A Usable Android Application Implementing Distributed Cryptography For Election Authorities

Stephan Neumann, Oksana Kulyk, Melanie Volkamer
Technische Universität Darmstadt / CASED, Darmstadt, Germany
name.surname@cased.de

***Abstract*—Although many electronic voting protocols have been proposed, their practical application faces various challenges. One of these challenges is, that these protocols require election authorities to perform complex tasks like generating keys in a distributed manner and decrypting votes in a distributed and verifiable manner. Although corresponding key generation and decryption protocols exist, they are not used in real-world elections for several reasons: The few existing implementations of these protocols and their corresponding interfaces are not designed for people with non technical background and thus not suitable for use by most election authorities. In addition, it is difficult to explain the security model of the protocols, but legal provisions generally require transparency. We implemented a smartphone application for election authorities featuring distributed key generation and verifiable distributed decryption of votes. In addition, we prepared education material throughout based on formulated metaphors for election authorities in order to explain the security of the application. We evaluated the usability of the application and understanding of the underlying security model, concluding that the application is usable for non-experts in computer science. While the participants were able to carry out the tasks, it became clear, that they did not have a clear understanding of the underlying security model, despite having viewed our educational material. We suggest improvements to this material as future work.**

I. Introduction

Electronic voting is increasingly used for different kinds of elections, e.g. parliamentary elections in Estonia [1] and Norway [2], and non-parliamentary ones like the presidential election at Université catholique de Louvain [3] and the election at the International Association for Cryptologic Research [4]. While different remote electronic voting systems are used, they all have in common that they are based on different cryptographic primitives and protocols in order to provide secrecy and verifiability at the same time. Most of these electronic voting systems are designed in the following way: The voter makes her selection, encrypts her selection with the election key, and sends her vote to a public web bulletin board. This board publishes the encrypted vote together with the voter's identity ensuring eligibility and supporting verifiability. At the end of the election, the encrypted votes are anonymized by the use of a mix-net or by building a homomorphic sum. Eventually, the anonymized votes (in case of the mix-net anonymization) or the sum (in case of the homomorphic sum) are/is decrypted.

If the decryption key were in the hands of a single election authority, this authority could decrypt the non-anonymized encrypted votes from the web bulletin board and thereby violate secrecy. Therefore, it is crucial to distribute the decryption key among several authorities in a way that an *ideal trade-off function* between secrecy and robustness is provided while robustness means the ability to decrypt successfully the votes (or the sum) also in case some of the election authorities are not available. An ideal trade-off function between secrecy and robustness is defined by the fact that secrecy improvements and robustness decrease (or vice versa) can be adjusted proportionally, so that a desired trade-off could be easily achieved by corresponding adjustments. Protocols providing such an ideal trade-off function have been proposed in the literature [5], [6] and implementations of these protocols are available [7], [8]. However, these protocols and implementations have not yet arrived in practice as the elections at the Université catholique de Louvain [3], in Austria [9], in Estonia [1], Spain [10], and Norway [2] confirm. For instance, during the election in Norway in 2011 [2], the election key was distributed among the members by the electoral board, yet generated by two entities, the collaboration of which could thus violate secrecy. We postulate that approaches with non-ideal trade-off functions are in place because existing implementations do not meet real-world election settings. For instance, election authorities might well posses smartphones but not laptops, or it might not be possible to rely on established public key infrastructures (PKI), or election authorities cannot be assumed to have a background in computer science.

In order to improve the situation in future, the *first goal of our research project is to develop a usable application implementing distributed cryptography addressing real-world election settings*. Elections are bound to rigorous legal requirements due to their high social impact. Apart from security requirements, transparency of the entire election process is of central importance for establishing trust in the election and its outcome [11], and its security needs to be communicated [12]. *Therefore, the second goal of our research project is to explain the need for such an application, as well as the security properties and security models underlying the application.*

As our contribution, we provide a smartphone application for (1) the generation, exchange and validation of personal public communication keys of the election authorities, (2) the distributed generation of a public key used in the election for encrypting the votes (in this paper referred to as the *election key*) and of the corresponding private key shares used for decryption; and (3) the verifiable distributed decryption of a set of encrypted votes which are downloaded from the web bulletin board. We furthermore developed education material for election authorities, which would accompany the

978-1-4799-4223-7/14 $31.00 © 2014 IEEE
DOI 10.1109/ARES.2014.35

application. Eventually, we initiated a user evaluation to assess the application's usability and its understandability. The results of the first evaluation group (five participants) show that the smartphone application is usable according to the established usability criteria and that participants properly performed the required checks. The participants were able to understand those parts of the security model that were addressed in the education material. However, they were less able to answer those questions requiring transfer skills which indicates that even more effective education is necessary. Therefore, we decided not to continue the evaluation until we improve the education process.

II. Election scenario

Our project setting is close to real-world elections: we consider elections (or handle election districts) with one thousand voters, since 824 is an average number of postal voters in a district in Germany [13] and five election authorities, because there must be at least five election authorities per electoral districts in Germany [14]. Additionally, election authorities are citizens without special information security knowledge. Furthermore, we assume election authorities not to have (national) eID, as not all countries have issued eID cards, and even in those that have, the percentage of population possessing the eID is not big enough[1]

We consider in our research project an abstract, yet widely implemented voting protocol based on El-Gamal[2]encryption: In the election setup phase, an El-Gamal election key (together with the corresponding private key shares) is generated by the election authorities with a corresponding distributed key generation protocol. During the vote casting, votes are encrypted with this public El-Gamal election key and submitted to a so-called web bulletin board (WBB); i.e. a remote web server with a database connection with the public having read access and observers taking copies in order to observe whether the WBB is not deleting votes. There, the encrypted votes are stored together with some information identifying the voter. We refer to these encrypted votes as personalized votes. In the tallying phase these personalized votes are anonymized using a verifiable re-encryption mix-net, e.g. the mix-net proposed in [16]. The choice of mix-net as a way to anonymize the votes has an impact on the efficiency of the decryption process. This stems from the fact that as opposed to one ciphertext decryption (as in the case of homomorphic sum tallying), all individual anonymized ciphertexts need to be decrypted. Yet we decided to build upon the mix-net approach because it has been used in parliamentary elections in Estonia [1] and Norway [2]. The election authorities download the anonymized votes from the WBB and run the verifiable distributed decryption protocol for calculating the election result.

The goal of our project is to support election authorities during election setup and tallying for the described abstract voting protocol. We assume that they either meet in person or make phone/video calls in both phases, i.e. there is an out-of-band channel to interchange information[3]. However, it might be that individual election authorities are not able to show up or no longer behave honestly. Therefore, we need to implement a threshold distributed key generation protocol and thus a threshold verifiable distributed decryption protocol, enabling the tallying of results even if only a threshold of election authorities is participating. For our particular setting we decided to choose a threshold of three, which we consider to provide an optimal trade-off between secrecy and robustness.

For practicability, it was decided to develop a smartphone application, as these devices are nowadays widespread, and the number of people using them still grows[4]. Furthermore, smartphones are equipped with mobile Internet that can be used for communications in case setting up a wireless network is not possible or requires too much organizational effort.

III. Prototype development

In this section we propose and discuss various design decisions for the development of the application.

A. Hardware and Software

We developed a prototype application on the Android platform due to Android's flexibility and openness. Correspondingly, the application is written in Java. We use two external libraries: aSmack [17] to implement the communication between the smartphones and SpongeCastle [18], which is an Android version of the BouncyCastle library, to implement the cryptographic operations, as these are, to our best knowledge, currently the most popular and well-developed libraries that are applicable for these purposes on Android.

B. Cryptographic Primitives and Protocols

First of all we had to decide which *distributed key generation protocol* we are going to implement. Several protocols for threshold secret sharing, which is sharing a secret between several participants, so that a threshold amount of them could reconstruct the secret by cooperating, have been proposed in the literature such as [6], [19]–[21]; some with central authority, some without. In our application, we use a protocol introduced by Pedersen [6]. One significant advantage of this protocol is that the keys shares are generated and distributed in fully decentralized manner; i.e. no single entity ever possesses the key which is important for elections (this entity could break secrecy of the vote). Furthermore, the protocol includes a commitment round, which allows for verifying the correct execution of the distributed key generation protocol. This is important to know before starting the election, as otherwise the key could be distributed incorrectly such that voters would cast their votes but these could not be decrypted and thus tallying would not be possible. To our best knowledge, this protocol is currently the most suitable for practical use on smartphones. Note, that Gennaro et. al show in [5] that the keys generated with Pedersen's protocol [6] are not random under

[1]As of 31.10.2012, only 17,5M of population in Germany possessed the eID card: http://http://www.personalausweisportal.de/SharedDocs/Pressemitteilungen/DE/2012/Wachsendes-Interesse-am-neuen-Personalausweis.html?nn=3043614.

[2]The El-Gamal parameters are 2048 bits for the generator and 224 bits for the multiplicative group order due to the standard recommendation in [15].

[3]This channel is needed at certain points for detecting misbehavior of the entities involved in protocol.

[4]This is mentioned, e.g., in http://www.technologyreview.com/news/427787/are-smart-phones-spreading-faster-than-any-technology-in-human-history.

certain circumstances. In the same paper they propose their own distributed verifiable secret sharing protocol, which solves the outlined problem, while being less efficient. However, Cortier et. al recently proved [8] that Pedersen's protocol can be used securely in the context of distributing El-Gamal keys.

For *verifiable distributed decryption*, we implement the protocol described by Cramer et al [22]. The protocol builds upon Pedersen's protocol adaptation for distributively generating an El-Gamal key pair, uses the private key shares to distributively decrypt the El-Gamal ciphertexts, and generates zero-knowledge proofs for the correctness of the decryption[5].

Both protocols rely on secure communication channels between authorities. To ensure *secure communication channels*, we decided to implement the Diffie-Hellman key agreement [23] for the authorities to exchange secret AES keys with each other. The AES keys have the length of 128 bits and are used for encrypting further communications in counter mode according to [24]. In addition, the RSA signatures are used for authentication purposes, in the Diffie-Hellman exchange, as well as in further communications. According to the recommendations in [15], the RSA keys have a bit length of 2048. As there is no established PKI, the *RSA key pairs need first to be generated and distributed*. Since we assume that there is an out-of-band communication channel between the authorities, they are able to use it to *verify* whether they received the proper public keys, i.e. there was no attacker running a man-in-the middle attack. For the RSA key distribution and verification, we implement a protocol based upon short authentication strings, described in [25] as symmetrised group protocol. A protocol using the same approach (albeit designed for agreement between two parties instead of a group) has been formally proven secure in [26]. A similar approach has also been implemented in [27] for their application. While the implementation in [27] uses SHA-1 as second pre-image resistant hash function, we rely on SHA-256, as recommended in [15].

The short passphrases, that are to be compared, are the 24-bit strings. For displaying the passphrases, the PGP Word List [28] is used, substituting each byte of the hash value with a specific word. Thus, instead of comparing the 24-bit strings, the users have to compare the equality of a passphrase that consists of three words, which improves usability greatly. As every election authority is expected to be aware that it is important to follow the procedures, we assume that they attentively perform the verification of passphrases.

Both the distributed key generation and the verifiable distributed decryption protocol need one entity to start the protocols and to invite the others to join. Therefore, it was decided to have a *head* of the group of election authorities who has these responsibilities. As this one can easily be replaced by another one (necessary due to the threshold) and does not have more power with respect to violating any security properties, we decided to leave the selection to the election authorities.

[5]The implementations of both of these protocols were proposed by [7], [8], but neither of them is usable for the setting we have determined.

C. Web Bulletin Board Involvement

At several stages in the setup and tallying, the WBB, as the central server of the Internet voting system, is involved. These stages are indicated and explained in the following: During the setup phase, the present, invited election authorities are announced on the WBB and the head labeled accordingly. This has two reasons: first it is more transparent who are the election authorities, and second it is only necessary to type this information once because otherwise each election authority has to enter this information manually in their smartphones upon starting the application. This improves the usability, while it has no effect on the security. Furthermore, all election authorities send the public key computed during the distributed key generation protocol to the WBB. This is necessary to load it into the main voting application and as such this key is available for the voter to encrypt her vote. In order to minimize the trust in the WBB, we decided to include in the setup phase a verification step; i.e. the election authorities verify the election key[6]. In order to enable the election authority in conducting this verification, the hash value of the public key is displayed by the smartphone and also accessible on the WBB[7]. In the tallying phase, the smartphone application downloads the list of encrypted and anonymized votes from the WBB. After the verifiable distributed decryption, the smartphone application of the head sends the decrypted votes, the results and the corresponding zero-knowledge proofs of the correctness of decryption to the WBB. Note, due to the proofs it is not necessary that everyone sends this information but the WBB as well es external observers can verify these proofs.

D. Communication Protocol

The different instances of the application have to communicate with each other in order to execute the before mentioned cryptographic protocols. There are in general several options to implement the communication between the smartphones such as Bluetooth, SMS, WiFi-Direct or (proprietary) instant message protocols like ICQ, MSN or Skype. We have to transfer large amounts of data in particular due to the number of voters and to enable communication regardless of the geographical location of the election authorities. Therefore, we decided to rely on one of the instant message protocols. We decided to use the open-source XMPP protocol for its flexibility, i.e. the possibility to add our own message types. In order to communicate with each other over XMPP, Jabber IDs are used[8].

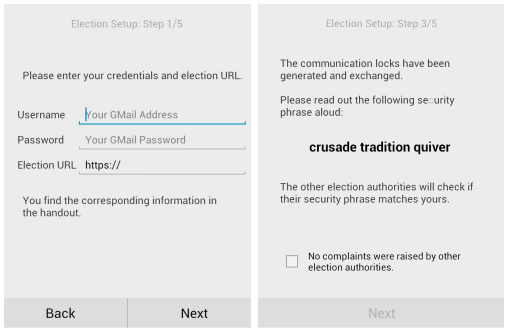
IV. Election procedure

Below we describe a walkthrough for the election setup and for the tallying phase with the developed application. The

[6]As mentioned above, it is assumed that the authorities would follow the procedures and verify the election key.

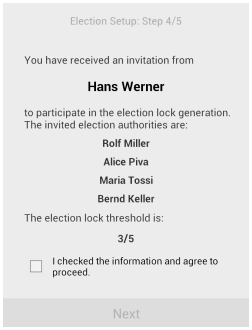
[7]Note, that due to the length of a hash value of a public key, for the sake of usability we chose not to output it as a passphrase of PGP words as used in the distributed key generation protocol. Instead we display the hash value as a hexadecimal string, truncated to 160 bits (40 characters).

[8]In our implementation we decided to use Gmail accounts as many people nowadays have such an account and to our best knowledge, it is currently the most widespread type of Jabber IDs. However, one could easily use accounts on any other XMPP server, including the private ones, that are established specifically for the purposes of the election.

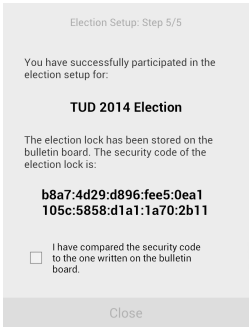


(a) Login.

(b) Verification of communication keys.



(c) Accepting invitation to participate in distributed key generation.



(d) Verification of election key.

Fig. 1. Distributed key generation on the smartphone application.

interfaces have been developed in an iterative process involving technical and non-technical potential election authorities, furthering the *goal of making the application usable*.

A. Setup Phase

Stage 1 - Tutorial: The election authorities get a tutorial on the functionality and security of the application and the Internet voting system (see Section VI).

Stage 2 - Announcing members and head: The election authorities announce their names and Jabber IDs. They also announce the head (after having discussed who should take the duties of the head). The administration staff of the WBB announces this information via the WBB.

Stage 3 - Installing the application: The election authorities download the application from the trusted public institution (TPI) they trust via secure HTTPS connection, or from the Google Play Store, if this institution is registered as a certified developer, and install it on their smartphone.

Stage 4 - Starting the application: The election authorities start the application and enter their Jabber credentials[9], and a WBB web address provided to them by the election organizers (see Figure 1(a)).

Stage 5 - Communication key exchange: After a successful login into the application (i.e. the election URL and the credentials are correct), the application displays the name of the election, as well as the name of the head of the election authorities as announced on the WBB. The election authorities have to confirm that they agree to participate in this election with the named person being the head of election authorities[10]. After the confirmation, a random RSA key pair is generated by each of the applications. The public keys are exchanged with all other authorities. At the end of this stage, each smartphone displays a passphrase (according to the protocol mentioned in Section III-B) (see Figure 1(b)). The instructions on the screen for the head are to read aloud this passphrase[11]. The instructions for the others are to compare and only continue if all have the same passphrase. In case the verification was successful, the authorities may continue to the next step.

Stage 6 - Distributed (election) key generation: In this stage, the head of the commission initiates the distributed key generation protocol. This is done by inviting the other election authorities (by selecting them from the displayed list). At the same time, the other authorities wait for receiving an invitation. Once they have received the invitation, the list of invited authorities, as well as the name and threshold value for the election is displayed (see Figure 1(c)). Then, they are asked whether they agree on this list and on the threshold. This is necessary to avoid the head inviting other people then the ones he is supposed to invite, or setting the wrong threshold value. After all election authorities accepted the invitation (as no misbehavior was detected), the distributed key generation protocol is executed. After its completion, each authority's smartphone sends a generated public key to the WBB.

Stage 7 - Verifying (election) key generation: The smartphone displays the hash value[12] of the public key it computed during the distributed key generation protocol. The instructions for all election authorities are to compare the displayed value with the one displayed on the WBB. If the value matches and all authorities confirm the matching, this key is loaded into the voting application to be used to encrypt votes[13]. The election authorities may now close the application.

B. Tallying Phase

Stage 1 - Starting the application and running the distributed verifiable decryption: The election authorities start the application and all anonymized votes are downloaded from the WBB. The head initiates the decryption by inviting all other election authorities to participate. At the same time the other election authorities wait for this invitation. Once all election authorities accepted the invitation[14], the decryption is started. The result of its successful completion is a list of decrypted votes, the number of votes per candidate together with corresponding proofs. The result is sent to the WBB from the head. For increasing robustness, the authorities are then

[9]These are stored in the application and are only used for XMPP login.

[10]This is done to confirm that the person announced as the head by the WBB is indeed the person the election authorities agreed on being the head.

[11]In the application the passphrase is called security phrase as pre-studies show that people do not know what passphrase means here.

[12]In the application this hash value is called security code for the purpose of understandability.

[13]The vote casting process is out of scope of our project.

[14]It is necessary to have the election authorities being involved because otherwise the decryption could be started before the end of the election.

encouraged to create a backup of the data generated by the application by exporting the data to external storage.

Stage 2 - Announcing the results: The result is displayed on the WBB but also by each application of the election authorities. The election authorities compare the result displayed on their smartphones with the one displayed by the WBB. They then close the application and in case they want to get access to the result again they just open the application again.

V. Security Properties and Security Model

In order to check whether the authorities understand the security properties and security model of the application, it is important to be very precise about both. The identified properties relevant for the Internet voting system – namely robustness, secrecy, and integrity – and the corresponding assumptions are determined and discussed in this section. For the analysis we consider each of the entities involved with the proposed application (each election authority and the WBB) as well as outsider attackers (reading, deleting, modifying messages that are interchanged between the involved entities) as well as combinations being able to violate each of these three properties. We first consider each election authority as one unit including the person, the smartphone, the operating system, and the application. We later discuss the different parts and their impact on the security model.

Robustness means that it is possible to decrypt the anonymized votes from the WBB. This is ensured if at least a threshold amount of election authorities is available and behaves correctly during the distributed decryption protocol. In addition, it needs to be assumed that a communication network with enough throughput is available, and we rely on the used cryptographic primitives in place.

Secrecy means that it should be impossible to decrypt the personalized, encrypted votes. This is ensured if at least a threshold amount of election authorities is honest and behaves correctly in any stage. This includes that they check the passphrase and that the hash value of the election key displayed on their smartphone and the one displayed on of the WBB match. In addition, secrecy can currently only be ensured if the WBB is trustworthy (if not, the WBB could send the personalized votes and would obtain the corresponding decrypted ones). From the voting system's components we need to trust that at least one mix node is honest, and, finally, we rely on the cryptographic primitives in place.

Integrity means that it should be detected if the decrypted and anonymized votes do not match. This is ensured if at least a threshold amount of election authorities is honest and behaves correctly at any stage. This includes that they check that the proper result is published on the WBB. In addition, integrity can currently only be ensured if the WBB is trustworthy. Finally, we rely on the used cryptographic primitives.

Different 'Parts' of the Election Authority. So far we only considered each election authority as a unit. However, actually it is not necessarily the person herself who might be dishonest and for instance not join the decryption phase or export the key share; it might also be the application which is not implemented by the election authority due to the necessary lack of technical knowledge; thus the downloaded application could be malicious in arbitrary ways. Therefore, it is recommended to have different trustworthy institutions programing their own application. Then each election authority downloads the application from a different institution. With this, the above security model also holds on the application level. The same problems also exist on the operating system and the smartphone manufacturer level. However, here it is not easy to find a solution matching the above mentioned security model. As the election authorities are supposed to use their own smartphones, we rely on whatever smartphones and operating systems they have. Thus, the security model on these levels actually depends on the concrete setting of an election, i.e. which smartphones and which operating systems are used by the election authorities.

Trustworthiness of the WBB. As seen, the current implementation builds upon the trustworthiness of the WBB. We assume, however, that the security of the WBB is taken care of outside of our application in real-world elections, especially the high-stake ones. As such, for ensuring integrity and secrecy of the votes, the PKI established among the voters and the mix nodes can be used. In such a case, for example, for ensuring secrecy the election authorities could be directed to decrypt the votes only if these votes are signed by a threshold amount of mix nodes. Furthermore, the WBB can be constantly supervised by a trusted third party in order to ensure that it does not arbitrarily change its content, thereby ensuring integrity.

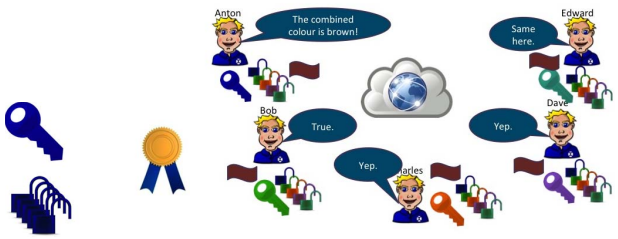
VI. Education material

Transparency is an important property of elections as for instance the German constitutional court stated referring to the use of voting machines in Germany [29]. In recent elections, most constituencies provide education material for poll workers, such as for instance the state of Berlin [30]. This material proves to be particularly crucial for Internet elections, since the election authorities in Internet elections usually have a high education, but are not specialists in computer science or information security [31]. We therefore prepared, in addition to the application itself, a training for election authorities. This training aimed for communicating the motivation for this application and a basic understanding of the application's objectives and security properties (including the underlying security model). We use a metaphoric approach like many others in security education literature, e.g. [32]–[34]. The challenge is to select appropriate metaphors, i.e. the election authorities build a mental model on how the application works which allows them to deduce the security properties and the security model properly.

In the following, we first introduce the metaphors used for the different cryptographic primitives and protocols. Then, we propose the structure of the education tutorial.

A. Metaphors for Cryptographic Primitives

Asymmetric encryption: For the concept of asymmetric encryption, we followed a well-established approach, namely the metaphor of padlocks and the corresponding key. This concept has for instance been used by Keller et al. [35]. As



(a) Locks, keys and seals.

(b) Verification of communication keys.

(c) Partially opening encrypted votes.

Fig. 2. Metaphors used for cryptography in the education material.

locks can only be used to protect one object (e.g. a letter inside an envelope), we introduced the concept of producing several locks for one key. Furthermore, key pairs can be identified by equal colours of keys and locks.

Digital signature: We use similar to many other approaches a seal imprint in the same color as the lock. The idea is that the seal imprint in one color can only be produced by the person owning the key in exactly that same color (see Figure 2(a)). This essentially corresponds to the concept of seals in the physical world. Seal imprints can only be produced by the owner of the corresponding seal (key).

B. Metaphors for Cryptographic Protocols

Voting protocol: As metaphor for encrypting and signing votes, we decided to follow the two envelope approach known and experienced from postal voting. The inner layer is a neutral envelope, identical for all voters, which is closed with the election lock. The outer layer is bound to the voter's identity. This metaphor is also used in Estonia to explain their Internet voting system [36]. The mix-net is represented by a box in which all the votes are put, closed and then shuffled. Again this is very similar to the recent tallying process. Here poll workers are supposed to first shuffle the ballot box and then open it. Eventually, the web bulletin board (WBB), which is used to announce any public data, is a metaphor in itself and as such can be directly represented by bulletin boards known from the physical world.

Verification of communication keys: The exchanged public keys must be validated by the authorities by comparing a hash value (of all other public keys). In order to avoid explaining second pre-image resistance of cryptographic hash functions, we use the following metaphor: First, authorities exchange colored public locks. Second, after all colored locks have been exchanged, the mix color of all lock colors is obtained. The mix color represents the outcome of the second pre-image resistant hash function applied to the concatenation of all public keys (see Figure 2(b)). This metaphoric approach has been used to communicate the concept of one-way functions used to teach Diffie-Hellman secret sharing [37]–[39]. Due to the conceptual similarity of one-way functions and second pre-image resistant hash functions, we consider the color metaphor to be a proper approach.

Cryptographic key sharing: Many works, see for instance [40], rely on the idea of splitting keys physically apart as metaphor for cryptographic key sharing. However, this concept does not work for schemes being robust against single entities being malicious. Therefore, we introduced the concept of each share being part of a description used to build the key. As these descriptions are not disjoint it is possible to reconstruct the key based on a subset of all shares. As it is necessary to build the key, we also explain that a computer program is necessary to be involved and the lock is stored as key information in a file.

Distributed decryption: One particular challenge of our approach, as opposed to the Estonian Internet voting system, has been the explanation of distributed decryption. We discussed several approaches and decided eventually that each election authority holding a key reconstruction description, opens the locked envelope partially on a specific position. Positions of different authorities overlap each other in order to communicate robustness of the protocol. If an envelope is partially opened by a sufficient (threshold) amount of election authorities, the envelope can be opened on the WBB and the election result can be computed (see Figure 2(c)).

C. Structure and Content of the Tutorial

The presentation is divided into the voting protocol and the main part, namely the distributed key generation and decryption. The voting protocol is explained in the following way: (a) We first outline the voter experience as follows: The WBB provides empty ballots and election locks which can be picked by voters. After a voter picked those items, she fills her ballot (resulting in a vote), puts the filled vote in an envelope and locks that envelope with the election lock. Afterwards, the voter puts the envelope into a second, personalized envelope. The voter sends this envelope to the WBB. Here, all other voters' envelopes are stored as well. (b) Then we continue with tallying: After the election has been finished, the outer envelopes are checked against the electoral register such that only eligible votes are included in the election result. Thereafter, the remaining envelopes are opened and the inner envelopes are taken out. These inner envelopes are shuffled with the mix-net, i.e. the shuffling box. Eventually, the anonymized, locked envelopes are opened with the election key and the election result can be computed.

For the main part, we decided not to immediately explain the different stages of the application but deduce the solution from identified problems step by step. The motivation behind this idea is that people better understand why the application is needed at all and why certain manual checks are necessary. Correspondingly, we start with the situation in which the election key is held by one single election authority. We demonstrate that this single authority can break secrecy in case it behaves maliciously (*problem 1*): Rather than decrypting anonymized votes, the malicious authority could decrypt personalized votes from the WBB.

As *solution* to this problem, it is shown that an election key can physically be split apart. The pieces of the election key can then be distributed to a set of election authorities such that only the collaboration of all authorities allows reconstructing the election key and tallying the election. In this case, it is then shown that a malicious election authority cannot break secrecy anymore. However, it is demonstrated that a malicious (or simply failing) election authority who does not participate in key reconstruction prevents the honest election authorities from tallying the election (*problem 2*).

As *solution* to this second problem, we introduce the notion of key shares being information to reconstruct a key such that any subset of a specified number (threshold) of election authorities have enough information to reconstruct the key. As such we move from the physical to the digital world. The education material shows that the previous problem is solved. However, it proceeds with the danger that arises from a single key distribution or key reconstruction computer. If one of these computers behaves maliciously, it might store the election key in its entirety and use it to open personalized votes, thereby violating secrecy (*problem 3*).

The *solution* to this problem is refraining from the computer in the construction and reconstruction phases, which has possession of the entire election key. To that end, rather than having any centralized key component, the entire key generation and tallying processes are conducted in a decentralized manner exchanging a couple of messages over the Internet (while so far the key shares were distributed in person). It is explained that this approach solves the previous problem. However, we then mention the problem of untrusted channels (such as the Internet) used to exchange messages. An adversary might read the exchanged messages and obtain the knowledge of each individual election authority. We explain how an adversary gets sufficiently many key reconstruction descriptions in order to obtain the entire election key, thereby breaking secrecy (*problem 4*).

As a *solution* to this problem, the establishment of secure channels between participants is explained which should then be used to securely interchange the messages. Therefore, each individual election authority generates her own personal key and the corresponding locks. Each election authority sends out her locks to all other election authorities. Eventually, each election authority can use another authority's lock to send confidential messages to that authority. However, the adversary is also able to change messages sent on the network. As such the adversary might replace locks sent out from one election authority by different (own) locks. If the adversary replaces all authorities' locks accordingly, he will obtain all authorities' knowledge and still violate secrecy as described above (*problem 5*). As a *solution*, we explain that each voter combines her individual generated lock with all other obtained locks in terms of the different colours.

The *solution* to problem 3 outlined above refrains from the reconstruction machine in the tallying phase. Correspondingly, it is demonstrated how tallying can be conducted without reconstructing the election key by applying the metaphor of partially opening envelope. Then it is explained that an attacker could replace single partially opened envelopes when sent from one authority to another or to the WBB (*problem 6*). As a *solution* seals were introduced. Thus, actually, after an authority partially opened the envelope, she puts her own seal on the envelope indicating that this has been opened by her. Eventually, it was outlined that one application can be provided as explained in the last solution. If, however, this application is implemented from one single institute, the institute might secretly introduce backdoors, thereby endangering secrecy of integrity (*problem 7*). Therefore, as final improvement we outlined that the application is therefore implemented by different TPIs from which it can be downloaded.

VII. Evaluation

The goal of the user evaluation was two-fold: The first goal was to evaluate the usability of the implemented application, i.e. that users are able to run all the protocol steps and conduct the checks they are asked to execute. The second goal was to evaluate the understandability, i.e. whether the used education material enabled the participants to understand the application in terms of its objectives and the underlying security model.

A. Evaluation Setup

The evaluation of the application has been conducted on five Samsung Galaxy S3 devices due to their widespread use. Based upon previous studies, five in Germany trusted public institutions were chosen, namely the Federal Office for Information Security, the Federal Constitutional Court, the Federal Electoral Management Body, the Organization for Security and Co-operation in Europe (OSCE), and the Federal Statistical Office. For the test election, a WBB has been set up on which relevant data of the test election has been published and shown to the participants. The test election included 208 test votes on the WBB in order to diminish the pure decryption time from 20 minutes to 5 minutes. We generated Gmail accounts for each participant throughout the evaluation.

B. Participants

We decided to conduct the evaluation separately for groups of five participants. We recruited the participants via emails sent out around the Technische Universität Darmstadt. These shall be persons with non-expert knowledge in computer science and information security in order to reliably represent election authorities of Internet elections.

C. Questionnaire

For the usability evaluation, we use the standard usability criteria defined in [41], namely effectiveness, efficiency, and satisfaction, as a basis. We measured the effectiveness of the application in terms of the fact whether participants achieved their goal of generating an election key in a distributed way correctly and of distributively tallying the election and by the number of questions raised to achieve their goal. Efficiency has been determined by the time it took the participants to finish their task. Satisfaction has been evaluated with the system usability scale (SUS) [42], which is a ten-item standard questionnaire. The questions regarding understandability were tested in a pre-evaluation and improved accordingly. Questions of the final questionnaire were as follows: The first question

regarded secrecy of the vote, the second one - the robustness of the system. The participants were asked to check all the options that would be able to violate secrecy or robustness respectively. The questionnaire's options for answering these questions were presented in a way that several responses should immediately follow from the educational part, while other responses required the transfer of learned concepts. We then evaluated the number of correct responses for each questionnaire. After each participant filled and handed in her individual questionnaire, the participants were asked to discuss their own understanding among themselves and fill *one* questionnaire together afterwards. This second questionnaire was motivated by the observation other fields or research, e.g., genetics [43] and biochemistry [44]. These works provide evidence that group discussions foster students' understanding of learned concepts.

D. Evaluation Process

After an informal welcome, the participants – who played the role of election authorities – were asked whether they agree to have their voices recorded. All of them accepted. All participants had to pass through four sequential phases.

First Phase. The first phase, the education phase, consisted of three parts. In the first part, participants were made familiar with the motivation and the goal of the evaluation process. The participants were told the cover story that the Technische Universität Darmstadt plans the implementation of Internet voting for the next University election. Then participants were asked to shortly introduce themselves. In the second part, the general concept of the Internet voting scheme as outlined in section II was introduced. Finally, the participants were asked whether there were questions. The third part was the presentation of the education material in terms of a tutorial, i.e. a presentation by the instructor. During the presentation, the instructor made 5 breaks for questions. Afterwards, participants were provided with their credentials in closed envelopes and a one-page guideline containing the names of the election authorities, parties from which they were delegated, the election threshold, and the election website.

Second Phase. In the second phase, participants used the developed application for a *test election scenario*. Participants were first introduced into the process and their duties. After entering their credentials and the election website, participants started the process. During its first execution, the application crashed during the initialization of the key generation protocol. One of the supervisors supported the participants to restart the application. After the restart the application continued in the step right after the verification of communication keys. After having generated the key shares in a distributed way, participants were told that for this evaluation the vote casting phase is skipped, and they already are in the next election phase, namely the tallying phase. Participants started the application again, used the application to distributively decrypt the test cast votes, and compared the results on their smartphone with the election result on the WBB. During the decryption phase participants were told that it might take some time, because of expensive mathematical operations executed for decrypting cast votes. Furthermore, participants were explained why they can trust the results shown on the WBB, as the general public can verify the correctness proofs of the decryption phase that are also posted on the WBB.

Third Phase. In the third phase, participants filled *questionnaires* regarding the understandability and usability of the application, covered in the first and the second phase. Furthermore, demographic data were collected. The demographic data were collected separately from the questionnaire, in order to ensure anonymity.

Fourth Phase. In the fourth and last phase, participants were *debriefed*, indicating that the application will not be used for the next University elections, but maybe in the near future this might be considered. Finally, the supervisors thanked the participants and compensated them with 16 Euro each.

VIII. Results

This section summarizes the findings of the evaluation with the first five participants. These were 1 female and 4 male students aged between 21 and 26. Two of them were Albanians, one was German, one Kenyan, and one Indian. The participants studied mechanical engineering, physics, architecture, electrical engineering, and materials science.

A. Usability

Effectiveness. In order to achieve their goal of generating the election key and tallying the election correctly, the participants had to solve four sub-goals in the setup phase, namely accepting to participate in the distributed key generation with the announced head, validating the exchanged communication keys, agreeing on the participating election authorities and the threshold, and verifying the election key. In the tallying phase, the participants had to solve two sub-goals, namely accepting the initiation of the tallying process and comparing the announced election result to the locally computed election result. Only if all six sub-goals are achieved, the overall goal is achieved. The observation of participants and the audio-record revealed that all six sub-goals were achieved such that the overall goal has been achieved by the participants.

Shortly after all participants entered their login data and the election website into the application, the application crashed due to network problems with one smartphone. The organizers had to intervene and restart the application in order to dissolve confusion among the participants. After restarting the app, the head of the election authorities was required to read out the security phrase aloud which was not clear to her such that a further question was asked to the organizers. Eventually, in the tallying phase, after the head of the election authorities initiated the tallying process and all other authorities were invited to participate in the tallying, one participant asked if she should join the tallying. After the tallying has been conducted in distributed manner, the result was shown on each individual smartphone and on the public WBB. One participant raised the question whether the results should be exported or if the application can be closed. The results show that the participants achieved their task of generating the key in a distributed way and tallying an election without major problems. Nevertheless, the questions raised by the participants indicate that both the communication key validation and election result verification processes must be made more clear within the application.

Efficiency. In order to achieve their goal of setting up an election key and tallying an election afterwards, the participants in collaboration needed 32 minutes. Given 824 postal votes within an average German district, the time needed to generate the election key and tally the election would increase to 47 minutes. Given the fact that complex ballots might be easily encoded within a single El-Gamal ciphertext, this time indicates significant efficiency improvements.

Satisfaction. The SUS questionnaire has been slightly adapted to our smartphone application. To prevent confusion, the term *system* has been replaced by the term *application*. Thereby, the participants were clearly pointed to the smartphone application rather than to the overarching Internet voting system. Scoring of the questionnaires resulted in an average value of 82.5. According to Sauro's normalization method [45], a value above 80.3 results in an A grading and as such the application is more likely to be recommended.

B. Understandability

Regarding secrecy, correct responses that should immediately follow from the education were *The institution which provided the mobile election authority application on the smartphone I used*, *An outsider controlling the wireless network*, *Intermediate Internet servers*, *Your research group*, *Internet Provider*, *Smartphone manufacturer*, and *The Shuffling component*. The matching of the mental security model of individual participants with the real security model was on average 56%. Interestingly, the later discussion directed the mental model of the group in the right direction such that a matching of 88% was obtained. Including transfer responses into the immediate responses results in an average individual matching of 50% and a group (after discussion) matching of 70%. With respect to robustness, the immediate responses slightly differ from immediate responses regarding secrecy[15]. The immediate responses for robustness are: *The smartphone I used*, *The institution which provided the mobile election authority application on the smartphone I used*, *Your research group*, *four*, *three*, *two*, and *one other election authority*, *If I collaborate with all others election authorities*, *Any voter*, *The Shuffling component*, *All voters together*. The average rate of correct responses, which reflects the matching of individual mental models with the real security model equals 71%, while the later group discussion resulted in a group matching of 100%. Including the transfer responses, the individual mental model matching is 58% and 78%.

As seen, the prepared education material had not the expected effect on the participants' understanding of the security model. Therefore, we decided not to continue the evaluation with more participants, but rather to revise education material before re-running the evaluation.

[15]This stems for instance from the fact that with respect to secrecy, we considered that election authorities might be corrupt and hire a information security expert to obtain election key reconstruction data from the smartphone, while the participants considered only the election authority's personal skills. On the other hand, with respect to robustness, even a non-expert authority might decide to destroy her smartphone thereby making her key reconstruction description unavailable.

IX. Conclusion and future work

The present work pursued two goals. The first goal was to develop a usable application for the tasks of election authorities that consists of distributed key generation and verifiable distributed decryption, thus furthering the implementation of electronic voting protocols that make use of these concepts. The second goal was to communicate the security model used in electronic voting protocols (with particular focus on our application) to laymen.

We developed education material using metaphors to explain the cryptographic concepts behind our application. The initiated evaluation has shown that the security model of the application has not been understood to a satisfactory extent on an individual level. Therefore, we decided not to continue the user evaluation, but rather revise our education material before running a complete user evaluation. Interestingly, we found that group discussion among the participants fostered their understanding such that the secrecy model for immediate responses was understood to 88% and the robustness model to 100%.

The developed application is promising as the initiated evaluation shows. The participants of our evaluation who had no prior knowledge of information security, despite the issues that occurred during the key generation stage, found the application easy to use, giving it an average SUS score of 82.5, which is well above the global average [42], [45]. While the number of participants in insufficient in drawing reliable statistical conclusions regarding the application usability, the findings, however, indicate a trend towards finding the application usable.

For the future, we intend to integrate the essential parts of the group discussion into our education material and run a complete user evaluation. Second, we intend to consider the impact of adversarial behavior on the application's usability. From a technical perspective, we are currently integrating authentication measures into the application in order to protect sensitive election information such as secret key shares. We are furthermore integrating the application into the Helios voting system [46] which has been extensively studied in the literature, see for instance [47]. The conducted research represents a further step towards secure and transparent electronic elections. We are convinced that prior to deploying electronic voting for high-stake elections, election authorities and the public need to understand the overall security of the voting system as well as the remaining risks to gain trust in the election and its result.

Acknowledgment

This paper has been developed within the project 'BoRoVo' Board Room Voting - which is funded by the German Federal Ministry of Education and Research (BMBF) under grant no. 01IS12054. The authors assume responsibility for the content. The authors would like to thank Jurlind Budurushi for helping with conducting user studies.

References

[1] Estonian National Electoral Committee, "E-Voting System General Overview," 2010. [Online]. Available: http://www.vvk.ee/public/dok/General_Description_E-Voting_2010.pdf

[2] O. Spycher, M. Volkamer, and R. Koenig, "Transparency and technical measures to establish trust in norwegian internet voting," in *E-Voting and Identity*. Springer, 2012, pp. 19–35.

[3] B. Adida, O. De Marneffe, O. Pereira, and J.-J. Quisquater, "Electing a university president using open-audit voting: analysis of real-world use of helios," in *Proceedings of the 2009 conference on Electronic voting technology/workshop on trustworthy elections*. USENIX Association, 2009, pp. 10–10.

[4] "International Association for Cryptologic Research 2012 Election," http://www.iacr.org/elections/2012/, 2012.

[5] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Secure distributed key generation for discrete-log based cryptosystems," *Journal of Cryptology*, vol. 20, no. 1, pp. 51–83, 2007.

[6] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Advances in Cryptology – CRYPTO91*. Springer, 1992, pp. 129–140.

[7] A. M. Davis, D. Chmelev, and M. R. Clarkson, "Civitas: Implementation of a Threshold Cryptosystem," Tech. Rep., 2008.

[8] V. Cortier, D. Galindo, S. Glondu, and M. Izabachene, "A generic construction for voting correctness at minimum cost-application to helios." *IACR Cryptology ePrint Archive*, vol. 2013, p. 177, 2013.

[9] A. Prosser, R. Kofler, R. Krimmer, and M. K. Unger, "E-voting Wahltest zur Bundespräsidentschaftswahl 2004," *Arbeitspapiere des Instituts Informationswirtschaft*, no. 01/2004, 2004.

[10] A. Riera and G. Cervelló, "Experimentation on secure internet voting in spain," *Electronic Voting in Europe Technology, Law, Politics and Society*.

[11] Council of Europe, "Guidelines on transparency of e-enabled elections," 2011.

[12] M. Volkamer, O. Spycher, and E. Dubuis, "Measures to establish trust in internet voting." in *ICEGOV*, 2011.

[13] D. Bernhard, S. Neumann, and M. Volkamer, "Towards a practical cryptographic voting scheme based on malleable proofs," Cryptology ePrint Archive, Report 2013/276, 2013, http://eprint.iacr.org/.

[14] Electoral officer and electoral board. Accessed: 2014-03-19. [Online]. Available: {http://www.bundeswahlleiter.de/en/glossar/texte/Wahlvorsteher_und_Wahlvorstand.html}

[15] E. Barker and A. Roginsky, "Transitions: Recommendation for transitioning the use of cryptographic algorithms and key lengths," *NIST Special Publication*, vol. 800, p. 131A, 2011.

[16] B. Adida and D. Wikström, "How to shuffle in public," in *Theory of Cryptography*. Springer, 2007, pp. 555–574.

[17] asmack library sources. Accessed: 2014-03-19. [Online]. Available: https://github.com/Flowdalic/asmack

[18] Spoungecastle library sources. Accessed: 2014-03-19. [Online]. Available: http://rtyley.github.io/spongycastle/

[19] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[20] P. Feldman, "A practical scheme for non-interactive verifiable secret sharing," in *Foundations of Computer Science, 1987., 28th Annual Symposium on*. IEEE, 1987, pp. 427–438.

[21] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch, "Verifiable secret sharing and achieving simultaneity in the presence of faults," in *Foundations of Computer Science, 1985., 26th Annual Symposium on*. IEEE, 1985, pp. 383–395.

[22] R. Cramer, R. Gennaro, and B. Schoenmakers, "A secure and optimally efficient multi-authority election scheme," *European transactions on Telecommunications*, vol. 8, no. 5, pp. 481–490, 1997.

[23] W. Diffie and M. E. Hellman, "New directions in cryptography," *Information Theory, IEEE Transactions on*, vol. 22, no. 6, pp. 644–654, 1976.

[24] M. J. Dworkin, "Sp 800-38a 2001 edition. recommendation for block cipher modes of operation: Methods and techniques," Tech. Rep., 2001.

[25] L. H. Nguyen and A. Roscoe, "Efficient group authentication protocol based on human interaction," in *Proceedings of Workshop on Foundation of Computer Security and Automated Reasoning Protocol Security Analysis*, 2006, pp. 9–31.

[26] S. Laur and K. Nyberg, "Efficient mutual data authentication using manually authenticated strings," in *Cryptology and Network Security*. Springer, 2006, pp. 90–107.

[27] M. Farb, M. Burman, G. Chandok, J. McCune, and A. Perrig, "Safeslinger: An easy-to-use and secure approach for human trust establishment," Technical Report CMU-CyLab-11-021, Carnegie Mellon University, Tech. Rep., 2011.

[28] P. R. Zimmermann, "Pgpfone: Pretty good privacy phone owner's manual," MIT, http://web. mit. edu/network/pgpfone/manual, 1995.

[29] Federal Constitutional Court, "Decisions of the german federal constitutional court (BVerfGE) 123, 39, (75)," 2009.

[30] Landeswahlleiterin Berlin, "Bundestagswahl 2013: Schulungsvideo für Wahlvorstände," Website, 2013, available online at https://www.wahlen-berlin.de/wahlvideo/index.html; Accessed: 2014-03-19.

[31] Barbara Simons, "Verified Voting Blog: Report on the Estonian Internet Voting System," Website, 2011, available online at https://www.verifiedvoting.org/report-on-the-estonian-internet-voting-system-2/; Accessed: 2014-03-19.

[32] F. Raja, K. Hawkey, S. Hsu, K.-L. C. Wang, and K. Beznosov, "A brick wall, a locked door, and a bandit: a physical security metaphor for firewall warnings." in *SOUPS*. ACM, 2011, p. 1.

[33] G. Markowsky and L. Markowsky, "Using the castle metaphor to communicate basic concepts in cybersecurity education," in *Proceedings of the 2011 International Conference on Security and Management*. Springer, 2011, pp. 507–511.

[34] T. H. Karas, J. H. Moore, and L. K. Parrott, "Metaphors for cyber security," Sandia National Laboratories, Tech. Rep., 2008.

[35] L. Keller, D. Komm, G. Serafini, A. Sprock, and B. Steffen, "Teaching public-key cryptography in school." in *ISSEP*, ser. Lecture Notes in Computer Science, vol. 5941. Springer, 2010, pp. 112–123.

[36] E. Maaten, in *Electronic Voting in Europe*.

[37] G. Taylor, "Understanding public key cryptography with paint," Website, 2007, available online at http://maths.straylight.co.uk/archives/108; Accessed: 2014-03-19.

[38] S. Zdancewic, "Lecture series: Introduction to networks and security, lecture: Introduction to networks and security," Website, 2006, available online at http://www.cis.upenn.edu/~cse331/lectures/CSE331-26.pdf; Accessed: 2014-03-19.

[39] Art of the Problem, "Public Key Cryptography: Diffie-Hellman Key Exchange," Website, 2012, available online at http://www.youtube.com/watch?v=3QnD2c4Xovk; Accessed: 2014-03-19.

[40] A. M. Froomkin, "The metaphor is the key: cryptography, the clipper chip, and the constitution," *University of Pennsylvania Law Review*, vol. 143, no. 3, pp. 709–897, 1995.

[41] International Organization for Standardization, "ISO 9241-11: Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs): Part 11: Guidance on Usability," 1998.

[42] J. R. Lewis and J. Sauro, "The factor structure of the system usability scale," in *Human Centered Design*. Springer, 2009, pp. 94–103.

[43] M. K. Smith, W. B. Wood, K. Krauter, and J. K. Knight, "Combining peer discussion with instructor explanation increases student learning from in-class concept questions," *CBE - Life Sciences Education*, vol. 10, no. 1, pp. 55–63, March 2011. [Online]. Available: http://www.editlib.org/p/53870

[44] Z. Bobby, B. C. Koner, M. G. Sridhar, H. Nandeesha, P. Renuka, S. Setia, S. S. Kumaran, and S. Asmathulla, "Formulation of questions followed by small group discussion as a revision exercise at the end of a teaching module in biochemistry." *Biochem Mol Biol Educ*, vol. 35, no. 1, pp. 45–8, 2007.

[45] J. Sauro, "Measuring usability with the system usability scale (SUS)," http://www.measuringusability.com/sus.php, 2011.

[46] B. Adida, "Helios: Web-based open-audit voting." in *USENIX Security Symposium*. USENIX Association, 2008, pp. 335–348.

[47] F. Karayumak, M. M. Olembo, M. Kauer, and M. Volkamer, "Usability analysis of helios-an open source verifiable remote electronic voting system," in *Proceedings of the 2011 USENIX Electronic Voting Technology Workshop/Workshop on Trustworthy Elections. USENIX*, 2011.