Towards A Practical JCJ / Civitas Implementation

Stephan Neumann[1], Christian Feier[1], Melanie Volkamer[1], and Reto E. Koenig[2]

[1] Security, Usability, and Society
Technische Universität Darmstadt
Hochschulstraße 10
64289 Darmstadt, Germany
stephan.neumann@cased.de
feier@rbg.informatik.tu-darmstadt.de
melanie.volkamer@cased.de

[2]Research Institute for Security in the Information Society
Bern University of Applied Sciences
Quellgasse 21
CH-2501 Biel, Switzerland
reto.koenig@bfh.ch

January 7, 2014

Abstract

Internet voting continues to enjoy wide interest from both research and practice. Among the Internet voting schemes developed over the last decades, JCJ / Civitas stands out from the masses due to its innovative approach to resist voter coercion. To achieve its ambitious goal, the scheme builds upon particularly restrictive assumptions and an abstract credential handling rendering the scheme impractical for real-world use. At ARES 2012, Neumann and Volkamer presented a proposal which implements several of these assumptions (voter-side assumptions) and the credential handling by the use of smart cards. While addressing these practical shortcomings of JCJ / Civitas, their proposal did not take performance into account, and accordingly its performance has not been evaluated. In the present work, we revise the ARES proposal from a performance perspective in a security-invariant manner. Based on the herein proposed revisions, we are able to conclude that the revised ARES proposal is feasible to be used in real-world elections.

1 Introduction

Internet voting continues to be a topic of interest and many states started conducting political elections over the Internet. In order to be compliant with fundamental election principles, Internet voting systems must meet a diversity of security criteria. Among the most significant security criteria are vote secrecy and vote integrity. Since the early 80s, starting with Chaum's seminal work on Internet voting [1], many scientific proposals have been developed to address these criteria. One approach promising a particular form of security is JCJ [2] and its derivation Civitas [3]. JCJ / Civitas ensures secrecy even in case the voter interacts with the adversary during the vote casting process, i.e., during vote casting, the adversary coerces the voter into casting a specific vote or

A preliminary version of this work has been published in Lecture Notes in Informatics, Volume P-220, INFORMATIK 2013 – Informatik angepasst an Mensch, Organisation und Umwelt, Ed.: Matthias Horbach, ISBN: 978-3-88579-614-5

the voter intends to convince the adversary about the content of her vote in order to get benefits. Furthermore, the scheme on the one side mitigates the risk of violating integrity by means of voting material buying, i.e., adversaries are discouraged from buying voting material to vote multiple times thereby maliciously influencing the election result. On the other side the scheme prevents adversaries from forcing voters to abstain from the election, as the adversary does not have any mechanisms to control the voter's compliance. To ensure security under such circumstances, the scheme foresees that each voter casts her vote together with a credential validating or invalidating her vote, whereas the adversary is left uncertain about the validity of the credential and consequently about the validity of the cast vote. Even though the security enforcement is elegantly addressed, the scheme relies on a number of assumptions and an abstract credential handling rendering it impractical for the real-world use.

To overcome these drawbacks, Neumann and Volkamer presented a proposal [4] implementing several of these assumptions[1] and the abstract credential handling of JCJ / Civitas by the use of smart cards. Even though their proposal addresses practical problems of the JCJ / Civitas scheme, their work has not considered performance and consequently is not tailored towards performance, thereby leaving the community in doubt about its practical impact. In the remainder of this work, we refer to their proposal as the *NV12 proposal*, whereas the JCJ / Civitas scheme extended by the proposal is referred to as *NV12 scheme*.

The present work takes up the NV12 scheme. As a first contribution, the scheme is revised from a performance perspective in a security-invariant manner, later on referred to as *revised NV12 scheme*. Throughout this revision process, we determine which smart card routines of the NV12 scheme can be replaced, removed, or outsourced to improve the overall performance without affecting the underlying security model. As a second contribution, based on recent smart card timings, the overall performance of the revised NV12 scheme is analyzed. Given the findings, we are able to conclude that the revisions pave the way for the real-world use of the revised NV12 scheme.

The remainder of this work is structured as follows: In Section 2, we provide a brief overview of the NV12 scheme and outline the underlying security model. In Section 3 we specify the project setting within which this work has been developed. We furthermore revise the NV12 scheme by modifying smart card routines and argue why these modifications do not affect the underlying security model. In Section 4, the revised NV12 scheme is analyzed with respect to its performance on recent smart cards. Therefore, first, we assess timings of basic smart card operations, second, we decompose the revised NV12 scheme into its basic smart card operations and show that the revised NV12 scheme is feasible to be used in real-world elections. The work is concluded in Section 5 and directions for future research are given.

2 The JCJ / Civitas and the NV12 Scheme

The JCJ voting scheme [2], developed in 2005, has been the first Internet voting scheme satisfying the criteria of *coercion-resistance*, i.e., 1) secrecy of the vote is ensured even if the adversary interacts with (coerces) the voter during the vote casting process, 2) the adversary cannot force voters into forwarding their voting material, and 3) the adversary cannot force voters into abstaining from the election. At the same time, the scheme provides some kind of evidence in the integrity of the declared election result. Due to its particular security guarantees, JCJ has gained wide interest in the research community. In 2008, the scheme has been extended to the Civitas scheme [3]. Civitas slightly enhances the JCJ scheme from a theoretical point of view while the main focus lies on the instantiation of cryptographic components and the implementation of the JCJ scheme. As such, the Civitas implementation might build the basis for future real-world improvements on the JCJ / Civitas scheme. In the remainder of this work, we use the term JCJ / Civitas as integration of theoretical concepts of JCJ and practical deployments of Civitas.

[1] In their work, the authors focus on assumptions that require the voter active and benign behavior to meet the scheme's security criteria, so-called voter-side assumptions.

2.1 Previous Improvements of the JCJ / Civitas Scheme

To settle our own contribution, we provide a short overview on works addressing the JCJ / Civitas scheme. Several works addressed JCJ / Civitas' drawback with respect to its tallying complexity: Among these works, there are the contributions of Smith [5], Weber et al. [6], Araujo et al. [7], Spycher et al. [8]. In summary, the tallying process of the JCJ / Civitas scheme has been reduced from quadratic to linear complexity in the number of cast votes.

Haenni et al. [9] addressed the vulnerability of board flooding attacks in JCJ / Civitas. Due to the fact that the scheme relies on an anonymous channel to cast votes, anybody can cast arbitrary many votes on the bulletin board, thereby slowing down or even blocking the tallying process[2]. To prevent these kind of attacks, the authors propose to provide each eligible voter with her real credential and furthermore with a random but fixed number of so-called dummy credentials.

In [10], Backes et al. presented a formalization and security proof for JCJ in the applied Pi-calculus. Smyth et al. [11] adopted the approach of [10] to the Civitas scheme. Küsters and Truderung [12] propose a coercion-resistance definition, which differs slightly from the original. Based on that definition, they analyzed Civitas and discovered two coercion-resistance flaws. Correspondingly, they suggested improvements of the scheme. Shirazi et al. [13] identified a robustness vulnerability of the Civitas scheme and proposed improvements addressing this drawback.

Bursuc et al. [14] introduced the concept of trial credentials in order to improve the overall understandability of verifiability of the JCJ / Civitas scheme. Neumann and Volkamer [4] and Mendes [15] addressed the problem of credential management in JCJ / Civitas by the use of smart cards.

2.2 The NV12 Scheme Overview

The NV12 scheme builds upon the robustness extension by Shirazi et al. [13]. The NV12 scheme is motivated by the fact that the JCJ / Civitas scheme relies on a number of abstract assumptions and an abstract credential handling. Similar to the work by Mendes [15], Neumann and Volkamer address several of these assumptions and the credential handling by the use of smart cards. The NV12 scheme comprises the following entities: A *supervisor* who is in charge of running the election and declaring election authorities; the *voter* who intends to cast her vote; the *voter's smart card* that serves as trusted device between the voter and the JCJ / Civitas system; a *registrar* who administrates the electoral register; a *supervised registration authority* and a set of *registration tellers* that provide the voter with her credential; a set of *tabulation tellers* that are in change of the tallying process; a set of *ballot boxes* to which voters cast their votes; and a *bulletin board* that is used to publish information. In the remainder of this subsection, the NV12 scheme is summarized, while a minor improvement with respect to side channel attacks is integrated. Rather than assigning PIN codes to a credential share directly, from the PIN code and the credential share a first degree monomial is generated in the manner of [16].

Setup Phase. The supervisor sets up the election and publishes details about the ballot design. The registrar publishes the electoral register together with the voters' public keys. The tabulation tellers distributively generate the election key pair and publish the corresponding public key pk_{EK}. Each registration teller thereafter generates randomly chosen private credential shares for all eligible voters. They encrypt these private credential shares with the public election key resulting in *public credential shares* and publish these public credential shares next to the voter's entry within the published electoral register. More formally, for a specific voter registration teller i publishes $S_i = \{c_{RT_i}\}^r_{pk_{EK}}$, where c_{RT_i} is the voter's credential share.

Registration Phase. As opposed to the original JCJ / Civitas scheme, the NV12 scheme distinguishes between an offline and an online registration phase. In the *offline phase*, a voter v personally consults a so-called *supervised registration authority* (*SRA*). The authority checks that the voter is not under direct influence of any coercers. The voter is requested to insert her smart

[2]Note that this attack applies to both linear and quadratic complexity tallying approaches.

card[3] into the smart card reader. The voter is invited to set her voting PIN. Afterwards, the supervised registration authority stores the private credential share c^v_{SRA} generated for that voter on the voter's smart card. In JCJ / Civitas manner, along with the private credential share, the authority generates a *designated-verifier re-encryption proof* (DVRP) that convinces only this voter's smart card about the fact that the public credential share published on the bulletin board in the setup phase is a re-encryption of the private credential share sent to the voter[4]. The smartcard will then calculate the coefficient c_p used for mapping the PIN (p) to the private credential share (c^v_{SRA}) ($c_p = \frac{c^v_{SRA}}{p}$). These calculations are done over a finite field of prime order, whereas the chosen order is n-times ($n \geq 1$) as big as the order of the set the private credential is element of. The calculated coefficient is the only value the smart card will store within this phase. The voter leaves the supervised registration authority and the offline registration phase is finished. In the *online phase*, the voter remotely connects to the election website, which allows her to finalize the registration process. The voter is asked to chose her preferred registration tellers out of the set of available registration tellers[5]. The voter's selection of registration tellers is forwarded to her smart card upon which the smart card asks the voter to confirm her selection over her smart card reader. Thereafter, the IDs of trusted registration tellers are stored on her smart card. For voter v, the trusted registration tellers are denoted by *TRT(v)*. Afterwards, the card establishes secure connections to the trusted registration tellers via the client machine and obtains the private credential shares $c^v_{RT_i}$, an encryption $S'_i = \{c_{RT_i}\}^{r'}_{pk_{EK}}$ of c_{RT_i} together with the DVRPs from each individual teller proving that S_i and S'_i contain the same message. After the card obtained all private credential shares and verified the DVRPs, voter v's card computes and stores v's credential factor as

$$c_{factor} = c_p \times \prod_{i \in TRT(v)} c_{RT_i}$$

Voting Phase. Once, the voter finalized the registration, she can start the voting process. Therefore, she visits the election website upon which a JavaScript is loaded. The voter can make her selection within the JavaScript. After the voter finalized her selection, the selection is forwarded to her smart card which randomly encrypts the voter's selection. In cut-and-choose manner (NV12 implements this with the Benaloh challenge [17]), the voter can audit the correctness of the running JavaScript, i.e., the voter can verify that the JavaScript forwarded really the voter's selection. After the voter is convinced about the correctness, she is asked to confirm her choice by inserting her voting PIN on the smart card reader. The card then uses this PIN and multiplies it with the credential factor calculated and stored during registration phase. Please note, that this calculation again is done within the finite field of prime order described above. If the voter has entered the real PIN, it will result in the voter's real credential which then is associated to her vote, otherwise the resulting random (invalid) credential will be used. Hence, the credential in use is calculated as follows

$$c = PIN \times c_{factor}$$

This allows a proper PIN handling without being prone to side-channel attacks. Formally, the voter's smart card generates a ballot of the form

$$\langle \{c\}_{pk_{EK}}, \{vote\}_{pk_{EK}}, \sigma, \phi \rangle.$$

[3]This might be a special-purpose smart card or an electronic ID card, which stores the voter's private key, the registration tellers' public keys, and the smart card algorithm outlined in the remainder of the section. Although not explicitly pointed out in the paper, the public election must be stored on the smart card. For the sake of robustness, we assume that each registration teller provides the public election key to the smart card to detect faulty behavior of individual registration tellers.

[4]The nature of this proof allows a coerced voter to replace her private credential share by a random number and forward this number to the coercer who is not able to tell real or a fake credential apart.

[5]According to [13], the voter must chose at least half of the available registration tellers.

The terms $\{c\}_{pk_{EK}}$ and $\{vote\}_{pk_{EK}}$ are a private credential and the voter's vote both encrypted with public election key. σ is a *proof of well-formedness* (PWF) which shows that the encrypted vote $\{vote\}_{pk_{EK}}$ contains a valid choice, while ϕ is a zero-knowledge proof (PKCV) which shows that the submitter knows both c and $vote$ in order to avoid replay attacks. The smart card computes the hash value $hash(\langle\{c\}_{pk_{EK}}, \{vote\}_{pk_{EK}}, \sigma, \phi\rangle)$ and outputs this on the smart card reader. Even though, the smart card implementation results in a more practical JCJ / Civitas implementation, it must be noticed that neither the adversary nor the voter obtain any integrity-assuring evidence after the PIN has been typed. Given the fact that human beings notoriously tend to mistype or forget PINs, passwords, etc. [18], the NV12 scheme bears new challenges from a practical point of view which have to be considered in the future. Thereafter, the smart card casts the prepared ballot anonymously to all available ballot boxes. Upon receipt, each ballot box computes the hash value of the obtained ballot and publishes this value on the bulletin board.

Tallying Phase. In the tallying phase, all tabulation tellers retrieve the ballots from all ballot boxes and the public credentials stored on the bulletin board. Zero-knowledge proofs are verified, duplicates (due to vote-updating) and unauthorized votes (due to the use of fake credentials) are eliminated. Finally, encrypted credentials of remaining ballots are discarded and the respective encrypted votes are distributively decrypted. Each step of the tabulation tellers is publicly verifiable based on a set of zero-knowledge proofs.

2.3 Security Model

This subsection is dedicated to the security model underlying the NV12 scheme. We use the secrecy and integrity definitions from Budurushi et al.'s work [19], while the forced-abstention resistance is inspired by the JCJ scheme [2]. For each criteria, we provide assumptions on which the respective criterion is built upon. While most of these assumptions trace back to the original NV12 scheme (assumption index of [4] is indicated in parentheses[6]), some assumptions are stated more precisely and their need is justified.

Secrecy. For each voter who casts a vote for an arbitrary candidate c, it holds that the adversary cannot get more evidence about the fact that the voter selected c or any other selection c' as he can get from the final tally. With respect to secrecy, the adversary is restricted as follows:

- Each voter trusts at least half of the remote registration tellers and the supervised registration authority. (TA1)
- The adversary is neither able to corrupt smart cards nor smart card readers. (TA2)
- The adversary is not able to corrupt more than k out of all n tabulation tellers. (TA6)
- The adversary cannot control the client machine.

  Justification: Generally, a voter has the chance to prepare several ballots over her client machine (even though, the adversary does not know which intention is associated to her real credential. Unless the voter does not prepare a ballot for all possible intentions, the adversary knows which intention has not been cast by a specific voter which consequently violates secrecy.

It should be noted that adversarial capabilities not listed here must be countered by the voting system. For instance, the voting system should maintain secrecy even in the case the adversary coerces the voter into preparing a specific ballot thereby proving the content of her cast vote. This criterion is usually referred to as receipt-freeness enriched by the exclusion of randomization attacks as part of Juels et al.'s [2] coercion-resistance definition.

Integrity. The aggregation of all eligible voters' intentions matches the declared election result. Following the definitions of Budurushi et al. [19], integrity is composed of the sub-criteria encoded-as-intended, cast-as-encoded, stored-as-cast, eligibility, and democracy integrity. With respect to integrity, the adversary is restricted as follows:

[6]Note that Neumann and Volkamer in [4] refer to trust assumptions rather then assumptions.

- The adversary is neither able to corrupt smart cards nor smart card readers. (TA2)
- The adversary is not able to corrupt all ballot boxes. (TA5)

Here, it should be noted that *simulation attacks*, defined as part of coercion-resistance, are covered by eligibility/democracy (depending on the fact if the adversary is also an eligible voter) integrity. Furthermore, *both* criteria build upon the following assumption:

- The adversary is restricted to probabilistic polynomial time computations and cryptographic primitives work. (TA7)

In [19], an Internet voting scheme is said to be *end-to-end verifiable* if integrity is ensured without posing restrictions on the adversary. We relax this statement and consider an Internet voting scheme end-to-end verifiable if integrity is ensured under the sole assumption that the adversary is restricted to probabilistic polynomial time computations and cryptographic primitives work. According to this definition, the NV12 scheme is not end-to-end verifiable.

Forced-abstention Resistance. The adversary does not get any evidence if the voter abstained from the election. Apart from randomization and simulation attacks, coercion-resistance as defined in [2] ensures resistance against forced-abstention attacks. To ensure forced-abstention resistance, the adversary is restricted in the same way as for secrecy, while one further assumption must be stated.

- The adversary cannot control or manipulate all nodes in the anonymization network. (TA4)
- There is a point in the voting phase, in which the voter is not consciously under adversarial control.

  Justification: If this assumption would not hold, the voter would never have a chance to cast her real intention and would implicitly prove to the adversary that she abstained from the election. Even though not mentioned in [4], this assumption has been outlined by Clarkson et al. [3] as part of their threat model.

3 Project Setting and Preliminary Considerations

In [4], Neumann and Volkamer address practical shortcomings of the JCJ / Civitas scheme by integrating smart cards. However, their work did not consider performance and consequently the performance of their proposal has not yet been investigated. In the remainder of this work, we merely consider smart card performance rather than client machine performance. This is justified by the fact that standard computers run routines many times faster than smart cards. As outlined in the previous section, the NV12 scheme involves the smart card in the registration and voting phases. Amongst others, the smart card is used for the generation and verification of zero-knowledge proofs, the encryption of data, establishing anonymous channels to the ballot boxes, and cut-and-choose techniques as the Benaloh challenge.

Before diving into the revisions of this work, we outline the project setting within which this work has been conducted. This setting allows us to slightly improve security and the overall performance of the NV12 scheme. In the second part of this section, we provide the reader with preliminary performance considerations and propose corresponding revisions to the NV12 scheme. We base our modifications on security arguments to prove that these modifications do not influence the security model underlying the NV12 scheme.

3.1 Project Setting

We first present the used smart card technology. Thereafter, we outline the type of elections considered in the remainder of this paper.

Smart Cards. A number of different smart card operating systems exist. Among the most established and wide-spread operating systems certainly, there are MULTOS[7] and the Java Card OS[8]. This paper has been developed as part of a research project in which Java Cards NXP JCOP J20A80G are available which are built upon Java Card version 2.2.2. Hence, it has been decided to rely on NXP JCOP J20A80G cards.

Simple Ballot Elections. The NV12 scheme provides encoded-as-intended integrity against the client machine by the fact that the voter's selection is encrypted by the smart card and can be audited in cut-and-choose manner. In the general case, this proceeding is adequate and recommendable. However, in the project setting, we consider simple ballots, e.g., single-vote plurality ballots. In the case of simple ballots, the cut-and-choose verification process can be simplified as follows: Rather than inserting her intention over her client machine, the voter inserts her intention directly to the smart card over her smart card reader[9]. In the latter case, one is able to save the computations of the Benaloh challenge. Note, the consequence of this simplification is the elimination of the assumption "*The adversary cannot control the client machine.*" with respect to secrecy. Hence, the client machine must only be trusted with respect to forced-abstention resistance. At this point, we do not see a way to refrain from this assumption for the following reason: If the machine, over which the voter's smart card is connected, would be under adversarial control, the adversary would be able to notice if the machine forwards data between the smart card and any Internet service, in particular the Internet voting service provider

3.2 Revising the NV12 Smart Card Routines

The NV12 scheme did not consider implementation-specific details. As a consequence thereof, the work proposed to implement most of the routines on the smart card. Based on the fact that smart cards are generally highly resource-restricted, it is advisable to lower the number of smart card operations if the underlying security model is not affected by these modifications. The goal of this subsection is therefore to revise the NV12 scheme with regard to smart card performance. This revision is a subtle process due to the fact that security must not be compromised by any modification. Therefore, each modification is substantiated by an argument relating it to the underlying security model.

In the remainder of this subsection, cryptographic operations are analyzed with respect to the need to execute these operations on the smart card.

Designated-Verifier Re-encryption Proof. The DVRP is used to convince the smart card about the fact that S_i is a re-encryption of S'_i which is shown to encrypt c_i. The proof prevents malicious registration tellers from providing invalid credential shares. The verification of a DVRP includes two subroutines [3]: On the one side, there is the ElGamal encryption of the received credential share c_i. Each ElGamal encryption builds upon one multiplication and two fast exponentiations (refer to [3], p. 34, Algorithm: ElGamal Encryption). On the other side, there is the essential verification to the proof. This part of the DVRP relies on four divisions, six fast exponentiations, two additions, and one multiplication (refer to [3], p. 39, Protocol: DVRP, Step 3). In view of the fact that the smart card is trusted with respect to secrecy, the proof generation (and proof verification) can be simplified: Rather than generating a DVRP, the registration teller only needs to prove to the card that the published S_i is an encryption of c_i. Because of the fact that the smart card is trusted and consequently there exists no routine to get any further information from the card, the DVRP can be reduced to an encryption of c_i by using the randomness r that was used to generate S_i. Hence, together with c_i, the registration teller outputs the randomness r and the card solely verifies if $\{c_i\}^r_{pk_{EK}}$ equals S_i and if S_i is signed by RT_i and found on the bulletin board[10].

[7] `http://www.multos.com/`

[8] `http://www.oracle.com/technetwork/java/javame/javacard/overview/getstarted/index.html`

[9] For this purpose, we assume that prior to the election, enumerations of choices are publicly announced.

[10] The smart card obtains the signed S_i from the bulletin board via the client machine.

Proof of Well-formedness. The PWF in the voting phase is used to prove that the voter's cast vote encodes one of the election options. As outlined by Haenni and Koenig [20], the PWF serves to prevent adversaries from forcing voters into casting uniquely spoilt ballots together with their real credentials. The PWF for ballots containing L candidates builds upon L ElGamal encryptions, $2L$ divisions, $4L$ fast exponentiations, $4L$ multiplications, $2L$ additions, and $2L$ subtractions (refer to [3], p. 41, Protocol: ReencPf, Step 1+2). It is furthermore worth mentioning that during the generation of PWF, there are $8L + 4$ numbers stored. Each number needs 1536 bits and for $L = 20$ the PWF needs 31488 Bytes memory. A modern smart card has 80 KB memory from which are 64 KB available. Ballots of the German Federal election in 2009 could not be handled properly; in Wiesbaden, voters had 117 possible combinations to vote[11] which would result in the smart card running out of memory. Given the fact that smart cards are trusted with respect to secrecy and forced-abstention resistance, smart cards only generate ballots encoding valid choices or one single invalid marking for all invalid choices. Therefore, the smart card must obtain the ballot form in the registration process, namely *SRA* provides the smart card with the signed ballot form such that the smart card can differentiate between invalid and valid votes. As a consequence thereof, one can refrain from the generation of PWFs.

Anonymous Channels. The NV12 scheme proposes the establishment of anonymous channels over the smart card to ensure forced-abstention resistance and to ease the burden on the voter of establishing anonymous channels. Given the fact that a malicious machine might always violate forced-abstention-resistance (see the argument in the *Simple Ballot* paragraph), there is no need to establish anonymous channels over the smart card. In accordance to the original JCJ / Civitas scheme, anonymous channels between the client-side and the ballot boxes are therefore entirely established over the client machine. Several anonymization implementations are currently available on the market, e.g., TOR[12] and I2P[13]. For the purpose of usability and to maintain forced-abstention resistance, we propose the integration of an anonymization implementation into the JavaScript.

Proof of Knowledge of Credential and Vote. The zero-knowledge proof of knowledge of the credential and the vote (PKCV) in the voting phase shows that the proving entity knows both the credential and the vote inside the ballot. Generating these proofs prevents observers from maliciously re-using credentials generated in the registration phase. The generation of PKCV proofs relies on two fast exponentiations, two multiplications, and two subtractions (refer to [3], p. 41, Protocol: VotePf, Step 1). Given the fact that no other smart card than the voter's smart card obtains full knowledge of the respective credential, this proof can intentionally only be generated by the smart card. Hence, the proof generation remains on the smart card.

Revised NV12 Scheme. Integrating the outlined modifications into the NV12 scheme results in the revised NV12 scheme. In the following section, the real-world feasibility of the revised NV12 scheme is investigated.

4 Performance Analysis of the Revised NV12 Scheme

After the setting has been specified, several revisions permitted us to decrease the smart card's computational effort. The goal of this section is to analyze the revised NV12 scheme with respect to its performance. To do so, first, timings for basic operations on smart cards are assessed based on the work by Bichsel et al. [21]. Secondly, we decompose the revised NV12 scheme into basic smart card operations allowing us to estimate the the performance of the revised NV12 scheme.

4.1 Timings of Smart Card Operations

In this section we obtain timings for the modular operations *addition* ($a + b \bmod p$), *subtraction* ($a - b \bmod p$), *multiplication* ($a \cdot b \bmod p$), and *fast exponentiation* ($a^b \bmod p$) on modern smart

[11] `http://www.bundestagswahl-2009.de/stimmzettel/`
[12] https://www.torproject.org/
[13] http://www.i2p2.de/

cards from Bichsel et al.'s work [21]. To benefit from hardware acceleration as much as possible, subtraction and addition are both implemented using the RSA-CRT encryption. Therefore we assume that addition and subtraction take approximately the same time. Furthermore, Bichsel et al. map modular multiplication on the crypto coprocessor. To calculate modular division $\frac{a}{b} \bmod p$, two approaches might be considered: In the first approach, one has to solve $a \equiv b \cdot x \bmod p$ using the extended euclidean algorithm, a standard algorithm for this task as described in [22]. This algorithm needs needs one addition, one subtraction, two multiplications and one integer division per step. The second approach relies on Fermat's little theorem:

$$a^{p-1} \equiv 1 \mod p$$

Therefore, it holds:

$$a^{-1} \mod p \equiv a^{-1} \cdot 1 \mod p \equiv a^{-1} \cdot a^{p-1} \mod p \equiv a^{p-2} \mod p$$

Hence, $\frac{a}{b} \bmod p$ corresponds to $a \cdot b^{p-2} \bmod p$, which corresponds to 1 subtraction, 1 multiplication, and 1 exponentiation, which results in 1.029 seconds for 1536 bit numbers.

The average time for 1536 bit numbers is provided in Table 1. We are aware of the fact that a modulus of length 1536 as given in [21] does not provide adequate security. We believe, however, that the progress in smart card technology keeps pace with the security requirements such that modern smart cards might perform similarly for a modulus of 2048 bit.

Bitlength \Operation	Addition	Subtraction	Multiplication	Fast Expo.	Division
1536	0.082	0.082	0.517	0.430	1.029

Table 1: Average Operation Times in Seconds (refer to [21])

One should notice that multiplication performs slightly slower than fast exponentiation. This stems from the fact that multiplication and fast exponentiation are both mapped on the crypto coprocessor. As opposed to the fast exponentiation, which is mapped onto RSA encryption, the multiplication is mapped onto RSA chinese remainder theorem (CRT) decryption, which has slightly lesser performance than RSA encryption. Due to this fact, one RSA decryption takes as long as a multiplication and one RSA encryption takes as long as fest exponentiation.

One might consider a further performance improvement, namely elliptic curve cryptography (ECC). In contrast to finite fields, elliptic curves over finite fields allow to decrease the key size and consequently increase the performance of cryptosystems without compromising security. ECC could be realized on a smart card even without direct crypto coprocessor support. Scalar multiplications on elliptic curves are a additions of the same point. An EC addition can be performed only by using addition and subtraction over a finite field. Due to its properties, ECC turns out to be highly valuable for low power devices. Nevertheless, we had to exclude ECC from our further considerations because the Civitas implementation in its current state does not integrate ECC and as a consequence the Civitas backend would have to be modified which is beyond the scope of this work.

4.2 Decomposing the Revised NV12 Scheme

In this section, we decompose the revised NV12 scheme into its basic smart card operations.

Registration Phase. In the registration phase, the NV12 scheme foresees to establish secure connections to each registration teller via the Needham-Schroeder-Lowe (NSL) protocol [23] and to verify one DVRP for each registration teller. Additionally the real credential is calculated right after all credential shares are received, namely $|TRT|$ credential shares are multiplied, while SRA's credential is merged with the voter's PIN resulting in one further division. The NSL protocol builds

upon one RSA encryption and one RSA decryption (refer to [3], p. 45, Protocol: Register, Step 1-8).

As justified in section 3, the DVRP of each trusted registration teller is replaced by a simple re-encryption of the obtained credential share. ElGamal builds upon one multiplication and two fast exponentiations (refer to section 3).

In summary, the number of operations needed throughout the registration phase is given as follows:

$$\begin{aligned} t_{registration} &= (1 \cdot t_{mul} + 2 \cdot t_{exp} + t_{RSAenc} + t_{RSAdec}) \cdot |TRT| + \\ & \quad t_{mul} \cdot |TRT| + \\ & \quad t_{div} \end{aligned}$$

Voting Phase. According to [4], the voting phase builds upon two ElGamal encryptions, one proof of well-formedness (PWF), and one zero-knowledge proof of knowledge of the credential and the vote (PKCV). ElGamal needs one multiplication and two fast exponentiations (see above). The PKCV needs two fast exponentiations, two multiplications, and two subtractions (refer to [3], p. 41, Protocol: VotePf, Step 1). Additionally, the voter's PIN is multiplied to the credential factor.

As justified in section 3, the PWF is removed in accordance to the security model. Apart from this modification, one might consider a further performance gain by initially selecting a unique fake credential and pre-computing randomized credential encryptions: Depending on the fact if the credential used for the last vote was correct, the fake or the real credential would need to be encrypted to vote. During the time the voter makes her selection, the smart card could already encrypt this credential, store both encrypted credentials (fake and real) and use one of the two encrypted credentials after the voter submitted her voting PIN. This approach is subtly flawed: The adversary could ask the voter to cast a vote for a specific candidate associated with her real and her fake credential. If the voter follows the adversary's instruction, throughout the tallying phase, no duplicate would be removed and the adversary could be sure that one vote will be tallied.

In conclusion, the performance of the revised voting phase is:

$$\begin{aligned} t_{voting} &= 2 \cdot t_{mul} + 4 \cdot t_{exp} + \\ & \quad 2 \cdot t_{mul} + 2 \cdot t_{exp} + 2 \cdot t_{sub} \\ & \quad t_{mul} \end{aligned}$$

4.3 Obtaining the Overall Performance of the Revised NV12 Scheme

After the smart card timings for the basic operations have been obtained and the number of basic smart card operations has been assessed, we are able to draw conclusions about the performance of the revised NV12 scheme. We assume five trusted registration tellers to be a reasonable choice for high-stake elections.

$$\begin{aligned} t_{registration} &= (1 \cdot 0.517 + 2 \cdot 0.430 + 0.430 + 0.517) \cdot 5 + \\ & \quad 0.517 \cdot 5 + 1.029 = 15.234 \text{ s} \\ t_{voting} &= 2 \cdot 0.082 + 4 \cdot 0.517 + 6 \cdot 0.430 + 0.517 = 5.329 \text{ s} \end{aligned}$$

The total time for the registration phase is around 15 seconds, while the time for the voting phase is around 5 seconds. It can be concluded that the performance of the revised NV12 scheme is feasible and can therefore be used in real-world elections.

5 Conclusion and Future Work

After decades of theoretical research on the topic of Internet voting, scientific Internet voting schemes come up trumps with promising security claims. One of the schemes providing resistance

to voter coercion is JCJ / Civitas [2, 3]. The scheme relies on a number of assumptions and poses an insurmountable hurdle to the voter when it comes to coercion due to the abstract credential handling. In 2012, Neumann and Volkamer presented a proposal [4] that addresses these practical challenges by incorporating smart cards into the JCJ / Civitas scheme. Certainly, the proposal serves as a step towards the real-wold use of the JCJ / Civitas scheme. Nevertheless, their work did not center on performance and as such their proposal is not tailored towards performance.

The present work is directed to close this gap. In the first part of our work, we revised the NV12 scheme from a performance perspective and were able to replace, remove, or outsource smart card operations in order to improve the overall performance. We showed that these modifications did not affect the security model underlying the NV12 scheme. Based on these revisions, in the second part of our work, we assessed smart card timings for basic operations from recent literature and decomposed the revised NV12 scheme into basic smart card operations. Summarizing these insights, we calculated smart card running times of around 15 seconds for the registration phase and 5 seconds for the voting phase of the revised NV12 scheme. We are convinced that these results prove the NV12 scheme feasible to be applied within real-world elections.

Nonetheless, we plan to improve the overall performance for the registration phase and the voting phase by further optimizations, e.g. the outsourcing of basic operations to the client. We furthermore strive for discarding the client machine assumption also with respect to forced-abstention. To date, the revised NV12 scheme assumes the voter not to mistype or forget her PIN. This assumption might be too strong and consequently should be reconsidered in future research. In the future, the revised NV12 scheme will be implemented and used within test elections. Ultimately, we plan to evaluate and improve the usability of the revised NV12 scheme by user studies.

Acknowledgment. We would like to thank Oliver Spycher and Rolf Haenni for the fruitful discussions that improved this work significantly. This paper has been developed within the project 'ModIWa2' - Juristisch-informatische Modellierung von Internetwahlen - which is funded by the Deutsche Forschungsgemeinschaft (DFG, German Science Foundation).

References

[1] Chaum, D.L.: Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM **24**(2) (1981) 84–90

[2] Juels, A., Catalano, D., Jakobsson, M.: Coercion-resistant electronic elections. In: ACM Workshop on Privacy in the Electronic Society, ACM (2005) 61–70

[3] Clarkson, M.R., Chong, S., Myers, A.C.: Civitas: Toward a secure voting system. In: IEEE Symposium on Security and Privacy, IEEE Computer Society (2008) 354–368

[4] Neumann, S., Volkamer, M.: Civitas and the real world: Problems and solutions from a practical point of view. In: International Conference on Availability, Reliability and Security (AReS 2012), IEEE Computer Society (2012) 180–185

[5] Smith, W.D.: New cryptographic election protocol with best-known theoretical properties. Frontiers in Electronic Elections (2005)

[6] Weber, S.G., Araujo, R., Buchmann, J.: On coercion-resistant electronic elections with linear work. In: International Conference on Availability, Reliability and Security (AReS 2007, IEEE Computer Society (2007) 908–916

[7] Araujo, R., Foulle, S., Traor, J.: A practical and secure coercion-resistant scheme for internet voting. In: Towards Trustworthy Elections. Volume 6000 of LNCS., Springer (2010) 330–342

[8] Spycher, O., Koenig, R.E., Haenni, R., Schläpfer, M.: A new approach towards coercion-resistant remote e-voting in linear time. In: Financial Cryptography. Volume 7035 of LNCS., Springer (2011) 182–189

[9] Koenig, R.E., Haenni, R., Fischli, S.: Preventing board flooding attacks in coercion-resistant electronic voting schemes. In: SEC, Springer (2011) 116–127

[10] Backes, M., Hritcu, C., Maffei, M.: Automated verification of remote electronic voting protocols in the applied pi-calculus. In: IEEE Computer Security Foundations Symposium, IEEE Computer Society (2008)

[11] Smyth, B., Ryan, M., Kremer, S., Kourjieh, M.: Towards automatic analysis of election verifiability properties. In: Joint Conference on Automated Reasoning for Security Protocol Analysis and Issues in the Theory of Security, Springer (2010) 146–163

[12] Kuesters, R., Truderung, T.: An epistemic approach to coercion-resistance for electronic voting protocols. In: 30th IEEE Symposium on Security and Privacy, IEEE Computer Society (2009) 251–266

[13] Shirazi, F., Neumann, S., Ciolacu, I., Volkamer, M.: Robust electronic voting: Introducing robustness in civitas. In: International Workshop on Requirements Engineering for Electronic Voting Systems (REVOTE), IEEE Computer Society (2011) 47 –55

[14] Bursuc, S., Grewal, G., Ryan, M.: Trivitas: Voters directly verifying votes. In: Third International Conference on E-voting and Identity. Volume 7187 of LNCS., Springer (2012)

[15] da Silva Mendes, J.M.B.: Trusted Civitas: Client trust in Civitas electronic voting protocol. Master's thesis, Instituto Superior Tcnico (2011)

[16] Koenig, R.E., Haenni, R.: How to store some secrets. Cryptology ePrint Archive, Report 2012/375 (2012) `http://eprint.iacr.org/`.

[17] Benaloh, J.: Simple verifiable elections. In: USENIX / Accurate Electronic Voting Technology Workshop 2006 on Electronic Voting Technology Workshop. (2006) 5–5

[18] Florencio, D., Herley, C.: A large-scale study of web password habits. In: International Conference on World Wide Web, ACM (2007) 657–666

[19] Budurushi, J., Neumann, S., Olembo, M., Volkamer, M.: Pretty understandable democracy. In: International Conference on Availability, Reliability and Security (AReS 2013), IEEE Computer Society (2013) to be published.

[20] Haenni, R., Koenig, R.E.: A generic approach to prevent board flooding attacks in coercion-resistant electronic voting schemes. Computers & Security **33** (2013) 59–69

[21] Bichsel, P., Camenisch, J., Groß, T., Shoup, V.: Anonymous credentials on a standard java card. In: ACM Conference on Computer and Communications Security. CCS '09, ACM (2009) 600–610

[22] Aboud, S.J.: Baghdad method for calculating multiplicative inverse. In: International Conference on Information Technology: Coding and Computing, IEEE Computer Society (2004) 816–819

[23] Lowe, G.: An attack on the needham-schroeder public-key authentication protocol. Inf. Process. Lett. **56**(3) (1995) 131–133