent to the decryption service, where the votes are mixed and decrypted and the final tally is calculated. An auditor supervises the entire process.

Since vote updating is allowed in the Norwegian system, voters can repeat the above vote casting procedure multiple times, in the same or in different MinID sessions.[5] It is also possible to cast a final paper vote at the polling station, where any electronic vote is overridden. Note that vote updating provides some protection against vote buying and coercion.

2.2 Adversary Model and Trust Assumptions

The main purpose of the verification code mechanism is to give voters some feedback on whether the vote has been cast and recorded as intended. Matching verification codes should provide voters with strong evidence that everything worked properly, at least with very high probability, whereas non-matching or missing verification codes should provide strong evidence that something went wrong somewhere, which should then encourage voters to vote again, possibly on paper. For this mechanism to work—in addition to the assumption that the code sheets were generated, printed, and mailed securely—it is necessary to assume that the adversary's capabilities are restricted as follows [1, 7]:

1. The server infrastructure is not under the adversary's control, in particular
   - the MinID authentication service,
   - the ballot box,
   - the receipt generator.
2. The SMS post-channel cannot be intercepted, interrupted, or manipulated by the adversary.
3. The component on the voter's device used to receive and display SMS messages is not compromised or controlled by the adversary.
4. The adversary is polynomially bounded and thus incapable of breaking cryptographic primitives.

Otherwise, we consider an external adversary capable of controlling the Internet voting channel and compromising an arbitrary number of voting computers. Internal adversaries are excluded by the first assumption in the above list.

As the security of the Norwegian Internet voting system depends strongly on the verification code mechanism, any violation of the above assumptions may potentially lead to incorrect votes and thus to an incorrect election outcome. Of particular importance is the assumed security and independence of the SMS post-channel, over which the verification codes are transmitted to the voters. Accordingly, the post-channel as shown in Figure 1 goes from the receipt generator directly to the voter. Note that this is clearly a simplistic view, which does not

[5] The available protocol specification does not define whether vote updating in the same MinID session is supported or not [6, 7]. Therefore, we take this possibility into account in our analysis.

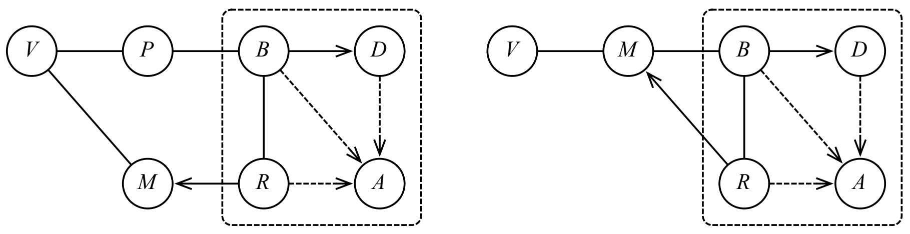

Fig. 2. A more complete picture of the communication channels in the Norwegian system: With two separate devices, a computer P and a mobile device M (left), or with a single mobile device M (right)

take into account that most users today use smartphones or tablet computers for sending and receiving SMS messages. Since these device are more and more comparable to ordinary computers, it is no longer legitimate to consider them immune against malware or other types of external attacks. Figure 2 shows two updates of Figure 1, one with an additional mobile device M and one with M replacing the voter's computer (e.g., when using a tablet computer for vote casting and as an SMS receiver). In comparison with Figure 1, this obviously creates a number of additional attack scenarios. We will discuss them in detail in the following section.

3 Attacking the SMS Channel in the Norwegian System

The verification mechanism in the Norwegian Internet voting system is thought to be a solution for the secure platform problem. It does not prevent malware from taking full control over the voter's computer or its web browser, but it aims at making such an attack detectable. In this section, we assume that the voter's computer is infected by malware, but that the voter is not aware of the infection. We suppose that the malware mainly resides inside the voter's web browser and is therefore capable of launching all sorts of *man-in-the-browser* (MITB) attacks against arbitrary web applications, including the web application of the Norwegian system. It may therefore modify the content of the displayed web pages, modify incoming or outgoing transactions, insert additional transactions, communicate with other computers over the network, or even use the computer's local WLAN, Bluetooth, IrDA (infrared communication), or audio interfaces. This can all happen in a completely covert fashion invisible to both the voter and the web application.

If such an MITB malware resides on the voter's computer, it can easily break the secrecy of the vote, for example by observing the voter's interaction with the web application and by transmitting a transcript to a remote computer. But this is not what the vote verification mechanism tries to avoid, its goal is only to protect the integrity of the vote. For this, the verification codes cannot simply be displayed on the screen of the voter's computer, because this would allow the malware to silently submit a vote update without the voter noticing. This is why the codes need to be delivered out-of-band.

In this section, we discuss three different attack scenarios for breaking the vote integrity in the Norwegian system. They correspond to the three pictures shown in Figures 1 and 2. In the first scenario, the SMS channel is attacked directly, by operating a fake GSM base transceiver station in the voter's proximity. In the second and third scenario, the mobile device used to receive SMS messages is attacked with additional malware. The practicability, scalability, and detectability of corresponding attacks is different in each scenario.

3.1 Attacking the Security of the SMS Channel

There are two different ways of attacking the SMS channel by an external adversary. Since SMS messages are transmitted over the GSM network, the airway traffic between the closest base transceiver station (BTS) and the mobile device is (optionally) encrypted with the weak and broken stream cipher A5/1 or A5/2. A *passive* attack consists in intercepting this traffic and by decrypting the A5-encrypted content. This attack requires only low-budget hardware (less than \$100) and open-source software such as AirProbe and OsmocomBB.[6]

The practicability of an *active* attack with a fake GSM base transceiver station (also called *IMSI catcher*) has been shown by Chris Paget during a live demonstration at the Defcon conference in 2010 with a \$1500 device made from of-the-shelf hardware and an open-source software called OpenBTS.[7] Once such a fake BTS is operating, it can serve as a proxy between the real GSM network and the GSM phones of any kind, covering an area of up to 35km radius. As GSM does not provide any kind of sender authentication, it is a simple task for a proxy to intercept, block, or fabricate SMS messages [12,17,20,24,26].

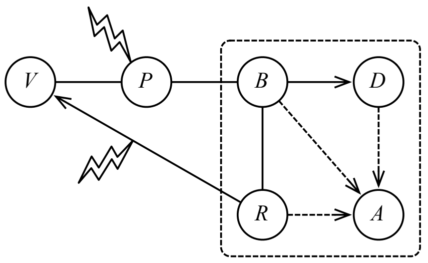

Fig. 3. A combined and synchronized attack against the voter's computer and the SMS channel

If the SMS post-channel in the Norwegian system is attacked as explained above, it can no longer protect the integrity of the vote, even if the SMS verification codes obtained after vote casting match. The attack scenario in Figure 3

[6] See `http://bb.osmocom.org` and `https://svn.berlin.ccc.de/projects/airprobe`.

[7] See `http://wush.net/trac/rangepublic`.

shows a situation where both the voter's computer and the SMS post-channel are under attack. If these attacks are coordinated by the same adversary, it is easy to bypass the verification code mechanism in the following way.

Attack 3.1. *Silent vote updating with blocked verification codes by fake BTS.*

1. *When a vote is cast, the MITB malware informs the fake BTS to withhold the second next SMS message from the receipt generator.*
2. *The fake BTS transmits the first SMS message from the receipt generator to the voter's mobile phone.*
3. *Within the same MinID session, but possibly with some delay to not attract too much attention, the MITB malware silently casts a second vote (vote updating).*
4. *The fake BTS blocks the second SMS message from the receipt generator.*

This attack is executed every time the voter casts a vote. The confirmation codes obtained over the SMS channel will always match and thus let the voter think that everything worked properly. As nothing suspicious happens on either side, except maybe for an increased percentage of vote updates, the attack is likely to remain unnoticed by both the voter and the voting system. However, the presence of a fake BTS is something that will draw somebody's attention sooner or later. In such a case, determining the location of the fake BTS is clearly not very difficult.

This attack clearly violates the vote integrity of the voters under attack, and thus leads to an incorrect election outcome. It is thus a serious security problem for the Norwegian Internet voting system. On the other hand, as the attack requires hardware infrastructure and maintenance from the adversary, and is geographically limited to the signal perimeter of the fake BTS, it has a very limited scalability and is therefore mainly applicable to local municipality elections.

3.2 Attacking the Independence of the SMS Channel

The key of the attack in the previous section is the adversary's ability to block the SMS message with the verification codes. This allows the MITB malware to silently submit a second vote without any noticeable consequences. The two attacks presented in this section are based on the same idea, but they do not require any hardware infrastructure to interrupt the SMS channel between the receipt generator and the voter. Figure 4 shows two different attack scenarios in which the required independence between the voting channel and the SMS post-channel is violated. This means that verification codes can no longer be transmitted out-of-band. In the first scenario, the adversary controls both the voter's computer P and the SMS-receiving mobile device M. In the second scenario, the adversary controls the single mobile device M used for casting votes and receiving SMS messages.

Attacking Two Devices. We consider now a combined attack against both the voter's computer and the voter's mobile device. Installing respective malware

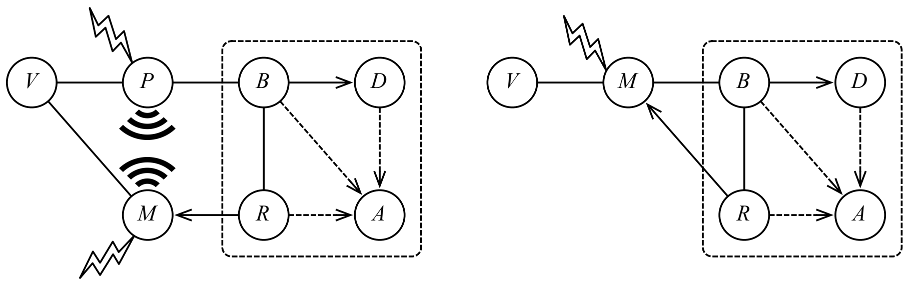

Fig. 4. Two attack scenarios of blocking the SMS post-channel for voters using a smart mobile device

on two different devices may appear nearly impossible, but one can easily think of a sophisticated social engineering attack, possibly launched by the malware of the first infected device to finally infect the other device with its malicious counterpart. For example, suppose the MITB malware on the voter's computer gets into possession of the voter's mobile phone number. Then it can send an SMS message to the voter's mobile device with a download invitation for a seemingly useful free app, for instance one the presents the political profiles of the candidates in the forthcoming election. If the voter follows the instructions, the double infection is established. The infection in the Eurograbber attack, where an SMS message instructed the victims to install a security update on their mobile device, was a social engineering attack of that type [14].

Recent technology advances may allow even simpler ways of infecting two devices simultaneously. A general trend pushed by the major technology providers today is to offer a uniform user experience on multiple devices with cloud-based synchronization of personal data and apps. Already today, both the Android and the iOS operating systems allow automatic app installation across multiple devices. The MITB malware on the voter's computer could therefore silently select its free malicious counterpart in the app store and therefore initiate the installation on the mobile device with ease, and without attracting too much attention.

If we suppose that both involved devices are infected by malware and thus controlled by the adversary, the attack is almost straightforward.

Attack 3.2. *Silent vote updating with blocked verification codes on the voter's mobile device.*

1. *When a vote is cast, the MITB malware informs its counterpart on the mobile device to withhold the second next SMS message from the receipt generator.*
2. *When the first SMS message from the receipt generator arrives, it is stored and displayed as usual on the mobile device.*

3. *Within in the same MinID session, but possibly with some delay to not attract too much attention, the MITB malware silently casts a second vote (vote updating).*
4. *When the second SMS message from the receipt generator arrives, it is blocked and immediately deleted by the malware on the mobile device.*

In Step 1 of this attack, the two malware counterparts need to setup a unidirectional communication channel from the voter's computer to the mobile device. We consider four attack options for doing so:

a) One of the simplest options is Bluetooth. Provided that the two devices are already paired, a Bluetooth connection can be established very easily and without the user noticing. Since Bluetooth connections are point-to-point, no traces are left on devices other than the ones involved in the attack. Sophisticated malware takes care of removing all possible local traces such as entries in log files or in the device's usage history.
b) A standard TCP/IP network connection is another simple option, provided that the mobile device is connected at the time of the attack. If both devices are connected to the same LAN, 'finding' each other is rather simple, for example by locally broadcasting an inconspicuous handshake message. If they are not connected to the same LAN, for instance if the mobile device is connected over a cellular data network (GPRS, UMTS, EDGE, LTE), then a remote server to which each malware connects is needed to establish the connection. In both cases, some inevitable traces are left along the path that the message has taken in the network. These traces may help to uncover the attack in suspicious cases.
c) If the malware on the voter's computer knows the mobile device's phone number, then sending an SMS message or making a call to alert the mobile device might be another easy-to-implement option. When the mobile phone receives such an alert SMS, it silently blocks and immediately deletes the message. If it receives a call, the call is suppressed and deleted from the list of incoming calls. Traces are only left at the voter's mobile network carrier, through which the SMS message or call is delivered.
d) Another interesting option is to use the audio output of the voter's computer to emit a ultrasonic audio signal (>20kHz), which is not audible by humans, but which can be captured by the mobile device's microphone.[8] This leaves no traces at all on both sides.

The fact that most of these options leave almost no evident traces, they are very hard to detect in real-time and almost impossible to uncover in retrospect. Furthermore, as no external command-and-control mechanism is needed in order to execute the attack on the voter's side, there is no channel back to the adversary.

[8] While almost all microphones are capable of capturing ultrasonic signals, only so-called *piezoelectric loudspeakers* are able to emit them. If no piezoelectric loudspeakers are available at the voter's computer, an audible acoustic signal could be used to carry out the attack. This may attract the voter's attention, but not necessarily raise much suspicion if obfuscated properly.

This simplifies the preparation and execution of the attack and decreases the chance of being detected.

Several of the above attack options have been implemented in the attack against mTAN-based online banking applications as presented in [15]. This report demonstrates that implementing such an attack is surprisingly simply and only requires basic knowledge in IT security and limited skills in programming mobile devices. For example, taking full control over the SMS functionality in the Android operating system is done in a few lines of code, including the deletion of corresponding log file entries.[9] Even sending and receiving messages over an ultrasonic channel is just a matter of using the right software libraries.[10]

Attacking a Single Device. Finally, we suppose that the voter uses the same device for casting the vote and for receiving the verification code. Today's tablet computers provide both a full-featured web browser with a sufficiently large display and GSM communication for sending and receiving SMS messages. This attack scenario with a single mobile device M is shown on the right hand side of Figure 4. Among the scenarios discussed in this section, it is clearly the simplest one. Compared to Attack 3.1, no additional hardware equipment is needed to execute the attack. Compared to Attack 3.2, only a single device needs to be infected by malware and no additional communication channel needs to be established. As the malware needs to control both the web browser and the SMS component of the voter's mobile device, it must be slightly more powerful than a simple MITB malware. The concrete steps for the malware to execute the attack are the following.

Attack 3.3. *Silent vote updating with blocked verification codes on a single mobile device.*

1. *When a vote is cast, the malware on the voter's mobile device starts monitoring the incoming SMS messages.*
2. *When the first SMS message from the receipt generator arrives, it is stored and displayed as usual on the mobile device.*
3. *Within in the same MinID session, but possibly with some delay to not attract too much attention, the malware silently casts a second vote (vote updating).*
4. *When the second SMS message from the receipt generator arrives, it is blocked and immediately deleted.*

Besides its simplicity, this attack has the advantage of only affecting the voter's mobile device and thus not leaving any traces at all at other places. Even the traces left on the infected device can be entirely removed, if the malware is sophisticated enough to delete all entries in corresponding log files, in the device's usage history, or in the list of notifications. Therefore, the adversary's risk of being detected is even smaller than in the two attacks presented before. On the

[9] *SMS Popup* provides free source code for programming SMS applications in the Android operating system, see `http://code.google.com/p/android-smspopup`.

[10] *SSCConnect* is an example of an ultrasonic communication tool for both the Android and the iOS operating systems, see `http://www.sonicom.co.kr/main_eng/m_3_1.php`.

other hand, as long as only a minority of voters is using a single mobile device for both vote casting and verification, the scalability of the attack is fairly limited.

3.3 Attacking the MinID Authentication Service

All three attacks presented so far are based on the assumption that the SMS post-channel is under the adversary's control. If this is actually the case, a more general attack with much farther-reaching consequences can be launched in a very similar way against the MinID authentication service. It is also based on blocking incoming SMS messages, the ones that contain the one-time passwords from the MinID authentication server. The attack scenario here is identical to the online banking attacks presented in [15].

The attack works in all three scenarios presented so far. As an example, we consider here the scenario with two devices, one for accessing the MinID-based web application and one for receiving the SMS message with the one-time password. As before, we suppose that both devices are infected by malware from the same adversary. To prepare an attack, the MITB malware intercepts the victim's MinID credentials during the login process of a MinID-based service. The credentials are either stored locally on the victim's computer, sent over the network to a remote server, or transmitted to the malware's counterpart on the victim's mobile device (using one of the unidirectional communication channels proposed in Attack 3.2). The place where the credentials are stored or sent to determines the location from where the actual attack will be launched.

The simplest possibility for executing the attack is the third one. In that case, the malware on the mobile device knows the MinID credentials and is able to block incoming SMS messages. It can then initiate the MinID login process, submit the victims's correct credentials, retrieve the one-time password from the incoming SMS message, block the SMS message from being displayed to the user, and finally submit the one-time password to complete the authentication process. The web application will then accept the malware as a legitimate user and give access to its resources and functionalities. All this happens silently in the background, possibly while the user is not using the device.

All MinID-based applications are affected equally by this attack. In an attack against the Norwegian voting system, it could be used to postpone the final vote cast to a different MinID session, which obfuscates the attack further. Note that the attack is even easier to implement in the scenario with a single device.

4 Preventing and Detecting the Attack

What should the people behind the Norwegian system do in the light of the attacks presented in the previous section? In this section, we suggest some countermeasures and enhancements, which may help detecting or even preventing such attacks. Recall that all attacks depend on the vote updating feature of the Norwegian system and on the adversary's ability to silently block and discard incoming SMS messages from the receipt generator. The proposition listed below can be implemented separately or in combination.

No Vote Updating. The simplest but most radical counter-measure against all proposed attacks is to discard vote updating on the electronic channel. This feature offers some protection against vote buying and coercion, but it is also responsible for undermining the verification code mechanism in the proposed attacks. Without vote updating, the first submitted vote is the one that counts. This does not prevent the malware on the voter's computer from submitting a different vote, but then the verification codes will no longer match. The malware on the voter's mobile device could also try to block the SMS message from the receipt generator, but this would immediately make the voter suspicious. In both cases, voters are instructed to cast a final vote on paper. This shows that the verification code mechanism perfectly works without vote updating. To allow or to discard vote updating is therefore a trade-off between solving the secure platform problem or preventing vote buying and coercion.

Vote Updating in Different Sessions. A less drastic counter-measure is to allow vote updating, but to require different MinID sessions for doing so.[11] This would clearly prevent the malware from executing the third step in each of the Attacks 3.1 to 3.3, but the adversary would still have the option of attacking the MinID authentication service as presented in Section 3.3. The people behind the Norwegian system could then argue that they are not responsible for the MinID security, but this does not solve the problem. Note that the MinID security could be improved by delivering multiple one-time passwords beforehand by postal mail (which is currently an option for users with no registered phone number) or on a secure hardware token.

Voting TAN. Another possible counter-measure is based on the fact that a secure pre-channel already exists for delivering the code sheets to the voters by postal mail. This channel could therefore be used for other purposes without much additional effort. We propose to include an indexed list of additional one-time passwords. Such a password plays the role of a *transaction authentication number* (TAN), which voters need to enter for casting a vote. As the malware will always be unaware of the correct next voting TAN, this would again prevent the third step in each of the Attacks 3.1 to 3.3. The adversary could then try to get into possession of some voting TANs with a phishing attack, but this would clearly lower the overall effectiveness of the attack. Enhancing the Norwegian voting system with voting TANs seems therefore to be a viable solution, even if it slightly decreases the usability. Note that it also restricts the number of vote updates, which could be exploited by a vote buyer or a coercer.

CAPTCHA. Each attack presented in this paper exploits the fact that the voting servers cannot distinguish if a vote has been cast by the voter or by the malware on the voter's computer. CAPTCHA is a widely applied challenge-response test to prevent automated software from performing actions in behalf

[11] From showing a draft of this paper to Christian Bull, the Chief Security Officer of the Norwegian e-voting project, we have learned that the actual implementation already prevents vote updating within the same MinID session.

of humans. Voters could therefore be asked to solve a CAPTCHA when casting a vote. Under the assumption that CAPTCHAs cannot be solved automatically by the malware (using sophisticated OCR software or cheap human labor), this would again prevent the execution of the third step in each of the Attacks 3.1 to 3.3. Note that enhancing the Norwegian system with CAPTCHAs is very easy to implement, but it would decrease the overall usability.

Trusted SMS Receiver. If we neglect the threat of Attack 3.1, in which a hardware infrastructure is needed to interrupt the SMS post-channel, we can assume that the SMS message from the receipt generator reaches the mobile device to which it is sent. The remaining problem then is the malware's ability to block the SMS message from being displayed, as proposed in Attack 3.2 and Attack 3.3. To solve this problem, voters could be equipped with a trusted tiny SMS receiver, which does nothing else than displaying incoming SMS messages from a dedicated set of certified senders (receipt generator, MinID authentication service). If produced in large numbers, we expect the price for such tiny SMS receivers with a small display and an integrated SIM card to be reasonably small.

An additional measure to prevent Attack 3.1 is to attach a one-time password to the SMS message from the receipt generator. After checking the verification codes, the voter needs to enter the password to finalize the vote cast. To prevent the adversary from intercepting this password when the SMS message passes through the fake BTS, it must be encrypted by the receipt generator and decrypted by the trusted SMS receiver. Another possible counter-measure against Attack 3.1 is to let the trusted SMS receiver automatically send a digitally signed SMS acknowledgment back to the receipt generator.

Trusted Hardware Token. In the previous solution with the trusted SMS receiver, we suggested a digitally signed SMS acknowledgment as an ultimate measure to prevent Attack 3.1. Note that the same idea works independently of using SMS as transmission channel. The SMS receiver could therefore be replaced by a trusted hardware token with the ability to receive and display encrypted messages and to return a signed acknowledgment to the receipt generator. Such devices are available on the market, for example the *Zone Trusted Information Channel* (ZTIC), which establishes an end-to-end TLS connection via the user's untrusted computer to a remote server [28, 29]. Under the assumption that such devices are trustworthy in displaying all incoming messages, the MITB malware could still block the final acknowledgment message, but this would attract attention on the server side where the acknowledgment is expected.

Analyzing Vote Update Patterns. The common denominator of the attacks presented in this paper is that they all exploit the vote updating feature of the Norwegian system. Shortly after a vote has been cast from an infected computer, a vote update from the same voter is initiated by the malware. This obviously generates statistical patterns in the electronic ballot box, which differ from normal usage patterns. Even the vote updating frequency may already be a good

Table 1. Overview of the strengths and weaknesses of the three attacks and corresponding options presented in this paper

		Practicability	*Scalability*	*Non-Detectability*	*Overall Risk*
Attack 3.1		low	low	low	low
Attack 3.2	*a.*	medium	medium	high	medium
	b.		high	medium	medium
	c.		high	medium	medium
	d.		high	high	high
Attack 3.3		high	medium	high	high

indicator for detecting an attack on a statistical basis. Corresponding plausibility tests could be added to the Norwegian system.[12]

5 Conclusion

In this paper, we analyzed the security of the verification code mechanism in the Norwegian Internet voting system. Our analysis takes the perspective of an adversary, who tries to interrupt the SMS post-channel, which is needed to transmit the verification codes to the voter after casting a vote. We investigated three different attack scenarios, in which the adversary is able to submit a fake vote update in the name of an eligible voter. The key for the adversary to execute such an attack is to block the SMS message from the receipt generator, which would make the voter suspicious when delivered. In each attack scenario, the interruption of the SMS channel is achieved in a different way.

In Table 1, we summarize the three attack scenarios by rating them with respect to the most relevant criteria: practicability, scalability, non-detectability. Attack 3.1 is clearly the least practical and scalable one, and the presence of a fake BTS is likely to attract attention. Hence, it receives our weakest overall rating. Much more practical and scalable is Attack 3.2, even if infecting two devices simultaneously with malware poses a supplementary challenge to get started. In Attack 3.3, only a single device needs to be infected, but the scalability is limited to the percentage of voters to which this attack scenario applies. Since Attack 3.2 (depending on the chosen option) and Attack 3.3 are also very hard to detect, we evaluate them both as highly risky for the Norwegian system.

Our final conclusion and general recommendation for the persons in charge of the Norwegian system is not to underestimate these types attacks. Similar attacks against online banking services have already demonstrated that they represent a real threat, and we know from our students that they are not very difficult to implement. Therefore, we recommend making the Norwegian system more resistant against these types of attack, for example by implementing some of the proposed counter-measures.

[12] In the 2011 pilots of the Norwegian Internet voting system, 3.6% of the voters eligible to vote electronically submitted multiple votes.

As a final note, we want to emphasize that the main threat results from the smartphone, which is an insecure platform per se. Implementing the verification code mechanism as a smartphone app is therefore not a viable solution.[13]

Acknowledgments. We thank the anonymous reviewers for their thorough reviews and appreciate the comments and suggestions. This research has been supported by the Swiss National Science Foundation (project No. 200021L_140650).

References

1. Ansper, A., Heiberg, S., Lipmaa, H., Øverland, T.A., van Laenen, F.: Security and trust for the Norwegian e-voting pilot project E-Valg 2011. In: Jøsang, A., Maseng, T., Knapskog, S.J. (eds.) NordSec 2009. LNCS, vol. 5838, pp. 207–222. Springer, Heidelberg (2009)
2. Benzmüller, R.: MalwareReport: Half-yearly report (January-June 2012). Tech. rep., G Data SecurityLabs (2012)
3. Chaum, D., Carback, R., Clark, J., Essex, A., Popoveniuc, S., Rivest, R.L., Ryan, P.Y.A., Shen, E., Sherman, A.T., Vora, P.L.: Scantegrity II: End-to-end verifiability by voters of optical scan elections through confirmation codes. IEEE Transactions on Information Forensics and Security 4(4), 611–627 (2009)
4. Chevallier, M., Warynski, M., Sandoz, A.: Success factors of Geneva's e-voting system. Electronic Journal of e-Government 4(2), 71–78 (2006)
5. Cortier, V., Wiedling, C.: A formal analysis of the Norwegian E-voting protocol. In: Degano, P., Guttman, J.D. (eds.) POST 2012. LNCS, vol. 7215, pp. 109–128. Springer, Heidelberg (2012)
6. Gebhardt Stenerud, I.S., Bull, C.: When reality comes knocking–Norwegian experiences with verifiable electronic voting. In: 5th International Workshop on Electronic Voting, EVOTE 2012, Bregenz, Austria, pp. 21–33 (2012)
7. Gjøsteen, K.: Analysis of an internet voting protocol. IACR Cryptology ePrint Archive 2010/380 (2010)
8. Haenni, R., Koenig, R.E.: Voting over the Internet on an insecure platform. In: Design, Development, and Use of Secure Electronic Voting Systems. IGI Global (accepted, 2013)
9. Heiberg, S., Lipmaa, H., van Laenen, F.: On e-vote integrity in the case of malicious voter computers. In: Gritzalis, D., Preneel, B., Theoharidou, M. (eds.) ESORICS 2010. LNCS, vol. 6345, pp. 373–388. Springer, Heidelberg (2010)
10. Helbach, J., Schwenk, J.: Secure internet voting with code sheets. In: Alkassar, A., Volkamer, M. (eds.) VOTE-ID 2007. LNCS, vol. 4896, pp. 166–177. Springer, Heidelberg (2007)
11. Helbach, J., Schwenk, J., Schäge, S.: Code voting with linkable group signatures. In: Krimmer, R., Grimm, R. (eds.) 3rd International Workshop on Electronic Voting, EVOTE 2008. Lecture Notes in Informatics, vol. P-131, pp. 209–222. Gesellschaft für Informatik E.V., Bregenz (2008)
12. Hubacher, I.: Management Demo: Intercepting SMS. Bachelor thesis, Bern University of Applied Sciences, Biel, Switzerland (2011)

[13] In a recent talk at the *Verifiable Voting Schemes Workshop* in Luxembourg on March 22nd, 2013, Jan Willemson suggested such a solution for the Estonian system.

13. Joaquim, R., Ribeiro, C., Ferreira, P.: Improving remote voting security with code-Voting. In: Chaum, D., Jakobsson, M., Rivest, R.L., Ryan, P.Y.A., Benaloh, J., Kutylowski, M., Adida, B. (eds.) Towards Trustworthy Elections. LNCS, vol. 6000, pp. 310–329. Springer, Heidelberg (2010)
14. Kalige, E., Burkey, D.: A case study of Eurograbber: How 36 million euros was stolen via malware. Tech. rep., Versafe & Check Point Software Technologie (2012)
15. Klaus, S., Brei, D.: Sicherheit von E-Banking auf Smart-Platforms. Bachelor thesis, Bern University of Applied Sciences, Biel, Switzerland (2013)
16. Lipmaa, H.: Two simple code-verification voting protocols. IACR Cryptology ePrint Archive 2011/317 (2011)
17. Meyer, U., Wetzel, S.: On the impact of GSM encryption and man-in-the-middle attacks on the security of interoperating GSM/UMTS networks. In: 15th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC 2004, Barcelona, Spain, vol. 4, pp. 2876–2883 (2004)
18. Øberg, M.W.: Improving the Norwegian Internet Voting Protocol. Master's thesis, Norwegian University of Science and Technology (2011)
19. Oppliger, R., Schwenk, J., Helbach, J.: Protecting code voting against vote selling. In: 4. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e.V., Sicherheit 2008, Saarbrücken, Germany, pp. 193–204 (2008)
20. Perez, D., Pico, J.: A practical attack against GPRS/EDGE/UMTS/HSPA mobile data communications. White paper, Taddong S.L. (2011)
21. Ryan, P.Y.A.: Prêt à voter with confirmation codes. In: Shacham, H., Teague, V. (eds.) Electronic Voting Technology Workshop/Workshop on Trustworthy Elections, EVT/WOTE 2011, San Francisco, USA (2011)
22. Ryan, P.Y.A., Teague, V.: Pretty good democracy. In: Christianson, B., Malcolm, J.A., Matyáš, V., Roe, M. (eds.) Security Protocols 2009. LNCS, vol. 7028, pp. 111–130. Springer, Heidelberg (2013)
23. Schläpfer, M., Volkamer, M.: The secure platform problem: Taxonomy and analysis of existing proposals to address this problem. In: 6th International Conference on Theory and Practice of Electronic Governance, ICEGOV 2012, Albany, USA (2012)
24. Song, Y., Zhou, K., Chen, X.: Fake BTS attacks of GSM system on software radio platform. Journal of Networks 7(2), 275–281 (2012)
25. Spycher, O., Volkamer, M., Koenig, R.: Transparency and technical measures to establish trust in Norwegian Internet voting. In: Kiayias, A., Lipmaa, H. (eds.) VoteID 2011. LNCS, vol. 7187, pp. 19–35. Springer, Heidelberg (2012)
26. van den Broek, F.: Catching and Understanding GSM-Signals. Master's thesis, Radboud University Nijmegen (2010)
27. von Bergen, P.: Analyse du code source de l'application d'e-voting de Genève. Project report, Bern University of Applied Sciences, Biel, Switzerland (2013)
28. Weigold, T., Hiltgen, A.: Secure confirmation of sensitive transaction data in modern Internet banking services. In: World Congress on Internet Security, WorldCIS 2011, London, U.K., pp. 125–132 (2011)
29. Weigold, T., Kramp, T., Hermann, R., Höring, F., Buhler, P., Baentsch, M.: The Zurich Trusted Information Channel – An efficient defence against man-in-the-middle and malicious software attacks. In: Lipp, P., Sadeghi, A.-R., Koch, K.-M. (eds.) Trust 2008. LNCS, vol. 4968, pp. 75–91. Springer, Heidelberg (2008)